Antibodies are proteins generated by the adaptive immune system to recognize and counteract a plethora of pathogens through specific binding. This adaptive binding is mediated by structural diversity in the six complementary determining region (CDR) loops (H1, H2, H3, L1, L2 and L3), which also makes accurate structural modeling of CDRs challenging. Both homology and de novo modeling approaches have been used; to date, the former has achieved greater accuracy for the non-H3 loops. The better performance of homology modeling in non-H3 CDRs is due to the fact that most of the non-H3 CDR loops of the same length and type can be grouped into a few structural clusters. Most antibody-modeling suites utilize homology modeling for the non-H3 CDRs, differing only in the alignment algorithm and how/if they utilize structural clusters. While RosettaAntibody and SAbPred do not explicitly assign query CDR sequences to clusters, two other approaches, PIGS and Kotai Antibody Builder, utilize sequence-based rules to assign CDR sequences to clusters. While the manually curated sequence rules can identify better structural templates, because their curation requires extensive literature search and human effort, they lag behind the deposition of new antibody structures and are infrequently updated. In this study, we propose a machine learning approach (Gradient Boosting Machine [GBM]) to learn the structural clusters of non-H3 CDRs from sequence alone. We argue the GBM method gives simplicity in feature selection and immediate integration of new data compared to manual sequence rules curation. We compare the classification results using the GBM method to that of RosettaAntibody in a 3-repeat 10-fold cross-validation scheme on the cluster-annotated antibody database PyIgClassify and we observe an improvement in the classification accuracy from $78.8\pm0.2\%$ to $85.1\pm0.2\%$. We find the GBM models can reduce the errors in specific cluster membership misclassifications if the involved clusters have relatively abundant data. Based on the factors identified, we suggest methods that can enrich structural classes with sparse data can possibly further improve prediction.
accuracy in future studies.
Non-H3 CDR template selection in antibody modeling through machine learning

Xiyao Long¹, Jeliazko R. Jeliazkov², Jeffrey J. Gray¹,²,³,⁴,※

¹Department of Chemical and Biomolecular Engineering, Johns Hopkins University, Baltimore, Maryland, USA
²Program in Molecular Biophysics, Johns Hopkins University, Baltimore, Maryland, USA
³Institute for NanoBioTechnology, Johns Hopkins University, Baltimore, Maryland, USA
⁴Sidney Kimmel Comprehensive Cancer Center, Johns Hopkins University, Baltimore, Maryland, USA

※Correspondence: Jeffrey J. Gray (jgray@jhu.edu)

Keywords: complementarity determining regions (CDRs), RosettaAntibody
Abstract

Antibodies are proteins generated by the adaptive immune system to recognize and counteract a plethora of pathogens through specific binding. This adaptive binding is mediated by structural diversity in the six complementary determining region (CDR) loops (H1, H2, H3, L1, L2 and L3), which also makes accurate structural modeling of CDRs challenging. Both homology and de novo modeling approaches have been used; to date, the former has achieved greater accuracy for the non-H3 loops. The better performance of homology modeling in non-H3 CDRs is due to the fact that most of the non-H3 CDR loops of the same length and type can be grouped into a few structural clusters. Most antibody-modeling suites utilize homology modeling for the non-H3 CDRs, differing only in the alignment algorithm and how/if they utilize structural clusters. While RosettaAntibody and SAbPred do not explicitly assign query CDR sequences to clusters, two other approaches, PIGS and Kotai Antibody Builder, utilize sequence-based rules to assign CDR sequences to clusters. While the manually curated sequence rules can identify better structural templates, because their curation requires extensive literature search and human effort, they lag behind the deposition of new antibody structures and are infrequently updated. In this study, we propose a machine learning approach (Gradient Boosting Machine [GBM]) to learn the structural clusters of non-H3 CDRs from sequence alone. We argue the GBM method gives simplicity in feature selection and immediate integration of new data compared to manual sequence rules curation. We compare the classification results using the GBM method to that of RosettaAntibody in a 3-repeat 10-fold cross-validation scheme on the cluster-annotated antibody database PyIgClassify and we observe an improvement in the classification accuracy from 78.8±0.2% to 85.1±0.2%. We find the GBM models can reduce the errors in specific cluster membership misclassifications if the involved clusters have relatively abundant data. Based on the factors identified, we suggest methods that can enrich structural classes with sparse data can possibly further improve prediction accuracy in future studies.
1 Introduction

Antibodies are central to adaptive immunity. They are responsible for recognizing a variety of target molecules known as antigens. They acquire the ability to recognize any one of a diverse set of targets through two biological mechanisms: V(D)J recombination and affinity maturation. These gene-editing mechanisms can produce an enormous quantity of unique sequences, in theory on the order of $10^{13}$ (Georgiou et al., 2014; DeKosky et al., 2016), though the antibody repertoire of any single individual is comprised of only a fraction of the possible sequences. Recent advances in high-throughput sequencing techniques are permitting unparalleled access to the human antibody repertoire, thus furthering our comprehension of immune response to vaccination, infection, and autoimmunity. Beyond sequence data, structural information can provide additional insights about the functions of antibodies. Yet, only a very small fraction of antibodies have solved crystal structures in the Protein DataBank, reported as 3087 structures (Dunbar et al., 2014) with a filtered set of 1,940 PDB antibody entries included in PyIgClassify as of August, 2017 (Adolf-Bryfogle et al., 2015). Most of these structures are murine (51.15%) and human (35.51%), while repertoire sequencing is rapidly expanding our knowledge of other species. It would be challenging and time-consuming to close the gap between structure and sequence knowledge through experimental structure determination methods. Computational modeling provides a feasible alternative. For example, in chronic lymphocytic leukemia, models of antibody structures added prognostic value over sequence data alone (Marcatili et al., 2013). Besides using modeling to develop biological understanding, docking studies of antibodies complexed with various antigens can reveal atomic details of antibody–antigen interactions (Kuroda et al., 2012b; Kilambi & Gray, 2017; Koivuniemi, Takkinen & Nevanen, 2017; Weitzner et al., 2017). Finally, in antibody design studies, computational approaches can enhance affinity or design an antibody de novo—without prior
sequence information (Lippow, Wittup & Tidor, 2007; Kuroda et al., 2012a; Dunbar et al., 2016; Baran et al., 2017; Adolf-Bryfogle et al., 2018). To be useful, however, computational methods must be able to accurately predict antibody structure.

