Published in final edited form as:
Cell Syst. 2019 April 24; 8(4): 292–301.e3. doi:10.1016/j.cels.2019.03.006.

End-to-end differentiable learning of protein structure

Mohammed AlQuraishi[1,2,*]

[1]Laboratory of Systems Pharmacology, Harvard Medical School, Boston, MA 02115.

[2]Department of Systems Biology, Harvard Medical School, Boston, MA 02115.

SUMMARY

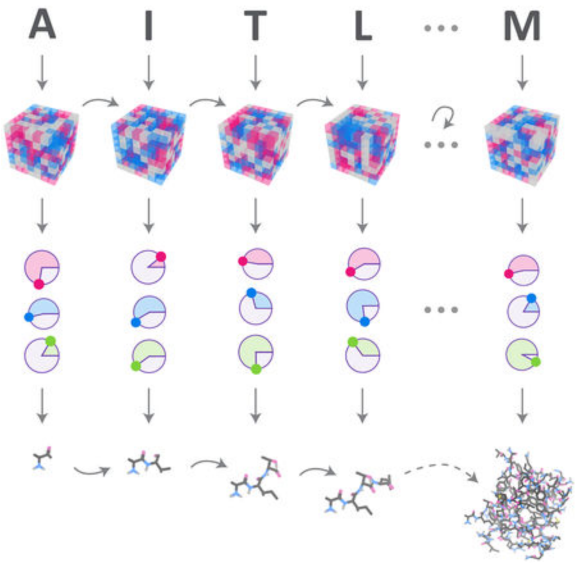
Predicting protein structure from sequence is a central challenge of biochemistry. Co-evolution methods show promise, but an explicit sequence-to-structure map remains elusive. Advances in deep learning that replace complex, human-designed pipelines with differentiable models optimized end-to-end suggest the potential benefits of similarly reformulating structure prediction. Here we introduce an end-to-end differentiable model for protein structure learning. The model couples local and global protein structure via geometric units that optimize global geometry without violating local covalent chemistry. We test our model using two challenging tasks: predicting novel folds without co-evolutionary data and predicting known folds without structural templates. In the first task the model achieves state-of-the-art accuracy and in the second it comes within 1–2Å; competing methods using co-evolution and experimental templates have been refined over many years and it is likely that the differentiable approach has substantial room for further improvement, with applications ranging from drug discovery to protein design.

Graphical Abstract

[*]Corresponding author and lead contact: alquraishi@hms.harvard.edu.
Author Contributions:
M.A. conceived the model, conducted the experiments, and wrote the paper.

Publisher's Disclaimer: This is a PDF file of an unedited manuscript that has been accepted for publication. As a service to our customers we are providing this early version of the manuscript. The manuscript will undergo copyediting, typesetting, and review of the resulting proof before it is published in its final citable form. Please note that during the production process errors may be discovered which could affect the content, and all legal disclaimers that apply to the journal pertain.

Declaration of Interests:
The author declares no competing interests.



eTOC Blurb

Prediction of protein structure from sequence is important for understanding protein function, but it remains very challenging, especially for proteins with few homologs. Existing prediction methods are human-engineered, with many complex parts developed over decades. We introduce a new approach based entirely on machine learning that predicts protein structure from sequence using a single neural network. The model achieves state of the art accuracy, and does not require co-evolution information or structural homologs. It is also much faster, making predictions in milliseconds vs. hours or days, which enables new applications in drug discovery and protein design.

INTRODUCTION

Proteins are linear polymers that fold into very specific and ordered three dimensional conformations based on their amino acid sequence (Branden and Tooze, 1999; Dill, 1990). Understanding how this occurs is a foundational problem in biochemistry. Computational approaches to protein folding not only seek to make structure determination faster and less costly; they aim to understand the folding process itself. Existing computational methods fall into two broad categories (Gajda et al., 2011b, 2011a). The first category builds explicit sequence-to-structure maps using computational procedures to transform raw amino acid sequences into 3D structures. This includes physics-based molecular dynamics simulations (Marx and Hutter, 2012), which are restricted by computational cost to small proteins, and fragment assembly methods (Gajda et al., 2011a), which find energy-minimizing conformations by sampling statistically-derived protein fragments. Fragment assembly usually achieves high accuracy only when homologous protein structures are used as templates. Such template-based methods use one or more experimental structures—found through homology searches—as the basis for making predictions.

The second category of methods eschews explicit sequence-to-structure maps and instead identifies co-evolving residues within protein families to derive residue-residue contact maps, using co-evolution as an indicator of contact in physical space (Hopf et al., 2014; Marks et al., 2011). With a large and diverse set of homologous sequences—typically tens to hundreds of thousands—co-evolution methods can accurately predict contact maps (Juan et al., 2013). A correct contact map can guide fragment assembly methods to an accurate 3D structure 25–50% of the time (Ovchinnikov et al., 2017). However, because co-evolutionary methods do no construct a model of the relationship between individual sequences and structures, they are unable to predict structures for which no sequence homologs exist, as in new bacterial taxa or *de novo* protein design. Moreover, even for well-characterized proteins, such methods are generally unable to predict the structural consequences of minor sequence changes such as mutations or indels, because they operate on protein families rather than individual sequences (they do however show promise in predicting the functional consequences of mutations (Hopf et al., 2017)). Thus, there remains a substantial need for new and potentially better approaches.

End-to-end differentiable deep learning has revolutionized computer vision and speech recognition (LeCun et al., 2015), but protein structure pipelines continue to resemble the ways in which computers tackled vision and speech prior to deep learning, by having many human-engineered stages, each independently optimized (Xu and Zhang, 2012; Yang et al., 2015) (Figure 1). End-to-end differentiable models replace all components of such pipelines with differentiable primitives to enable joint optimization from input to output. In contrast, use of deep learning for structure prediction has so far been restricted to individual components within a larger pipeline (Aydin et al., 2012; Gao et al., 2017; Li et al., 2017; Lyons et al., 2014; Zhao et al., 2010), for example prediction of contact maps (Liu et al., 2017; Wang et al., 2016). This stems from the technical challenge of developing an end-to-end differentiable model that rebuilds the entire structure prediction pipeline using differentiable primitives. We have developed such a model by combining four ideas: (i) encoding protein sequence using a recurrent neural network, (ii) parameterizing (local) protein structure by torsional angles, to enable a model to reason over diverse conformations without violating their covalent chemistry, (iii) coupling local protein structure to its global representation via recurrent geometric units, and (iv) using a differentiable loss function to capture deviations between predicted and experimental structures. We find that the new approach outperforms other methods, including co-evolution ones, when predicting novel folds even though it uses only primary sequences and position-specific scoring matrices (PSSMs) that summarize individual residue propensities for mutation. We also find that when predicting known folds, the new approach is on average within 1–2Å of other approaches, including template-based ones, despite being template-free.

RESULTS

Recurrent geometric networks

Our model takes as input a sequence of amino acids and PSSMs and outputs a 3D structure. It is comprised of three stages—computation, geometry, and assessment—that we term a recurrent geometric network (RGN). The first stage is made of computational units that, for each residue position, integrate information about its amino acid and PSSM with information coming from adjacent units. By laying these units in a recurrent bidirectional topology (Figure 2), the computations for each residue integrate information from residues upstream and downstream all the way to the N- and C-terminus, covering the entire protein. By further stacking units in multiple layers (not shown), the model implicitly encodes a multi-scale representation of proteins. Each unit outputs three numbers, corresponding to the torsional angles of the residue. We do not specify *a priori* how angles are computed. Instead, each unit's computation is described by an equation whose parameters are optimized so that RGNs accurately predict structures.

The second stage is made of geometric units that take as input the torsional angles for a given residue and the partially completed backbone resulting from the geometric unit upstream of it, and output a new backbone extended by one residue, which is fed into the adjacent downstream unit (AlQuraishi, 2019a; Parsons et al., 2005). The last unit outputs the completed 3D structure of the protein. During model training, a third stage computes deviations between predicted and experimental structures using the distance-based root mean square deviation (dRMSD) metric. The dRMSD first computes pairwise distances between all atoms in the predicted structure and all atoms in the experimental one (separately), and then computes the root mean square of the distance between these sets of distances. Because dRMSD is distance-based, it is invariant to reflections, which can lead RGNs to predict reflected structures (effectively wrong chirality) that must be corrected by a counter-reflection. RGN parameters are optimized to minimize the dRMSD between predicted and experimental structures using backpropagation (Goodfellow et al., 2016). Hyperparameters, which describe higher-level aspects of the model such as the number of computational units, were determined through manual exploration of hyperparameter space. See Supplementary Text for a complete mathematical treatment.

