Chainsaw: protein domain segmentation with fully convolutional neural networks

Jude Wells[1], **Alex Hawkins-Hooker**[1], **Nicola Bordin**[2], **Brooks Paige**[1], **and Christine Orengo**[2*]

[1]University College London, Centre for Artificial Intelligence
[2]University College London, Institute of Structural & Molecular Biology

ABSTRACT

Protein domains are fundamental units of protein structure and play a pivotal role in understanding folding, function, evolution, and design. The advent of accurate structure prediction techniques has resulted in an influx of new structural data, making the partitioning of these structures into domains essential for inferring evolutionary relationships and functional classification. Currently, the most performant computational methods for parsing protein domains are structure-based methods that rely on unsupervised heuristics to determine the optimal segmentation. Given the availability of large curated databases of domain annotations, we find it worthwhile to revisit structure-based approaches but adopt a supervised learning approach. This paper presents Chainsaw, a novel supervised learning approach to domain parsing that achieves accuracy that surpasses current state-of-the-art methods. The Chainsaw method uses a residual convolutional neural network which is trained to predict the adjacency matrix over residues, where element a_{ij} is an estimate for the probability that residues i and j are in the same domain. Domain assignments are then derived from this soft-adjacency matrix using a non-learned algorithm that is equivalent to maximising the likelihood of the assignments given the set of pairwise probabilities expressed in the soft-adjacency matrix. We benchmark the method against recent state-of-the-art structure-based methods and show that it outperforms them, achieving an NDO score of 0.87 vs 0.80 for the next closest method when predicting a non-redundant test set of multi-domain proteins.

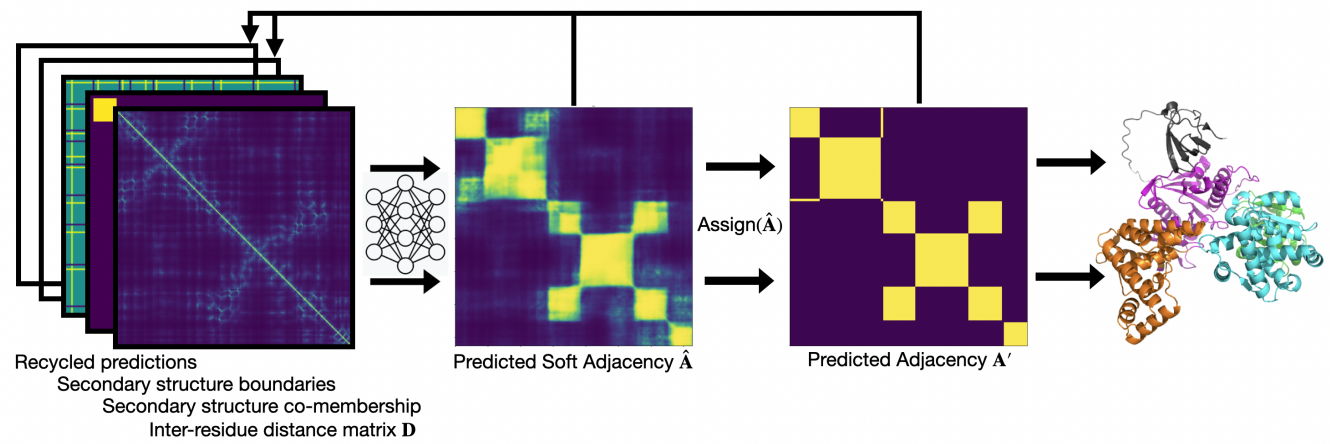


Figure 1. Chainsaw model overview

1 Introduction

The definition of a protein domain is open to some interpretation. However, typically definitions point to self-stabilizing units, usually consisting of multiple secondary structure components which are packed together with a hydrophobic core. From an evolutionary perspective, the protein domain is the level at which homology and functional groups are understood. From a biological standpoint, globular domains are key players in protein activity, from binding small molecules and nucleic acids to interactions within the cellular and external environments. Domains can exist as individual units, or in conjunction with additional domains within a protein chain often creating functional units where an active site is created at the interface between two or more domains. Drug design often targets the disruption of interfaces between domains, or pockets within

them by either designing binders (1) that would elicit the activation or deactivation of the domain's activity. The identification of correct domains and their interactions are at the core of protein function prediction, helping to detect various phenomena such as moonlighting (2), convergent evolution of function in different domains (3), as well as architecture conservation (4). From a structural characterisation standpoint, experimental methods for determining protein structure often require that the sequence is trimmed to the domain boundaries or split into separate domains to facilitate expression and crystallization. Structure-based protein domain databases such as CATH (5), SCOP (6), SCOPe (7) and ECOD (8) are essential for advancing scientific understanding of the protein universe. In 2022 DeepMind released the AlphaFold predicted structures for over 200 million proteins increasing the number of available structures by multiple orders of magnitude (9). This presents the challenge of organizing the data to include it in protein domain databases. An essential first step to discovering novel domains and evolutionary links is to identify domain boundaries and parse these 200 million structures into constituent domains with high accuracy. Code implementation of Chainsaw is available at `github.com/JudeWells/chainsaw`.

1.1 The landscape of domain boundary predictors

Protein domain boundary prediction techniques can be broadly classified into two categories: sequence-based approaches and structure-based approaches. As expected, approaches utilizing structural input outperform those relying solely on sequence information (10). With the advent of high-quality predicted structures from AlphaFold2 (11), obtaining a 3D structure is no longer a significant constraint. In order to integrate the 200 million AlphaFold structures into protein domain databases, it is logical to exploit the predicted structure as an input for enhancing domain boundary prediction. Historically, most structure-based approaches have used unsupervised algorithms applied to contact maps or pairwise residue distances. These approaches are grounded in the physical intuition that residues within domains exhibit a high degree of contact, while inter-domain contacts are comparatively fewer. The growth of protein structure databases, complete with domain annotations, presents an opportunity to recast the problem as a supervised learning task. Deep learning models have the potential to capture complex structural relationships and exploit these to achieve higher accuracy than the heuristics-based unsupervised methods, yet this approach has remained relatively unexplored in the literature. This paper aims to recast the domain segmentation problem as a supervised learning task, in which a neural network is trained to estimate the probability that a given pair of residues share the same domain. We show that our proposed approach, Chainsaw, outperforms state-of-the-art methods.

1.1.1 Unsupervised domain parsers

The main category of domain parsers are structure-based unsupervised methods that use the pairwise residue contact matrix or pairwise residue distance matrix to derive scores for proposed domain splits. This category includes PUU (12), PDP (13), SWORD (14), SWORD2 (15), FUPred (16) and UniDoc (17). Each of these methods have slightly different approaches for how the domain split scores are assessed and different ways to iterate and terminate the splitting process and different approaches to handling discontinuous domains. FUPred and UniDoc stand out in their ability to take a sequence as input and predict a contact map as a preliminary step before using an unsupervised approach to domain splitting based on the predicted contact map. CATHEDRAL (18) and DPAM (19) augment the unsupervised approach with the ability to match against a library of known domains. Both approaches search for matches in domain structure and sequence libraries. DPAM uses a fixed formula to calculate the domain splitting score as function of inter-residue distance, predicted co-membership via matching a known domain and AlphaFold's Predicted Aligned Error (PAE). The dependence on input features based on matching the target protein against a library of known domains is a potential limitation of these domain template matching approaches. In DPAM, matches against domain structures are generated using DALI (20) and sequence matches using HHSuite (21). This increases the inference time of the method and may limit the ability of the approach to generalise to novel domains.

