Enhancing Antibody CDR Coordinate Prediction with Graph Neural Networks

SSUDI: ssyoung, limkifw

December 11, 2024

1 Preamble

By Sonny Young and Kif Lim as part of the Stanford CS224W course project. This implementation has not been previously reported to our knowledge and contributes to the field of protein structure prediction.

2 Motivation

We explored a novel contribution to modeling antigen-antibody interactions using graph neural networks. Our goal is to create a better generative model that predicts complementary-determining regions (CDRs) in antibodies as inspired by the paper "Iterative Refinement Graph Neural Network for Antibody Sequence-Structure Co-design" [1]. Doing so can serve as a high potential therapeutic tool that generates higher affinity and immunogenic antibodies.

In the original base paper, the RefineGNN model generates antibodies CDR coordinates but lacks constraints related to the antigen's structure and spatial configuration, which is left as future work. To address this, we propose integrating an antigen constraint while incorporating the following improvements upon this architecture:

- Using an augmented dataset with more proper CDR masking and up to date PDB structures from the Structural Antibody Database (SAb-Dab) [2]
- Explore the importance of the quality and size of dataset on initial model performance.
- Define a probability distribution $P_{\text{compl\_antig}}$ representing possible spatial configurations of CDR loops complementing a given antigen. This distribution serves as input to the RefineGNN model, guiding spatial configurations of node and edge features. We enforce compliance by including this distribution as a penalty term in the model's loss function.
- Develop a parallel ML algorithm using a diffusion model that, given an antigen configuration, outputs the $P_{\text{compl\_antig}}$ distribution.

3 Antibody CDR dataset preparation and Expected Output

RefineGNN model was trained on the 2022 version of SAbDab, which included 4736 antibody structures. The current SAbDab dataset as of 2024 has been updated to include 1705 unbound structures and 7191 antigen-bound structures for a total of 8896 structures. Due to memory constraints, we work with a dataset that contains 5044 structures using a 80-10-10 train, validation, test split. We format the dataset to the following:

```
data_format_summary = {
    "pdb": "PDB identifier for the structure",
    "antibody_seq": "Antibody amino acid sequence",
    "antibody_cdr": "Mapping of residues to complementarity-determining regions (CDRs)",
    "antibody_coords": "3D coordinates for each residue in the antibody",
```

```
    "antibody_atypes": "Atomic types for the antibody",
    "antigen_seq": "Antigen amino acid sequence",
    "antigen_coords": "3D coordinates for each residue in the antigen",
    "antigen_atypes": "Atomic types for the antigen"
}
```

Running the model without augmented data yields CDR coordinate predictions that correspond to the plots below, with a final loss of 188.391 and final RMSD of 2.433. After sampling the best 1000 CDR coordinates, we can also see what a generated loop would look like.

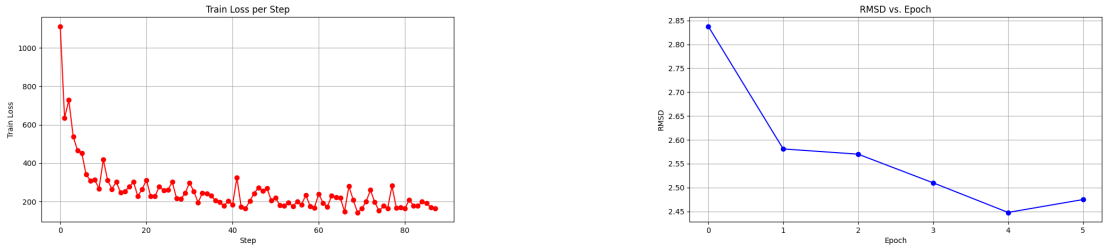

Figure 1: Trained on original 4300 antibody structures. The model predicted CDR loops given the antibody sequence and coordinates given in its respective PDB file with the H1, H2, and H3 loops indicated in the masked amino acid sequence.

