EpiCLIP: Learning Antibody-Antigen Interactions from Approximate Interfaces

Joseph Boen [1 2] **Imee Sinha** [2] **Samuel Stanton** [2] **Pranav Rao** [2] **Rob Alberstein** [2] **Simon Kelow** [2]

Abstract

Modeling antibody-antigen interactions in therapeutic discovery remains challenging because structurally resolved complexes are scarce, exact interfaces are unavailable at inference, and pairwise structure-based prediction does not scale. We introduce EpiCLIP, a contrastive dual-encoder framework that casts antibody-antigen interaction modeling as dense retrieval over approximate epitopes and paratopes. EpiCLIP trains on synthetic epitope candidates designed to match the noisy search space encountered at deployment, enabling both binder retrieval and epitope mapping in a shared embedding space. Empirically, EpiCLIP achieves strong performance on both tasks while operating directly on approximate interface candidates. These results suggest that retrieval over approximate interfaces is a scalable alternative to exact-interface supervision or exhaustive pairwise structural modeling for antibody-antigen interaction prediction. Code for EpiCLIP can be found at https://github.com/prescient-design/epiclip

1. Introduction

Antibodies are a major therapeutic modality, making antibody-antigen interaction modeling important for rational discovery and candidate prioritization (Lu et al., 2020). Antigen recognition is mediated primarily by the six complementarity-determining regions (CDRs) in the antibody variable domains, denoted H1-H3 and L1-L3, which account for most of the sequence and structural diversity involved in binding. The subset of antibody residues that directly contacts the antigen is called the *paratope*, while the corresponding antigen residues define the *epitope*. Paratope residues are typically concentrated within the CDRs, though framework residues can also contribute to binding (Sela-Culang et al., 2013). Because antibody-antigen recognition is mediated by localized interactions, understanding binding requires reasoning not only about whether an antibody binds a target, but also where that interaction is likely to occur (Dang et al., 2023).

In this work, we focus on two related tasks: **binder retrieval**, which ranks antibodies likely to bind a target antigen, and **epitope mapping**, which identifies the binding region for a known antibody-antigen pair. Binder retrieval may be either *blind*, when the relevant binding site is unknown, or *guided*, when antibodies are prioritized for a desired antigen region. Experimental methods for resolving antibody-antigen complexes and interfaces remain too costly to apply at scale, motivating computational methods that can model antibody-antigen interactions efficiently while remaining sensitive to binding-site information (Dang et al., 2023).

Computational modeling is challenging for three reasons. First, accurate prediction depends on modeling antibody-antigen compatibility rather than either molecule in isolation, so screening over N antibodies and M antigens requires $\mathcal{O}(MN)$ pairwise evaluations. Second, true paratope-epitope interfaces are rarely known at deployment, leaving only approximate candidate regions derived from weak biological or structural priors (Jespersen et al., 2019). Third, experimentally resolved antibody-antigen complexes remain scarce, limiting direct structural supervision (Dunbar et al., 2014). To address these challenges, we introduce EpiCLIP, a unified model for binder retrieval and epitope mapping. EpiCLIP recasts antibody-antigen modeling as dense retrieval over approximate epitopes and paratopes, with independent candidate encoding and lightweight similarity-based retrieval. This replaces $\mathcal{O}(MN)$ pairwise evaluations with $\mathcal{O}(M+N)$ candidate encodings and aligns training with the noisy interface candidates available at inference.

2. Background and Related Work

Antigen-Only Epitope Prediction Methods such as DiscoTope (Høie et al., 2024), BepiPred (Clifford et al., 2022), and related models predict epitopes from antigen sequence, structure, or local physicochemical context alone. These approaches are useful for identifying generally likely binding sites, but do not condition on a specific antibody partner. As

[1]Stanford University [2]Prescient Design, Genentech. Correspondence to: Joseph Boen <tboen@stanford.edu>, Simon Kelow <simon.kelow@gmail.com>.

