Finding Antibodies in Cryo-EM densities with `CrAI`

Vincent Mallet [1], **Chiara Rapisarda** [2], **Hervé Minoux** [2], and **Maks Ovsjanikov**[1]

[1]LIX, Ecole Polytechnique, IPP Paris
[2]Sanofi
maks@lix.polytechnique.fr

Abstract

Therapeutic antibodies have emerged as the most prominent class of new drugs due to their high specificity and their ability to target several molecules at the same time, in the case of multi-specific bio-therapeutics. Once an antibody has been identified, an optimization phase follows that includes affinity maturation, often based on the structure of the interface between the target and the antibody.

Cryo-EM is quickly emerging as the most efficient method to obtain such structures. It provides a noisy tridimensional map of probability of presence of electrons. The number, nature and structure of antibodies have to be inferred from the map. Automated methods exist for solving the more general problem of finding any protein in a density, but have a limited accuracy and usually require additional inputs, high resolution maps, and a long running time.

We propose a method dedicated to finding antibodies in cryo-EM densities : `CrAI`. This machine learning method leverages the conserved structure of antibodies and a dedicated novel database to solve this problem. We show that our method performs favorably compared to other methods, alleviating aforementioned limitations. Running a prediction takes only a few seconds and requires nothing but the cryo-EM density.

1 Introduction

Since the first monoclonal antibody entered the clinic in 1986 [14], antibody-based therapeutics have made considerable progress. With over a hundred compounds approved and forty just in the last three years [22, 42], the use of antibodies currently appears as one of the most promising approaches for designing new treatments for patients. One of the critical aspects of antibody-based therapeutics remains the binding of an Antibody (Ab) to a target molecule (the antigen) with high specificity [28], through their tips called the Complementarity Determining Regions (CDRs). The most classical approach has been to use monoclonal antibodies which contain two interacting domains called the Antigen Binding Fragment (Fab). More recently, an antibody fragments consisting of single variable antibody domain, originally engineered from heavy-chain antibodies generated by camelids [1] (nAbs or VHH) - have been rising in interest as an alternative form of therapeutics [12, 20]. Whether for Abs of VHHs, initial hits are found using immunization [19, 24] or phage display [30, 41]. These hits need to be optimized, to improve their capacity to selectively bind their antigen and their biological function but also to improve their efficacy, manufacturability and safety, before entering clinical studies. A specific binding to the antigen remains a critical property optimized in the early phase of the optimization process and is scrutinized until drug candidate selection. The usual rational way to optimize binding is to get the structure of the antibody [7], preferably in complex with the antigen. Indeed, knowledge of the atomic coordinates at the contact points between the Ab and its target, improves the understanding of its mode of action and guides the optimization of the affinity of binding.

Cryogenic Electron Microscopy (cryo-EM) is becoming the most common way to experimentally obtain protein structures of therapeutic antibodies bound to their target. In cryo-EM, the target protein is embedded in ice and exposed to an electron beam, resulting in raw noisy images of individual particles. These raw 2D images are then aligned and transformed into a 3D electron probability density [35, 39] from which atomic coordinates are inferred. Recent advances in data collection hardware and software, as well as better data processing pipelines have allowed an increase in data output and has resulted in more academic labs and big pharma [31] adopting the technology. Lately, a new data collection workflow has been shown to produce 3-4 Å structures of a pharmaceutically relevant target protein with 1 hour of instrument time and several hours of processing, thus allowing the theoretical resolution of potentially 24 structures a day [10]. While data collection is rapid, nowadays the data processing requires a lot of user input for simple decisions and requires days and weeks, also depending on availability of computer resources.

As automation allowed X-ray crystallography to become a key technique in the structure-based drug discovery pipeline [4], so it should for cryo-EM, making it cheaper and faster and freeing the time of researchers from button pressing tasks, to structure interpretation and drug engineering. The process of automation is being accelerated by the application of machine learning to the different stages of the pipeline, from data acquisition [5, 16], to preprocessing of micrographs [38], particle picking [2, 44, 45, 46, 47], 2D class selection [23] and 3D heterogeneity deconvolution [29, 48].

The last step of attribution of the map (going from 3D map to protein structure) is a challenging problem from a machine learning point of view, making off-the-shelf computer vision methods irrelevant. First, the training data is challenging : it is limited in size (about 10k aligned pairs of density maps and structures), redundant, noisy, with varying resolution, grid step and grid size. Second, Cryo-EM maps do not have a preferred orientation so any developed method must be able to handle arbitrary global rotations. Third, we aim to detect a pose (orientation) of our object and not solely its localization. Fourth, the different predictions interact with one another : for instance two antibodies cannot overlap.

For those reasons, the rigid body fitting of atomic coordinates of a protein into the density is still a largely manual process using ChimeraX [32], followed by the local optimization of atomic coordinates using Coot [15]. Some automatic procedures were developed to help automate this process [26], although with limited accuracy. Tools based on machine learning to trace the sequence in the density were recently developed with good results [18, 34], but are for now limited to resolutions of 4Å and better. Moreover, these tools display a long running time, incompatible with high throughput imaging. In the context of using cryo-EM for optimization of therapeutic Abs, we can solve a simpler task : finding Abs in the density and segmenting the map around them. Our goal is to automate that step using a machine learning approach. We leverage the conserved structure of Abs as a prior information to make our learning problem simpler [9]. Hence, we reduce the problem of finding Abs in the density to an instance of 3D object detection [36, 49] : learning a rigid transform for a template Ab.

In this paper, we address the problem of finding Abs (Fabs and VHHs) automatically and quickly, at all resolutions better than 10Å, without manual intervention or any additional inputs beyond the density. We formulate and parametrize the problem in section 2.2. We then gather a clean database of aligned structures and densities in section 2.3. Finally, we solve our problem by using the database to build an object detection machine learning method in section 2.4. We successfully find Abs in over 90% of systems, widely outperforming `dock_in_map` both in terms of speed and performance.

2 Methods

2.1 Overview

The design of our approach is motivated by aforementioned methodological challenges. First, due to the challenges inherent in the limited scale and significant noise present in the data, we incorporate the conserved structure of antibodies as a prior information to make learning more data-efficient and robust. Second, to facilitate antibody pose finding in the presence of limited data, we use a custom parametrization of the rotation that takes into account an "expected" pose, while being flexible to allow arbitrary orientations. Third, we introduce a fully convolutional design to accommodate arbitrary grid sizes that might be present at inference time. We also train the network with rotation augmentation to approximate rotation equivariance. Finally, we introduce a custom loss when training our model. Our loss incorporates a formulation based on Optimal Transport along with persistence diagrams to better capture the geometric aspects of our problem, such predicting non-overlapping objects and including *distance*-based penalties between our prediction and the ground truth.

2.2 Problem formulation

Starting from an input cryo-EM map, we want our method to output the 3D coordinates of one or several Abs. Because of the highly conserved structure of Abs, we can simplify our problem by only predicting how to align a fixed antibody template $\mathbf{T}$ with the Abs, without deformations.

More formally : Let $\mathcal{X}$ be the cryo-EM density we consider, $n_{\mathcal{X}}$ the number of Abs contained in this density and $\mathcal{A}_{\mathcal{X}} = \{\mathbf{A}_i, 0 \leq i < n_{\mathcal{X}}\}$ the set of such Abs. Given an Ab $\mathbf{A}$ and a registration objective d, let $R^*_{\mathbf{A}}, T^*_{\mathbf{A}} = \min_{R,T\in(SO_3(\mathbb{R})\times\mathbb{R}^3)} d(\mathbf{A}, R\mathbf{T}+T)$ be the translations and rotations that best align $\mathbf{A}$ onto $\mathbf{T}$. Finally let $\mathcal{S}_{\mathcal{X}} = \{(R^*_{\mathbf{A}_i}, T^*_{\mathbf{A}_i}), 0 \leq i < n_{\mathcal{X}}\}$ be the set of optimal alignments. In this paper, we aim to predict $\mathcal{S}_{\mathcal{X}}$ instead of $\mathcal{A}_{\mathcal{X}}$.