1.1.2 Supervised domain parsers

DNN-Dom (10) trains a model to predict which residues occur at domain boundaries using only the protein sequence and features derived from a multiple sequence alignment (MSA). However, the DNN-Dom authors report that the approach performs significantly worse than unsupervised structure-based methods such as PDP (13), DomainParser (22), DDomain (23) and SWORD (14). DistDom (24) uses supervised learning but unlike DNN-Dom it leverages features derived from the target structure. Similar to DNN-Dom, the authors take a per-residue boundary classification approach. One limitation of the residue-classification approach is that the method cannot predict discontinuous domains. The DistDom authors report similar performance to the unsupervised method FUpred (with FUpred leading on some datasets while DistDom leads on others).

1.2 Novel contributions of our approach

While previous approaches have sought to estimate the probabilities that pairs of residues belong to the same domain and other methods have used supervised learning to predict which residues occur at the domain boundaries, our approach is unique in applying supervised learning to directly predict the residue-pairwise-domain co-occurrence. The advantage of this representation is that it can be used to directly supervise the learning model. This approach avoids the problem of severe

class imbalance (between boundary and non-boundary residues) that arises in supervised methods which formulate the task as per-residue boundary classification. The second contribution of our approach is an algorithm to resolve the predicted pairwise soft-adjacency matrix into final domain assignments. We achieve this using a greedy algorithm that maximises the likelihood of the domain assignments under the probabilities expressed in the soft-adjacency matrix which is the output of the neural network. Finally, we demonstrate improved performance by self-conditioning the network's predictions. We achieve this by recycling the network's predictions combined with the algorithm's output as additional input features into the neural network.

2 Methods

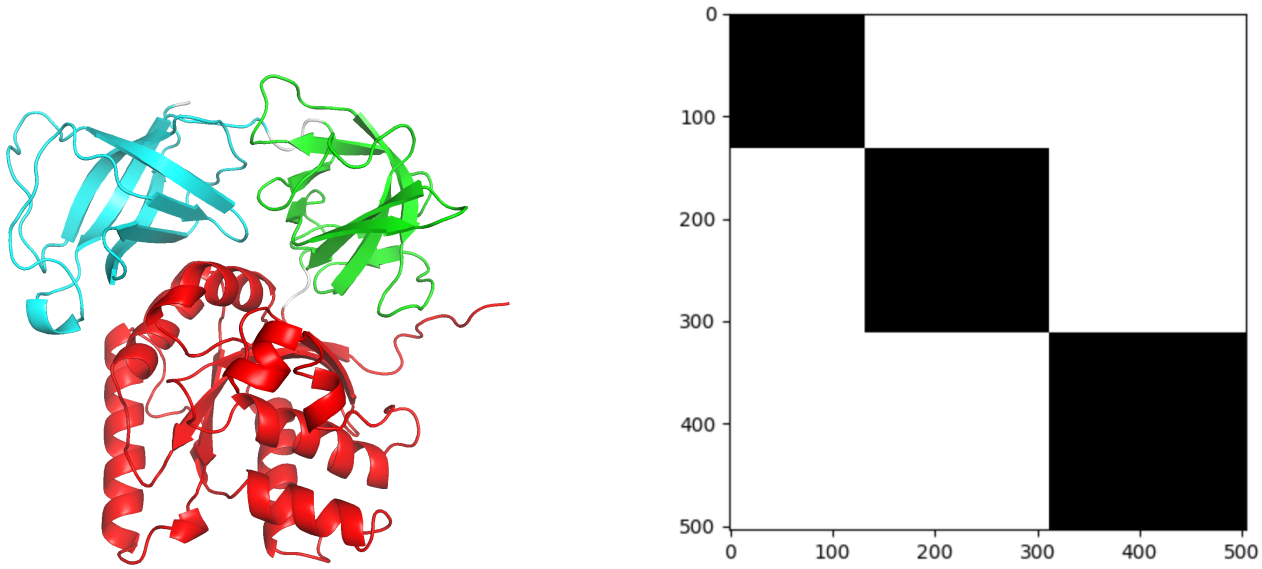
We formulate the task of domain prediction as a supervised machine-learning problem, where the model predicts for each pair of residues whether they are in the same domain. This approach has several natural advantages over recently published supervised-learning methods: it can naturally predict discontinuous domains which occur in 15-20% of the PDB structures annotated by CATH (18). The approach allows the prediction of an arbitrary number of domains and can also classify non-domain residues which commonly occur as disordered regions or linker regions. We avoid any dependency on matching against previously observed domains as a means to speed up inference times and to ensure that the model is able to detect completely novel domains that may exist in the AlphaFold and ESMAtlas models databases.

2.1 Adjacency matrices as domain assignments

The domain assignments of an L residue protein can be uniquely represented by an $L \times L$ adjacency matrix $\mathbf{A}$, where element

$$a_{ij} = \begin{cases} 1 & \text{if residues } i \text{ and } j \text{ are in the same domain} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

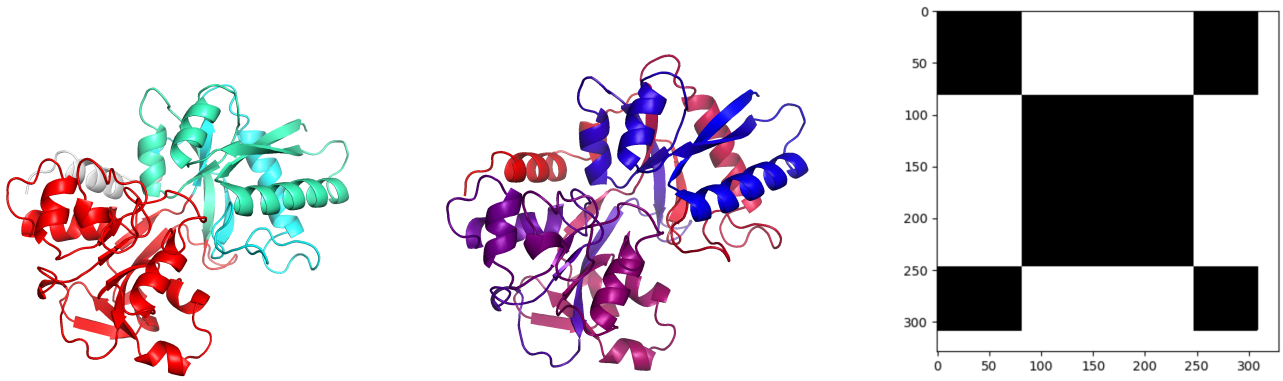
This adjacency matrix has a low-rank structure where the rank is equal to the number of domains. This characteristic is due to the fact that all residues within the same domain are represented with the same co-membership vector in the matrix $\mathbf{A}$. We later exploit this low-rank structure in the domain parsing algorithm (see Algorithm 1). For domains with residues that are continuous in sequence, this will result in an adjacency matrix that takes the form of a block-diagonal matrix, (see Figure 2). Domains which are discontinuous in sequence result in blocks occurring in the off-diagonal; (see Figure 3).



(a) PDB 1b23P with domains colored

(b) Pair-wise domain labels for 1b23P

Figure 2. Protein 1b23P with 3 domains over 504 residues.



(a) Domain 1 in green/cyan and domain 2 in red

(b) Color gradient: n-terminus (blue) to c-terminus (red).

(c) Off-diagonal blocks indicate discontinuity in the first domain.

Figure 3. Protein 1a8eA: a two-domain protein where the first domain is discontinuous.

2.2 Input features

The Chainsaw neural network takes a 3D structural representation of a protein, such as a PDB file. From the 3D structure, we generate five feature channels comprised of a residue pairwise distance matrix and four channels representing predicted secondary structure using STRIDE (25). The pairwise residue distance matrix is an $L \times L$ matrix $\mathbf{D}$ where element d_{ij} is the distance, in angstroms, between the α-carbon atoms of residues i and j. The predicted secondary structure is represented in two formats. The first is a co-membership matrix $\mathbf{C}$ where element c_{ij} is 1 if residues i, j are in the same secondary structure component, 0 otherwise. The second format indicates which residues occur at the start and end of secondary structure components. Each of the secondary structure representations is instantiated independently for helices and strands resulting in four secondary structure feature channels in total. In addition to the pairwise distance and secondary structure features, we include an additional two input channels which are initialised with zeros and subsequently filled with the model's predictions upon prediction recycling (see algorithm 2.)