Accepted at the 2026 Workshop on Generative and Agentic AI for Biology (ICML 2026)

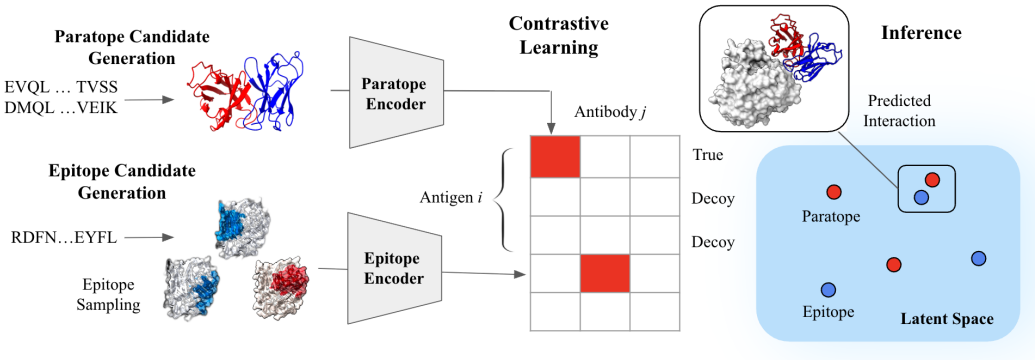

Figure 1. EpiCLIP is a dual-encoder framework for antibody-antigen interaction prediction. Antibody and antigen sequences are converted into approximate paratope and epitope candidates, encoded jointly by contrastive learning, and retrieved in a shared embedding space for binder ranking and epitope mapping.

a result, they are not suited to interaction-focused tasks such as binder retrieval or antibody-specific epitope mapping (Jespersen et al., 2019).

Complex Structure Prediction At the other extreme are pairwise structural models that explicitly represent both binding partners, including co-folding systems such as AlphaFold-3 (Abramson et al., 2024) and Boltz-2 (Passaro et al., 2025). These methods directly predict antibody-antigen complexes and can in principle support epitope mapping and binder ranking. However, this expressivity comes at substantial computational cost since each antibody-antigen hypothesis requires a fresh structure prediction, making large-scale screening over many candidate pairs prohibitively expensive in practical discovery settings (Pedotti et al., 2011).

Contrastive Learning for Dense Retrieval EpiCLIP is inspired by CLIP-style contrastive learning and dense retrieval, where two objects are embedded independently and compared with a simple similarity function (Radford et al., 2021). This provides a scalable middle ground between antigen-only epitope scoring and fully pairwise structural modeling. Concurrent with our work, CALM (Lee et al., 2026) explores contrastive learning for antibody-antigen specificity prediction; EpiCLIP differs in its explicit focus on approximate epitope and paratope candidates and on supporting epitope mapping, guided binder retrieval, and blind binder retrieval within a single framework.

3. Problem Definition

We formulate antibody-antigen modeling as representation learning over approximate interface regions. Let x denote an antigen sequence of length L_x, and let y denote the concatenated heavy- and light-chain antibody variable region of length L_y. Let $E \subseteq \{1, \dots, L_x\}$ and $P \subseteq \{1, \dots, L_y\}$ denote the true epitope and paratope residue sets. At inference time, however, E and P are unknown. Instead, EpiCLIP operates on candidate interface regions: an epitope candidate is specified by (x, m), where $m \in \{0,1\}^{L_x}$ is a binary mask over antigen residues, and a paratope candidate is specified by (y, c), where $c \in \{0, \mathrm{H1}, \mathrm{H2}, \mathrm{H3}, \mathrm{L1}, \mathrm{L2}, \mathrm{L3}\}^{L_y}$ is a categorical per-residue annotation indicating whether each residue lies outside the candidate paratope or belongs to one of the six CDR loops. Both are approximate: m may only coarsely localize the true epitope, and c typically contains, but is not identical to, the true paratope.

EpiCLIP learns an epitope encoder f and a paratope encoder g that map candidate regions into a shared d-dimensional embedding space,

$$e = f(x, m) \in \mathbb{R}^d, \qquad p = g(y, c) \in \mathbb{R}^d, \tag{1}$$

and scores candidate compatibility by cosine similarity,

$$s\big((x, m), (y, c)\big) = \left\langle \frac{e}{\|e\|_2}, \frac{p}{\|p\|_2} \right\rangle, \tag{2}$$

where s denotes the interaction score in the shared embedding space.

This formulation recasts antibody-antigen interaction modeling as dense retrieval over approximate epitope and paratope candidates. Given an antigen x, an antibody y, a set of candidate epitope masks $M(x) = \{m_i\}_{i=1}^{K}$, a library of antibodies Y, each with candidate paratope annotation c_y, and optionally a target epitope m_{target}, we induce three retrieval tasks:

$$m^* = \arg\max_{m \in M(x)} s\big((x, m), (y, c_y)\big), \tag{3}$$

$$y^* = \arg\max_{y \in Y} s\big((x, m_{\text{target}}), (y, c_y)\big), \tag{4}$$

$$y^* = \arg\max_{y \in Y} \max_{m \in M(x)} s\big((x, m), (y, c_y)\big). \tag{5}$$

Eq. 3 corresponds to epitope mapping, Eq. 4 to guided binder retrieval, and Eq. 5 to blind binder retrieval.