The optimal rotations mentioned above can be parametrized in many ways. Let $\vec{p}_{\mathbf{A}}$ denote the unit vector oriented from the center of mass of an Ab $\mathbf{A}$ towards its antigen. We believe it is easier to predict $\vec{p}_{\mathbf{A}}$ than the rotation around $\vec{p}_{\mathbf{A}}$ from the density. Therefore, we decomposed $R^*_{\mathbf{A}}$ into a rotation transforming $\vec{p}_{\mathbf{T}}$ into $\vec{p}_{\mathbf{A}}$ and one 2D rotation around $\vec{p}_{\mathbf{T}}$ of angle $\theta_{\mathcal{A}}$. The generality of this decomposition is established in Supplemental A.1. In practice, we chose the orientation of the template to have $p_{\mathbf{T}} = \vec{u}_z$. We assess the relevance of this decomposition in the results section.

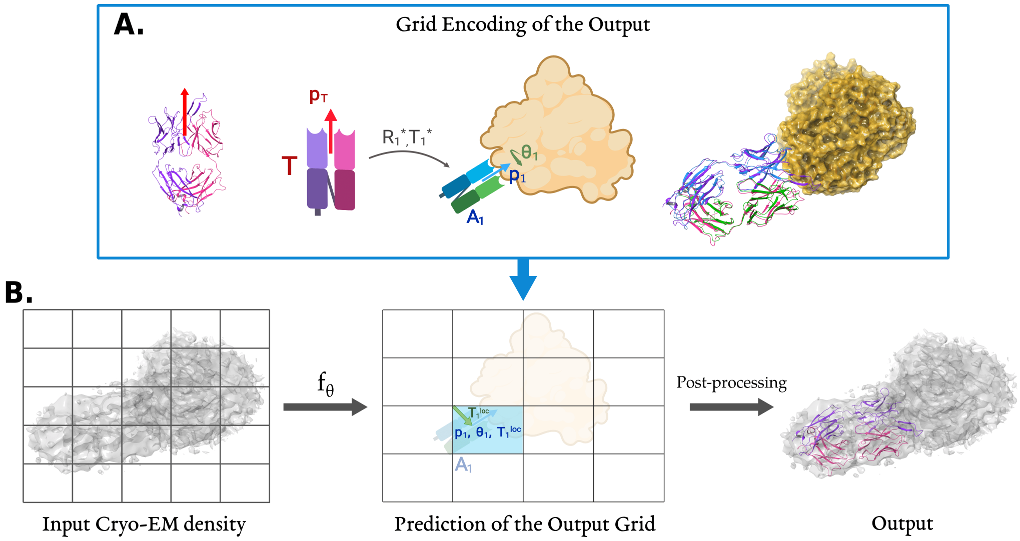


Figure 1: `CrAI` pipeline : From an input density, we train a network to predict alignments of a template encoded as a grid, and then process this grid to get a prediction as a protein structure.
A. Encoding of the output of `CrAI`. (*Left to right*) : The structure of our template (pdb `7lo8`) is displayed next to its cartoon, with $\vec{u}_z$ shown in red. We compute optimal alignments $R^*_{\mathbf{A}}, T^*_{\mathbf{A}}$ of our template onto Abs and decompose the rotation into $R^*_{\mathbf{A}} = (\vec{p}_{\mathbf{A}}, \theta_{\mathbf{A}})$. Finally, we show half the structure of system `6bf9` : antigen (chain A in orange) and Fab (blue, green), along with our aligned template (red, purple). As can be seen, our template aligns well to other antibodies (RMSD = 1.8Å). (*center bottom*) : we decompose the translations into a position in a grid $T^{cell}_{\mathbf{A}}$ and an offset from the grid corner $T^{loc}_{\mathbf{A}}$. We thus obtain a grid with zero values except for cells containing an Ab, in those cells, we have $T^{loc}_{\mathbf{A}}$, $\vec{p}_{\mathbf{A}}$ and $\theta_{\mathbf{A}}$.
B. We then train a network to take as input a cryo-EM density grid and to predict the output grid generated as explained above. At test time, we only use the grid as input. A post-processing is then applied to transform the predicted output grid into predicted antibody locations. Details on these steps are provided in section 2.4.

Our problem is now formulated as an object detection problem. Following the best practice in this domain, we overlay an occupancy grid $\mathcal{G}^{\mathcal{X},\mathbf{s}}$ of size $\mathbf{s}$ over our input : this grid contains ones for cells encompassing an Ab and zeros elsewhere. We decompose each translation into two parts : one going to the corner of the occupied cell and one from this corner to the Ab : $T^*_{\mathbf{A}} = T^{cell}_{\mathbf{A}} + T^{loc}_{\mathbf{A}}$, as shown in Figure 1A.

2.3 Database

Building a database To solve our problem, we start by building a curated antibody database. Fabs are composed of one constant and one variable domain of each of the heavy and the light chain. The two variable domains are denoted as the Fv variable fragment (Fv). Our database will include both Fabs and Fvs. The reason why we put these two together is that the constant region of Fabs is often missing in the structure file, but present in the density. Hence we will aim to predict the Fv of either Fabs or Fvs. The data is originally fetched from *SabDab*[13]. We fix a few broken annotations, notably for systems containing both Fabs and VHHs - more details in Section A.3. Using the PDB[3], we find the corresponding cryo-EM maps and download all corresponding maps and structures. Then, we remove systems with resolution below 10Å or ones with no antibodies or antigen chains, and group all hits pertaining to the same structure together, yielding a total of 1032 systems. We then split this data set in 722, 155 and 155 systems in the train, validation and test splits respectively. To split the data, we compared two approaches : a randomly shuffled one (`random`) and a temporal one with more recent systems in the test split (`sorted`). We will sometimes monitor performance at the individual Fab level, and refer to it as `ab`, in opposition to aggregating the performance by systems, which we will refer to as `sys`. The number of Fab per split is 1627, 320 and 374 respectively, so there is a mean number of 2.25 Fabs per system in our database.

Once equipped with those splits we are ready to do machine learning. However, the map files can be enormous (up to 10^9 grid cells). In addition, the maps are not centered for viruses : the asymmetric unit only occupies a fraction of the map, which would create false negatives (some parts of the density are associated with proteins that were omitted in the structure file, for instance pdb `7kcr`). To deal with this, we replace the original maps with ones centered around the structure with a margin of 25A that covers most of the normal cases while limiting those false negatives. Of course, at use time, users can provide a cropped map or a full one. Each of these maps is resampled to get a fixed voxel size of two Å. Then we normalize the map values by zeroing out negative values and dividing by the maximum of the map. We now have a split database of cropped, resampled, normalized cryo-EM density maps.

Template management We now need an antibody template to transform other antibodies into translations and rotations. To get this template, we pick a Fab system (we used pdb `7lo8`) and manually select the Fv residues. Then we shift the system so that the Fv is centered at the origin and align its main axis with the $u_z = (0, 0, 1)$ vector using PCA. We save these Fab and Fv as templates.

For a given system, we align the 3D structure in its density. We then label each Ab it contains as either a Fab or an Fv, based on their number of residues. Finally, we compute the optimal alignments between these labeled Ab and their corresponding templates, using `pymol align`[11]. We provide an example alignment of our templates in Figure 1A.

VHHs All of these steps apply to the production of VHH data. We started from *SAbDab-nano*[40] and applied the same filters on the raw data, resulting in 398 systems. Then, we performed the *random* and *sorted* splitting procedures as above. This amounts to 278, 60, 60 `systems` in each split and 458, 74, 86 when counted as `ab`, for a mean of 1.55 VHHs per system. We then used the VHH domain of the Fv template as a VHH template.