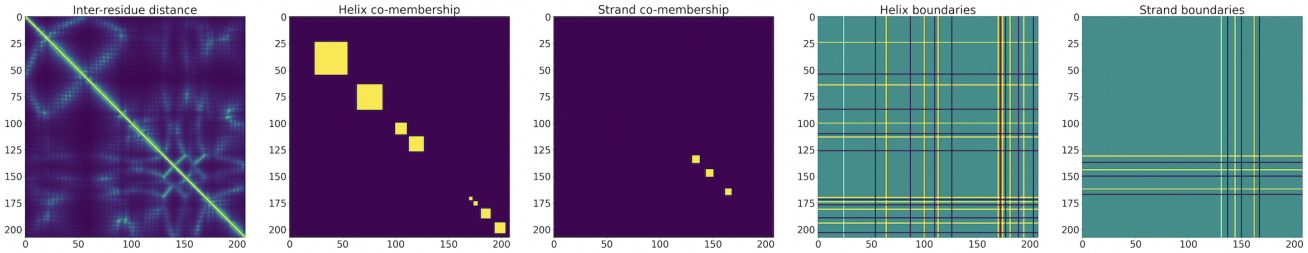


Figure 4. Input features for the Chainsaw neural network

2.3 Model architecture

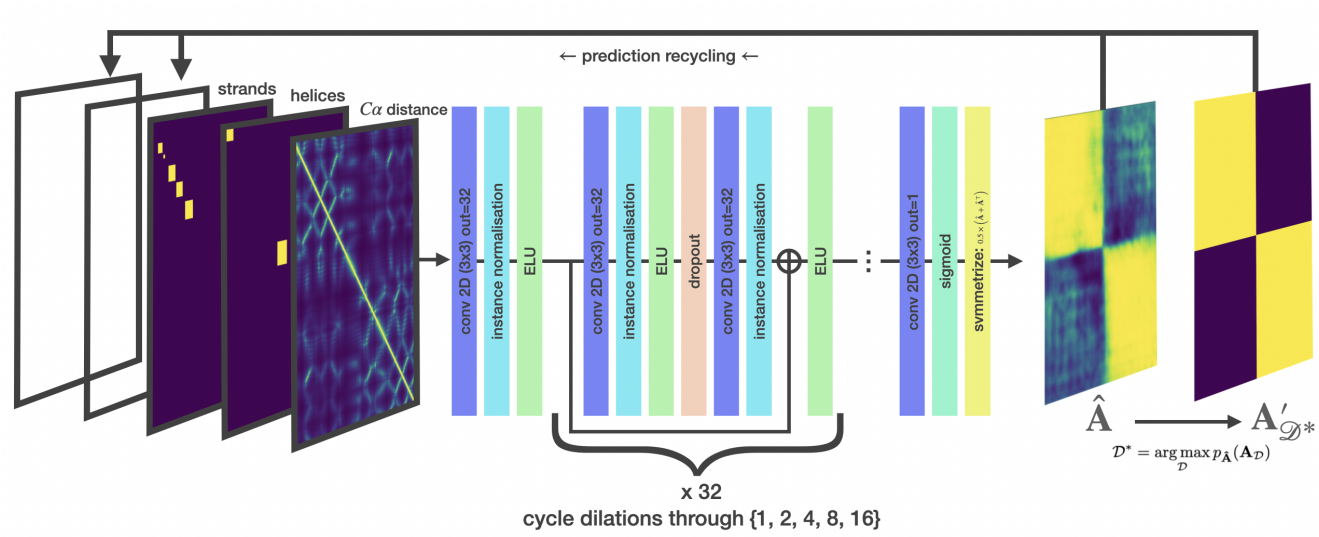


Figure 5. Chainsaw model diagram

The learning objective is to minimize the binary-cross entropy of the predicted soft adjacency matrix and the true adjacency matrix which represents the probability that residues co-occur in the same domain. To use the language of group theory, one advantage of this pairwise distance input representation lies in its SE(3) invariance, signifying that the representation remains unaltered under rigid transformations of the 3D structure, which consist of rotations, translations and reflections within the original coordinate space. An important theoretical motivation for using this representation is that a unique configuration of points in 3D space, up to rigid transformation, is fully specified by the 2D pairwise distance representation. Consequently, this implies that the 3D coordinates do not encode any supplementary valuable information in comparison to the pairwise distances. To put this another way, the 2D representation captures all relevant spatial geometry while ignoring nuisance factors such as the arbitrary selection of the coordinate system that is used to represent the 3D structure. Based on this fact, we propose that if a human annotator can assign domain boundaries by looking at the backbone 3D structure, then it should be possible for a supervised learning method to do the same based on 2D inter-residue distances.

We formulate the supervised learning problem as a 2D to 2D task: transforming the 2D input features into a pairwise probability matrix. As such, a deep convolutional network architecture with skip connections to overcome vanishing gradients is a natural choice. Due to the fact that domains are formed of contiguous segments of residues the output features exhibit strong local correlation which further motivates the use of convolutional layers. We used a modified version of the trRosetta architecture (26), a model originally developed for protein structure prediction. trRosetta is a residual network, whose blocks employ convolutional layers with progressively increasing dilation rates to achieve a wide receptive field. We enforce a symmetric-output constraint by adding the transpose of the final layer to itself to give the final symmetrized $\hat{\mathbf{A}}$. We truncate the trRosetta architecture to 31 blocks and reduce the number of filters from 64 to 32 but otherwise follow the original model in all details (26).

2.4 Domain assignment algorithm

The output of the neural network is an $L \times L$ matrix $\hat{\mathbf{A}}$, with entries $\hat{a}_{ij}$ representing the probability that residue i and residue j belong to the same domain. Let $A_{\mathscr{D}}$ be the binary matrix associated with a given domain assignment $\mathscr{D}$. To identify the most likely domain assignment $\mathscr{D}^*$ given a predicted set of adjacency probabilities $\hat{\mathbf{A}}$, we seek to find the $\mathscr{D}$ with maximum probability under $\hat{\mathbf{A}}$,

$$\mathscr{D}^* = \arg\max_{\mathscr{D}} p_{\hat{\mathbf{A}}}(\mathbf{A}_{\mathscr{D}}). \tag{2}$$

The probability of a domain assignment given the predicted values $\hat{\mathbf{A}}$ is given by the product of the entries

$$p_{\hat{A}}(A) = \prod_{i<j} p_{\hat{a}_{ij}}(a_{ij}) = \prod_{i<j} \hat{a}_{ij}^{a_{ij}} (1-\hat{a}_{ij})^{(1-a_{ij})} \tag{3}$$

where each $\hat{a}_{ij}$ represents the probability $a_{ij} = 1$. To perform the maximisation, we use a greedy algorithm, which exploits the fact that $\mathbf{A}_{\mathscr{D}}$ has a low-rank structure induced by the domain assignment $\mathscr{D}$. Let $\mathbf{v}_1, ..., \mathbf{v}_K$ be a sequence of binary indicator

vectors for each of the K domains. Let $V_{\mathscr{D}}$ be the matrix whose columns are the $\mathbf{v_i}$, hence $V_{\mathscr{D}}$ is a binary matrix with dimensions $L \times K$. The requirement that no residue can be assigned to more than one domain ensures that the rows of this matrix are K-dimensional one-hot vectors indicating domain assignments for each residue. Given this matrix of domain assignments, $V_{\mathscr{D}}$, the elements a_{ij} of the adjacency matrix $A_{\mathscr{D}}$ are generated as

$$A_{\mathscr{D}} = \sum_{k=1}^{K} \mathbf{v}_k \mathbf{v}_k^T \tag{4}$$

$$= V_{\mathscr{D}} V_{\mathscr{D}}^T \tag{5}$$

Thus our maximisation problem becomes to find a set of K vectors, such that the probability of domain assignment induced by the vectors is maximised

$$V_{\mathscr{D}}^* = \underset{V_{\mathscr{D}}}{\arg\max}\, p_{\hat{A}}(V_{\mathscr{D}} V_{\mathscr{D}}^T) \tag{6}$$

where the optimal number K of domains is itself unknown *a priori* and is therefore determined jointly with the domain assignments.