Typical approaches to antibody structure prediction decompose the problem into three parts based on known antibody-structural features (Almagro et al., 2014). Antibodies are typically comprised of a light and heavy chain, both having variable (V) and constant (C) regions (Fig. 1A). While the constant region is important for signaling, it does not vary across antibodies and does not greatly affect the antigen-binding function. On the other hand, the variable region can differ between antibodies and is responsible for recognizing antigens. The variable region can be further divided into a framework region (FR), with greek-key β-barrel topology, and six complementarity-determining regions (CDRs), which are solvent-exposed loops connecting the β-strands comprising the aforementioned β-barrel (Fig 1B–C). The FR is conserved and has a low rate of mutation across antibodies, whereas the CDRs, and in particular the CDR H3, are highly mutable in order to be able to bind a wide variety of antigens (Schroeder, Cavacini & Cavacini, 2010). Thus, the antibody modeling problem is often decomposed into (1) homology modeling of light and heavy FRs, (2) homology modeling of the non-H3 CDR loops, and (3) de novo modeling of the CDR-H3 loop.

Of these three modeling problems, modeling the CDR-H3 loop is the most challenging. For example, an average backbone RMSD of 2.8±0.4 Å was reported over eleven test antibodies and seven modeling approaches in a recent blind assessment (Almagro et al., 2014). By comparison, FR modeling was found to achieve sub-angstrom accuracy, on average, for both the light and heavy chains. The quality of the modeling of the non-H3 CDRs was uneven, with average backbone RMSDs ranging from 0.5±0.1 to 1.3±1.1 Å for RosettaAntibody models of targets in the same assessment (Weitzner et al., 2014). This result was surprising since previous studies have found that, when divided by loop type and length (e.g. H1-10), non-H3
CDRs can be structurally clustered and a majority (85%) of the loops assume structures similar to just a few loops structures, called the cluster examplars (North, Lehmann & Dunbrack, 2011). Whether antibody-modeling methods have been using this structural information effectively remains an open question.

In the four most popular methods SAbPred (Dunbar et al., 2016), PIGS (Marcatili et al., 2014), Kotai Antibody Builder (Yamashita et al., 2014) and RosettaAntibody (Weitzner et al., 2017), non-H3 CDR loops are generally modeled by homology: a CDR loop with a known structure is chosen as a template structure based on its sequence similarity to the query CDR loop. However, the use of additional structure-based rules, the scoring matrix used to determine sequence similarity, and the database of possible templates all vary among methods.

First, PIGS and Kotai Antibody Builder both use sequence-based rules to identify the structural cluster of the query CDR sequence. If a potential cluster or clusters can be identified, the methods constrain the template search to these clusters. While sequence rules are easy to interpret and can offer deterministic cluster assignments, they are limited in their adaptability and their power—as the number of known antibody structures and sequences grows, analysis by hand becomes more challenging. For example, the current PIGS method uses curated rules from a variety of previous studies (Marcatili et al., 2014). For the CDR H1 loop, it has 4 canonical clusters from 4 different loop lengths with sequence rules, but according to North et al. 2011 study there are now 17 structural clusters and 6 loop lengths for the CDR-H1 loop (North, Lehmann & Dunbrack, 2011). Another issue is that some clusters lack deterministic, human-identified rules. Kotai Antibody Builder (Yamashita et al., 2014) devised sequence rules for cluster identification in accordance with the clusters identified by North et al., but in that publication there are not clear sequence rules for distinguishing among H1 clusters. In fact, only a fraction of the remaining non-H3 CDR clusters (26/56) have sequence rules (Shirai et al., 2014) and, worryingly, not all sequence rules are comprehensive. For example, under the
Chothia numbering convention (Chothia et al., 1989), an arginine at position 71 in length 10 CDR-H2 loops can indicate membership to either the H2-10-1 or H2-10-2 cluster, but not all sequences belonging to the H2-10 cluster have that arginine: only 8 out of 155 CDRs in H2-10-1 and 38 out of 42 CDRs in H2-10-2 do. To address this problem and the problem of inadequate sequence-based rule coverage, Kotai Antibody Builder built position-specific-substitution-matrix (PSSM) profiles for each cluster, so that when sequence rules fail, PSSM-based scoring can be used to suggest a cluster (Shirai et al., 2014). When assessed using the PylgClassify antibody dataset, Kotai Antibody Builder correctly identified the cluster in 90% (Shirai et al., 2014) of all CDR loops, including the CDR H3. However, it is not clear whether the tested data was excluded from the construction of the PSSM profiles, so the reported accuracy might have been overestimated.