Assessment of model error

Machine learning models must be trained against as large a proportion of available data as possible to fit model parameters and then evaluated against a distinct test set to assess accuracy. Reliable evaluation is frequently complicated by unanticipated information leakage from the training set into the test set, especially for protein sequences which share an underlying evolutionary relationship. Partly to address this problem, the Critical Assessment of Protein Structure Prediction (CASP) (Moult et al., 1995) was organized to assess methods in a blinded fashion, by testing predictors using sequences of solved structures that have not been publicly released. To assess RGNs we therefore sought to recreate the conditions of past CASPs by assembling the ProteinNet datasets (AlQuraishi, 2019b). For every CASP from 7 through 12, we created a corresponding ProteinNet test set comprised of CASP structures, and a ProteinNet training set comprised of all sequences and structures publicly available prior to the start of that CASP. Using multiple CASP datasets enables a deeper and more thorough assessment that spans a broad range of dataset sizes than relying on the most recent CASP alone. We also adopted the CASP division of test structures into free modeling (FM) targets that assess prediction of novel folds, and template-based (TBM and TBM-hard) targets that assess prediction of folds with known homologs in the Protein Data Bank (PDB) (Bernstein et al., 1977). We set aside a subset of the training data as a validation set, to determine when to stop model training and to further insulate training and test data.

ProteinNet datasets were used for all analyses described here. RGN hyperparameters were fit by repeated evaluations on the ProteinNet 11 validation set followed by three evaluations on the ProteinNet 11 test set. Once chosen, the same hyperparameters were used to train models on ProteinNet 7–12 training sets, with a single evaluation made at the end on each test set (excepting ProteinNet 11) to generate Table 1. Subsequently additional test set evaluations were made to generate Table S1, with one evaluation per number reported. No additional test set evaluations were made. Overall, this represents a rigorous approach to evaluation with the lowest possible risk of information leakage.

Predicting new folds without co-evolution

We first assessed RGNs on a difficult task that has not consistently been achieved by any existing method: predicting novel protein folds without co-evolutionary data. FM structures served as targets for this exercise. Table 1 compares the average dRMSD of RGN predictions on FM structures to the top five automated predictors in CASP 7–12, known as “servers” in CASP parlance (“humans” are server/human-expert pipelines—we do not compare against this group as our processing is automated). In Figure 3a we break down the predictions by target against the top performing server and in Figure 3c against the dRMSD distribution of all CASP servers.

On all CASPs, RGNs had the best performance, even compared to servers that use co-evolution data (in CASP 11 (Kryshtafovych et al., 2016; Ovchinnikov et al., 2016) and CASP 12 (Schaarschmidt et al., 2017)). RGNs outperformed other methods at both short and long, multi-domain proteins, suggesting their performance is not limited to one regime (e.g. short single domain proteins), despite having no explicit knowledge of domain boundaries. While the margin between RGNs and the next best server is small for most CASPs, such small gaps are representative of the differences between the top five performers in Table 1. In general, small gains in accuracy at the top end are difficult, with only minimal gains obtained over a ten-year time span from CASP 6 to CASP 11 (Kryshtafovych et al.). More substantial gains were seen in CASP 12 due to the use of co-evolutionary information (Moult John et al., 2018), but RGNs match these advances without using co-evolutionary data and by operating in a fundamentally distinct and complementary way. The accuracy gap between RGNs and other servers is highest on CASP 11, which benefits from having the RGN hyperparameters fit on the ProteinNet11 validation set, suggesting similar gains may be had by optimizing RGN hyperparameters for each dataset (this would not correspond to overfitting, as only the validation set is used to fit hyperparameters, but would require substantially more compute resources for training.) ProteinNet datasets of earlier CASPs are smaller which may have also reduced accuracy. To assess the contribution of dataset size to model error, we used RGNs trained on earlier ProteinNet datasets to predict later CASP test sets (Table S1). As expected, accuracy drops as datasets shrink.

The dRMSD metric does not require structures to be pre-aligned, and is consequently able to detect regions of high local concordance even when global concordance is poor. Because dRMSD assesses predictions at all length scales however, it penalizes large global deviations in proportion to their distance, which can result in very high error for far apart regions. To obtain a complementary assessment of model accuracy, we also tested RGNs using TM scores (Zhang and Skolnick, 2004), which are defined by the following equation:

$$TM\ score = \max\left[\frac{1}{L_{target}}\sum_{i}^{L_{aligned}}\frac{1}{1+\left(\frac{d_i}{d_0\left(L_{target}\right)}\right)^2}\right]$$

where L_{target} and $L_{aligned}$ are the lengths of the full protein and the aligned region, respectively, d_i is the distance between the ith residues in the experimental and predicted structures, and $d_0\left(L_{target}\right) = 1.24\sqrt[3]{L_{target} - 15} - 1.8$ is used to normalize scores. TM scores do require structures to be pre-aligned, and thus can penalize predictions with high local concordance if a global alignment cannot be found, but they are less sensitive to large deviations because they only compute error over the aligned regions. TM scores range from 0 to 1, with a score of < 0.17 corresponding to a random unrelated protein, and > 0.5 generally corresponding to the same protein fold (Xu and Zhang, 2010). Since TM scores are not invariant to reflections, we compute them for both the original and reflected RGN structures and use the higher of the two. Table S2 compares TM scores of RGN predictions to CASP servers. In general, RGNs rank among the top five servers, but do not consistently outperform all other methods as they do on dRMSD, possibly reflecting the lack of partial credit assignment by TM scores.

Predicting known folds without templates

We next assess RGNs on predicting known protein folds without experimental templates, a challenging task that provides an advantage to template-based methods (Zhou et al., 2010). TBM structures served as targets for this purpose. Table 1 and Table S2 compare RGN predictions to top CASP servers using dRMSD and TM score, respectively, while Figure 3b breaks down predictions by target and Figure 3c shows the distribution over all CASP servers. A representative sampling of the full quality spectrum of FM and TBM predictions is shown in Figure 3e. In general, RGNs underperform the very top CASP servers, all of which use templates, although ~60% of predictions are within 1.5Å of the best-performing server.

Since RGNs do not use templates, this suggests that they learn generalizable aspects of protein structure, and their improved accuracy on TBM targets relative to FM reflects denser sampling in TBM regions of protein space. To investigate this possibility, we partitioned ProteinNet validation sets into groups based on maximum sequence identity to the training set, and computed dRMSDs within each group across CASPs 7–12 (Figure 3d) and by individual CASP (Figure S1). RGN performance robustly transfers to sequences with >40% sequence identity, predicting structures with a median dRMSD of ~5Å, and then begins to deteriorate. There was little difference in dRMSD between 50% and 90% sequence identity, with substantial error remaining at 90%, which is suggestive of underfitting.

Template-based methods are particularly accurate where template and query sequences overlap, and are inaccurate where they do not; unfortunately, non-overlapping regions are often the regions of high biological interest. Errors in these critical non-overlapping regions can be masked by large overlapping regions, inflating overall accuracy (Contreras-Moreira et al., 2005; Dill and MacCallum, 2012; Liu et al.; Perez et al., 2016). To determine whether RGNs suffer from similar limitations, we split TBM domains into short fragments ranging in size from 5 to 50 residues and computed the RMSD for every fragment (with respect to the experimental structure) from the best template, the best CASP prediction, and the RGN prediction (Figure 4). We found CASP predictions to be correlated (average $R^2 = 0.44$) with template quality across length scales as previously reported (Kryshtafovych et al.), while RGN predictions were not (average $R^2 = 0.06$). This distinction persists even when predictions with >3Å accuracy are excluded (average $R^2 = 0.49$ for best CASP predictions; average $R^2 = 0.02$ for RGN predictions). Thus RGNs perform equally well on regions of proteins with experimental templates and on those without.

RGNs learn an implicit representation of protein fold space

Applications of deep learning in sensory domains often result in models whose internal representation of the data is interpretable, e.g. placing semantically similar words nearby in a natural language model. To ascertain whether RGNs behave similarly, we extracted the internal state of their computational units after processing each protein sequence in the ProteinNet12 training set. For each protein, we obtained multiple high-dimensional vectors, one per layer / direction of the RGN. We then used linear dimensionality reduction techniques to visualize these vectors in two dimensions, separately for each layer / direction (Figure 5a), and by concatenating all layers together (Figure 5b). When we color each protein (dot) according to the fraction of secondary structure present in its original PDB structure, clear visual patterns emerge (Figure 5b). This is notable because secondary structure was neither used as input to aid model prediction nor as an output signal to guide training; i.e. the model was not explicitly encoded with the concept of secondary structure, yet it uses secondary structure as the dominant factor in shaping its representation of protein fold space.