Algorithm 1 Greedy domain assignment

```
Â = nn(X)                                          ▷ predicted soft adjacency matrix
K = K_init                                         ▷ initial number of domains
V = zeros(L, K)                                    ▷ initialise domain assignments
for i < N_iter do
    for j < L do
        k* = argmax_k 𝓛_{j→k}(V, Â)                ▷ highest scoring domain assignment for residue j
        if 𝓛_{j→k*}(V, Â) > 𝓛_{j→∅}(V, Â) then     ▷ if assignment is better than null
            V_j ← 0                                ▷ remove any previous assignment
            V_{jk*} ← 1                            ▷ make new assignment
        end if
        if k* = K then
            K ← K + 1                              ▷ add an extra domain
            V ← concat(V, zeros(L, 1))
        end if
    end for
end for
```

Our domain assignment algorithm proceeds as follows. First V_D is initialised as an $L \times K_{init}$ matrix of zeros, where K_{init} is an initialisation for the number of domains ($K_{init} = 4$). For each residue in turn, we score each of the K possible domain assignments using the score $p(V_D V_D^T)$, and assign the residue to the domain which produces the maximal score, as long as the maximal score is greater than the score under the current assignment. In case of ties, the first domain is selected. After a complete pass through the sequence, the process is iterated a number of times to allow for corrections in initial assignments. As soon as some residue is assigned to the K_{init}^{th} domain, we add an extra column of zeros, corresponding to an 'overflow' domain which can be assigned to subsequently. This allows the algorithm to predict an arbitrary number of domains. The procedure is summarised in Algorithm 1. We note that the algorithm is incentivised to predict the correct number of domains: given a perfect predictor, then for any ground truth adjacency matrix A_D, all assignments other than the correct assignment are guaranteed to receive lower scores, including assignments which over- or under-predict the number of domains. Indeed for a perfect predictor, our algorithm is guaranteed to recover the correct domain assignment. Not all residues have to be assigned to a domain. The algorithm only makes an assignment if the score of the best possible domain assignment is greater than the score under a null assignment. Since each residue is initialised with a null assignment (corresponding to a row of zeros in V), residues for which no better assignment is found will maintain a null assignment throughout the procedure.

2.5 Self-conditioned predictions: prediction recycling

One restriction of our domain assignment algorithm is that it is non-differentiable and this prevents us from training the network using a loss that is derived from the final domain assignments. In order to mitigate this problem we trained the model to generate predictions based on the output of the predicted domain assignments. This is achieved by doing a first pass through the network followed by the assignment algorithm and using the results as additional inputs in the second pass through the network. A similar approach was demonstrated to have a significant positive impact on the performance of AlphaFold 2 (11)

and has been shown to be effective in similar problems (27). In our ablation tests, we observed that prediction recycling caused consistent improvements in domain prediction performance (See figure 7).

Algorithm 2 Self-conditioned training algorithm

```
R                                                  ▷ Max no. of recycles: set to 1 in our case
sample integer r uniformly from the set {0, 1, ..R}
A' ← 0                                             ▷ initialize predicted adjacency matrix with zeros
Â ← 0                                              ▷ initialize predicted soft adjacency matrix with zeros
for i < r do
    Â ← nn(D, Â, A')                               ▷ predict soft-adjacency matrix without gradients
    A' ← DomainAssigner(Â)
end for
Â ← nn(D, Â, A')                                   ▷ with gradients
Calculate loss
Update nn parameters with training step
```

2.6 Datasets

We generated a training dataset of chains with domain annotations from CATH 4.3. Following previous work (28), we use a topology-based split of chains to ensure minimal structural overlap between the training set and validation and test datasets. For validation and test datasets we use the chains belonging to the corresponding splits provided by (28), which were constructed by first splitting CATH topology codes into train, validation and test sets, then assigning chains of up to 500 residues in length containing domains in the CATH 4.3 40% sequence identity set to each split. Chains containing topology codes in multiple splits were excluded. Since the training split produced by (28) is restricted to chains containing domains in the 40% sequence identity non-redundant set, it sacrifices a significant amount of the fine-grained diversity of domain annotations in CATH. We therefore choose to re-generate a more complete set of training chains, by including in our training set all chains in CATH that contain no domain with a topology code belonging to either of the validation or test splits. To account for the additional redundancy introduced by this choice, during training we sample chains to train on in a redundancy-aware manner, by making use of CATH's sequence-identity-based clustering of domains. As such, one epoch is defined as one pass through all of the sequence clusters (clustered at 60% sequence identity). We additionally experiment with varying the relative frequency with which single and multi-domain chains are sampled as training datapoints. Our final approach sampled multi-domain proteins with probability 0.65 and single domain with probability 0.35. Hyper-parameter selection was based on performance on the validation set, final performance is reported on the non-redundant test set.

In order to show that our method is not overfitting to CATH assignments and to test that the approach generalizes to alternative domain assignments we evaluate an additional test set using the domain assignments from CASP 6. We choose CASP 6 because, unlike later CASP competitions, CASP 6 has a complete set of structures with well-annotated domain boundaries. Furthermore, in this competition, domain boundary prediction was specifically assessed by the organizers with published results (29). The CASP 6 domain boundary assessors point out that in a minority of cases, there are multiple ways that a protein chain can be divided into domains. In such cases, multiple sets of domain boundary definitions may be considered. In their proposed approach the prediction is scored according to the annotation that has the highest agreement with the prediction (29). However, we report scores based on the primary assignment only.

2.7 Assessment Metrics

The normalised domain overlap (NDO) score is a metric introduced in the CASP 6 domain parsing paper (29). Intuitively, this is a score roughly represents the proportion of residues that are assigned to the correct domain. The maximum score of 1 is achieved when all domain-labelled residues are assigned to their correct domains. A detailed description of the algorithm for calculating the NDO score can be found in (29). The domain boundary distance score was used in conjunction with NDO to assess domain boundary predictions in CASP 7. A detailed description of how the score is calculated is provided in (30). At a high level: each boundary is scored independently. Predicting within 1 residue of the true boundary scores 8 points, within 2 residues scores 7 and so on until the distance is 9 residues or more, at which point the score is 0. Each boundary score is divided by 8 so that scores per boundary are between 0 and 1. The final boundary distance score for the entire chain is then calculated as the sum of individual predicted boundary scores divided by the total number of domain boundaries. The number of domain boundaries comes from either the target or the prediction, whichever is higher. In this way, over-prediction is penalized.

3 Results

We find that Chainsaw outperforms UniDoc, PUU and SWORD2 on both the CATH domain test set and the CASP 6 test set. We note that the choice of metric (NDO, DBD or Domain Count Accuracy) can change the relative performance gaps for the models. In spite of this, Chainsaw attains the maximum score in 13 of the 15 dataset and metric combinations that we measured.