2.4 `CrAI`: a model to find antibodies in cryo-EM densities

Architecture and learning procedure We aim to solve the object detection problem stated above with our data set. To do so, we will follow the YOLO [37] framework. Given a cryo EM density of size s_i, $\mathcal{X} \in \mathbb{R}^{s_i}$ and its corresponding occupancy grid of size $\mathbf{s}$, $\mathcal{G}^{\mathcal{X},\mathbf{s}}$, our network is a function $f_\theta : \mathbb{R}^{s_i} \to \mathbb{R}^{10\times\mathbf{s}}$ such that $\hat{y}(i, s) = f_\theta(\mathcal{X}) \in \mathbb{R}^{10\times\mathbf{s}}$ is our prediction for $\mathcal{X}$. The first dimension of this output is a prediction of the occupancy grid $\mathcal{G}^{\mathcal{X},\mathbf{s}}$. The other dimensions are predictions in each cell relative to the putative Ab contained in it. The detailed computation of the loss is detailed below.

The architecture of our model is a simple 3D UNet [8] with a depth of 4. While dedicated models were trained on Fabs, we found that VHH prediction performance benefited from training on both data. The network is trained using data augmentation, with the eight possible rotations over a grid and random cropping inside the input grid up to three cells. We used Adam optimizer over 1000 epochs. Hyper-parameters and exact architecture were not extensively optimized. Finer details about the ones we used can be found in Supplementary A.2.

Prediction of the right cells using optimal transport Let $\hat{\mathcal{G}}_s = \hat{y}(0, s)$ our predicted occupancy grid. For convenience in this section, we drop indices and denote $\mathcal{G}^{\mathcal{X},\mathbf{s}}$ as $\mathcal{G}$. In order to make our prediction closer to $\mathcal{G}$, we will use two loss terms.

Because most grid cells are unoccupied, our prediction is very imbalanced. Hence, our first loss term is a weighted binary focal loss [27] :

$$\mathcal{L}_1(\hat{\mathcal{G}}, \mathcal{G}) = \sum_s (\lambda * \mathcal{G}_s * (1 - \hat{\mathcal{G}}_s)^\gamma * log(\hat{\mathcal{G}}_s) + (1 - \mathcal{G}_s) * \hat{\mathcal{G}}_s^\gamma * log(1 - \hat{\mathcal{G}}_s)). \tag{1}$$

We used $\lambda = 30$ and $\gamma = 4$ in our experiments.

However, this loss does not consider the distance between our prediction and the ground truth. If we predict the neighbor pixel, we have the same value as if we had predicted the opposite side of the map. To address this issue, we add an optimal transport term to our loss [33]. It gives meaningful supervision to all voxels of our grids that depends on the distance to the closest occupied voxel. We used the *GeomLoss* package [17] to compute this loss and refer to it as $\mathcal{L}_2(\hat{\mathcal{G}}_s, \mathcal{G}_s)$ without detailing its mathematical expression here. The relevance of this loss is assessed in the Results section.

Prediction within each cell Let us now detail the loss terms that are used for the other nine dimensions of our output. We emphasize that these will only be applied on pixels containing an antibody. Let $\mathbf{A}$ be an antibody in our system, $s_\mathbf{A}$ its position and $\hat{y}^i_\mathbf{A} = \hat{y}(i, s_\mathbf{A})$ the prediction at this position. We use the notation $\hat{y}^{j:j+k-1}_\mathbf{A}$ to denote the k dimensional vector obtained from the concatenation of $\hat{y}^j_\mathbf{A}, \hat{y}^{j+1}_\mathbf{A} \cdots \hat{y}^{j+k-1}_\mathbf{A}$.

To predict the right pose of $\mathbf{A}$ in the cell, we use eight values. The three first are used to predict the offset from the corner of the grid cell, $T^{loc}_\mathbf{A}$. We use a mean squared error loss to learn these values, $\mathcal{L}_3$. The following three values are used to predict $\vec{p}_\mathbf{A}$ by directly predicting its coordinates. The corresponding loss, $\mathcal{L}_4$, is composed of a dot product term to control the direction of the prediction along with a term to make this vector a unit one. Finally, the remaining two values are used to predict the angle $\theta_\mathbf{A}$. Instead of directly predicting the angle, we aim to predict $u_\mathbf{A} \in \mathbb{R}^2$, the vector of polar coordinates $(1, \theta_\mathbf{A})$. This formulation avoids singularities and was shown to be beneficial when predicting angles [21]. $\mathcal{L}_5$ has a similar form than $\mathcal{L}_4$ in two dimensions to predict $u_\mathbf{A}$. We end up with the following losses :

$$\mathcal{L}_3(\hat{y}^{1:3}_\mathbf{A}, T^{loc}_\mathbf{A}) = \texttt{mse}(\hat{y}^{1:3}_\mathbf{A}, T^{loc}_\mathbf{A}), \tag{2}$$

$$\mathcal{L}_4(\hat{y}^{4:6}_\mathbf{A}, p_\mathbf{A}) = 1 - \langle \hat{y}^{4:6}_\mathbf{A}, p_\mathbf{A} \rangle + \texttt{mse}(\|\hat{y}^{4:6}_\mathbf{A}\|, 1), \tag{3}$$

$$\mathcal{L}_5(\hat{y}^{7:8}_\mathbf{A}, u_\mathbf{A}) = 1 - \langle \hat{y}^{7:8}_\mathbf{A}, u_\mathbf{A} \rangle + \texttt{mse}(\|\hat{y}^{7:8}_\mathbf{A}\|, 1). \tag{4}$$

Finally, we use a single model for both Fabs and VHHs, hence we have a final term that represents the probability that the object contained in the grid cell is a VHH and not a Fab. Let $\delta_n(x)$ be the indicator function for VHHs (one if VHH else zero). We again used a weighted focal loss, with a weight of $\lambda_n = 1000/400$ corresponding to the ratio of VHHs to Fabs, yielding the last loss,

$$\mathcal{L}_6(\hat{y}^9_\mathbf{A}, \mathbf{A}) = \lambda_n * \delta_n(\mathbf{A}) * (1 - \hat{y}^9_\mathbf{A})^\gamma * log(\hat{y}^9_\mathbf{A}) + (1 - \delta_n(\mathbf{A})) * (\hat{y}^9_\mathbf{A})^\gamma * log(1 - \hat{y}^9_\mathbf{A})). \tag{5}$$

To train our network we use a weighted sum ($\lambda_s = 0.2$) of previous loss terms as the final loss. We sum the first two global ones and the sum of the four others over each antibody in our system. :

$$\mathcal{L}_{tot}(\hat{y}, \mathcal{X}) = \sum_{i=1}^{2} \mathcal{L}_i(\hat{\mathcal{G}}, \mathcal{G}) + \lambda_s * \sum_{i=3}^{6} \sum_{\mathbf{A} \in \mathcal{A}_\mathcal{X}} \mathcal{L}_i(\hat{y_\mathbf{A}}, \mathbf{A}). \tag{6}$$

Post processing A well known problem with object detection is the possibility that the network predicts overlapping objects. Suppose grid cells edges are 3Å long, adjacent cells cannot both contain a Fab. Hence, high values for adjacent cells amount to the detection of the same object. Non Maximal Suppression (NMS) algorithms are used to discard such redundant predictions. In this paper we used `persistence diagrams` [43]. Simply put, we decrease a threshold probability value from the maximum of the grid and keep track of cells above this threshold. When the value of a cell goes over the threshold, either it has no neighbors belonging to a visited connected component, giving *birth* to a new one, or all neighboring components are merged into the one with lowest initial values and merged ones *die*. The difference between the values of death and birth are called lifetimes. We return connected components sorted by lifetimes.