We next used the CATH database (Dawson et al., 2017), which hierarchically classifies proteins into structural families, to partition data points into CATH classes and visualize their distribution in RGN space. At the topmost CATH level, divided into “Mainly Alpha”, “Mainly Beta”, “Alpha Beta”, and “Few Secondary Structures”, we see clearly demarcated regions for each class (represented by differently colored contour plots), with “Alpha Beta” acting unsurprisingly as the bridge (leftmost panel in Figure 5c.) We then reapplied dimensionality reduction to data in each class and visualized the distributions of their respective second-level CATH categories (three right panels in Figure 5c.) We again see contiguous regions for each category, albeit with greater overlap, likely owing to the continuous nature of protein structure space and reduction of RGN space to just two dimensions. These visualizations suggest RGNs are learning a useful representation of protein sequence space that may yield insights into the nature of protein structure space.

RGNs are 6–7 orders of magnitude faster than existing methods

Existing structure prediction pipelines are multi-staged (Figure 1), first detecting domains that can be separately modelled, and running multiple algorithms to estimate secondary structure propensities, solvent accessibility, and disordered regions. Co-evolutionary methods use multiple sequence alignments to predict contact maps, and template-based methods search the PDB for templates. Their predictions are converted into geometric constraints to guide a conformation sampling process, where fragments are swapped in and out of putative structures to minimize an expertly-derived energy model. Due to this complexity, prediction times range from hours to days, and require codebases as large as several million lines of code (Leaver-Fay et al., 2011).

In contrast, a trained RGN model is a single mathematical function that is evaluated once per prediction. Computation of this function implicitly carries out domain splitting, property finding, energy minimization, and conformational sampling simultaneously. We found that 512 concurrent RGN-based predictions, with sequence length ~700, can be made in ~5.4 seconds on a single GPU, i.e. ~10 milliseconds / structure. Table 2 compares training and prediction speeds of RGNs to established methods that rely heavily on simulation with limited learning (first row), and deep learning plus co-evolution-based contact prediction methods that rely on learning (second row), combined with CONFOLD (Adhikari et al., 2015) to convert predicted contact maps into tertiary structures. While training RGNs can take weeks to months, once trained, they make predictions 6–7 orders of magnitude faster than existing pipelines. This speed enables new types of applications, such as the integration of structure prediction within docking and virtual screening in which ligand-aware RGNs could output distinct protein conformations in response to distinct ligand poses.

DISCUSSION

A key limitation of explicit sequence-to-structure maps, including molecular dynamics and fragment assembly, is a reliance on fixed energy models that do not learn from data; a second limitation is the exclusive use of single-scale atomic or residue-level representations. In contrast, modern co-evolution methods leverage learning and multi-scale representations to substantially improve performance (Liu et al., 2017; Wang et al., 2016). RGNs go one step further by building a fully differentiable map extending from sequence to structure with all of the steps in existing prediction pipelines implicitly encoded and learnable from data. Through their recurrent architecture, RGNs can capture sequence-structure motifs and multiple scales from residues to domains (Alva et al., 2015; Ponting and Russell, 2002). When tracking structure prediction during RGN training (Movie S1), RGNs appear to first learn global aspects of protein folds, then refine their predictions to generate more accurate local structure.

RGNs are multi-representational, operating on three distinct parameterizations of protein structure. The first is torsional, capturing local relationships between atoms with bond lengths and angles held fixed, and torsional angles as the immediate outputs of computational units. This virtually guarantees that predictions are structurally correct at a local level. The second is Cartesian, built by geometric units and capturing the global coordination of multiple atoms in 3D space, the catalytic triad of an enzyme’s active site for example, even if the residues are distant along the protein chain. Future augmentations—e.g. 3D convolutional networks that operate directly on the Cartesian representation—may further improve the detection and quality of long-range interactions. The third parameterization, built in the dRMSD stage, is the matrix of inter-atomic distances, and is simultaneously local and global. It is useful for optimizing RGN parameters *de novo*, as we have used it, but can also be used to incorporate prior knowledge expressible in terms of atomic distances; such knowledge includes physical features (e.g. electrostatics) and statistical data on interactions (e.g. evolutionary couplings).

One limitation of current RGNs is their reliance on PSSMs, which we have found to be helpful to achieving high accuracy predictions. PSSMs are much weaker than multiple sequence alignments as they are based on single residue mutation frequencies and ignore how each residue mutates in response to all other residues. Co-evolutionary couplings require pairwise frequencies, resulting in quadratically rather than linearly scaling statistical cost. Nonetheless, removing PSSMs and relying exclusively on raw sequences could robustify RGNs for many applications, including prediction of genetic variants. Achieving this may require more data-efficient model architectures. For protein design, RGNs can be used as is, by fixing the desired structure and optimizing the raw sequence and PSSMs to match it (i.e. by computing derivatives of the inputs—as opposed to model parameters—with respect to the dRMSD between predicted and desired structures.) Co-evolution methods do not have this capability as their inputs are the inter-residue couplings themselves, making the approach circular.

The history of protein structure prediction suggests that new methods complementary to existing ones are eventually incorporated into hybrids. RGNs have this benefit, being an almost entirely complementary modeling approach. For example, structural templates or co-evolutionary information could be incorporated as priors in the distance-based parameterization or even as raw inputs for learning. RGNs can also include secondary structure predicted by other algorithms. This is likely to be advantageous since the RGNs described here often predict global fold correctly but do less well with secondary structure (e.g. T0827 in Figure 3e). RGNs can also be made to predict side-chain conformations, by outputting a branched curve in lieu of the current linear curve, and are applicable to a wide range of other polymers (e.g. RNA tertiary structure.) Our demonstration that state of the art performance in structure prediction can be achieved using an end-to-end differentiable model will make available to protein folding and biophysics very rapid improvements in machine learning across a wide range of scientific and technical fields. We predict that hybrid systems using deep learning, co-evolution as priors, and physics-based approaches for refinement will soon solve the long-standing problem of accurate and efficient structure prediction. It is also possible that the use of neural network probing techniques (Alain and Bengio, 2016; Koh and Liang, 2017; Nguyen et al., 2016; Shrikumar et al., 2017; Simonyan et al., 2013) with RGNs will provide new insight into the physical chemistry of folding and the sorts of intermediate structures that proteins use to sample conformational space.

STAR METHODS

CONTACT FOR REAGENT AND RESOURCE SHARING

Further information and requests for resources and reagents should be directed to and will be fulfilled by the Lead Contact, Mohammed AlQuraishi (alquraishi@hms.harvard.edu).

METHOD DETAILS

Model—We featurize a protein of length L as a sequence of vectors $(x_1, \dots, x_L)$ where $x_t \in \mathbb{R}^d$ for all t. The dimensionality d is 41, where 20 dimensions are used as a one-hot indicator of the amino acid residue at a given position, another 20 dimensions are used for the PSSM of that position, and 1 dimension is used to encode the information content of the position. The PSSM values are sigmoid transformed to lie between 0 and 1. The sequence of input vectors are fed to an LSTM (Hochreiter and Schmidhuber, 1997), whose basic formulation is described by the following set of equation.

$$i_t = \sigma\left(W_i\left[x_t, h_{t-1}\right] + b_i\right)$$

$$f_t = \sigma\left(W_f\left[x_t, h_{t-1}\right] + b_f\right)$$

$$o_t = \sigma\left(W_o\left[x_t, h_{t-1}\right] + b_o\right)$$

$$\tilde{c}_t = tanh\left(W_c\left[x_t, h_{t-1}\right] + b_c\right)$$

$$c_t = i_t \odot \tilde{c}_t + f_t \odot c_{t-1}$$

$$h_t = o_t \odot tanh\left(c_t\right)$$

W_i, W_f, W_o, W_c are weight matrices, b_i, b_f, b_o, b_c are bias vectors, h_t and c_t are the hidden and memory cell state for residue t, respectively, and $\odot$ is element-wise multiplication. We use two LSTMs, running independently in opposite directions (1 to L and L to 1), to output two hidden states $h_t^{(f)}$ and $h_t^{(b)}$ for each residue position t corresponding to the forward and backward directions. Depending on the RGN architecture, these two hidden states are either the final outputs states or they are fed as inputs into one or more LSTM layers.