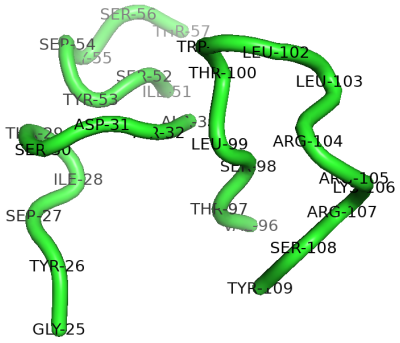

Figure 2: Predicted CDR loop from the PDB entry 1c12. The three loops correspond to CDR H1, H2, and H3, with each loop labeled by amino acid types.

The model was trained for 10 epochs with a hidden dimension of 256, 100 batch tokens, 9 k-neighbors, batch size of 8, dropout of 0.1, learning rate of 0.001. In order to optimize for memory, we used gradient checkpointing and mixed precision training. Model training for the base model took approximately 2 hours on one A100 80GB GPU. Evaluation metric included perplexity and RMSD between the predicted CDR and the ground truth masked coordinates.

4 RefineGNN model architecture and Antigen structure design

The antibody graph $G(s) = (V, E)$ is defined with node features $V = \{v_1, \ldots, v_n\}$ representing dihedral angles $(\phi_i, \psi_i, \omega_i)$ of each residue i, and edge features $E = \{e_{ij}\}_{i \neq j}$. Edge features e_{ij} encode spatial relationships via:

$$e_{ij} = \left(E_{\text{pos}}(i-j), \text{RBF}(\|x_{i,\alpha} - x_{j,\alpha}\|), O_i^T \frac{x_{j,\alpha} - x_{i,\alpha}}{\|x_{i,\alpha} - x_{j,\alpha}\|}, q(O_i^T O_j) \right).$$

We use a K-nearest neighbors graph with $K = 8$. The refinement process initializes a graph $G^{(0)}$ and iteratively refines it through message passing with parameters θ and $\tilde{\theta}$:

$$\{h_1^{(t)}, \ldots, h_n^{(t)}\} = \text{MPN}_\theta(G^{(t)}), \quad h_i^{(t,l+1)} = \text{LayerNorm}\left(\sum_j \text{FFN}(h_i^{(t,l)}, h_j^{(t,l)}, E(s_j), e_{ij})\right),$$

predicting amino acid types via:

$$p_{t+1} = \text{softmax}(W_a h_{t+1}^{(t)}).$$

The coordinates are refined with:

$$\{h_1^{(t+0.5)}, \ldots, h_n^{(t+0.5)}\} = \text{MPN}_{\tilde{\theta}}(G^{(t+0.5)}), \quad x_i^{(t+1),e} = W_e x h_i^{(t+0.5)}, \quad e \in \{\alpha, c, n\}.$$

Structure and sequence networks are disentangled for distinct prediction tasks.

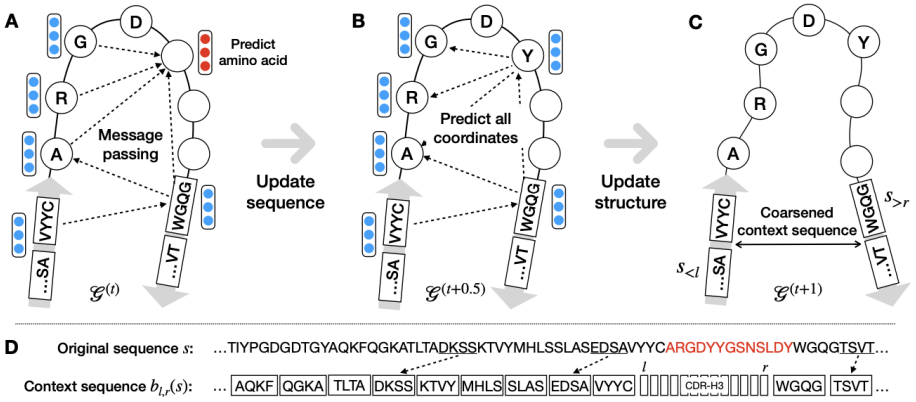

Figure 3: This is how the paper illustrates their CDR prediction. We enhance the prediction by incorporating antigen coordinates to generate an inverse probability map.