4. Methods

4.1. Architecture

EpiCLIP employs a CLIP-style dual-encoder architecture that maps epitope and paratope candidates into a shared embedding space (Appendix A). In each encoder, a frozen pretrained residue-level backbone first produces contextual residue embeddings, after which a lightweight candidate-conditioned aggregation module summarizes the proposed interface region and a small projection head maps the result into the shared embedding space. This design keeps the trainable architecture lightweight while allowing the model to refine coarse candidate regions rather than treating all candidate residues uniformly.

For epitope encoding, the aggregator receives antigen residue embeddings together with a binary mask m specifying the candidate region, and uses a bottleneck self-attention module followed by learned pooling to model interactions within the candidate patch efficiently. For paratope encoding, the candidate representation c_y exposes additional structure as each residue is annotated by CDR loop identity. The paratope aggregator therefore uses a hierarchical design: it first summarizes residues within each of the six CDR loops, then models interactions among the resulting loop-level embeddings, and finally pools them into a single paratope representation. Together, these modules allow EpiCLIP to refine coarse local candidates while respecting their differing organization.

4.2. Sampling Synthetic Epitopes

In EpiCLIP, a natural prior for paratope candidates is CDR regions, but no analogous epitope candidate set is immediately available. We therefore generate synthetic epitope candidates to approximate the coarse regional proposals available at deployment. This aligns training with inference-time candidates and expands supervision by generating multiple plausible and non-binding regions on the same antigen.

Conceptually, we view synthetic epitope generation as sampling binary masks $m \sim P(m \mid x)$ over plausible antigen regions. Because directly sampling residue subsets is intractable, we factor this process through a residue-level latent interface prior $s \in \mathbb{R}^{L_x}$,

$$P(m \mid x) = \int P(m \mid s, x)\, P(s \mid x)\, ds. \tag{6}$$

Here, $P(s \mid x)$ is an antibody-agnostic prior over plausible interaction residues, and $P(m \mid s, x)$ is a geometry-aware sampler over spatially coherent antigen patches. In our experiments, we instantiate $P(s \mid x)$ with relative solvent accessibility (RSA) (Tien et al., 2013), a deliberately weak antibody-agnostic prior based only on residue exposure and chosen to reduce leakage from a stronger learned epitope predictor. We instantiate $P(m \mid s, x)$ with a geometric patch sampler that produces a finite set of candidate epitopes $M(x) = \{m_i\}_{i=1}^{K}$. The same mechanism is used during training and inference: during training, $M(x)$ provides targeted augmentation and intra-antigen negatives, while at inference it defines the search space for epitope mapping and binder retrieval (Appendix B).

4.3. Contrastive Objective

During training, a batch contains B antibody-antigen complexes. For each antibody y_i, we form a single paratope candidate from its CDR annotation and embed it as $p_i = g(y_i, c_i)$. For each antigen x_i, we generate a finite set of synthetic epitope candidates $M(x_i) = \{m_{ik}\}_{k=1}^{K}$, which are embedded as $e_{ik} = f(x_i, m_{ik})$. This yields an asymmetric batch consisting of B paratope and $B \times K$ epitope embeddings. Synthetic epitope candidates are labeled by their overlap with the true epitope of the corresponding complex, so a single paratope may have multiple valid positive epitope candidates. Remaining patches from the same antigen act as intra-antigen negatives, while candidates from other complexes provide inter-antigen negatives (Appendix B.4).

We train EpiCLIP with a bidirectional InfoNCE objective (Oord et al., 2018) adapted to this asymmetric, multi-positive setting. Let $s_{ij} = p_i^\top e_j$ denote the similarity between paratope i and epitope candidate j, let $P(i)$ denote the set of positive epitope candidates for paratope i, and let $\gamma > 0$ be a learnable temperature parameter. In the paratope-to-epitope direction, each paratope must assign high similarity to all synthetic positives:

$$\mathcal{L}_i^{p\to e} = -\log \frac{\sum_{j\in P(i)} \exp(\gamma s_{ij})}{\sum_{j=1}^{BK} \exp(\gamma s_{ij})}. \tag{7}$$