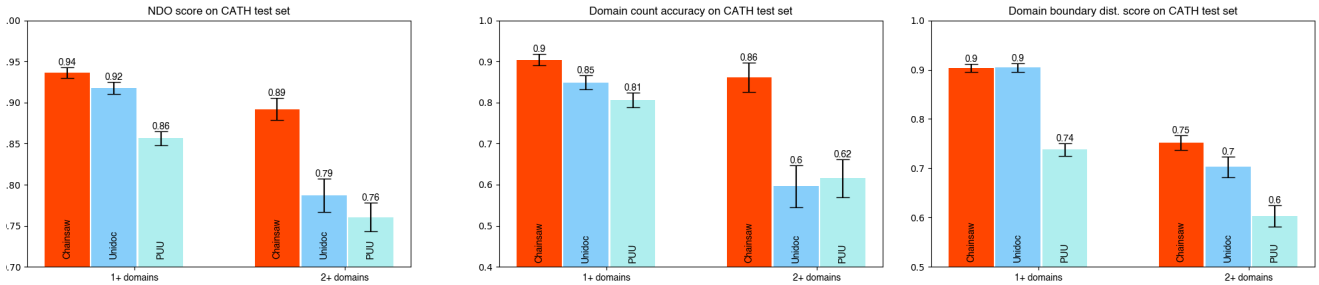


Figure 6. NDO score and domain count accuracy and domain boundary distance score with 95% CI

Table 1. Comparison of methods on CATH and CASP 6 datasets

Dataset	NDO Score			
	Chainsaw	UniDoc	PUU	SWORD2
CATH (n. domains $\geq$ 1)	**0.94**	0.92	0.89	-
CATH (n. domains $\geq$ 2)	**0.89**	0.79	0.76	-
CATH (n. domains $\geq$ 3)	**0.84**	0.83	0.74	-
CASP 6 (n. domains $\geq$ 1)	**0.92**	0.91	0.90	0.91
CASP 6 (n. domains $\geq$ 2)	**0.86**	0.80	0.81	0.83

Table 2. Comparison of methods on CATH and CASP 6 datasets

Dataset	Domain Count Accuracy			
	Chainsaw	UniDoc	PUU	SWORD2
CATH (n. domains $\geq$ 1)	**0.90**	0.85	0.81	-
CATH (n. domains $\geq$ 2)	**0.86**	0.60	0.61	-
CATH (n. domains $\geq$ 3)	**0.67**	0.59	0.48	-
CASP 6 (n. domains $\geq$ 1)	0.86	0.82	**0.87**	0.86
CASP 6 (n. domains $\geq$ 2)	**0.74**	0.61	**0.74**	0.70

Table 3. Comparison of methods on CATH and CASP 6 datasets

Dataset	Domain Boundary Distance Score			
	Chainsaw	UniDoc	PUU	SWORD2
CATH (n. domains $\geq$ 1)	**0.90**	**0.90**	0.76	-
CATH (n. domains $\geq$ 2)	**0.75**	0.70	0.61	-
CATH (n. domains $\geq$ 3)	0.73	**0.76**	0.59	-
CASP 6 (n. domains $\geq$ 1)	**0.87**	0.83	0.80	0.85
CASP 6 (n. domains $\geq$ 2)	**0.73**	0.62	0.62	0.66

3.1 Testing on large CATH chains

One limitation of the test split provided in (28) is that it is restricted to proteins with less than 500 residues and within this set the maximum domain count is in a single chain is 4. We, therefore, benchmark on an additional test set which consists of all CATH chains with more than 500 residues or more than four domains where at least one domain has a CATH topology code included the CATH test set.

Table 4. Comparison of methods on chains with more than 500 residues or more than four domains

Metric	CATH large test set	
	Chainsaw	UniDoc
NDO score	**0.76**	0.74
Domain count acc.	**0.45**	0.35
DBD score	**0.55**	0.54

Due to irresolvable errors in SWORD for a significant proportion of the CATH test set, we are unable to directly compare this method.

3.2 Predicting on AlphaFold predicted structures

When predicting domain boundaries on AlphaFold predicted structures as opposed to PDB structures we do not have a ground-truth set of labels. To overcome this problem we use SIFTS (31) to map PDB structures to their corresponding predicted structures in the AlphaFold database. This enables us to map a subset of the CATH test set domain annotations onto AlphaFold predicted structures. Using this approach we compare Chainsaw's performance on 1039 samples from the CATH test set. Table 5 shows that we do not see any significant decrease in performance when predicting on AlphaFold predicted structures as opposed to PDB structures. However, it is important to note that this subset of AlphaFold predicted structures is not representative of the AlphaFold database as a whole due to the fact that it only contains segments that have PDB structures and have been annotated by CATH.

Table 5. Comparing Chainsaw performance on AlphaFold predicted structures versus experimental structures from the PDB

Dataset	NDO Score	
	Chainsaw on AlphaFold	Chainsaw on PDB
CATH (n. domains $\geq$ 1)	0.93	0.94
CATH (n. domains $\geq$ 2)	0.91	0.90
CATH (n. domains $\geq$ 3)	0.94	0.95

3.3 Ablation analysis

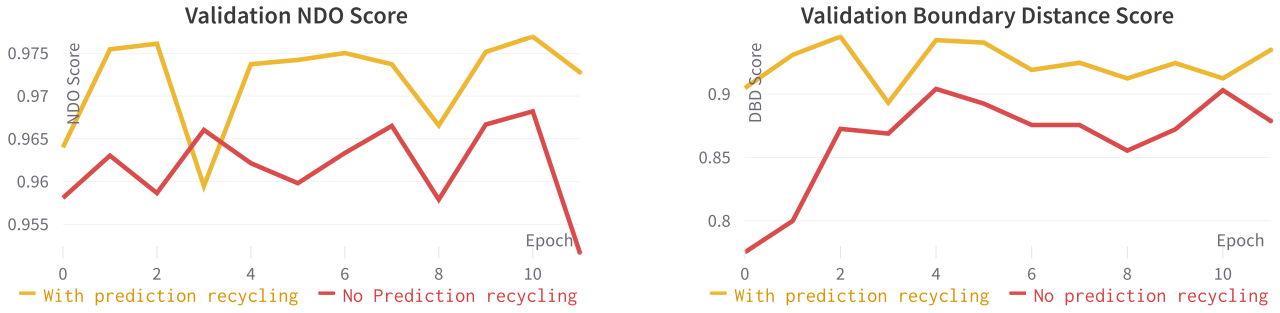
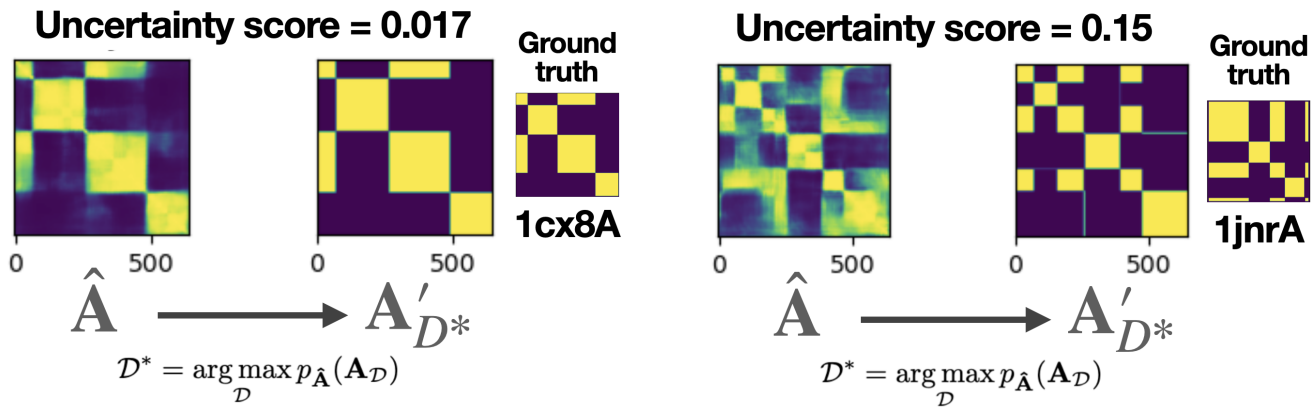


Figure 7. Prediction recycling improves model performance on the CATH validation data set

3.4 Uncertainty quantification

A natural approach to uncertainty quantification is to consider the output of the neural network $\hat{A}$ as an $L \times L$ multivariate Bernoulli distribution. Then we can consider the output of the final assignment A' as an observation from the $\hat{A}$ distribution and calculate the likelihood (normalised by the number of residues). Figure 8 compares instances where the model has high confidence (figure 8a) with other cases where the model confidence is lower suggesting that alternative domain assignments may be valid (figure 8b).