This procedure takes into account both the value of a minimum and its distance to other minima. If we suppose that the number of objects to find is known, we keep proceeding until it is reached (and refer to this setting as and `num`). Otherwise we use a lifetime threshold and retain all predictions above this threshold. We use a threshold of 0.2 that was found to work best on the validation set (see Supplementary A.4). We will denote this setting `thresh`. Interestingly, this means that we can automatically detect the numbers of Abs in a density !

From our density map, we now have a list of predicted values. For each of those, we choose a template based on the classification in Fabs and VHHs and move the template to the predicted location and pose. We save the result as a file in PDB format.

3 Results

3.1 Experimental setup

We have trained independent models for our random and sorted splits. We make inference of those models on their respective test sets and over the maps obtained using the procedure described above. This ends up with a PDB with one or several predicted Fv positions. We then use the Hungarian algorithm [25] to match several systems together based on the distance between their center of mass. We report the results for individual Abs (`ab`) as well as aggregated by systems (`sys`) so that a system with many Fabs does not influence the results too much.

To our knowledge, no tool enables predicting the position of Fabs solely from any density. We benchmark against `dock_in_map` [26], a tool that takes the density along with the structures of objects that need to be docked in the map. At test time, we normally do not have access to the ground truth structure. We ran `dock_in_map` in two setups : the template based just aims to dock a fixed template Fab or Fv and the ground truth based that uses the actual structure giving an upper bound of the performance.

First, we focus on the distances between centers of mass as they discriminate hard failure from approximate success. We present a capped histogram of the distances predicted by our model and by the tool in Figure S.2. As expected, we notice that `dock_in_map` performs significantly better when used with the ground truth Fab structures than with templates. Moreover, our tool seems to get better distances and less failures than `dock_in_map`. Finally, we observe that the predicted distances are bimodal : a first peak corresponds to systems predicted successfully with distances below 10Å and another spread mode corresponds to failed prediction.

Based on these observations, we will now only compare to the more challenging scenario of `dock_in_map` ran with ground truth. We will also report our results in terms of hit rates : a prediction is deemed as a hit if it is closer than 10Å to the ground truth. As can be seen in the figure, performance is not very sensitive to the choice of this threshold. When the number of Fabs is wrongly predicted, we add the difference as failed cases.

3.2 `CrAI` accurately finds Abs

The results of Fab detection are presented in Table 1. Looking at the first two lines, we see that `CrAI` drastically outperforms `dock_in_map` in terms of hit rates. This holds in all settings for an overall hit rate going from approximately 69% to 95%. In terms of distance between the centers of mass prediction and the ground truth, both methods give results below 2.5Å which is close to optimal considering the map resolutions. `dock_in_map` appears to be a little bit more precise, even though this mean is only computed on successful systems and thus does not include 25% that are included in the `CrAI` average. One could combine both tools by first screening a map with a high hit rate, then precisely refine the result.

Method	Hit rate						Distance	Runtime	
	random		sorted		mean				
	sys	ab	sys	ab	sys	ab		CPU	GPU
`dock_in_map`	67.5	62.2	74.8	71.1	71.2	66.7	**0.81A**	883s	-
`CrAI`	**93.5**	**93.8**	**97.3**	**97.1**	**95.4**	**95.4**	2.15A	**0.41s**	**0.031s**
`CrAI thresh`	91.5	88.8	95.6	93.8	93.5	91.3	2.09A	**0.41s**	**0.031s**

Table 1: Performance of our tool at detecting Fabs. We report the performance of the benchmark tool and of ours, provided or not with the ground truth number of objects. We report the Hit Rate, defined as the number of prediction below a threshold of 10Å, the mean distance of these successful predictions, and the mean run time of the tool

`CrAI` can be used with unknown number of Fabs If we now compare the second and third lines of the Table, we see that we retain most of the performance when providing the network with *only the density*. This is to compare to the result of `dock_in_map` that is additionally given the experimental structures of all antibodies that are to be found in the map. We believe this to be the main result of our paper.

`CrAI` runs fast, at all resolutions The average runtime of `dock_in_map` on our validation set is a prohibitive 883s/system. This mean is optimistic because we stopped a few systems after 5 hours of computations (letting them run longer would increase the mean). In comparison, our tool runs in 0.031s/system, thirty thousand times faster when using a GPU. When using only one CPU, our tool runs in 0.41s/system, two thousands times faster than `dock_in_map` and fast enough for this computational step to integrate seamlessly in an analysis pipeline. This can be explained by the fact that our algorithm is linear with regards to the grid size.

Supplementary B.2 shows how our performance depends on resolution of the input map, over all of our data sets. We do not see a strong dependency and claim that `CrAI` is robust to low resolution. This is a novel result as machine learning methods for tracing such as ModelAngelo [18] only work for resolutions below 4Å.

`CrAI` finds VHHs We now turn to the prediction of VHHs. They are more challenging for a machine learning model, because we have less training data. Moreover, the binding modes of VHHs are less canonical which breaks the decomposition of our rotation. Finally, VHHs are smaller and tend to be more buried into the density - as opposed to Fabs often sticking out. Our results are presented in Table 2.

Method	Hit Rate		Hit rate sorted			Hit rate Mean			Distance
	random		sorted			mean			
	sys	ab	sys	ab	ab_2	sys	ab	ab_2	
`dock_in_map`	76.7	67.4	87.5	**84.9**	81.4	82.1	76.2	74.4	**0.7A**
`CrAI num`	**93.1**	**90.7**	**88.7**	70.9	**85.7**	**90.9**	**80.8**	**88.2**	2.2A
`CrAI thresh`	92.9	86.1	87.1	64.0	77.1	90.0	75.0	81.6	2.1A

Table 2: Performance at detecting VHHs. The convention used is the same as above. We add the setting ab_2 that corresponds to removing one outlier system from the validation.

We achieve a positive result with a 10% hit rate boost over `dock_in_map` overall. Beyond the intrinsic difficulties outlined above, we underline three additional factors to explain why this result is weaker than for Fabs. `dock_in_map` performs much better for VHHs than for Fabs, which could stem from the reduced size of the objects. It performs even better for the `sorted` split (performance over the whole data is similar to *random*), which can be explained by the variance of estimators on small sized test sets. Finally, among the 60 systems in our test set (84 VHHs), one system (pdb 8gq5) contains 16 VHHs arranged in an unusual way, surprisingly well captured by `dock_in_map` but missed by our tool. This single system biases the `ab` statistics against `CrAI`. We include the statistics obtained when ignoring this outlier and refer to it as $\texttt{ab}_2$ in the Table.

3.3 Further analysis

Ablation study To assess the relevance of different design choices, we retrain a few models without those features. Because this means retraining a model every time, we only perform this analysis in the Fab `random` split setting. We report the performance in the `sys` setting, but results are consistent in the `ab` setting. We try removing Persistence Diagrams (PD) and replacing them with a naive NMS. We also tried training without using the Optimal Transport (OT). We present the results in Table 3.

	Hit Rate	
Method	`num`	`thresh`
without PD	**93.8**	89.9
without OT	90.3	55.9
$u_z \rightarrow u_y$	91.7	90.6
`CrAI`	93.5	**91.5**

Table 3: Ablation study of our method on the Fab `random` split, in the `sys` setting.

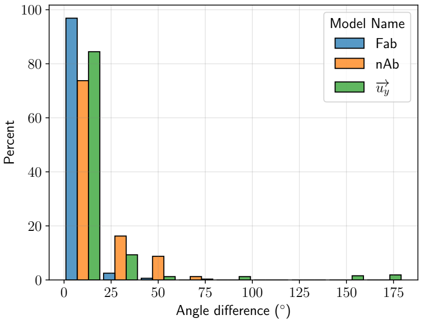


Figure 2: Histograms for the values of the angle between the predicted and experimental vector for Fabs, VHHs and $\vec{u}_y$ model

When removing optimal transport, the ability to predict the number of objects collapses. Finally, persistence diagrams seem to work better in the `num` setting and worse otherwise, with a limited impact.