The outputs from the last LSTM layer form a sequence of a concatenated hidden state vectors $\left(\left[h_1^{(f)}, h_1^{(b)}\right], \cdots, \left[h_L^{(f)}, h_L^{(b)}\right]\right)$. Each concatenated vector is then fed into an angularization layer described by the following set of equations:

$$p_t = softmax\left(W_\varphi\left[h_t^{(f)}, h_t^{(b)}\right] + b_\varphi\right)$$

$$\varphi_t = arg\left(p_t exp(i\Phi)\right)$$

W_φ is a weight matrix, b_φ is a bias vector, Φ is a learned alphabet matrix, and arg is the complex-valued argument function. Exponentiation of the complex-valued matrix $i\Phi$ is performed element-wise. The Φ matrix defines an alphabet of size m whose letters correspond to triplets of torsional angles defined over the 3-torus. The angularization layer interprets the LSTM hidden state outputs as weights over the alphabet, using them to compute a weighted average of the letters of the alphabet (independently for each torsional angle) to generate the final set of torsional angles $\varphi_t \in S^1 \times S^1 \times S^1$ for residue t (we are overloading the standard notation for protein backbone torsional angles, with φ_t corresponding to the (ψ, φ, ω) triplet). Note that φ_t may be alternatively computed using the following equation, where the trigonometric operations are performed element-wise:

$$\varphi_t = atan2\left(p_t sin(\Phi), p_t cos(\Phi)\right)$$

In general, the geometry of a protein backbone can be represented by three torsional angles φ, ψ, and ω that define the angles between successive planes spanned by the N, C$^\alpha$, and C’ protein backbone atoms (Ramachandran et al., 1963). While bond lengths and angles vary as well, their variation is sufficiently limited that they can be assumed fixed. Similar claims hold for side chains as well, although we restrict our attention to backbone structure. The resulting sequence of torsional angles (φ_1, … , φ_L) from the angularization layer is fed sequentially, along with the coordinates of the last three atoms of the nascent protein chain (c_1, …, c_{3L}), into recurrent geometric units that convert this sequence into 3D Cartesian coordinates, with three coordinates resulting from each residue, corresponding to the N, C$^\alpha$, and C’ backbone atoms. Multiple mathematically-equivalent formulations exist for this transformation; we adopt one based on the Natural Extension Reference Frame (Parsons et al., 2005), described by the following set of equations:

$$\tilde{c}_k = r_{k\ mod\ 3}\begin{bmatrix} cos\left(\theta_{k \bmod 3}\right) \\ cos\left(\varphi_{\lfloor k/3 \rfloor, k \bmod 3}\right) sin\left(\theta_{k \bmod 3}\right) \\ sin\left(\varphi_{\lfloor k/3 \rfloor, k \bmod 3}\right) sin\left(\theta_{k \bmod 3}\right) \end{bmatrix}$$

$$m_k = c_{k-1} - c_{k-2}$$

$$n_k = m_{k-1} \times \widehat{m_k}$$

$$M_k = \left[\widehat{m_k}, \widehat{n_k} \times \widehat{m_k}, \widehat{n_k}\right]$$

$$c_k = M_k \tilde{c}_k + c_{k-1}$$

Where r_k is the length of the bond connecting atoms k − 1 and K, θ_k is the bond angle formed by atoms $k-2$, $k-1$, and k, $\varphi_{\lfloor k/3 \rfloor, k \bmod 3}$ is the predicted torsional angle formed by atoms $k-2$ and $k-1$, C_k is the position of the newly predicted atom k, $\widehat{m}$ is the unit-normalized version of m, and × is the cross product. Note that k indexes atoms 1 through $3L$, since there are three backbone atoms per residue. For each residue t we compute C_{3t-2}, C_{3t-1}, and C_{3t} using the three predicted torsional angles of residue t, specifically $\varphi_{t,j} = \varphi_{\left\lfloor \frac{3t}{3} \right\rfloor, (3t+j) \bmod 3}$ for $j = \{0,1,2\}$. The bond lengths and angles are fixed, with three bond length (r_0, r_1, r_2) corresponding to N-C$^{\alpha}$, C$^{\alpha}$-C', and C'-N, and three bond angles (θ_0, θ_1, θ_2) corresponding to N-C$^{\alpha}$-C', C$^{\alpha}$-C'-N, and C'-N-C$^{\alpha}$. As there are only three unique values we have $r_k = r_{k \bmod 3}$ and $\theta_{6k} = \theta_{k \bmod 3}$. In practice we employ a modified version of the above equations which enable much higher computational efficiency (AlQuraishi, 2019a).

The resulting sequence (C_1, …, C_{3L}) fully describes the protein backbone chain structure and is the model's final predicted output. For training purposes a loss is necessary to optimize model parameters. We use the *dRMSD* metric as it is differentiable and captures both local and global aspects of protein structure. It is defined by the following set of equations:

$$\tilde{d}_{j,k} = \left\| c_j - c_k \right\|_2$$

$$d_{j,k} = \tilde{d}_{j,k}^{(\mathrm{exp})} - \tilde{d}_{j,k}^{(\mathrm{pred})}$$

$$dRMSD = \frac{\|D\|_2}{L(L-1)}$$

Where $\{d_{j,k}\}$ are the elements of matrix D, and $\tilde{d}_{j,k}^{(\mathrm{exp})}$ and $\tilde{d}_{j,k}^{(\mathrm{pred})}$ are computed using the coordinates of the experimental and predicted structures, respectively. In effect, the *dRMSD* computes the ℓ_2-norm of the distances over distances, by first computing the pairwise distances between all atoms in both the predicted and experimental structures individually, and then computing the distances between those distances. For most experimental structures, the coordinates of some atoms are missing. They are excluded from the dRMSD by not computing the differences between their distances and the predicted ones.

Hyperparameters—RGN hyperparameters were manually fit, through sequential exploration of hyperparameter space, using repeated evaluations on the ProteinNet11 validation set and three evaluations on ProteinNet11 test set. Once chosen the same hyperparameters were used to train RGNs on ProteinNet7–12 training sets. The validation sets were used to determine early stopping criteria, followed by single evaluations on the ProteinNet7–12 test sets to generate the final reported numbers (excepting ProteinNet11).

The final model consisted of two bidirectional LSTM layers, each comprised of 800 units per direction, and in which outputs from the two directions are first concatenated before being fed to the second layer. Input dropout set at 0.5 was used for both layers, and the alphabet size was set to 60 for the angularization layer. Inputs were duplicated and concatenated; this had a separate effect from decreasing dropout probability. LSTMs were random initialized with a uniform distribution with support [−0.001, 0.01], while the alphabet was similarly initialized with support [$-\pi$, π]. ADAM was used as the optimizer, with a learning rate of 0.001, $\beta_1 = 0.95$ and $\beta_2 = 0.99$, and a batch size of 32. Gradients were clipped using norm rescaling with a threshold of 5.0. The loss function used for optimization was length-normalized dRMSD (i.e. dRMSD divided by protein length), which is distinct from the standard dRMSD we use for reporting accuracies.

RGNs are very seed sensitive. As a result, we used a milestone scheme to restart underperforming models early. If a dRMSD loss milestone is not achieved by a given iteration, training is restarted with a new initialization seed. Table S3 summarizes the milestones, which were determined based on preliminary runs. In general, 8 models were started and, after surviving all milestones, were run for 250k iterations, at which point the lower performing half were discarded, and similarly at 500k iterations, ending with 2 models that were usually run for ~2.5M iterations. Once validation error stabilized we reduced the learning rate by a factor of 10 to 0.0001, and run for a few thousand additional iterations to gain a small but detectable increase in accuracy before ending model training.

Dataset—We use the ProteinNet dataset for all analyses (AlQuraishi, 2019b). ProteinNet recreates the conditions of past CASP assessments by restricting the set of sequences (for building PSSMs) and structures used to those available prior to the start of each CASP assessment. Each ProteinNet entry is comprised of two inputs, the raw protein sequence, represented by a one-hot vector, and the protein’s PSSM and information content profiles, derived using 5 iterations of JackHMMer with an e-value threshold of 10^{-10}. PSSM values are normalized to lie between 0 and 1. The output for each ProteinNet entry is comprised of the Cartesian coordinates of the protein’s backbone atoms, annotated by metadata denoting which atoms are missing from the experimental structure. These atoms are excluded from the dRMSD loss calculation, which enables use of partially resolved experimental structures that would otherwise be excluded from the dataset.

For ProteinNet7–11, the publicly available CASP structures were used as test sets. For ProteinNet12, the publicly available CASP12 structures are incomplete, as some structures are still embargoed. We obtained a private set of structures from the CASP organizers that includes all structures used in CASP12 (except one or two), and we used this set for model assessment. For training all RGN models, the 90% “thinning” version of ProteinNet was used.

DATA AND SOFTWARE AVAILABILITY

TensorFlow (Abadi et al., 2016) code for training new RGN models, as well as pre-trained RGN models used in reporting results for CASP 7–12, are available on GitHub at https://github.com/aqlaboratory/rgn.