(a) High confidence model prediction where $\hat{A}$ is very close to A'.

(b) Low confidence prediction implies alternative assignments.

Figure 8. Negative log-likelihood of $\mathbf{A}'$ under $\hat{\mathbf{A}}$ is a good metric for model uncertainty

We observe that Chainsaw's uncertainty score, which is proportional to the residue-averaged negative log-likelihood of $\mathbf{A}'$ has good correlation with the loss (see figure 9). The uncertainty score achieves a Spearman's correlation score of 0.84 with the negative log-likelihood loss compared with the ground-truth label, and -0.70 with the NDO score when measured on the combined CATH and CATH large data sets.

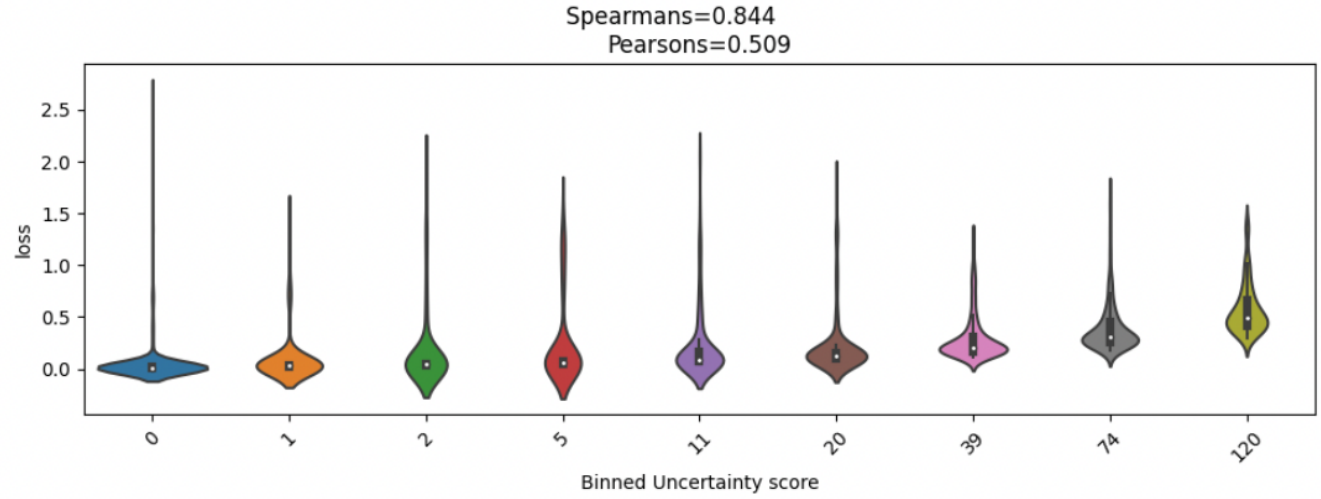


Figure 9. Correlation between Chainsaw's uncertainty score and log-likelihood loss on the CATH test set.

3.5 Inference times

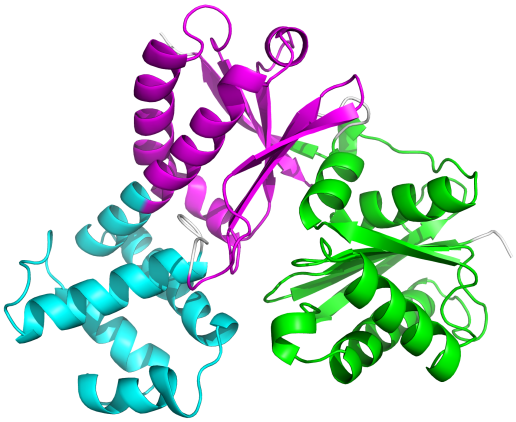
The inference time for each model was calculated by measuring prediction times on the test split of CATH data where the mean sequence length was 164 residues and the max sequence length was 500 residues.

Table 6. Inference times of different models

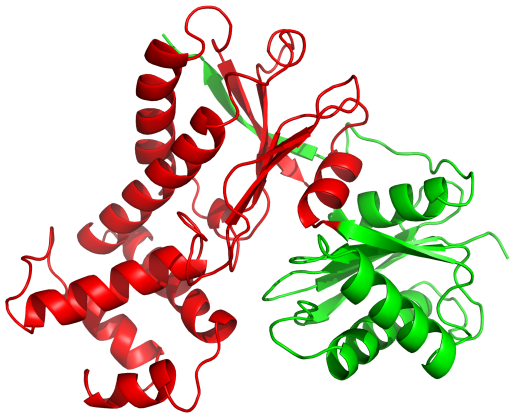
Mean run-time per chain (seconds)					
Chainsaw (CPU)	Chainsaw (GPU)	UniDoc (CPU)	SWORD (CPU)	SWORD2 (CPU)	PUU (CPU)
2s	0.8s	0.01s	3.2s	45s	0.01s

3.6 Illustrative examples where Chainsaw outperforms UniDoc

Due to the fact that UniDoc works by clustering secondary structure components, it is unable to predict domain boundaries on secondary structure residues. This can lead to domain parsing errors such as Figure 10b which are well handled by Chainsaw (Figure 10a).

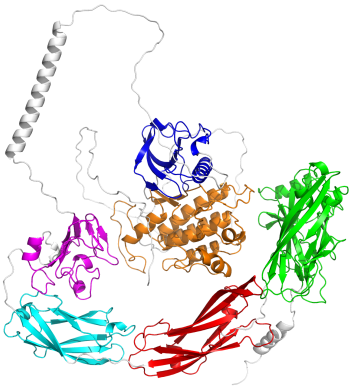


(a) Chainsaw prediction for AF-Q9UJ70-F1-model_v4

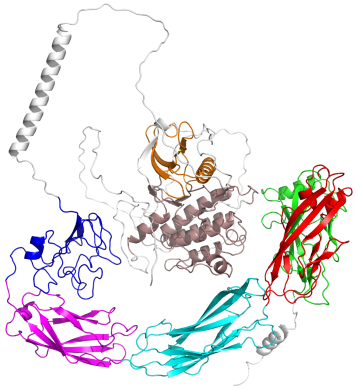


(b) UniDoc prediction for AF-Q9UJ70-F1-model_v4

Figure 10. UniDoc is unable to predict boundaries within secondary structure components leading to parsing errors, that are well-handled by Chainsaw.