`CrAI` finds meaningful poses We have validated the position of our predictions, now let us comment on the predicted poses : are Abs in the right orientation ? Using the decomposition of rotations into predicting a vector $\vec{p}_{\mathbf{A}}$ and an angle $\theta_{\mathbf{A}}$, we can compute the angles between predicted values and actual ones. For the Fab data, the angle between $\vec{p}_{\mathbf{A}}$ and its prediction is on average of 6.5° and the one for $\theta_{\mathbf{A}}$ is 12°, low enough values to make the prediction almost overlap with ground truth. For the VHH data, those values are on average 15° and 25° respectively : predicting the VHH pose seems to be more challenging, which can be explained by the less conserved binding mode.

Finally we tried removing the rotation encoding that puts forward the $\vec{p}_{\mathbf{T}} = \vec{u}_z$ axis and replaced it by one that puts $\vec{u}_y$ instead, and retrained a model with this supervision. This model has an angle error of respectively 17° and 13°. We see that predicting $\vec{u}_z$ is easier than $\vec{u}_y$, which validates our assumption and formulation. We provide histograms for the distribution of these angles in Figure 2, to show that most systems are predicted accurately. We also provide a few examples of predicted systems in Figure S.4.

Case study As a case study for our tool, we take the system `8hij` (corresponding map `34818` has a resolution of 3.54Å). It is a well known example of interaction between Abs, with a VHH bound to a Fab bound to an antigen. We use the model trained on the `sorted` split which is supposedly the least successful one based on the above results. We run our algorithm using only the raw density, relying on the automatic thresholding and classifications. Results are presented in Figure 3.

Under two seconds on a CPU, we get a prediction close to the ground truth. We predict a Fab at the right epitope, in the right orientation. We also predict the VHH binding to the Fab with the right epitope. Notably, our training data set does not include any example of systems with both a VHH and a Fab. We believe this use case to be particularly eloquent.

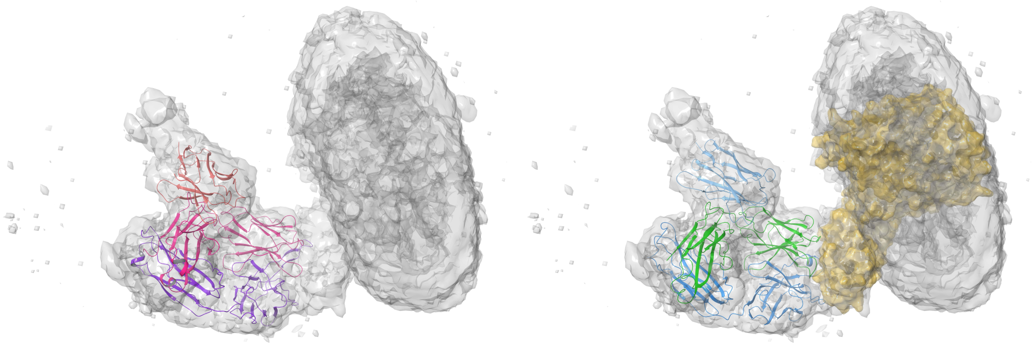


Figure 3: Prediction of our tool on the `8hij` system. Our prediction (left) and the experimental structure (right) are shown in the cryo-EM density. Antigen is displayed in orange, native antibodies in blue/green and predictions in purple/red. The VHH is displayed in lighter colors. We predict the correct positions and orientations for both the Fab and the VHH.

4 Conclusion, Future Work

In this paper, we addressed the problem of automatically finding the Abs in cryo-EM densities. This step currently constitutes a tedious, manual step for structural biologists, thus presenting significant opportunities and benefits for automation.

To achieve this goal, we proposed a customized solution, which exploits the structural properties and addresses the specific challenges of the problem, such as handling significant data scarcity and heterogeneity. Specifically, we leveraged the conserved structure of Fabs to cast this problem as an object detection one. We then gathered and cleaned a database to solve this problem with machine learning, designing a customized pose representation and loss based on optimal transport, which all help integrate prior information, while remaining efficient and flexible. Using our approach, no extra input is required to predict the number, position and pose of Fabs and VHHs in a density.

We validate our results on experimental maps and find the Ab positions with a hit rate of over 90%, which presents a 25% (resp 9%) improvement on Fabs (resp VHHs) over existing methods, while requiring no extra inputs and being thousands of times faster even when run on a single CPU. We show that the predicted pose correlates well with experiments, especially for Fabs, and illustrate our tool's performance on a particularly challenging `8hij` system.

We believe that this tool can reduce the burden on structural biologists working with cryo-EM densities of Abs and accelerate the resolution of 3D complexes of Abs bound to their antigen. While accurate and efficient, our approach has several limitations. Most importantly, it is currently limited to a single conserved family (Fabs and VHHs). Thus, it will be interesting to expanding our approach to other conserved families. More broadly, this approach could be expanded to dock several folded domains into a density, which has already been done without machine learning, [6, 26]. We could replace a fixed template by several embeddings of the folded domains. Finally, the throughput of our tool opens the door to finding Abs in the output of heterogeneous reconstruction or even continuous distribution of maps, and thus to capture several modes of antibody binding.

Acknowledgments and Disclosure of Funding

This research was supported by Sanofi. V. M. and M. O. are supported by DataIA, ANR AI Chair AIGRETTE and the ERC Starting Grant EXPROTEA. C.R. and H.M. are employees of Sanofi and may hold shares in the company.

References

[1] M. Arbabi Ghahroudi, A. Desmyter, L. Wyns, R. Hamers, and S. Muyldermans. Selection and identification of single domain antibody fragments from camel heavy-chain antibodies. *FEBS letters*, 414(3):521–526, 1997.

[2] T. Bepler, A. Morin, M. Rapp, J. Brasch, L. Shapiro, A. J. Noble, and B. Berger. TOPAZ: A positive-unlabeled convolutional neural network CryoEM particle picker that can pick any size and shape particle. *Microscopy and Microanalysis*, 25(S2):986–987, aug 2019. doi: 10.1017/s143192761900566x. URL `https://doi.org/10.1017%2Fs143192761900566x`.

[3] H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne. The protein data bank. *Nucleic acids research*, 28(1):235–242, 2000.

[4] T. L. Blundell and S. Patel. High-throughput x-ray crystallography for drug discovery. *Current Opinion in Pharmacology*, 4(5):490–496, oct 2004. doi: 10.1016/j.coph.2004.04.007. URL `https://doi.org/10.1016%2Fj.coph.2004.04.007`.

[5] J. Bouvette, Q. Huang, A. A. Riccio, W. C. Copeland, A. Bartesaghi, and M. J. Borgnia. Automated systematic evaluation of cryo-EM specimens with SmartScope. *eLife*, 11, aug 2022. doi: 10.7554/elife.80047. URL `https://doi.org/10.7554%2Felife.80047`.

[6] L. Chang, F. Wang, K. Connolly, H. Meng, Z. Su, V. Cvirkaite-Krupovic, M. Krupovic, E. H. Egelman, and D. Si. Deeptracer-id: De novo protein identification from cryo-em maps. *Biophysical Journal*, 121(15):2840–2848, 2022.

[7] M. L. Chiu, D. R. Goulet, A. Teplyakov, and G. L. Gilliland. Antibody structure and function: the basis for engineering therapeutics. *Antibodies*, 8(4):55, 2019.

[8] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger. 3d u-net: learning dense volumetric segmentation from sparse annotation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2016: 19th International Conference, Athens, Greece, October 17-21, 2016, Proceedings, Part II 19*, pages 424–432. Springer, 2016.