Supplementary Material

Refer to Web version on PubMed Central for supplementary material.

Acknowledgments

We are indebted to Peter Sorger for his mentorship and support, and thank him for extensive editorial feedback on this manuscript. We thank Jasper Snoek and Adrian Jinich for their editorial comments and many helpful discussions, and Uraib Aboudi, Ramy Arnaout, Karen Sachs, Michael Levitt, Nazim Bouatta, and Jinbo Xu for their feedback. We also thank Martin Steinegger and Milot Mirdita for their help with using the HHblits and MMseqs2 packages, Sergey Ovchinnikov for discussions about the manuscript and help with metagenomics sequences, Andriy Kryshtafovych for his help with CASP structures, Sean Eddy for his help with using the JackHMMer package, and Raffaele Potami, Amir Karger, and Kristina Holton for their help with using the HPC resources at Harvard Medical School. Finally we thank the anonymous reviewers for their constructive feedback. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPUs used for this research.

Funding:

This work was supported by NIGMS Grant P50GM107618.

References

Abadi M, Barham P, Chen J, Chen Z, Davis A, Dean J, Devin M, Ghemawat S, Irving G, Isard M, et al. (2016). TensorFlow: A system for large-scale machine learning In 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16), pp. 265–283.

Adhikari B, Bhattacharya D, Cao R, and Cheng J (2015). CONFOLD: Residue-residue contact-guided ab initio protein folding. Proteins 83, 1436–1449. [PubMed: 25974172]

Alain G, and Bengio Y (2016). Understanding intermediate layers using linear classifier probes. ArXiv:1610.01644 [Cs, Stat].

AlQuraishi M (2019a). Parallelized Natural Extension Reference Frame: Parallelized Conversion from Internal to Cartesian Coordinates. Journal of Computational Chemistry 40, 885–892. [PubMed: 30614534]

AlQuraishi M (2019b). ProteinNet: a standardized data set for machine learning of protein structure. ArXiv:1902.00249 [Cs, q-Bio, Stat].

Alva V, Söding J, and Lupas AN (2015). A vocabulary of ancient peptides at the origin of folded proteins. ELife 4, e09410. [PubMed: 26653858]

Aydin Z, Thompson J, Bilmes J, Baker D, and Noble WS (2012). Protein Torsion Angle Class Prediction by a Hybrid Architecture of Bayesian and Neural Networks In 13th International Conference on Bioinformatics and Computational Biology, p.

Bernstein FC, Koetzle TF, Williams GJ, Meyer EF, Brice MD, Rodgers JR, Kennard O, Shimanouchi T, and Tasumi M (1977). The Protein Data Bank: a computer-based archival file for macromolecular structures. J. Mol. Biol 112, 535–542. [PubMed: 875032]

Branden C, and Tooze J (1999). Introduction to Protein Structure (New York, NY: Garland Science).

Contreras-Moreira B, Ezkurdia I, Tress ML, and Valencia A (2005). Empirical limits for template-based protein structure prediction: the CASP5 example. FEBS Lett. 579, 1203–1207. [PubMed: 15710414]

Dawson NL, Lewis TE, Das S, Lees JG, Lee D, Ashford P, Orengo CA, and Sillitoe I (2017). CATH: an expanded resource to predict protein function through structure and sequence. Nucleic Acids Res 45, D289–D295. [PubMed: 27899584]

Dill KA (1990). Dominant forces in protein folding. Biochemistry 29, 7133–7155. [PubMed: 2207096]

Dill KA, and MacCallum JL (2012). The Protein-Folding Problem, 50 Years On. Science 338, 1042–1046. [PubMed: 23180855]

Gajda MJ, Pawlowski M, and Bujnicki JM (2011a). Protein Structure Prediction: From Recognition of Matches with Known Structures to Recombination of Fragments In Multiscale Approaches to Protein Modeling, Kolinski A, ed. (New York, NY: Springer New York), pp. 231–254.

Gajda MJ, Pawlowski M, and Bujnicki JM (2011b). Multiscale Approaches to Protein Modeling (New York, NY: Springer New York).

Gao Y, Wang S, Deng M, and Xu J (2017). Real-value and confidence prediction of protein backbone dihedral angles through a hybrid method of clustering and deep learning. ArXiv:1712.07244 [q-Bio].

Goodfellow I, Bengio Y, and Courville A (2016). Deep Learning (Cambridge, Massachusetts: The MIT Press).

Hochreiter S, and Schmidhuber J (1997). Long Short-Term Memory. Neural Computation 9, 1735–1780. [PubMed: 9377276]

Hopf TA, Schärfe CPI, Rodrigues JPGLM, Green AG, Kohlbacher O, Sander C, Bonvin AMJJ, and Marks DS (2014). Sequence co-evolution gives 3D contacts and structures of protein complexes. Elife 3.

Hopf TA, Ingraham JB, Poelwijk FJ, Schärfe CPI, Springer M, Sander C, and Marks DS (2017). Mutation effects predicted from sequence co-variation. Nat Biotech 35, 128–135.

de Juan D, Pazos F, and Valencia A (2013). Emerging methods in protein co-evolution. Nature Reviews Genetics 14, 249–261.

Koh PW, and Liang P (2017). Understanding Black-box Predictions via Influence Functions. ArXiv: 1703.04730 [Cs, Stat].

Kryshtafovych A, Monastyrskyy B, and Fidelis K (2016). CASP11 statistics and the prediction center evaluation system. Proteins 84, 15–19. [PubMed: 26857434]

Kryshtafovych A, Monastyrskyy B, Fidelis K, Moult J, Schwede T, and Tramontano A Evaluation of the template-based modeling in CASP12. Proteins: Structure, Function, and Bioinformatics 86, 321–334.

Leaver-Fay A, Tyka M, Lewis SM, Lange OF, Thompson J, Jacak R, Kaufman K, Renfrew PD, Smith CA, Sheffler W, et al. (2011). ROSETTA3: an object-oriented software suite for the simulation and design of macromolecules. Meth. Enzymol 487, 545–574. [PubMed: 21187238]

LeCun Y, Bengio Y, and Hinton G (2015). Deep learning. Nature 521, 436–444. [PubMed: 26017442]

Li H, Hou J, Adhikari B, Lyu Q, and Cheng J (2017). Deep learning methods for protein torsion angle prediction. BMC Bioinformatics 18, 417. [PubMed: 28923002]

Liu T, Ish-Shalom S, Torng W, Lafita A, Bock C, Mort M, Cooper DN, Bliven S, Capitani G, Mooney SD, et al. Biological and functional relevance of CASP predictions. Proteins: Structure, Function, and Bioinformatics 86, 374–386.

Liu Y, Palmedo P, Ye Q, Berger B, and Peng J (2017). Enhancing Evolutionary Couplings with Deep Convolutional Neural Networks. Cels 0.

Lyons J, Dehzangi A, Heffernan R, Sharma A, Paliwal K, Sattar A, Zhou Y, and Yang Y (2014). Predicting backbone Cα angles and dihedrals from protein sequences by stacked sparse auto-encoder deep neural network. J Comput Chem.

Marks DS, Colwell LJ, Sheridan R, Hopf TA, Pagnani A, Zecchina R, and Sander C (2011). Protein 3D structure computed from evolutionary sequence variation. PLoS ONE 6, e28766. [PubMed: 22163331]

Marx D, and Hutter J (2012). Ab initio molecular dynamics: basic theory and advanced methods (Cambridge: Cambridge University Press).

Moult J, Pedersen JT, Judson R, and Fidelis K (1995). A large-scale experiment to assess protein structure prediction methods. Proteins 23, ii–iv. [PubMed: 8710822]

John Moult, Krzysztof Fidelis, Andriy Kryshtafovych, Torsten Schwede, and Anna Tramontano (2018). Critical assessment of methods of protein structure prediction (CASP)—Round XII. Proteins: Structure, Function, and Bioinformatics 86, 7–15.

Nguyen A, Yosinski J, and Clune J (2016). Multifaceted Feature Visualization: Uncovering the Different Types of Features Learned By Each Neuron in Deep Neural Networks. ArXiv: 1602.03616 [Cs].