We introduce $p(x)$ as an antigen-defined complementary probability distribution to constrain the model, ensuring that the GNN-generated antibody structures respect the spatial configurations dictated by the antigen. Our two approaches include: (1) a naive inverse probability map matrix and (2) a diffusion model to learn the inverse probability map.

4.1 Naive Inverse Probability Map Matrix

The first approach computes a complementary probability distribution $p(x)$ using a Gaussian kernel over pairwise distances between antibody and antigen coordinates. This deterministic inverse probability map uses the antibody coordinates $A \in \mathbb{R}^{B \times N_x \times 3}$ and antigen coordinates $G \in \mathbb{R}^{B \times N_y \times N_{\text{atoms}} \times 3}$, where B is the batch size, N_x is the number of antibody residues, N_y is the number of antigen residues, and N_{atoms} is the number of atoms per antigen residue, the process begins by computing the Euclidean distance between every antibody residue i and every antigen residue j for each atom k in the antigen. The pairwise distance matrix is given by:

$$d_{i,j} = \frac{1}{N_{\text{atoms}}} \sum_{k=1}^{N_{\text{atoms}}} d_{i,j,k}, \quad \text{where} \quad d_{i,j,k} = \|A[i,:] - G[:,j,k,:]\|.$$

These distances are then transformed into probabilities using a Gaussian kernel:

$$p_{i,j} = \exp\left(-\frac{d_{i,j}^2}{2\sigma^2}\right),$$

where σ is a hyperparameter controlling the decay rate, ensuring that closer residues correspond to higher probabilities. Finally, the inverse probability map is computed to highlight regions where residues are less likely to interact. The inverse probability is defined as:

$$\mathrm{P}_{i,j} = 1 - p_{i,j}.$$

This results in the final inverse probability map inv_prob $\in \mathbb{R}^{B \times N_x \times N_y}$. We transform this probability matrix into a loss term by penalizing residue predictions that are in disallowed regions. We include a regularization term that encourages node and edge features to align with the distribution $p(x)$. This can be achieved through additional loss terms as described earlier, which penalize deviations from $p(x)$ during the optimization process.

$$L_{\text{GNN}} = L_{\text{refineGNN}} + \lambda L_{\text{antigen}} = L_{\text{distance}} + L_{\text{dihedral}} + L_{C\alpha} + \lambda L_{\text{antigen}}$$

Since we take pairwise distances in computing the inverse probability map, our model remains both rotation and translation invariant. Another snippet of code for the inverse probability map is shown below.

```
def generate_inverse_probability_map(self, antibody_coords, antigen_coords):
    """
    Generate an inverse probability map based on distances.
    """
    ...
    # Compute pairwise distances from processed PDBs
    # Shape: [B, 32, 5, 3]
    dX = antibody_coords.unsqueeze(2) - antigen_coords_reduced.unsqueeze(1)
    # Shape: [B, 32, 5]
    distances = torch.norm(dX, dim=-1)
    # Apply a Gaussian kernel to derive an inverse probability map
    sigma = 5.0
    inv_prob_map = torch.exp(-distances**2 / (2 * sigma**2))
    inv_prob_map = 1 - inv_prob_map
    return inv_prob_map
```

4.2 Diffusion Model to Learn the Probability Map

The diffusion generative model is employed to dynamically learn the probability map $p(x)$, leveraging graph neural networks (GNNs) to process antigen-antibody configurations. The forward diffusion process progressively perturbs the input data by adding Gaussian noise at each timestep t, which can be expressed as:

$$\mathbf{x}_t = \sqrt{\alpha_t}\mathbf{x}_{t-1} + \sqrt{1-\alpha_t}\mathbf{z}, \quad \mathbf{z} \sim \mathcal{N}(0, I),$$

where α_t is a noise schedule that controls the amount of noise added at step t. The reverse process aims to iteratively remove the noise and generate the probability map by predicting the added noise. This reverse step is modeled as:

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \sigma_t^2 I),$$

where $\mu_\theta(\mathbf{x}_t, t)$ is the predicted mean parameterized by the neural network, and σ_t^2 is the variance.