Recent assessments of antibody structural modeling report varying accuracy of non-H3 CDR modeling. When RosettaAntibody was benchmarked in a recent study on 54 antibody targets (Weitzner et al., 2014), non-H3 CDR loop modeling achieved sub-angstrom backbone RMSD between the homology-modeled and crystal-structure CDRs in 42/54 (L1), 50/54 (L2), 37/54 (L3), 36/54 (H1), and 42/54 (H2) cases. Meanwhile, using a set of 689 antibody structures and leave-one-out-cross-validation (LOOCV), PIGS (Marcatili et al., 2014) was found to capture ~50% of the modeled non-H3 CDRs with sub-angstrom backbone RMSDs to the native CDRs structures. Finally, SAbPred (Dunbar et al., 2016) was tested on the same set of 54 antibodies as RosettaAntibody and resulted in average backbone RMSDs as 1.09, 0.59, 1.00, 0.88 and 0.90 Å for 5 non-H3 CDRs (Choi & Deane, 2011). Despite mostly sub-angstrom average RMSDs for all methods and benchmarks, individual models with RMSDs much greater than an angstrom were not rare (Choi & Deane, 2011; Weitzner et al., 2014; Almagro et al., 2014), suggesting a need for special handling of these fail-prone cases. We propose introducing an extra step to non-H3 CDR modeling, where a machine learning approach is used to predict
cluster membership and template structures are only selected from the predicted cluster, improve accuracy by preventing templates coming from a structurally distinct cluster, typically with a large structural distance to the query CDR loop.

Machine learning has been used extensively in protein classification problems, for example it has accurately predicted protein function (Radivojac et al., 2013), folding rate (Corrales et al., 2015), super-family levels for fold recognition (Jain, Garibaldi & Hirst, 2009), enzyme classes (Kumar & Choudhary, 2012), and functional binding sites (Si, Zhao & Wu, 2015). Of many machine learning methods, Gradient Boosting Machine (GBM) was recently shown to yield the best accuracy for structural classification of proteins in the Structural Classification Of Protein database (SCOP) (Jain, Garibaldi & Hirst, 2009). We apply GBM to non-H3 CDR loop cluster prediction because it is a similar classification problem: amino acids serve as predictors for structural classes.

To assess the performance of GBM, we need to have a fair comparison with prior work. However, several problems arise when comparing CDR modeling accuracy between methods. First, each modeling method may use a different set of template antibodies, confounding comparison by introducing disparity in the template library size. For example, the smaller percentage of sub-angstrom accuracy models in PIGS than in RosettaAntibody could be partially due to the smaller template library in PIGS, featuring only 689 template immunoglobulins versus the 1915 templates used by RosettaAntibody. Second, when benchmarking, comparison across the studies is limited by the differing test sets. A small or incomplete data set can elide template selection problems that would otherwise be observed with a larger test set. Additionally, an unbalanced set containing loops more similar to the canonical cluster centers would be easier to model than one comprised solely of outliers. Third, these evaluations typically assess the quality of the final models rather than the structural
template, failing to compare the similarity of the template to the native structure as the final
structures are often subject to energy refinement (Weitzner et al., 2014).

In this work, we attempted to increase the quality of CDR structural template selection by
using the machine learning method GBM. For a relevant and fair assessment, we evaluated the
quality of template selection rather than that of the final model and the assessment was done on
the comprehensive dataset PylgClassify (Adolf-Bryfogle et al., 2015) for both the original
RosettaAntibody structural template identification method and the GBM method developed
herein. As the disparities of cluster member sizes can affect the performance of GBM (Sun et
al., 2007), we surveyed various techniques for overcoming the data imbalance problem.

Approaches vary, from down-sampling the majority class to up-sampling the minority classes, or
even adding synthetic members to balance the size of the clusters (Chawla et al., 2002; Blagus
et al., 2013). Previous results suggest that the best approach depends on the specific data set
and size (Kuhn & Johnson, 2013a). In our study, we utilized the up-sampling strategy to balance
the cluster sample sizes in the training data.

We show that (1) GBM can find better identify the query CDR’s structural cluster than
RosettaAntibody and (2) selecting structural templates from within the query cluster results in
lower RMSD templates than selecting outside the cluster. The GBM models also recapitulate
previously known sequence motifs and identify new ones. The GBM models find that the
presence or absence of a single residue on its own is not sufficient to assign a sequence to a
specific structural cluster. Rather the combination of residues in the query sequence is
important for assigning a probable cluster. These findings suggest that incorporating machine
learning methods may achieve closer-to-native templates selection during non-H3 CDR
homology modeling and realize an automated feature selection, surpassing the manual curation
of sequence rules.
2 Materials and Methods

Structural class prediction approaches

We employed two methods, blindBLAST and GBM, for CDR structural class prediction. For comparison, we constructed a random model that assigned structural classes to query CDRs based on their naturally occurring frequency. The blindBLAST approach comes from the current version of RosettaAntibody (Weitzner et al., 2017), which identifies template non-H3 CDR loops through a BLAST search against CDRs of the same length and type using the PAM30 matrix to rank sequence similarity. The BLAST parameters used are:

- substitution_matrix PAM30 -word_size 2 -max_target_seqs 3000 -evalue 2000

The template loop with the most sequence similarity to the query is then selected for grafting and further modeling. We refer to this approach as “blindBLAST”, as it does not utilize CDR structural cluster information but rather identifies the structural class of a CDR loop implicitly by choosing a template with the highest bitscore. On the other hand, we trained supervised GBMs for each combination of CDR loop and length type. Each model learns to predict the structural class (synonymous to the structural cluster) from the labelled CDR sequences, including the 10 flanking residues on either side. Sequences were vectorized by one-hot-encoding (Beck & Woolf, 2000): the observed amino acid is represented by a 1 and the other possible 19 amino acids are 0s. Thus, a CDR loop of length 10 is represented by a 30*20 matrix. To optimize gradient boosted tree model performance, we trained models with hyper-parameter tuning using 3-repeat 10-fold-stratified cross-validation. The parameter grid yielding the highest estimated model accuracy was used to train the final model. The cross-validation scheme was used instead of a single split between a training and a testing set because of the data sparsity in multiple structural classes. The fold-stratified division ensures that the composition of CDR clusters in every query set and in the training set resembles the entire dataset to minimize the
model variance (Kohavi, 1995). Among the 10 folds, a single fold comprises a query (validation) set and the remaining nine folds comprise a template (training) set. To counter the unbalanced sample problem, in each training set, cases belonging to each of the less popular classes were resampled to match the case number of the most popular class (Dittman et al.; Sun, Kamel & Wang, 2006). The 10-fold division is repeated three times followed by the same training protocol. The model complexity that gives the highest average accuracy in the validation sets is chosen as the best tuned model. All machine learning was performed by GBM method implemented in the Caret package (Kuhn, 2008).