[9] T. Cohen, M. Halfon, and D. Schneidman-Duhovny. Nanonet: Rapid and accurate end-to-end nanobody modeling by deep learning. *Frontiers in immunology*, 13:958584, 2022.

[10] V. I. Cushing, A. F. Koh, J. Feng, K. Jurgaityte, A. K. Bahl, S. Ali, A. Kotecha, and B. J. Greber. High-resolution cryo-electron microscopy of the human cdk-activating kinase for structure-based drug design. *bioRxiv*, pages 2023–04, 2023.

[11] W. L. DeLano. The pymol molecular graphics system. *http://www. pymol. org/*, 2002.

[12] S. Duggan. Caplacizumab: first global approval. *Drugs*, 78:1639–1642, 2018.

[13] J. Dunbar, K. Krawczyk, J. Leem, T. Baker, A. Fuchs, G. Georges, J. Shi, and C. M. Deane. Sabdab: the structural antibody database. *Nucleic acids research*, 42(D1):D1140–D1146, 2014.

[14] C. Emmons and L. Hunsicker. Muromonab-cd3 (orthoclone okt3): the first monoclonal antibody approved for therapeutic use. *Iowa Medicine: Journal of the Iowa Medical Society*, 77(2): 78–82, 1987.

[15] P. Emsley, B. Lohkamp, W. G. Scott, and K. Cowtan. Features and development of coot. *Acta Crystallographica Section D: Biological Crystallography*, 66(4):486–501, 2010.

[16] Q. Fan, Y. Li, Y. Yao, J. Cohn, S. Liu, S. M. Vos, and M. A. Cianfrocco. Cryorl: Reinforcement learning enables efficient cryo-em data collection. *arXiv preprint arXiv:2204.07543*, 2022.

[17] J. Feydy, T. Séjourné, F.-X. Vialard, S.-i. Amari, A. Trouve, and G. Peyré. Interpolating between optimal transport and mmd using sinkhorn divergences. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 2681–2690, 2019.

[18] K. Jamali, L. Kall, R. Zhang, A. Brown, D. Kimanius, and S. Scheres. Automated model building and protein identification in cryo-em maps. *bioRxiv*, pages 2023–05, 2023.

[19] P. T. Jones, P. H. Dear, J. Foote, M. S. Neuberger, and G. Winter. Replacing the complementarity-determining regions in a human antibody with those from a mouse. *Nature*, 321(6069):522–525, 1986.

[20] I. Jovčevska and S. Muyldermans. The therapeutic potential of nanobodies. *BioDrugs*, 34(1): 11–26, 2020.

[21] J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021.

[22] H. Kaplon, S. Crescioli, A. Chenoweth, J. Visweswaraiah, and J. M. Reichert. Antibodies to watch in 2023. In *MAbs*, volume 15, page 2153410. Taylor & Francis, 2023.

[23] D. Kimanius, L. Dong, G. Sharov, T. Nakane, and S. H. W. Scheres. New tools for automated cryo-EM single-particle analysis in RELION-4.0. *Biochemical Journal*, 478(24):4169–4185, dec 2021. doi: 10.1042/bcj20210708. URL `https://doi.org/10.1042%2Fbcj20210708`.

[24] G. Köhler and C. Milstein. Continuous cultures of fused cells secreting antibody of predefined specificity. *Nature*, 256(5517):495–497, 1975.

[25] H. W. Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

[26] D. Liebschner, P. V. Afonine, M. L. Baker, G. Bunkóczi, V. B. Chen, T. I. Croll, B. Hintze, L.-W. Hung, S. Jain, A. J. McCoy, et al. Macromolecular structure determination using x-rays, neutrons and electrons: recent developments in phenix. *Acta Crystallographica Section D: Structural Biology*, 75(10):861–877, 2019.

[27] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[28] R.-M. Lu, Y.-C. Hwang, I.-J. Liu, C.-C. Lee, H.-Z. Tsai, H.-J. Li, and H.-C. Wu. Development of therapeutic antibodies for the treatment of diseases. *Journal of biomedical science*, 27(1): 1–30, 2020.

[29] S. Matsumoto, S. Ishida, M. Araki, T. Kato, K. Terayama, and Y. Okuno. Extraction of protein dynamics information from cryo-EM maps using deep learning. *Nature Machine Intelligence*, 3(2):153–160, feb 2021. doi: 10.1038/s42256-020-00290-y. URL `https://doi.org/10.1038%2Fs42256-020-00290-y`.

[30] J. McCafferty, A. D. Griffiths, G. Winter, and D. J. Chiswell. Phage antibodies: filamentous phage displaying antibody variable domains. *Nature*, 348(6301):552–554, 1990.

[31] M. Peplow. Cryo-electron microscopy makes waves in pharma labs. *Nature Reviews Drug Discovery*, 16(12):815–818, 2017.

[32] E. F. Pettersen, T. D. Goddard, C. C. Huang, G. S. Couch, D. M. Greenblatt, E. C. Meng, and T. E. Ferrin. UCSF chimera?a visualization system for exploratory research and analysis. *Journal of Computational Chemistry*, 25(13):1605–1612, 2004. doi: 10.1002/jcc.20084. URL `https://doi.org/10.1002%2Fjcc.20084`.

[33] G. Peyré, M. Cuturi, et al. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019.

[34] J. Pfab, N. M. Phan, and D. Si. Deeptracer for fast de novo cryo-em protein structure modeling and special studies on cov-related complexes. *Proceedings of the National Academy of Sciences*, 118(2):e2017525118, 2021.

[35] A. Punjani, J. L. Rubinstein, D. J. Fleet, and M. A. Brubaker. cryosparc: algorithms for rapid unsupervised cryo-em structure determination. *Nature methods*, 14(3):290–296, 2017.

[36] C. R. Qi, O. Litany, K. He, and L. J. Guibas. Deep hough voting for 3d object detection in point clouds. In *proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9277–9286, 2019.

[37] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[38] R. Sanchez-Garcia, J. Segura, D. Maluenda, C. Sorzano, and J. Carazo. MicrographCleaner: A python package for cryo-EM micrograph cleaning using deep learning. *Journal of Structural Biology*, 210(3):107498, jun 2020. doi: 10.1016/j.jsb.2020.107498. URL `https://doi.org/10.1016%2Fj.jsb.2020.107498`.

[39] S. H. Scheres. Relion: implementation of a bayesian approach to cryo-em structure determination. *Journal of structural biology*, 180(3):519–530, 2012.

[40] C. Schneider, M. I. Raybould, and C. M. Deane. Sabdab in the age of biotherapeutics: updates including sabdab-nano, the nanobody structure tracker. *Nucleic acids research*, 50(D1):D1368–D1372, 2022.

[41] G. P. Smith. Filamentous fusion phage: novel expression vectors that display cloned antigens on the virion surface. *Science*, 228(4705):1315–1317, 1985.

[42] TheAntibodySociety. The antibody society. therapeutic monoclonal antibodies approved or in review in the eu or us. 8/23; www.antibodysociety.org/resources/approved-antibodies., 2023.

[43] C. Tralie, N. Saul, and R. Bar-On. Ripser.py: A lean persistent homology library for python. *The Journal of Open Source Software*, 3(29):925, Sep 2018. doi: 10.21105/joss.00925. URL `https://doi.org/10.21105/joss.00925`.

[44] T. Wagner, F. Merino, M. Stabrin, T. Moriya, C. Antoni, A. Apelbaum, P. Hagel, O. Sitsel, T. Raisch, D. Prumbaum, D. Quentin, D. Roderer, S. Tacke, B. Siebolds, E. Schubert, T. R. Shaikh, P. Lill, C. Gatsogiannis, and S. Raunser. SPHIRE-crYOLO is a fast and accurate fully automated particle picker for cryo-EM. *Communications Biology*, 2(1), jun 2019. doi: 10.1038/s42003-019-0437-z. URL `https://doi.org/10.1038%2Fs42003-019-0437-z`.