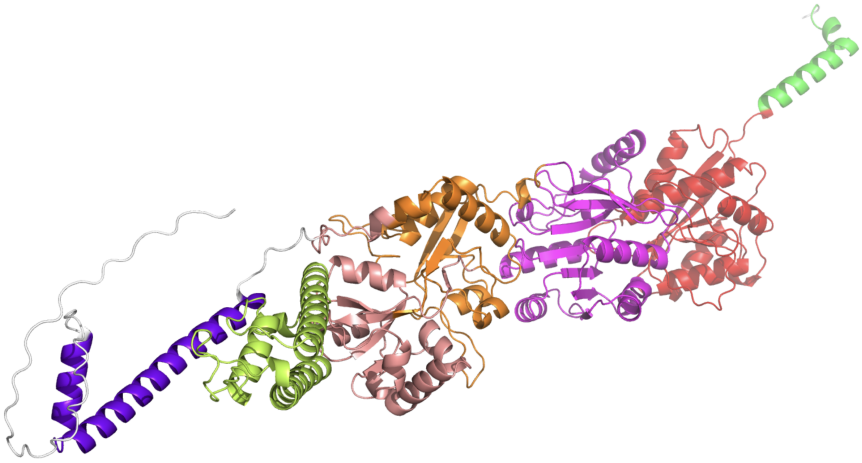
(a) Chainsaw prediction for AF-P07949-F1-model_v4



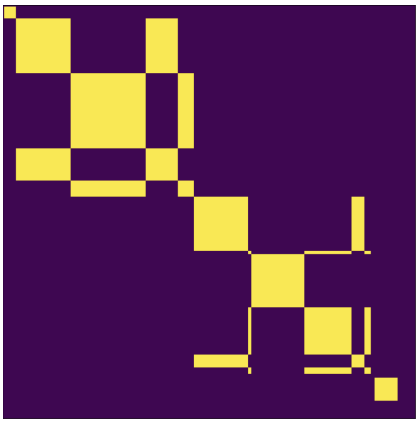
(b) UniDoc prediction for AF-P07949-F1-model_v4

Figure 11. Illustrative example where UniDoc over splits the green Chainsaw predicted domain (red/green for UniDoc predicted domain)

3.7 Illustrative examples of complex domain parsing on AlphaFold structures

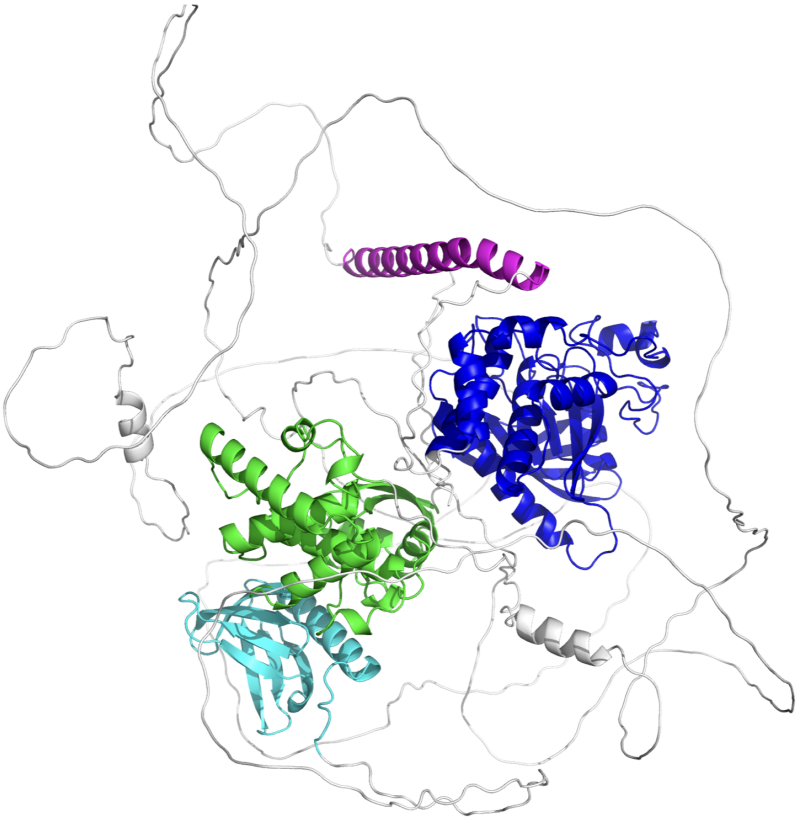


(a) AF-Q13002-F1-model_v4

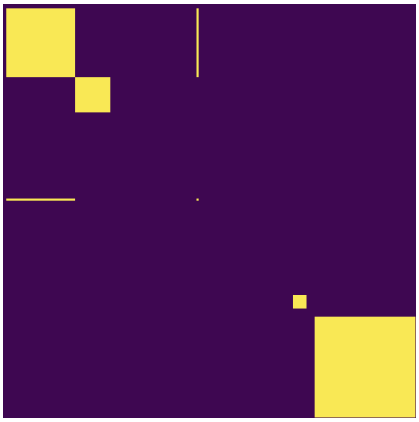


(b) $\mathbf{A}'$ for AF-Q13002-F1-model_v4

Figure 12. Chainsaw predicted domain assignment on AlphaFold predicted structure for Q13002

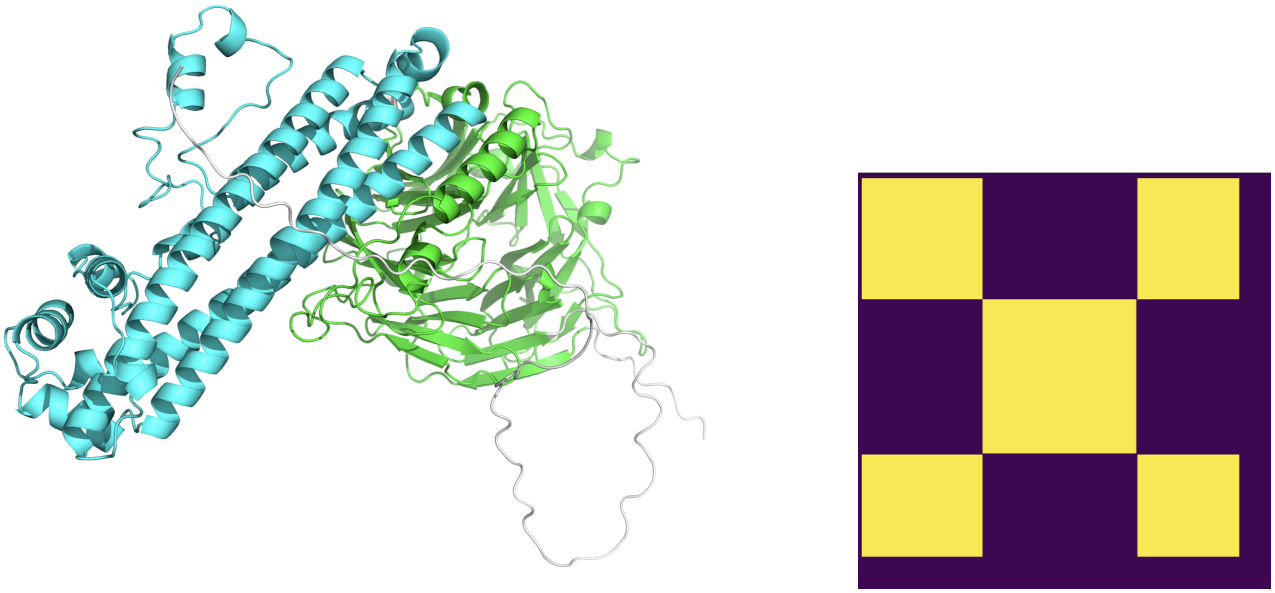


(a) AF-Q15678-F1-model_v4



(b) $\mathbf{A}'$ for AF-Q15678-F1-model_v4

Figure 13. Chainsaw predicted domain assignment on AlphaFold predicted structure for Q15678



(a) AF-Q9UJX5-F1-model_v4

(b) $\mathbf{A}'$ for AF-Q9UJX5-F1-model_v4

Figure 14. Chainsaw predicted domain assignment on AlphaFold predicted structure for Q9UJX5 showing successful prediction of a discontinuous domain (in green)

3.8 Comparison of Chainsaw and CATH-Resolve-Hits on Alphafold structures

To further evaluate the performance of Chainsaw on AlphaFold structures we conducted a comparison between Chainsaw domain predictions and the predictions generated by CATH-Resolve-Hits (CRH) method described in (32). The CRH approach uses sequence and structure-based domain matching and classification using embedded representations from protein language models. For the comparison, we sampled 100 AlphaFold structures with domain assignments allocated according to the CRH method and visually inspected the results.