Comparisons of different methods

We compared the performance of GBM, blindBLAST, and a null model in CDR structural class prediction on the non-redundant CDR loops in the PyIgClassify database (http://dunbrack2.fccc.edu/PyIgClassify/Download/Download.aspx). The structures and clusters were downloaded in February, 2017 by selecting the “CDRs and clusters of non-redundant sequences for a given CDR” database. The database contains antibody structures from the PDB with 2.8 Å or better resolution, 0.3 R-factor cutoff, and excludes non-proline cis loops or loops with highly improbable conformations (North, Lehmann & Dunbrack, 2011). The set of non-redundant canonical CDR loops was obtained from the database, in which the CDR loops are partitioned by their type and length. In PyIgClassify, CDR loop structures in each loop and length partition are clustered such that the members of each cluster are more structurally similar to the cluster exemplar than to the exemplar of any other cluster, with the exemplar as defined in (North, Lehmann & Dunbrack, 2011). The distribution of CDR cluster membership is unbalanced, with each CDR loop and length pair having one well-populated or dominant cluster and many sparsely populated or non-dominant clusters. In our study, clusters that have no more than three members were merged into a “none” cluster. The cluster member size distribution by CDR loop and length type is shown in Figure 1.
BlindBLAST was evaluated in the same data division scheme (3-repeat 10-fold cross-validation) as GBM. Each time, one fold was used as the query set and the remaining nine folds were used as the template set. For both blindBLAST and GBM, an error case was identified when the query cluster did not match to the predicted (template) cluster. The number of error cases and the corresponding accuracy were calculated for each loop and length type for each repeat and then averaged over the three repeats. To further analyze failures, we counted and compared the specific misclassifications (i.e. the number times a cluster A to cluster B misclassification occurred) for both GBM and blindBLAST.

Since raw error cases counts are confounded by the member size differences between structural classes, we compare results to a null model. In the null model, we assign cluster membership to each query CDR sequence by randomly matching it to any other CDR sequence of the same loop and length type. Such assignment is not deterministic, so we repeat this assignment process one thousand times, generating a distribution of random error counts (Supplementary Figure 1) for each misclassification. We then employ a significance test to determine which blindBLAST misclassifications significantly differ from random. The significance test is described in Equation 1 as a two-tailed hypothesis test at an 0.05 significance level. The null hypothesis is that blindBLAST did not classify more or less accurately than the random model. For a particular misclassification (e.g. H2-10-1 incorrectly classified as H2-10-4 is one misclassification), the test compares the average blindBLAST error count over 3 repeats of 10-fold cross validation ($\epsilon_{\text{blindBLAST}}^{X \rightarrow Y}$) to the error count ($\epsilon_{\text{random}}^{X \rightarrow Y}$) for each of the 1,000 random simulations. The significance value, $p$, is determined as the probability of observing an error more frequently by blindBLAST than at random:

$$p(\epsilon_{\text{blindBLAST}}^{X \rightarrow Y}) = \frac{\sum_{n=1}^{1000} I(\epsilon_{\text{blindBLAST}}^{X \rightarrow Y} < \epsilon_{\text{random}}^{X \rightarrow Y})}{1000},$$

(1)
where $I(\epsilon_{\text{blindBLAST}} \geq \epsilon_{\text{n, random}}) = \begin{cases} 1, & \text{if } \epsilon_{\text{blindBLAST}} \geq \epsilon_{\text{n, random}} \\ 0, & \text{if } \epsilon_{\text{blindBLAST}} < \epsilon_{\text{n, random}} \end{cases}$.

We reject the null hypothesis, if $p \leq 0.025$ (meaning blindBLAST misclassifies with significantly lower error than random) or if $p \geq 0.975$ (meaning blindBLAST misclassifies with significantly higher error than random).

Structure and sequence metrics

To better understand why misclassification may have occurred, we computed three structural metrics. First, for each loop and length type, we found the dihedral angle distance as defined in (North, Lehmann & Dunbrack, 2011) between the exemplars of the correct cluster and the cluster into which the loop was misclassified. Second, for each loop and length type, we calculated the dihedral angle distance of every query case to the exemplar of its corresponding cluster and compared the distance distributions for correctly and incorrectly classified cases. Third, we counted the number of structural neighbors in the cognate cluster for all CDR loops. A structural neighbor is defined to be any CDR loop with dihedral angle distance to the query less than $1/15$th of the radius of that cluster, where the radius is the largest dihedral distance between the cluster exemplar and any CDR loop in the cluster. We compared the distributions of the number of structural neighbors for the correctly and incorrectly classified cases.

In addition to investigating structural features, we extracted sequence features from the tuned GBM models based on the scaled feature importance. Absolute importance was calculated by determining how much a decision tree split reduces Gini impurity (Louppe et al., 2013) and then summing over all node-size-weighed reductions on splits corresponding to that feature over all boosting trees (Kuhn & Johnson, 2013b). The importance was then scaled to values from 0 to 100.
The code for the model generation and analysis can be found in

3 Results

BlindBLAST is not always more accurate compared to the null model

In blindBLAST, cluster assignment accuracies varied among the different CDR loop and
length types from below 50% to almost 100% according to 3-repeat 10-fold cross-validation, as
shown in Figure 2A. In most of the cases where the clusters of the query and the template CDR
did not match, a more near-native structural template could be found if the BLAST search was
restricted to within the query cluster (Figure 2B). This suggests that identification of the query
cluster could lead to selection of lower-RMSD templates.