4.2.1 Architecture

The diffusion model operates on a graph representation of antigen-antibody configurations and consists of the following components. We choose to include both antibody and antigen information rather than just the antigen so that the model can learn interactions between the two proteins that may have an influence on the structure. We created a basic fully connected graph where nodes are atoms and edges are pairwise interactions instead of the K-nearest neighbors in the rest of the model.

```
def construct_graph_edges(self, combined_coords, batch_size):
    """
    Construct edges (graph will be a fully connected graph) between Ab/Ag
    and batch indices for the graph representation.

    Args:
        combined_coords: Combined antibody and antigen coordinates [B, N_combined, 3].
        batch_size: Number of graphs in the batch.

    Returns:
        edge_index: Edge list [2, total_edges].
        batch: Batch indices [total_nodes].
    """
    B, N, _ = combined_coords.size()
    total_nodes = B * N

    edge_index = []
    batch = []
    for i in range(B):
        nodes = torch.arange(i * N, (i + 1) * N, device=combined_coords.device)
        edges = torch.combinations(nodes, r=2).t()
        edge_index.append(edges)
        batch.extend([i] * N)

    edge_index = torch.cat(edge_index, dim=1)
    batch = torch.tensor(batch, device=combined_coords.device)
    return edge_index, batch
```

To learn the probability matrix the other features include:

- **Time Embedding:** A learnable embedding layer encodes the timestep t into a vector representation, which is used to condition the diffusion process.
- **Input Projection:** A linear layer maps the noisy node features (3D coordinates) from the input space to a higher-dimensional hidden space.
- **Graph Neural Network (GNN):** Two layers of Graph Attention Networks (GATConv) iteratively refine the node features by modeling interactions between antibody and antigen residues. The graph structure is defined using node connectivity and spatial relationships.
- **Output Projection:** A linear layer maps the refined node features from the hidden space back to the 3D coordinate space, providing the denoised predictions for the next timestep.

The diffusion model iteratively processes the noisy input features using the GNN layers, conditioned on the time embeddings, to predict the mean $\mu_\theta(\mathbf{x}_t, t)$ for the reverse diffusion step. This enables the model to progressively refine the antigen-antibody configurations and generate the probability map $p(x)$, which captures regions of high complementarity.

4.2.2 Loss Function

The diffusion model is trained using a denoising score-matching loss:

$$L_{\text{denoise}} = \mathbb{E}_{\mathbf{x}_0, \mathbf{z}, t}\left[\|\mathbf{z} - \nabla_{\mathbf{x}_t} \log p_\theta(\mathbf{x}_t|t)\|^2\right],$$

along with the antigen constraint regularization term:

$$L_{\text{antigen}} = \frac{1}{B \cdot N_x} \sum_{b=1}^{B} \sum_{i=1}^{N_x} \mathbf{d}_{\text{weighted},i} + \lambda \|\mathbf{p}_{\text{inv, antibody}} - 1\|^2,$$

where $\mathbf{d}_{\text{weighted},i}$ represents the distance between predicted and true coordinates, weighted by the learned $p(x)$.

5 Results

5.1 Model Performance on Perplexity and RMSD

The trained model, utilizing a total of 5044 antibody-antigen structures, demonstrates a decreasing perplexity (PPL) and root mean square deviation (RMSD) during training and validation. Perplexity, a metric reflecting the likelihood of predicted amino acid sequences, decreased consistently across epochs. RMSD, measuring structural deviation between predicted and true residue coordinates, also improved, showing the model's growing capability to predict accurate 3D antibody structures.