Ovchinnikov S, Kim DE, Wang RY-R, Liu Y, DiMaio F, and Baker D (2016). Improved de novo structure prediction in CASP11 by incorporating coevolution information into Rosetta. Proteins 84, 67–75. [PubMed: 26677056]

Ovchinnikov S, Park H, Varghese N, Huang P-S, Pavlopoulos GA, Kim DE, Kamisetty H, Kyrpides NC, and Baker D (2017). Protein structure determination using metagenome sequence data. Science 355, 294–298. [PubMed: 28104891]

Parsons J, Holmes JB, Rojas JM, Tsai J, and Strauss CEM (2005). Practical conversion from torsion space to Cartesian space for in silico protein synthesis. J. Comput. Chem 26, 1063–1068. [PubMed: 15898109]

Perez A, Morrone JA, Brini E, MacCallum JL, and Dill KA (2016). Blind protein structure prediction using accelerated free-energy simulations. Sci Adv 2, e1601274. [PubMed: 27847872]

Ponting CP, and Russell RR (2002). The Natural History of Protein Domains. Annual Review of Biophysics and Biomolecular Structure 31, 45–71.

Ramachandran GN, Ramakrishnan C, and Sasisekharan V (1963). Stereochemistry of polypeptide chain configurations. J. Mol. Biol 7, 95–99. [PubMed: 13990617]

Schaarschmidt J, Monastyrskyy B, Kryshtafovych A, and Bonvin AMJJ (2017). Assessment of contact predictions in CASP12: co-evolution and deep learning coming of age. Proteins.

Shrikumar A, Greenside P, and Kundaje A (2017). Learning Important Features Through Propagating Activation Differences In PMLR, pp. 3145–3153.

Simonyan K, Vedaldi A, and Zisserman A (2013). Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. ArXiv:1312.6034 [Cs].

Wang S, Sun S, Li Z, Zhang R, and Xu J (2016). Accurate De Novo Prediction of Protein Contact Map by Ultra-Deep Learning Model. BioRxiv 073239.

Xu D, and Zhang Y (2012). Ab initio protein structure assembly using continuous structure fragments and optimized knowledge-based force field. Proteins 80, 1715–1735. [PubMed: 22411565]

Xu J, and Zhang Y (2010). How significant is a protein structure similarity with TM-score = 0.5? Bioinformatics 26, 889–895. [PubMed: 20164152]

Yang J, Yan R, Roy A, Xu D, Poisson J, and Zhang Y (2015). The I-TASSER Suite: protein structure and function prediction. Nat. Methods 12, 7–8. [PubMed: 25549265]

Zhang Y, and Skolnick J (2004). Scoring function for automated assessment of protein structure template quality. Proteins 57, 702–710. [PubMed: 15476259]

Zhang C, Mortuza SM, He B, Wang Y, and Zhang Y Template-based and free modeling of I-TASSER and QUARK pipelines using predicted contact maps in CASP12. Proteins n/a–n/a.

Zhao F, Peng J, and Xu J (2010). Fragment-free approach to protein folding using conditional neural fields. Bioinformatics 26, i310–i317. [PubMed: 20529922]

Zhou Y, Duan Y, Yang Y, Faraggi E, and Lei H (2010). Trends in template/fragment-free protein structure prediction. Theoretical Chemistry Accounts 128, 3–16. [PubMed: 21423322]

Highlights

- Neural network predicts protein structure from sequence without using co-evolution
- Model replaces structure prediction pipelines with one mathematical function
- Achieves state of the art performance on novel protein folds
- Learns a low-dimensional representation of protein sequence space

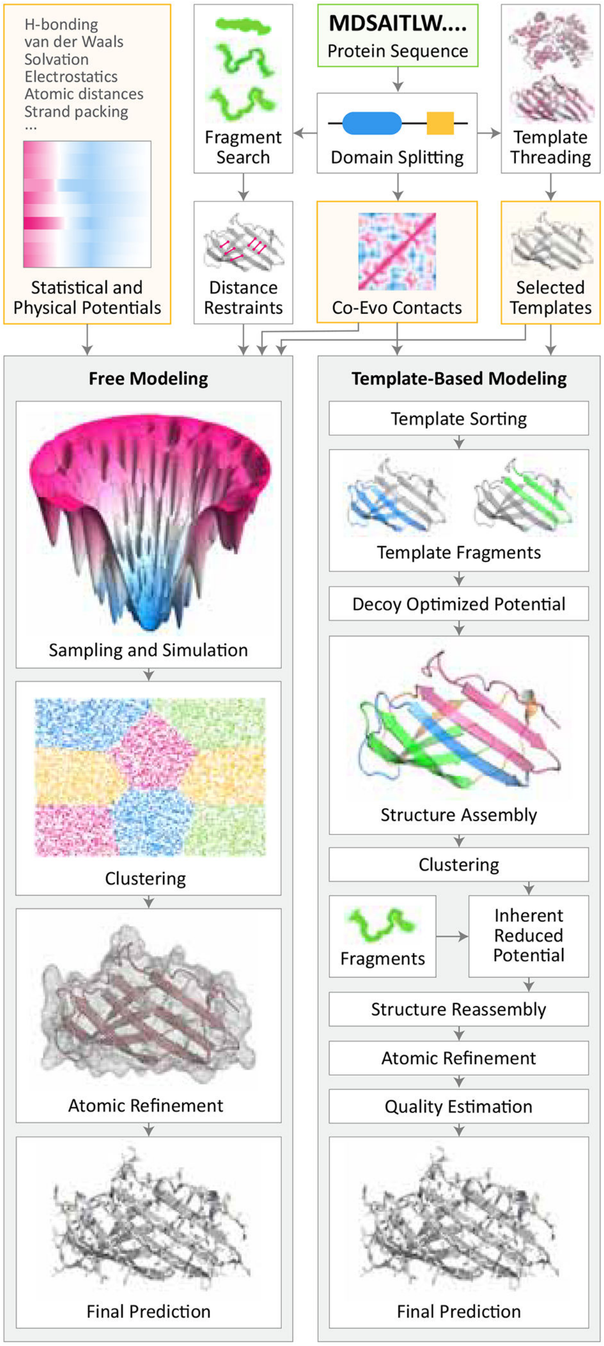


Figure 1: Conventional pipelines for protein structure prediction.

Prediction process begins with query sequence (top, green box) whose constituent domains and co-evolutionary relationships are identified through multiple sequence alignments. In free modeling **(left)**, fragment libraries are searched to derive distance restraints which, along with restraints derived from co-evolutionary data, guide simulations that iteratively minimize energy through sampling. Coarse conformations are then refined to yield the final structure. In template-based modeling **(right pipeline)**, the PDB is searched for templates. If found, fragments from one or more templates are combined to assemble a structure, which is then optimized and refined to yield the final structure. Orange boxes indicate sources of input information beyond query sequence, including prior physical knowledge. Diagram is modeled on the I-Tasser and Quark pipelines (Zhang et al.).

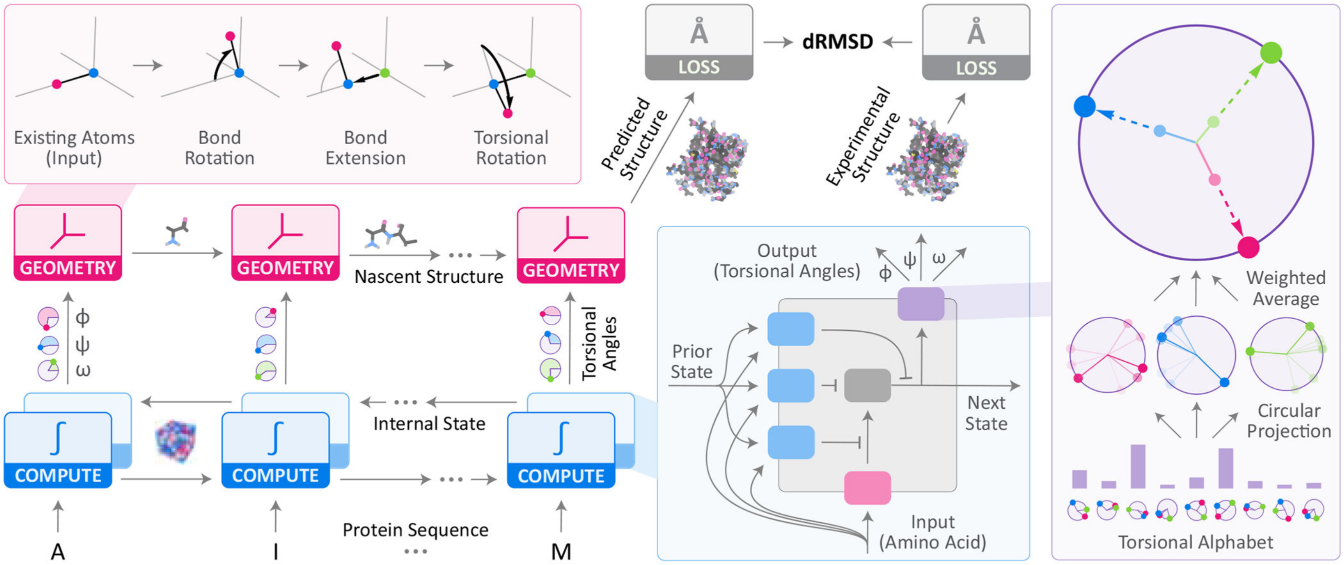


Figure 2: Recurrent geometric networks.