To improve the accuracy with which we identify query sequences’ CDR clusters, we first
sought to understand why the accuracy of CDR cluster identification varies across loop lengths
and types. We found that accuracy is affected by (1) the number of clusters in each loop length
and type, (2) the number of loops populating each cluster, and (3) the total number of loops of a
given length and type (Supplementary Figure 3). First, we found that loops with a larger
number of clusters tended to have lower accuracy. For example, H1-13 has eight clusters and a
blindBLAST assignment accuracy of 78.6±0.4% whereas H2-9 has three clusters and an
accuracy of 91.2±0.5%. Second, we found that loops with uniform populations among clusters
had lower accuracy. For example, H2-10 and H1-13 both have eight clusters, so based on our
first observation we expected their accuracy to be similar. It is not: H2-10 has an accuracy of
73.3±0.1% whereas H1-13 has an accuracy of 78.6±0.4%. Analyzing the populations of the
clusters for each loop, we observed that clusters H2-10-1 and H2-10-2 have a similar number of
CDRs whereas clusters H1-13-1 have many more CDRs than any other H1-13 cluster (Figure
1). Third, accuracy can be limited by sparse data. We have observed lower accuracies for loops
with a small number of structures for a length and type. This is exemplified by H1-14, L1-12, L3-8, L3-10 having the worst accuracies among all loops: 45.1±0.1%, 72.9±4.8%, 65.4±3.7%, 62.0±2.8% (with a total number of 30, 43, 62, 72 loops), respectively.

In addition to understanding how the accuracy of cluster assignment varies among loops, we are also interested in the types of misclassifications; a misclassification from cluster X to cluster Y is inherently different from misclassifying cluster X to cluster Z. From the significance testing for blindBLAST against the random model, we identified distinctive performances of different misclassifications, categorized by whether the difference is significant or not, if blindBLAST performs better or worse than random, and the number of times that misclassification error is observed. **Table 1** describes the four categories and **Figure 3** shows them visually. **Supplementary Table 2** lists all the misclassifications.

BlindBLAST classifications with good performance were identified as those with significantly fewer error counts than random assignment and a raw error count under 3. For these cases, we computed the dihedral-angle distances between the correct and incorrect cluster exemplars. We found that dihedral-angle distance was greater or equal to 20 degrees except for the H1-13-1 to H1-13-7 misclassification pair, indicating that blindBLAST performs well when the cluster exemplars are distant in dihedral space. This is further evidenced by the observation of larger dihedral distances between cluster pairs in this category than the other misclassification categories, except for the H1-13-5 to H1-13-1 misclassification (**Table 2**).

Based on these results, we decided to further study how the “position” of the query CDR in its cluster affects misclassification. We quantified query position with two metrics: (1) the dihedral distance of the query CDR to its cluster exemplar and (2) the number of structural neighbors to the query CDR. The distributions of query–exemplar dihedral-angle distances are illustrated in **Figure 4** and suggest that query CDRs that are more distant from their corresponding cluster exemplars are more likely to be misclassified by blindBLAST. The
distributions of structural neighbor counts are illustrated in Figure 5 and suggest that for some well populated clusters, such as H1-13-1, H2-9-1 and L3-9-cis7-1, CDRs with fewer neighbors in the same cluster are more likely to be misclassified. Taken together, these data indicate that query CDRs that are located centrally within their cluster — those having a small dihedral distance from the cluster exemplar and many neighbors — are more accurately classified.

**GBM can improve cluster identification accuracy over blindBLAST**

Compared to blindBLAST, GBM models improve average query cluster identification accuracy from 78.8±0.2% to 85.1±0.2% (Figure 6 and Supplementary Figure 2). The increased overall accuracy of the GBM models is significant, with improvements in accuracy for each individual loop and length types. The difference between GBM and blindBLAST accuracy is greater than the standard deviation calculated across each repeat of the 3 repeats in the 10-fold cross-validation scheme. In fact, variance arises from the sparsity of data and not the GBM models as evidenced by its persistence in the blindBLAST set (Supplementary Figure 3D).

Next, to determine where the improvement in GBM accuracy is achieved, we decomposed the overall error count into changes in individual misclassification counts (Figure 7) and compared potential sequence rules to key features extracted from the GBM models (Figures 8–9).

Decomposing the error counts into their constituent misclassifications provided a few insights into how GBM models outperform the blindBLAST method. For H2-10 loops, the GBM model improved H2-10-2 → H2-10-X misclassifications over blindBLAST. For example the H2-10-2 → H2-10-1 error count was reduced from 14 to 8 (Fig 7A). Correspondingly, H2-10-X → H2-10-2 misclassifications increased. For example the H2-10-6 → H2-10-2 error count increased from 4 to 7. Taken together, these observations indicate that blindBLAST was failing to properly classify CDR loops as H2-10-2. Similarly, we examined L1-11 loops, which are similar to H2-10 in that the second cluster is well populated (Figure 1). Yet, L1-11 loop...
classification improvements came from fewer L1-11-1 loops being misclassified as L1-11-2, rather than fewer L1-11-2 loops being misclassified as L1-11-X. Finally, we investigated improvements for L3-9 and L3-10 loops, both loops that occasionally contain cis peptide bonds (Fig 7B). For both loops, we observed that the most drastic improvements came from cases where blindBLAST was incorrectly assigning loops, some without prolines, to cis clusters. To BLAST, the penalty of misaligning a proline at the cis position is more or less equal to the penalty of misaligning any other position, but such parity is not required by a GBM model, which can assign greater importance to having a proline in the cis position.