The model represents antibody-antigen interactions using a graph representation where residues are nodes and their spatial relationships form edges. This approach enables accurate prediction of CDR amino acid coordinates. Notably, the model also predicts the identity of the amino acids in the CDR, enabling insights into antibody functionality. This highlights the utility of the model in generating realistic antibody structures suitable for downstream applications.

The dataset size experiment reveals that increasing the dataset size drastically improves both PPL and RMSD. Larger datasets provide more diverse structural examples, enhancing the model's ability to generalize to unseen antibody-antigen pairs. Given that the current Structural Antibody Database has thousands of more structures and is actively growing, we leave testing this model on the most updated structures as future work.

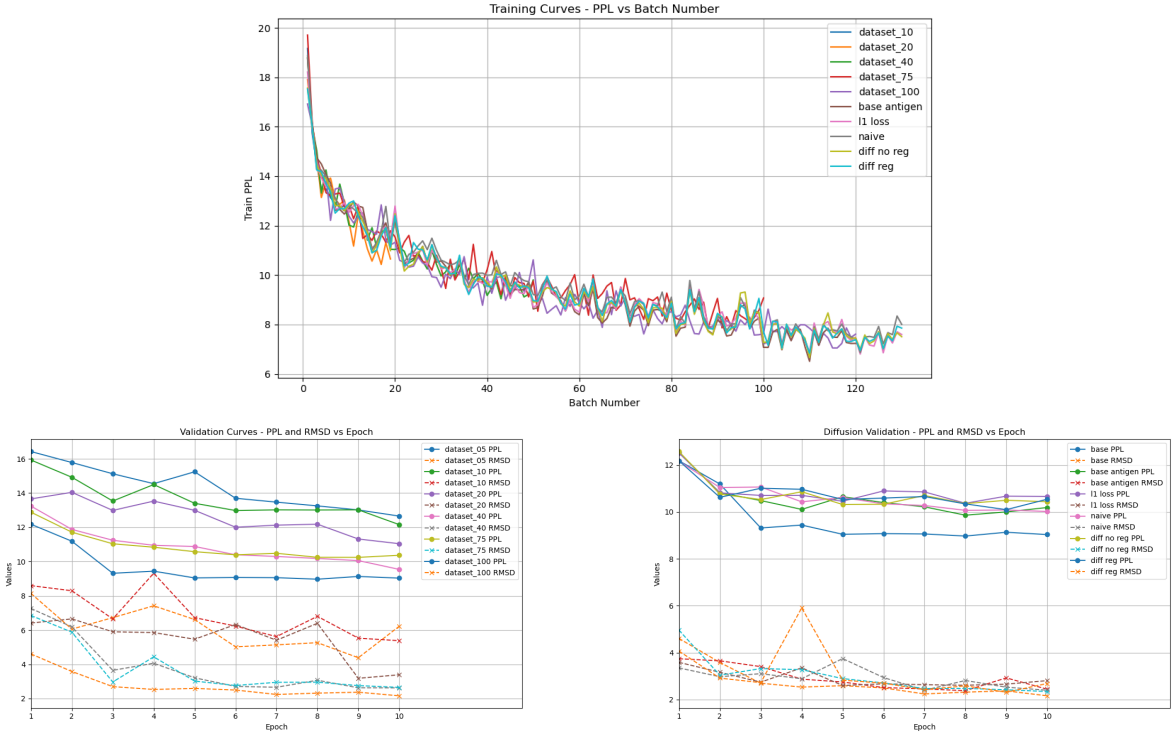

Figure 4: Training curve for all experiments. Validation plots of perplexity and RMSD for dataset experiments (left) and different antigen information-incorporated models.

Incorporating antigen information through diffusion-based modeling improved performance marginally in terms of PPL and RMSD. However, the improvements were modest for the following reasons:

1. More Data vs. Antigen Information: The observed improvement may be primarily attributed to the use of a larger dataset rather than directly incorporating antigen features. This suggests that the model already learns spatially relevant structural constraints effectively from the dataset.