Protein sequences are fed one residue at a time to the computational units of an RGN (bottom-left), which compute an internal state that is integrated with the states of adjacent units. Based on these computations, torsional angles are predicted and fed to geometric units, which sequentially translate them into Cartesian coordinates to generate the predicted structure. dRMSD is used to measure deviation from experimental structures, serving as the signal for optimizing RGN parameters. **Top-Left Inset:** Geometric units take new torsional angles and a partial backbone chain, and extend it by one residue. **Bottom-Right Inset:** Computational units, based on Long Short-Term Memory (LSTMs) (Hochreiter and Schmidhuber, 1997), use gating units (blue) to control information flow in and out of the internal state (gray), and angularization units (purple) to convert raw outputs into angles. **Rightmost Inset:** Angularization units select from a learned set of torsion angles (“alphabet”) a mixture of torsions, which are then averaged in a weighted manner to generate the final set of torsions. Mixing weights are determined by computational units.

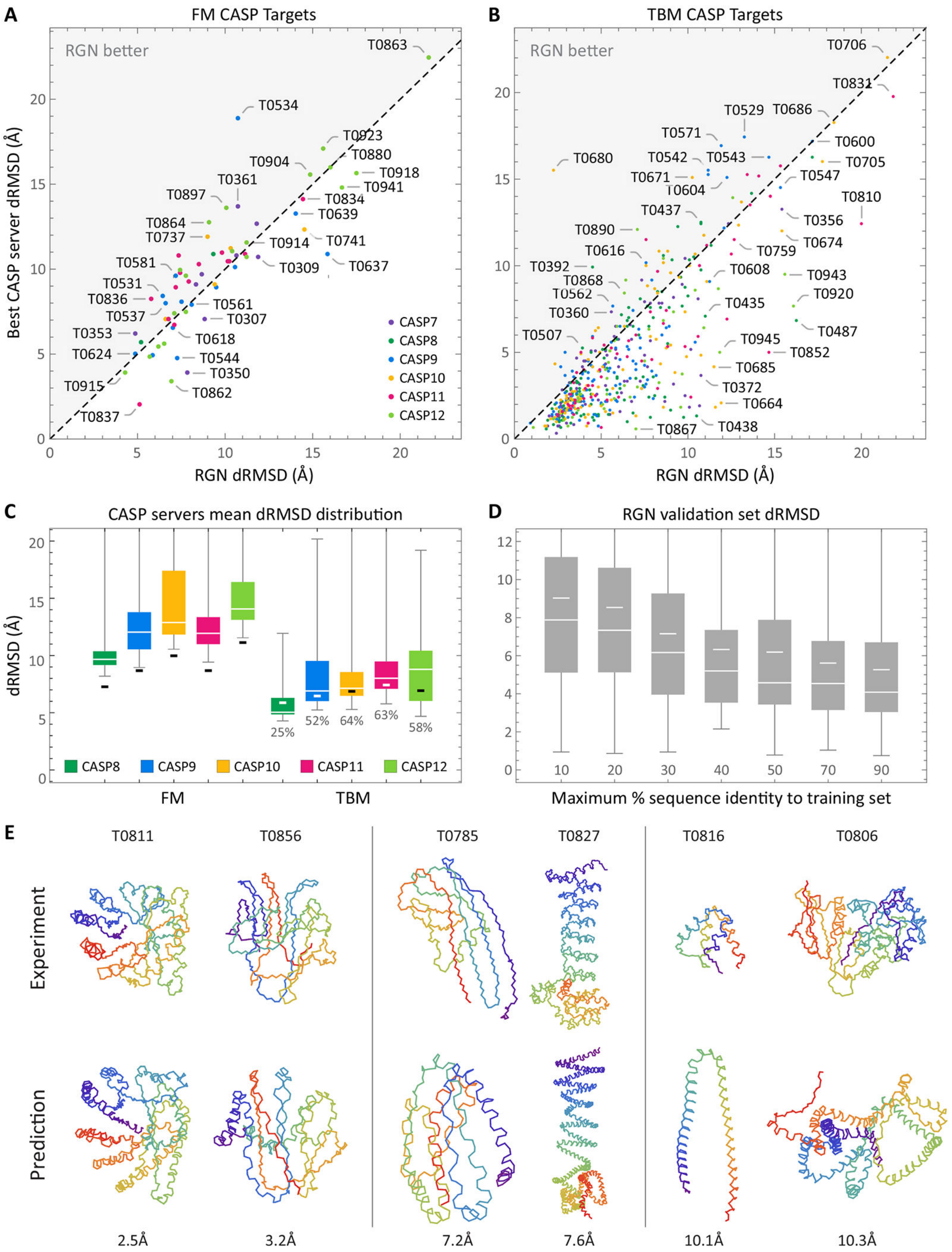


Figure 3: Results overview.

Scatterplots of individual FM **(A)** and TBM **(B)** predictions made by RGN and top CASP server. Two TBM outliers (T0629 and T0719) were dropped for visualization purposes. **(C)** Distributions of mean dRMSD (lower is better, white is median) achieved by servers predicting all structures with >95% coverage at CASP 8–12 are shown for FM (novel folds) and TBM (known folds) categories. Thick black (white on dark background) bars mark RGN dRMSD. RGN percentile rankings are shown for the TBM category (below whiskers). CASP 7 is omitted due to lack of server metadata. **(D)** Distribution of RGN dRMSDs on

ProteinNet validation sets grouped by maximum % sequence identity to training set over all CASPs (medians are wide white lines, means are short white lines.) **(E)** Traces of backbone atoms of well **(left)**, fairly **(middle)**, and poorly **(right)** predicted RGN structures are shown **(bottom)** along with their experimental counterparts **(top)**. CASP identifier is displayed above each structure and dRMSD below. A color spectrum spans each protein chain to aid visualization. See also Figure S1.

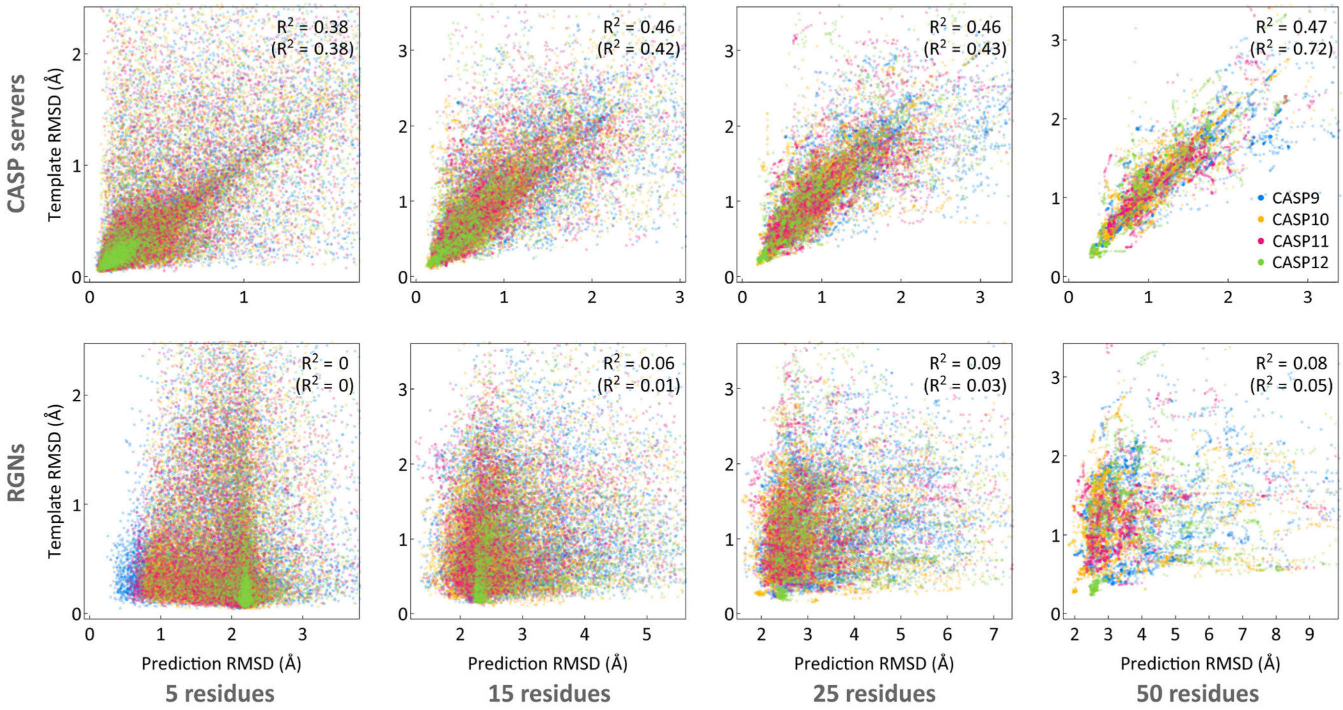


Figure 4: Correlation between prediction accuracy and template quality.