[45] F. Wang, H. Gong, G. Liu, M. Li, C. Yan, T. Xia, X. Li, and J. Zeng. DeepPicker: A deep learning approach for fully automated particle picking in cryo-EM. *Journal of Structural Biology*, 195(3):325–336, sep 2016. doi: 10.1016/j.jsb.2016.07.006. URL `https://doi.org/10.1016%2Fj.jsb.2016.07.006`.

[46] R. Yao, J. Qian, and Q. Huang. Deep-learning with synthetic data enables automated picking of cryo-EM particle images of biological macromolecules. *Bioinformatics*, 36(4):1252–1259, oct 2019. doi: 10.1093/bioinformatics/btz728. URL `https://doi.org/10.1093%2Fbioinformatics%2Fbtz728`.

[47] J. Zhang, Z. Wang, Y. Chen, R. Han, Z. Liu, F. Sun, and F. Zhang. PIXER: an automated particle-selection method based on segmentation using a deep neural network. *BMC Bioinformatics*, 20(1), jan 2019. doi: 10.1186/s12859-019-2614-y. URL `https://doi.org/10.1186%2Fs12859-019-2614-y`.

[48] E. D. Zhong, T. Bepler, B. Berger, and J. H. Davis. CryoDRGN: reconstruction of heterogeneous cryo-EM structures using neural networks. *Nature Methods*, 18(2):176–185, feb 2021. doi: 10.1038/s41592-020-01049-4. URL `https://doi.org/10.1038%2Fs41592-020-01049-4`.

[49] Z. Zou, K. Chen, Z. Shi, Y. Guo, and J. Ye. Object detection in 20 years: A survey. *Proceedings of the IEEE*, 2023.

A Methods

A.1 Proof of the validity of the decomposition

Following our notations, we are looking for $R^*_{\mathbf{A}}$ that aligns $\mathbf{A}$ onto $\mathbf{T}$. We want to show that $R^*_{\mathbf{A}}$ can be parametrized by the transformation of $p_{\mathbf{T}}$ into $p_{\tilde{\mathbf{A}}}$ along with a rotation around $p_{\mathbf{T}}$.

By denoting $\tilde{\mathbf{A}} = \mathbf{A} - T^*_{\mathbf{A}}$, we have by construction : $\tilde{\mathbf{A}} = R^*_{\mathbf{A}}\mathbf{T}$, hence, $p_{\tilde{\mathbf{A}}} = R^*_{\mathbf{A}} p_{\mathbf{T}}$.

Let M be a rotation matrix of axis $p_{\tilde{\mathbf{A}}} \times p_{\mathbf{T}}$ and of angle ρ the angle between these vectors, so that it transforms one into the other. We can write :

$$p_{\tilde{\mathbf{A}}} = M p_{\mathbf{T}} \tag{7}$$

$$M^{-1} R^*_{\mathbf{A}} p_{\mathbf{T}} = p_{\mathbf{T}}. \tag{8}$$

Therefore, $M^{-1}R^*_{\mathbf{A}}$ is a rotation around the axis $p_{\mathbf{T}}$ by some angle θ, and we can write :

$$R^*_{\mathbf{A}} = M Rot_{p_{\mathbf{T}},\theta}. \tag{9}$$

A.2 Detailed architecture

We used a 3D Unet architecture. First let us define a `conv_block`(i,o) as three series of convolutions (with kernel size three and stride one), batch normalization and PreLu activations. The first convolution goes from i to o channels and the two others have o as a fixed number of channels.

In the encoding stage, we use a series of four `conv_blocks` of successive dimensions (1, 32, 64, 128, 256), alternated by a max pooling of kernel size 2 and stride 2.

In the decoding stage, we apply a transpose convolution (256, 256) with kernel size of 3 and stride 2 to the output of the encoder. We concatenate the result with the output from the third encoding `conv_block` to get 384 channels. We then run a `conv_block`(384, 128) on this concatenation.

Finally, we apply a last `conv_block`(128, 128) with only two series on the resulting tensor and replace the third one by a linear layer going from dimension 128 to dimension 10. The classification output dimensions are followed by sigmoid activation.

A.3 Modifications to the SabDab database

We removed a few non canonical systems, such as fusion proteins as well as the obsolete 7ny5. We then fixed edgy situations, often encountered when VHHs and Fabs are present in the system.

- When querying the Fabs, some chains were wrongly considered separate : `7YVN chain HI 7YVP chains IJ, 8HHX chains HI, 8HHY chains FW-GI, 8DXS chains FI-GJ, 7XOD chains RS-UV-XY, 7MLV chains KF, 6XJA (missing chain AB and missing antigen)`
- VHHs were present when querying Fabs. They were annotated as having just one chain and no antigen partner : `7zlg-h-i-j chain K bound to L, 8hii-j-k chain N bound to L, 7xw6 chain N bound to AB, 7sk7 chain K bound to C, 7sk5 chain E bound to C, 7zyi chain K bound to L, 7xod chain T-W-Z bound to S-V-Y, 6ni2 chain A bound to B-V, 7wpe chain W-Z bound to V-Y, 6ww2 chain K bound to L, 7jhh chain N bound to L, 7ul3 chain C bound to A, 7tuy chain K bound to L`
- Fabs were present when querying VHHs (often interacting with this VHH) : `7pij chain HL bound to N, 7wpd chain XY bound to A, 7wpf chain RS, UV, XY, 7wpf chain AC-HL bound to EN, 7php chain HL bound to N, 7m74 chain HL bound to A`
- VHHs with no antigens were present in the VHH file (actually always bound to a Fab) : `7pij K bound to L, 7wpd Z bound to Y, 7nkr F bound to A, 7wpf T-W-Z bound to S-V-Y, 7wpf chain K-F bound to L-C, 7php-7m74-7jhg chain K, N, N bound to L, 7nis-7nk1-7nk2-7nk6-7nkc-7nj4-7nka chain F bound to A/B-A-A/B-A/B-B/C-A-A`

A.4 Finding the optimal threshold

We present the error in the number of Abs detected for different thresholds in Figure S.1. Throughout datasets, persistence diagrams seem to give a better estimate. The overall best value is achieved for a threshold of approximately 0.2 and is not very sensitive to this threshold.

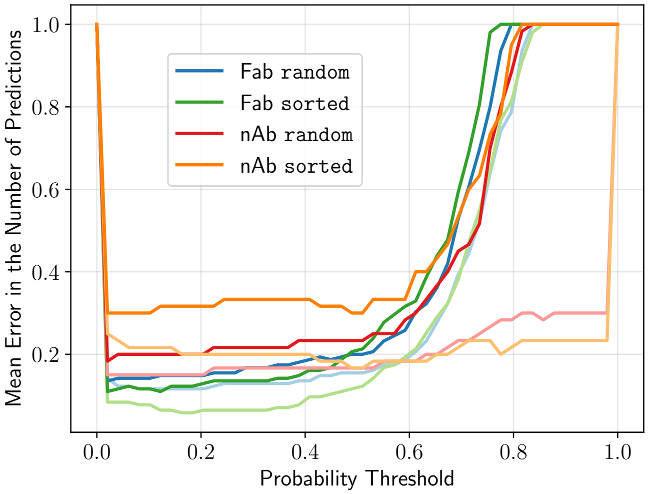


Figure S.1: Finding the best threshold for selecting the right number of system. x-axis represents the threshold value. y-axis represents the mean absolute error in the number of predicted systems over the data set. Different colors represent different data sets. Darker lines represent a naïve approach for NMS while the lighter ones are computed using persistence diagrams.