The proline observation raised the question: were there any other sequence features missed by BLAST, but identified by GBM models? To this end, we constructed sequence logos (Crooks et al., 2004) of select CDR loop clusters (Figure 8) and compared them to residue features deemed important in our GBM models (Figure 9). For L3-9, we observed proline at position 7 in the loop to be key in both cis clusters and the most important feature for our GBM model. The GBM model additionally identified a key threonine residue at position 6, which is never present in L3-9-1. Results are similar for L3-10, with proline residues at cis positions being by far the most important. However for H2-10, there are no dominant sequence features in either the sequences or the important GBM features, suggesting that the accuracy improvements arise not from a single residue presence or absence but from the combination of many features with varying weights in the training process.

4 Discussion

In the current implementation of RosettaAntibody, blindBLAST is used to select templates for the non-H3 CDR loops, so we are interested in investigating and improving cases where blindBLAST identifies templates with high RMSD. We found that one way to improve template RMSD is to take advantage of the fact that non-H3 CDR loops cluster and to search for templates within the query cluster. BlindBLAST does not consider cluster information explicitly,
instead it selects templates of the same length and loop type based on sequence similarity alone. When we tested the ability of blindBLAST to identify clusters compared to a random model, we found multiple and diverse sources of error in cluster classification, so we turned to machine learning, and in particular a GBM model, to improve classification accuracy.

When comparing blindBLAST to a random model, we observed various potential sources, but found no single cause, for inaccurate cluster assignment. First, some loops with few structures, such as H1-14, H1-15, L2-12, and L3-11, were more difficult to classify than loops with many structures. These loops accounted for 4 of the 15 misclassifications for which blindBLAST performs worse than the random model (Supplementary Table 2). Second, other loops with many potential and unbalanced clusters (i.e. where most loops belong to one cluster and few loops belong to the remaining clusters) clusters, such as H1-13, resulted in low accuracy. For these loops, we speculate that error arises when the query originates from a sparsely populated cluster but the template is searched for in all the clusters; there are many more potential templates from the most populated cluster than the other clusters and BLAST alignments of short sequences do not differ greatly (Supplementary Figure 4), so it is likely to find a more similar sequence from the most populated cluster than the other clusters. Indeed, we observe misclassifications from sparsely populated clusters to the most populated cluster frequently (6/15 times, Supplementary Table 2) when blindBLAST performs worse than random. Third, low accuracy was observed when there were two clusters with approximately equal membership. H2-10 loops, which have two highly populated clusters (1 and 2), are such an example and account for 2 of the 15 misclassifications when blindBLAST performs worse than random. Additionally, blindBLAST misclassifies loops where clusters have small dihedral-angle distance between exemplars, such as between L2-8-1 and L2-8-2 (4.5) and between L2-8-1 and L2-8-4 (8.8). Furthermore, in many loop and length types, queries lying at a greater
dihedral-angle distance to the cluster exemplar and with a smaller number of structural neighbors were found to have a greater chance to be misclassified.

With no single clear source for blindBLAST misclassifications, we turned to GBM models to improve classification accuracy. As shown in Figure 7, GBM can better distinguish some cluster pairs with even relatively small amounts of structural data. For example, misclassifications from L3-9-2 to L3-9-cis7-1, from H2-10-1 to H2-10-6, and from L2-8-1 to L2-8-4 have reduced error counts despite small dihedral-angle distances between their cluster exemplars (6.9, 6.8 and 8.8, respectively). However, better performance was not observable for misclassifications involving clusters L2-8-1 and L2-8-2 with only 4.45 dihedral-angle distance between exemplars.

For clusters with relatively large between-clusters-dihedral-angle-distance, GBM models may still not offer any improvement, such as the misclassification between cluster pairs H1-13-4 & H1-13-1 or H1-13-6 & H1-13-1 with dihedral-angle distances of 17 and 23 between their exemplars, respectively. Having the lack of improvements for such cases in mind, along with the fact that most misclassifications with reduced error counts with GBM models involve clusters that have relatively abundant sample number, we propose that the abundance of data in the non-dominant clusters of the cluster pairs affects how effectively GBM models can improve the blindBLAST performance.

Overall, our results suggest that relative to blindBLAST, GBM is able to better capture features and assign more sensible feature importance with only limited data. GBM models test results have reduced error count (>3) in 9 out of 15 listed blindBLAST “worse than random” misclassifications, in 14 out of 33 listed blindBLAST “random-like” misclassifications, and 12 out of 21 listed blindBLAST “better than random (high error)” misclassifications.
5 Conclusions

In summary, our study has demonstrated that a CDR template from the corresponding structural cluster generally has lower RMSD than a template from the wrong cluster. We have examined the ability of blindBLAST, which is the method used by RosettaAntibody, to identify non-H3 CDR loop clusters implicitly. We trained a GBM model for each CDR loop and length type, and cumulatively improved the canonical structural cluster identification accuracy from 78.8±0.2% test accuracy using the blindBLAST approach in RosettaAntibody to 85.1±0.2% test accuracy using GBM models. When we excluded queries from the extremely sparsely populated clusters, the blindBLAST identification accuracy was 84.0% and the GBM accuracy was 87.2%. The GBM model reduce error counts in all categories of misclassification we benchmarked for blindBLAST. However, most of the misclassifications with GBM reduced error counts involve clusters with relatively abundant sample sizes, especially the non-dominant clusters. Thus, the bottlenecks to further improvement are primarily the member size imbalance between clusters and data sparsity in clusters. Methods that can generate valid data to enrich clusters with sparse data may improve the estimation accuracy of the GBM model. A set of structures that lie within the cluster radius constraint could be generated using Rosetta, emulating the SMOTE method (Chawla et al., 2002) for enriching samples in underpopulated classes. Another approach that serves to increase the member sizes of these clusters is to use semi-unsupervised learning to incorporate the sequenced antibodies without solved structures.