2. Naive Inverse Probability Map: By explicitly constructing regions where the antibody does not need to bind (via hardcoded distance matrix probability maps), a small improvement was observed. While there is a small improvement, the probability matrix is sparse, which makes it difficult to penalize certain regions of 3D space in an effective way.

3. Diffusion Model Results: Using the diffusion model, further marginal gains were observed, indicating its utility in capturing spatial constraints. However, the antibody constraint loss term, which enforces antigen-antibody complementarity, vanishes quickly during training. Introducing a variable penalization parameter to reinforce this term yielded modest additional improvements.

	Test PPL	Test RMSD
Micro	13.021	6.399
Tiny	12.732	4.757
Small	11.982	3.262
Medium	10.313	2.491
Large	10.724	2.624
Full dataset (no antigen)	8.160	2.433
Full dataset (antigen)	10.224	2.348
L1 Loss	10.650	2.550
Naive IPM	10.459	2.349
Diffusion IPM no reg.	10.594	2.833
Diffusion IPM reg.	**10.152**	**2.278**

Figure 5: Comparison of Test PPL and RMSD across different models. Micro, Tiny, Small, Medium, Large correspond to dataset sizes that are 5%, 10%, 20%, 40%, and 75% of the full dataset, respectively. IPM = inverse probability map.

5.2 Antigen-Informed Model: Biological Relevance

An antigen-informed model provides a more realistic representation of real-world antibody functionality. Antibodies inherently interact with target antigens, and their structural relevance is contingent on this interaction. Without a target antigen, the antibody's structure lacks biological context or functional relevance. Incorporating antigen data allows the model to learn realistic spatial and functional constraints. It also facilitates the prediction of antibody sequences and structures that are optimized for binding specific antigens.

6 Conclusion and Future Directions

The presented model demonstrates robust performance in predicting CDR sequences and 3D structures. The dataset size experiment underscores the importance of training on diverse, large-scale datasets, while diffusion-based methods offer potential for capturing antigen-antibody complementarity. Future work should focus on:

- Incorporating all the latest structures from the Structural Antibody Database to yield better antigen-informed predictions.
- Enhancing the antigen constraint loss term to retain its impact during training.
- Exploring additional antigen-informed features, such as electrostatics or hydrophobic interactions, to improve complementarity predictions.
- Extending the model's application to de novo design of antibodies tailored for specific antigens. Other models can be considered to generate probability distribution landscapes like flows models or othe graph-based models.

Overall, this work highlights the importance of incorporating antigen features for biologically realistic antibody structure predictions and provides a foundation for further advancements in antibody design. Future work to build on this model is being explored.

7 Code Release

Custom scripts to generate our own datasets after obtaining raw PDB files from the SAb-Dab database. For the experiments we've run, all code can be viewed at: https://github.com/ssyoung26/cs224w_project

8 Acknowledgments

We thank the kind support from Simone Surdi who contributed ideas and advice for the formulation and execution of this project.

9 References

References

[1] Wengong Jin et al. *Iterative Refinement Graph Neural Network for Antibody Sequence-Structure Co-design*. 2022. arXiv: 2110.04624 [q-bio.BM]. URL: https://arxiv.org/abs/2110.04624.

[2] James Dunbar et al. "SAbDab: the structural antibody database". In: *Nucleic Acids Research* 42.D1 (Nov. 2013), pp. D1140–D1146. ISSN: 0305-1048. DOI: 10.1093/nar/gkt1043. eprint: https://academic.oup.com/nar/article-pdf/42/D1/D1140/3538157/gkt1043.pdf. URL: https://doi.org/10.1093/nar/gkt1043.

[3] Matthias Fey and Jan E. Lenssen. "Fast Graph Representation Learning with PyTorch Geometric". In: *ICLR Workshop on Representation Learning on Graphs and Manifolds*. 2019.