Scatterplots of fragment RMSDs, ranging in size from 5 to 50 residues, comparing the best CASP templates to the best CASP server predictions **(top)** and RGN predictions **(bottom)**. R^2 values are computed over all data points (non-parenthesized), and over data points in which predictions achieved <3Å accuracy (parenthesized). TBM domains were used (excluding TBM-hard which do not have good templates), and only templates and predictions covering >85% of full domain sequences were considered. Templates and predictions were selected based on global dRMSD with respect to experimental structure. CASP 7 and 8 are omitted due to lack of full template information.

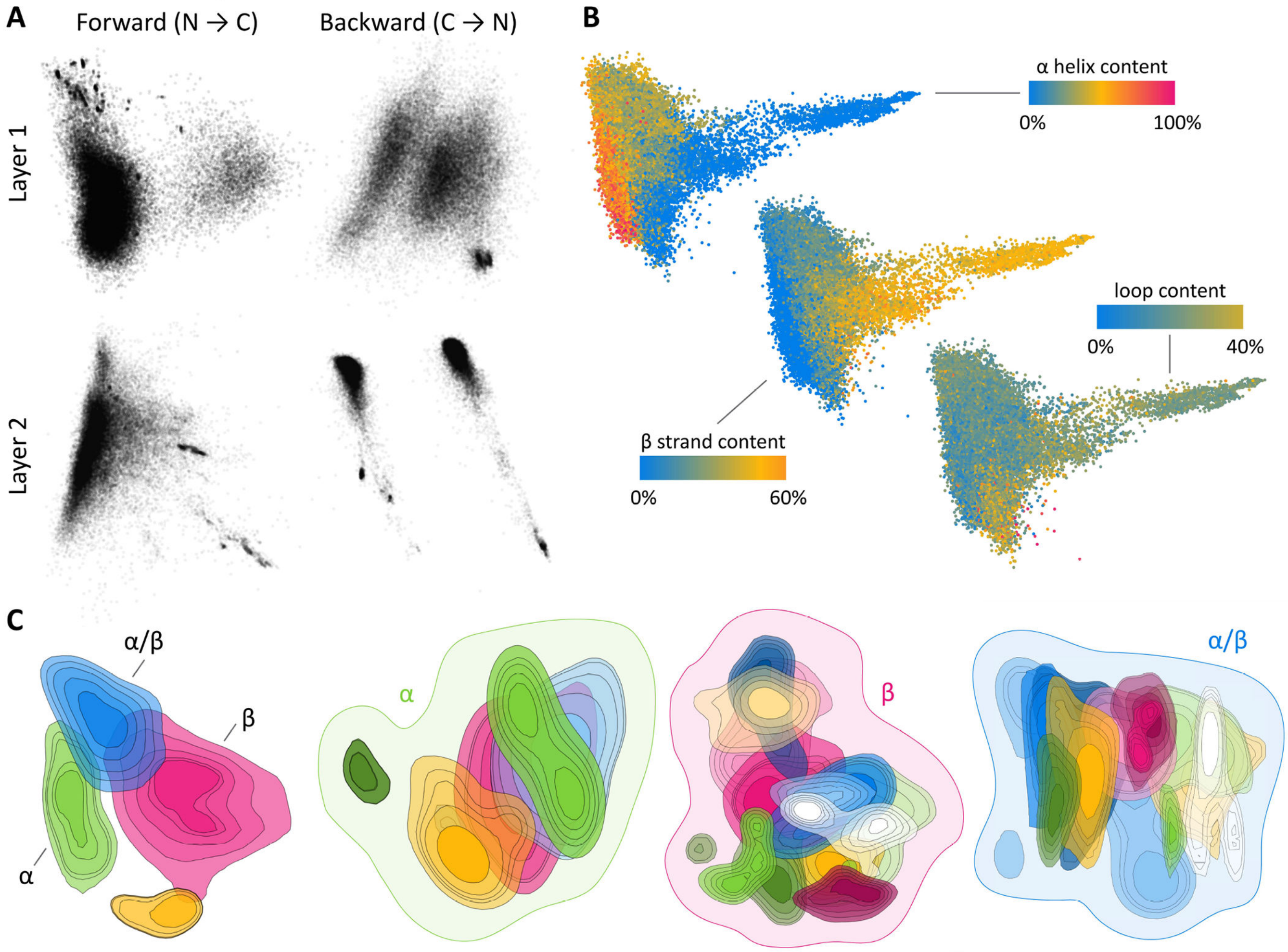


Figure 5: The latent space of RGNs.

2D projection of the separate **(A)** and combined **(B)** internal state of all RGN computational layers, with dots corresponding to individual protein sequences in the ProteinNet12 training set. **(B)** Proteins are colored by fractional secondary structure content, as determined by annotations of original protein structures. **(C)** Contour plots of the probability density (50–90% quantiles) of proteins belonging to categories in the topmost level of the CATH hierarchy **(first from left)** and proteins belonging to categories in the second-level CATH classes of "Mainly Alpha" **(second)**, "Mainly Beta" **(third)**, and "Alpha Beta" **(fourth)**. Distinct colors correspond to distinct CATH categorizations; see Figure S2–S5 for complete legends. The topmost CATH class "Few Secondary Structures" is omitted because it has no subcategories.

Table 1:

Comparative accuracy of RGNs using dRMSD.

The average dRMSD (lower is better) achieved by RGNs and the top five servers at each CASP is shown for the novel folds **(left)** and known folds **(right)** categories. Numbers are based on common set of structures predicted by top 5 servers during each CASP. A different RGN was trained for each CASP, using the corresponding ProteinNet training set containing all sequences and structures available prior to the start of that CASP. See also Table S1–S3.

	FM (novel folds) category (Å)						**TBM (known folds) category (Å)**					
	CASP7	**CASP8**	**CASP9**	**CASP10**	**CASP11**	**CASP12**	**CASP7**	**CASP8**	**CASP9**	**CASP10**	**CASP11**	**CASP12**
RGN	**9.3**	**7.3**	**8.7**	**10.0**	**8.5**	**10.7**	5.6	5.9	6.5	6.9	7.4	6.9
1st Server	9.3	8.3	9.0	10.3	9.3	11.0	**4.0**	**4.3**	**5.2**	**5.3**	**5.8**	**4.7**
2nd Server	9.9	8.6	9.1	10.6	9.6	11.2	4.0	4.6	5.2	5.4	6.0	4.8
3rd Server	10.0	9.2	9.7	10.9	11.2	11.3	4.1	4.8	5.4	5.7	6.5	5.6
4th Server	10.1	9.9	10.1	11.7	11.7	11.4	4.2	5.0	5.4	5.9	6.8	5.8
5th Server	10.4	10.4	13.5	12.0	12.9	13.0	4.8	5.0	5.5	7.2	6.9	5.9

Table 2:
Prediction and training speeds of structure prediction methods.

Top row corresponds to the most complex and established set of methods, which rely heavily on simulation and sampling, and typically have only a minimal learning component. Second row corresponds to methods combining co-evolution-based contact prediction with deep learning, which rely on a learning procedure, plus the CONFOLD method to convert predicted contact maps into tertiary structures. Time estimates are based on workflows used for CASP predictions, which (excepting RGNs) generate a large ensemble of structures, increasing prediction time. RGN predictions are deterministic and thus necessitate only a single prediction. All time estimates exclude MSA generation times.

Model	Prediction Speed	Training Time
Rosetta, I-Tasser, Quark	hours to days	N/A
Raptor X, DeepContact + CONFOLD	one to few hours	hours
Recurrent geometric networks (RGNs)	milliseconds	weeks to months

KEY RESOURCES TABLE

REAGENT or RESOURCE	SOURCE	IDENTIFIER
Software and Algorithms		
TensorFlow	Abadi et al., 2016	tensorflow.org
ProteinNet	AlQuraishi, 2019b	https://github.com/aqlaboratory/proteinnet