Conversely, for each positive epitope candidate j, let $t(j)$ denote the index of its matching paratope. The epitope-to-paratope direction requires each positive candidate to identify the correct paratope among the B in the batch:

$$\mathcal{L}_j^{e\to p} = -\log \frac{\exp(\gamma s_{t(j),j})}{\sum_{i=1}^{B} \exp(\gamma s_{ij})}. \tag{8}$$

The total InfoNCE loss is the convex combination,

$$\mathcal{L}_{\text{InfoNCE}} = \lambda\, \mathcal{L}^{p\to e} + (1-\lambda)\, \mathcal{L}^{e\to p}, \quad \lambda \in [0,1] \tag{9}$$

Together, these losses encourage a representation space that supports both epitope mapping and binder retrieval over local, approximate interface candidates (Appendix D).

5. Results

Data We train and evaluate EpiCLIP on antibody-antigen complexes from the Antibody-Antigen Complex Database

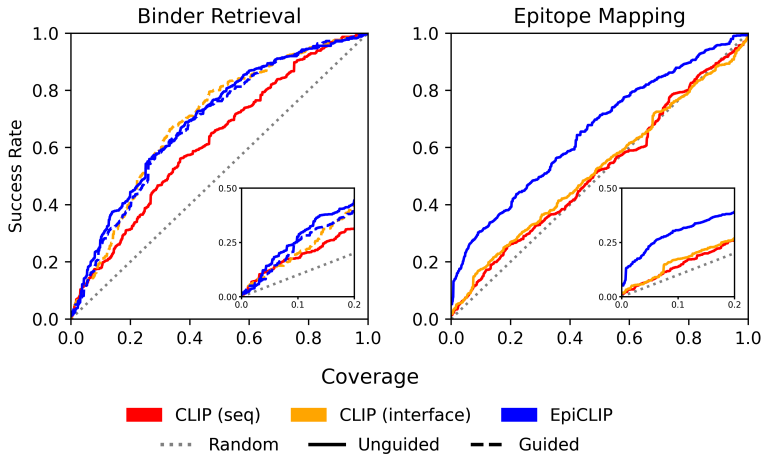

Figure 2. Binder retrieval (left) and epitope mapping (right), shown as success-rate versus coverage curves with low-coverage insets. EpiCLIP performs best overall, with especially strong epitope mapping and a small guided-versus-unguided gap in binder retrieval.

(AACDB) (Zhou et al., 2025). Paratope candidates are constructed from AHo-numbered antibody sequences using AntPack (Parkinson & Wang, 2024), and synthetic epitope candidates are generated from ESMFold (Lin et al., 2023)-predicted antigen structures to avoid leakage from antibody-conditioned holo complex geometry. Ground-truth epitope and paratope labels are derived from resolved antibody-antigen contacts using a 5.5 Å interface threshold. To evaluate generalization to unseen antigen families, we split the dataset by antigen sequence clusters using MMseqs2 (Steinegger & Söding, 2017) at 40% sequence identity with a 50/25/25 train/validation/test partition. After preprocessing and curation, this yields 3332 complexes (Appendix C).

Experiments We compare EpiCLIP against two CLIP-style baselines designed to isolate the value of learning from approximate interfaces. All methods use frozen ESM-2 (Lin et al., 2023) and IgBert (Kenlay et al., 2024) residue embeddings with a shared 128-dimensional embedding space. The CLIP (seq) baseline performs contrastive learning over full antibody/antigen sequences , while the CLIP (interface) uses exact interface annotations derived from resolved complexes. These two baselines correspond to the methods introduced in concurrent work CALM (Lee et al., 2026). As the CALM code is not publicly available, we treat the baselines as informative comparisons rather than exact reproductions. For binder retrieval, each model is evaluated in the inference regime it natively supports. For epitope mapping, each model utilizes its native paratope representation to rank the same set of candidate antigen binding regions (Appendix E).

Analysis Figure 2 shows that EpiCLIP is the strongest overall model, with particularly strong epitope mapping and competitive binder retrieval performance. On binder retrieval, when given exact interface annotations, CLIP (interface) substantially outperforms the sequence-only baseline, indicating that antibody-antigen specificity depends strongly on localized binding information rather than global representations. However, this advantage is brittle, because it relies on test-time access to exact interfaces that are unavailable in realistic deployment settings. This limitation becomes most apparent in epitope mapping, where both CLIP baselines perform only marginally better than random, suggesting that they learn only weak binding-site localization. By contrast, EpiCLIP is trained on approximate candidates and paired with aggregation modules designed to refine noisy local regions, allowing it to exploit local structure without becoming dependent on exact supervision. This is reflected both in its small guided-versus-unguided retrieval gap and in its strong epitope mapping performance, suggesting that the model learns not only which antibodies bind, but also where binding is likely to occur.