- 27 assignments where Chainsaw was preferable
- 3 assignments where CRH was preferable
- 31 assignments where Chainsaw and CRH agree
- 28 assignments where the methods disagree but look equally plausible
- 8 assignments where both methods look implausible

Overall we found the assignments from Chainsaw to look more structurally plausible. Assignments generated by Chainsaw left fewer secondary structure elements as unclassified. Chainsaw does not assign domains to regions that are poorly modelled with few secondary structure residues, CRH can still assign domains to these regions based on sequence matches.

4 Conclusion

This study presents Chainsaw, a novel supervised learning approach to protein domain prediction, we leverage a residual convolutional neural network to estimate the soft adjacency matrix over residues and combine this with a novel algorithm converting the soft-adjacency matrix into domain assignments. We find that recycling the output of the domain assignments algorithm through the network as an additional feature improves the quality of the final assignments. Our approach outperforms state-of-the-art structure-based methods, even when predicting domains not observed during training. The ability of the Chainsaw model to predict non-domain residues and also the ability to predict domain boundaries within secondary structure components make it an attractive alternative to UniDoc (17). Another advantage of Chainsaw when compared with models such as DPAM (19) is the lack of dependencies on databases of known domains. Given that there are often multiple valid ways to parse a protein into domains we contend that domain parsing can likely be improved by joining an ensemble of methods to reach consensus or diversity of predictions. Domain predictors such as Chainsaw can be utilised in conjunction with structure-matching tools such as FoldSeek (33) to facilitate the detection of structural homology.

References

1. Hunt, A. C. *et al.* Multivalent designed proteins neutralize sars-cov-2 variants of concern and confer protection against infection in mice. *Sci. translational medicine* **14**, eabn1252 (2022).

2. Jeffery, C. J. Protein moonlighting: what is it, and why is it important? *Philos. transactions Royal Soc. B: biological sciences* **373**, 20160523 (2018).

3. Gough, J. Convergent evolution of domain architectures (is rare). *Bioinformatics* **21**, 1464–1471 (2005).

4. Devos, D. *et al.* Components of coated vesicles and nuclear pore complexes share a common molecular architecture. *PLoS biology* **2**, e380 (2004).

5. Orengo, C. A. *et al.* Cath–a hierarchic classification of protein domain structures. *Structure* **5**, 1093–1109 (1997).

6. Murzin, A. G., Brenner, S. E., Hubbard, T. & Chothia, C. Scop: a structural classification of proteins database for the investigation of sequences and structures. *J. molecular biology* **247**, 536–540 (1995).

7. Chandonia, J.-M. *et al.* Scope: improvements to the structural classification of proteins–extended database to facilitate variant interpretation and machine learning. *Nucleic acids research* **50**, D553–D559 (2022).

8. Cheng, H. *et al.* Ecod: an evolutionary classification of protein domains. *PLoS computational biology* **10**, e1003926 (2014).

9. Varadi, M. *et al.* Alphafold protein structure database: massively expanding the structural coverage of protein-sequence space with high-accuracy models. *Nucleic acids research* **50**, D439–D444 (2022).

10. Shi, Q. *et al.* Dnn-dom: predicting protein domain boundary from sequence alone by deep neural network. *Bioinformatics* **35**, 5128–5136 (2019).

11. Jumper, J. *et al.* Highly accurate protein structure prediction with alphafold. *Nature* **596**, 583–589 (2021).

12. Holm, L. & Sander, C. Parser for protein folding units. *Proteins: Struct. Funct. Bioinforma.* **19**, 256–268 (1994).

13. Alexandrov, N. & Shindyalov, I. Pdp: protein domain parser. *Bioinformatics* **19**, 429–430 (2003).

14. Postic, G., Ghouzam, Y., Chebrek, R. & Gelly, J.-C. An ambiguity principle for assigning protein structural domains. *Sci. advances* **3**, e1600552 (2017).

15. Cretin, G. *et al.* Sword2: hierarchical analysis of protein 3d structures. *Nucleic Acids Res.* **50**, W732–W738 (2022).

16. Zheng, W. *et al.* Fupred: detecting protein domains through deep-learning-based contact map prediction. *Bioinformatics* **36**, 3749–3757 (2020).

17. Zhu, K., Su, H., Peng, Z. & Yang, J. A unified approach to protein domain parsing with inter-residue distance matrix. *Bioinformatics* **39**, btad070 (2023).

18. Redfern, O. C., Harrison, A., Dallman, T., Pearl, F. M. G. & Orengo, C. A. Cathedral: a fast and effective algorithm to predict folds and domain boundaries from multidomain protein structures. *PLoS computational biology* **3**, e232 (2007).

19. Zhang, J., Schaeffer, R. D., Durham, J., Cong, Q. & Grishin, N. V. Dpam: A domain parser for alphafold models. *Protein Sci.* **32**, e4548 (2023).

20. Holm, L. Dali server: structural unification of protein families. *Nucleic acids research* **50**, W210–W215 (2022).

21. Söding, J., Biegert, A. & Lupas, A. N. The hhpred interactive server for protein homology detection and structure prediction. *Nucleic acids research* **33**, W244–W248 (2005).

22. Guo, J.-t., Xu, D., Kim, D. & Xu, Y. Improving the performance of domainparser for structural domain partition using neural network. *Nucleic acids research* **31**, 944–952 (2003).

23. Zhou, H., Xue, B. & Zhou, Y. Ddomain: Dividing structures into domains using a normalized domain–domain interaction profile. *Protein science* **16**, 947–955 (2007).

24. Mahmud, S., Guo, Z., Quadir, F., Liu, J. & Cheng, J. Multi-head attention-based u-nets for predicting protein domain boundaries using 1d sequence features and 2d distance maps. *BMC bioinformatics* **23**, 283 (2022).

25. Heinig, M. & Frishman, D. Stride: a web server for secondary structure assignment from known atomic coordinates of proteins. *Nucleic acids research* **32**, W500–W502 (2004).

26. Yang, J. *et al.* Improved protein structure prediction using predicted interresidue orientations. *Proc. Natl. Acad. Sci.* **117**, 1496–1503 (2020).

27. Chen, T., Zhang, R. & Hinton, G. Analog bits: Generating discrete data using diffusion models with self-conditioning. *arXiv preprint arXiv:2208.04202* (2022).

28. Hsu, C. *et al.* Learning inverse folding from millions of predicted structures. In *International Conference on Machine Learning*, 8946–8970 (PMLR, 2022).

29. Tai, C.-H., Lee, W.-J., Vincent, J. J. & Lee, B. Evaluation of domain prediction in casp6. *PROTEINS: Struct. Funct. Bioinforma.* **61**, 183–192 (2005).

30. Tress, M. *et al.* Assessment of predictions submitted for the casp7 domain prediction category. *Proteins: Struct. Funct. Bioinforma.* **69**, 137–151 (2007).

31. Dana, J. M. *et al.* Sifts: updated structure integration with function, taxonomy and sequences resource allows 40-fold increase in coverage of structure-based annotations for proteins. *Nucleic acids research* **47**, D482–D489 (2019).

32. Bordin, N. *et al.* Alphafold2 reveals commonalities and novelties in protein structure space for 21 model organisms. *Commun. biology* **6**, 160 (2023).

33. van Kempen, M. *et al.* Fast and accurate protein structure search with foldseek. *Nat. Biotechnol.* 1–4 (2023).