B Results

B.1 Fab detection performance

We compute the distances from prediction to ground truth obtained from our tool and `dock_in_map`. For a nicer visualization, we cap all distances to 20Å. We also count failed systems and errors as 20Å predictions so that the bar at 20Å represents all failures of a tool. We show the result in Figure S.2

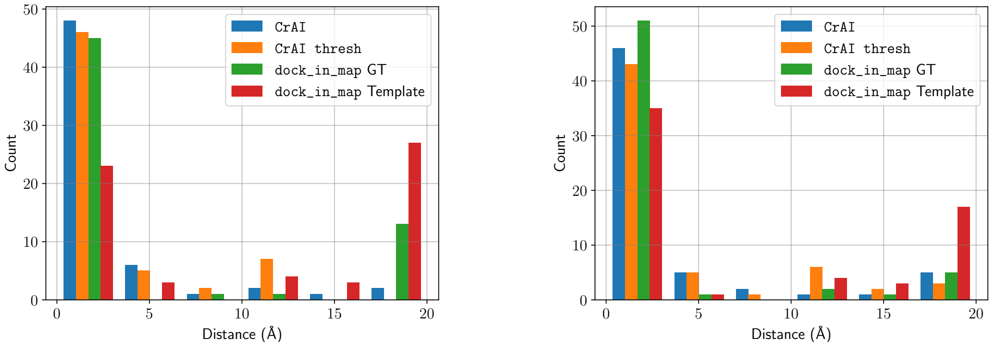


Figure S.2: Capped histogram of predicted distances on Fabs. `random` is on the left and `sorted` on the right.

As can be seen, our tool widely outperforms `dock_in_map`. Under the hit rate metric, we have the following performances (by order of tools) : `random` : 94.1, 88.8, 62.2, 19.4 `sorted` : 97.3, 93.9, 71.1, 8.5.

B.2 Dependence in resolution

We concatenate the predictions for all our systems and scatter them as a function of the resolution of the maps. We cap the predicted distances to 10Å and add a best fit curve. We show the result in Figure S.3. We see a moderate dependency in resolution, and many successful ones for resolutions above 4

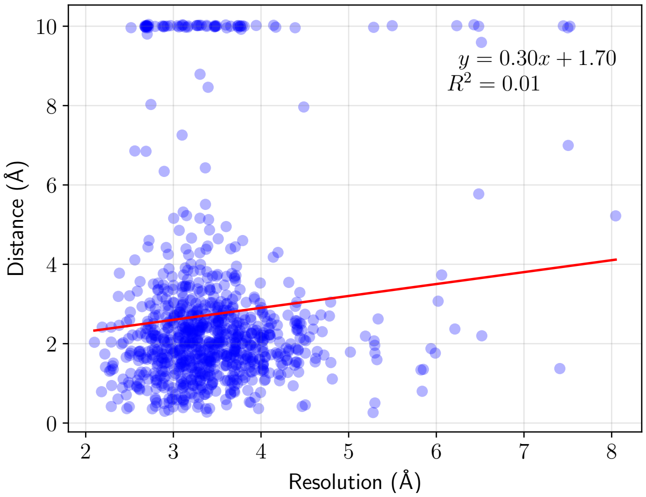
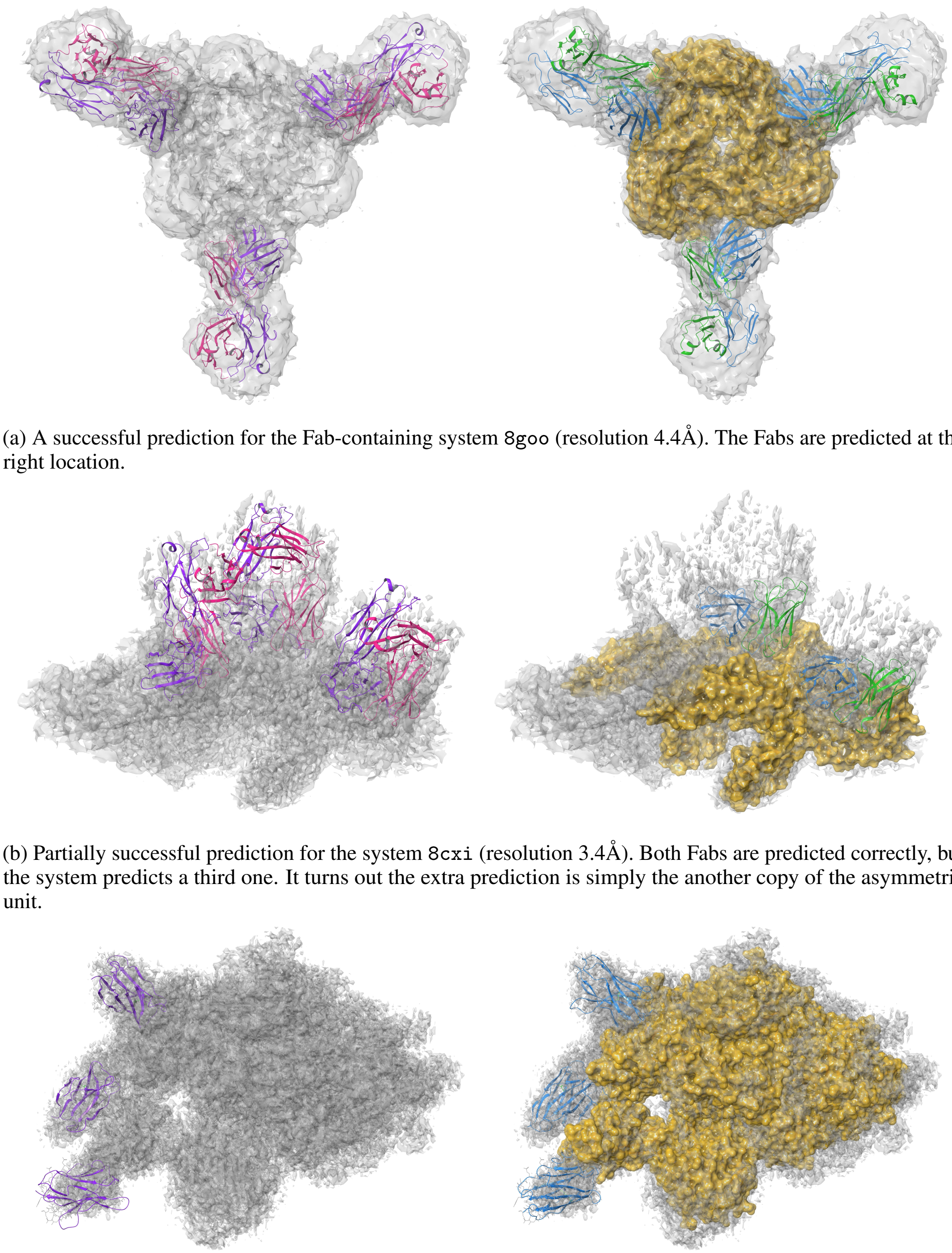


Figure S.3: x-axis is the resolution of the maps. y-axis is the distance between a prediction and its corresponding system, capped to 10Å for failed systems.

B.3 Prediction of the poses

We compute the angles between the actual and predicted poses for successful hits (below 10Å). We then plot a histogram of the results for three settings : Fabs, VHHs and the model trained on $\vec{u}_y$. We show the results in Figure S.3.



(a) A successful prediction for the Fab-containing system `8goo` (resolution 4.4Å). The Fabs are predicted at the right location.

(b) Partially successful prediction for the system `8cxi` (resolution 3.4Å). Both Fabs are predicted correctly, but the system predicts a third one. It turns out the extra prediction is simply the another copy of the asymmetric unit.

(c) A successful prediction for the systems `7z85` (resolution 3.1Å). All three VHHs are predicted correctly.

Figure S.4: A few prediction examples. Our prediction (left) and the experimental structure (right) are shown in the cryo-EM density. Antigens are displayed in orange, native antibodies in blue/green and predictions in purple/red.