Furthermore, the GBM models are found incapable of further reducing errors in misclassifications involving clusters with small dihedral angle distance such as between L2-8-1 and L2-8-2. To utilize the limitation, we may wish to reflect the differences of distances between cluster pairs in the loss function in the machine learning training process, so that mismatches between clusters of greater structural differences can be penalized more heavily. On the other hand, the sampling and learning process can be adjusted in ways other than generating
447 synthetic data for sparse clusters, such as training each weak learner with an under-sampled
dominant cluster rather than oversampling the non-dominant clusters used in this study.

449 6 Conflict of Interest

450 JJG is an unpaid board member of the Rosetta Commons. Under institutional participation
agreements between the University of Washington, acting on behalf of the Rosetta Commons,
Johns Hopkins University may be entitled to a portion of revenue received on licensing Rosetta
software including programs described here. As a member of the Scientific Advisory Board of
Cyrus Biotechnology, JJG is granted stock options. Cyrus Biotechnology distributes the Rosetta
software, which may include methods described in this paper.

456 7 Author Contributions

457 XL performed the research. XL, JRJ, and JJG designed the research, analyzed the data,
and wrote the paper.

459 8 Funding

460 XL, JRJ, and JJG were funded by NIH R01-GM078221. JRJ was additionally funded by
NIH F31-GM123616.

462 9 Acknowledgments

463 We would like to thank Drs. Roland Dunbrack (ORCID 0000-0001-7674-6667) and Jared
Adolf-Bryfogle for help with the PyIgClassify database.

465 10 References

466 Adolf-Bryfogle J., Xu Q., North B. et al. 2015. PyIgClassify: a database of antibody CDR structural

general framework for computational antibody design. PLOS Computational Biology 14:e1006112. DOI: 10.1371/journal.pcbi.1006112.


Boyd SD., Crowe JE. Deep sequencing and human antibody repertoire analysis Sequence analysis techniques for antibody variable genes. DOI: 10.1016/j.coi.2016.03.008.


Marcatili P., Olimpieri PP., Chailyan A. et al. 2014. Antibody structural modeling with prediction of


**Figure 1** (on next page)

Query-template RMSD is smaller, if a template is selected from the query cluster, but the query cluster cannot always be correctly identified.

(A) IgG cartoon structure highlighting the variable heavy (VH, red) and light (VL, blue) domains which bind antigen through their CDR loops. (B) Count of non-redundant CDR loops in the PyIgClassify database for each VH loop-length and -type cluster, with a gray background indicating adequate numbers for GBM modeling and a white background indicating inadequate numbers, and a cartoon highlighting the VH beta-strand connectivity and CDR loop location. The CDR H3 is excluded due to its highly variable nature. (C) Analogous to (B), but for the VL.
Query-template RMSD is smaller, if a template is selected from the query cluster, but the query cluster cannot always be correctly identified.

(A) Cluster assignment accuracy comparison between blindBLAST and random assignment. Error bars show the standard deviation of the accuracy, for blindBLAST this is calculated across the 3-repeat 10-fold cross validation, whereas for the random model it is the standard deviation of accuracies over 1,000 iterations. In all but one case (H1-14) blindBLAST determines clusters more accurately than random. The average accuracy for blindBLAST is 78.8% whereas the average for the random model is 57.8%. (B) Comparison of query-template RMSD when the loop is selected by BLAST from the incorrect versus the “corrected” cluster. In this case, incorrect loops are loops with templates from clusters other than the query and they are “corrected” by sequence alignment only to templates within their cluster. In most cases, BLAST finds lower-RMSD templates within the cognate loop cluster than outside of it, indicating that correct cluster determination from sequence can lead to a better structural template. However, two loop types (H1-15 and L3-11) do not have lower RMSD templates in their cognate cluster.

Figure 2 (on next page)
BlindBLAST misclassifications can be categorized based on comparison to random assignment (i.e. better, the same, or worse than random) and their error counts (more or fewer than three misclassifications).

The significance value, as defined in Equation 1, is used for comparing blindBLAST and random assignment for any specific misclassification. Each point representing a misclassification is plotted with the error count from the random assignment and blindBLAST as x value and the y value. These categories cluster when the blindBLAST error count is compared to the random error count.
CDRs that are misclassified have a relatively large dihedral angle distance to their cluster exemplars.

For loop lengths and types with more than 5 members both correctly and incorrectly classified, the density of dihedral angle distances between the query loops and their cluster exemplar loops is plotted. For most loop lengths and types (e.g. H1-13, H2-9, L1-15, L3-8, L3-9, and L3-10), the misclassified distribution has more density at larger dihedral angle distances with respect to the correctly classified. The skewedness indicates that for many loops, if a query CDR is distant to its corresponding structural exemplar, then it is more likely to be incorrectly classified using the blindBLAST method.
To-exemplar Dihedral Angle Distance

blindBLAST Cases
- Incorrect Classification
- Correct Classification
Figure 5 (on next page)

CDRs that are misclassified have fewer structural neighbors, in some clusters.

For loop lengths and types with more than 5 members both correctly and incorrectly classified, the density of the number of structural neighbors is plotted. Structural neighbors, as defined in the Methods, are all CDRs with dihedral angle distances equal to or less than 1/15th of the cluster radius to a given CDR. In many clusters, including H1-13-1, H1-13-4, H1-13-5, H1-14-1, H2-10-6, L1-12-1, L3-10-1, L3-9-1, L3-9-cis7-1, L3-9-cis7-2, the misclassified CDRs have greater density at lower numbers of structural neighbors, with respect to the correctly classified CDRs. This indicates that the number of structural neighbors may affect the chance of correct template selection for a query structure.
The GBM model has higher accuracy and lower error count than blindBLAST.

(A) Comparison of blindBLAST (red) and GBM (blue) accuracy in assigning CDR sequences to clusters. Error bars represent the standard deviation in accuracy from cross validation (see Methods). (B) Comparison of the number of erroneously assigned clusters in blindBLAST and GBM error count. The GBM model universally lowers error count.
Figure 7 (on next page)

A detailed dissection of error count reduction by the GBM model.

(A) All misclassifications with a difference of three or more in error count between GBM and blindBLAST are plotted. A misclassification labeled as 1 → 2 denotes the number of cases of CDR queries belonging to cluster 1 but are classified as cluster 2 using the method. (B) Misclassifications involving at least one cis cluster with corresponding blindBLAST and GBM error counts differing by at least one are plotted.
Figure 8 (on next page)

Sequence logos of selected CDR clusters show readily available sequence rules.

(A) The amino acid compositions of the L3-9 clusters. There are no universal distinguishing residues, except for L3-9-1 which does not contain a proline at position 7. (B) Similarly, for H2-10-1, -2, and -6 there is not a universal difference in sequences.
Relative importance can be extracted from GBM models, permitting the identification of key sequence features.

For the best GBM model of each loop and length type (though only L3-9 [A], H2-10 [B], and L3-10 [C] are shown), the training features were ranked by how much they can help to reduce the training error on a scale of least to most important (1–100). As expected for L3-9 and L3-10, two proline containing loops, the presence of a proline at key positions is the most important feature. By comparison, for H2-10, importance values are more uniform and there is no single key residue.
Table 1 (on next page)

Categorization criteria of misclassification.

The misclassifications observed in blindBLAST results can be divided into four categories, based on p-value (from our significance test, Equation 1) and error.
<table>
<thead>
<tr>
<th>Misclassification Category</th>
<th>p-value</th>
<th>Error Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Worse than random</td>
<td>p &gt; 0.975</td>
<td>&gt;3</td>
</tr>
<tr>
<td>Better than random (high error)</td>
<td>p &lt; 0.025</td>
<td>&gt;3</td>
</tr>
<tr>
<td>Better than random (low error)</td>
<td>p &lt; 0.025</td>
<td>≤3</td>
</tr>
<tr>
<td>Random-like accuracy</td>
<td>0.025 &lt; p &lt; 0.975</td>
<td>&gt;3</td>
</tr>
</tbody>
</table>
Table 2 (on next page)

Dihedral angle differences for loop cluster pairs with low (left) and high (right) counts of misclassifications.

Cluster pairs with few blindBLAST misclassification errors are generally structurally distant. Left and right columns list cluster pairs for blindBLAST dihedral distances by performance category. Each misclassification is identified by the query cluster and the (incorrectly assigned) template cluster. The third column is the dihedral distance between the cluster exemplars of the two clusters.
<table>
<thead>
<tr>
<th>Query</th>
<th>Template</th>
<th>Dihedral-Angle Distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>H1-13-1</td>
<td>H1-13-5</td>
<td>27.2</td>
</tr>
<tr>
<td>H1-13-6</td>
<td>H1-13-1</td>
<td>23.2</td>
</tr>
<tr>
<td>H1-13-7</td>
<td>H1-13-1</td>
<td>11.8</td>
</tr>
<tr>
<td>H1-13</td>
<td></td>
<td></td>
</tr>
<tr>
<td>H1-13-1</td>
<td>H1-13-2</td>
<td>11.1</td>
</tr>
<tr>
<td>H1-13-1</td>
<td>H1-13-3</td>
<td>15.9</td>
</tr>
<tr>
<td>H1-13-2</td>
<td>H1-13-1</td>
<td>11.1</td>
</tr>
<tr>
<td>H1-13-3</td>
<td>H1-13-1</td>
<td>15.9</td>
</tr>
<tr>
<td>H1-13-4</td>
<td>H1-13-1</td>
<td>16.6</td>
</tr>
<tr>
<td>H1-13-5</td>
<td>H1-13-1</td>
<td>27.2</td>
</tr>
<tr>
<td>H1-13</td>
<td></td>
<td></td>
</tr>
<tr>
<td>H2-10-4</td>
<td>H2-10-1</td>
<td>19.2</td>
</tr>
<tr>
<td>H2-10-5</td>
<td>H2-10-1</td>
<td>21.4</td>
</tr>
<tr>
<td>H2-10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>H2-10-1</td>
<td>H2-10-2</td>
<td>11.3</td>
</tr>
<tr>
<td>H2-10-1</td>
<td>H2-10-3</td>
<td>9.4</td>
</tr>
<tr>
<td>H2-10-1</td>
<td>H2-10-6</td>
<td>6.8</td>
</tr>
<tr>
<td>H2-10-2</td>
<td>H2-10-1</td>
<td>11.3</td>
</tr>
<tr>
<td>H2-10-2</td>
<td>H2-10-6</td>
<td>8.8</td>
</tr>
<tr>
<td>H2-10-3</td>
<td>H2-10-1</td>
<td>9.4</td>
</tr>
<tr>
<td>H2-10-4</td>
<td>H2-10-2</td>
<td>8.6</td>
</tr>
<tr>
<td>H2-10-6</td>
<td>H2-10-1</td>
<td>6.8</td>
</tr>
<tr>
<td>L3-9-1</td>
<td>L3-9-cis7-1</td>
<td>20.9</td>
</tr>
<tr>
<td>L3-9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>L3-9-2</td>
<td>L3-9-cis7-1</td>
<td>6.9</td>
</tr>
<tr>
<td>L3-9-cis7-1</td>
<td>L3-9-cis7-2</td>
<td>9.7</td>
</tr>
<tr>
<td>L3-9-cis7-1</td>
<td>L3-9-cis7-3</td>
<td>10.4</td>
</tr>
<tr>
<td>L3-9-cis7-2</td>
<td>L3-9-cis7-1</td>
<td>9.7</td>
</tr>
<tr>
<td>L3-9-cis7-3</td>
<td>L3-9-cis7-1</td>
<td>10.4</td>
</tr>
</tbody>
</table>