6. Conclusion

EpiCLIP frames antibody-antigen interaction modeling as dense retrieval over approximate epitopes and paratopes, combining a lightweight dual-encoder architecture with synthetic epitope generation. This formulation supports binder retrieval and epitope mapping while remaining aligned with the noisy candidate regions available at deployment. Empirically, EpiCLIP achieves strong performance on both tasks and provides evidence that the learned representation supports binding-site localization in addition to binder ranking. These results suggest that retrieval over approximate interface candidates is a scalable alternative to exact-interface supervision or exhaustive pairwise structural modeling for antibody-antigen interaction prediction.

References

Abramson, J., Adler, J., Dunger, J., Evans, R., Green, T., Pritzel, A., Ronneberger, O., Willmore, L., Ballard, A. J., Bambrick, J., et al. Accurate structure prediction of biomolecular interactions with alphafold 3. *Nature*, 630 (8016):493–500, 2024.

Clifford, J. N., Høie, M. H., Deleuran, S., Peters, B., Nielsen, M., and Marcatili, P. Bepipred-3.0: Improved b-cell epitope prediction using protein language models. *Protein Science*, 31(12):e4497, 2022.

Dang, X., Guelen, L., Lutje Hulsik, D., Ermakov, G., Hsieh, E. J., Kreijtz, J., Stammen-Vogelzangs, J., Lodewijks, I., Bertens, A., Bramer, A., et al. Epitope mapping of monoclonal antibodies: a comprehensive comparison of different technologies. In *MAbs*, volume 15, pp. 2285285. Taylor & Francis, 2023.

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

Høie, M. H., Gade, F. S., Johansen, J. M., Würtzen, C., Winther, O., Nielsen, M., and Marcatili, P. Discotope-3.0: improved b-cell epitope prediction using inverse folding latent representations. *Frontiers in immunology*, 15:1322712, 2024.

Jespersen, M. C., Mahajan, S., Peters, B., Nielsen, M., and Marcatili, P. Antibody specific b-cell epitope predictions: leveraging information from antibody-antigen protein complexes. *Frontiers in immunology*, 10:298, 2019.

Kenlay, H., Dreyer, F. A., Kovaltsuk, A., Miketa, D., Pires, D., and Deane, C. M. Large scale paired antibody language models. *PLOS Computational Biology*, 20(12): e1012646, 2024.

Lee, H., Castro, K., Renwick, S., Stalder, L., Glanzer, W., Kumar, R., Chen, N., Scheck, A., Yermanos, A., Mason, D., et al. Contrastive learning for antibody-antigen sequence-to-specificity prediction. *bioRxiv*, pp. 2026–02, 2026.

Lin, Z., Akin, H., Rao, R., Hie, B., Zhu, Z., Lu, W., Smetanin, N., Verkuil, R., Kabeli, O., Shmueli, Y., et al. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637): 1123–1130, 2023.

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

Oord, A. v. d., Li, Y., and Vinyals, O. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Parkinson, J. and Wang, W. For antibody sequence generative modeling, mixture models may be all you need. *Bioinformatics*, 40(5):btae278, 2024.

Passaro, S., Corso, G., Wohlwend, J., Reveiz, M., Thaler, S., Somnath, V. R., Getz, N., Portnoi, T., Roy, J., Stark, H., et al. Boltz-2: Towards accurate and efficient binding affinity prediction. *BioRxiv*, 2025.

Pedotti, M., Simonelli, L., Livoti, E., and Varani, L. Computational docking of antibody-antigen complexes, opportunities and pitfalls illustrated by influenza hemagglutinin. *International journal of molecular sciences*, 12(1):226–251, 2011.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

Sela-Culang, I., Kunik, V., and Ofran, Y. The structural basis of antibody-antigen recognition. *Frontiers in immunology*, 4:302, 2013.

Steinegger, M. and Söding, J. Mmseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nature biotechnology*, 35(11):1026–1028, 2017.

Tien, M. Z., Meyer, A. G., Sydykova, D. K., Spielman, S. J., and Wilke, C. O. Maximum allowed solvent accessibilites of residues in proteins. *PloS one*, 8(11):e80635, 2013.

Zhou, Y., Liu, W., Huang, Z., Gou, Y., Liu, S., Jiang, L., Yang, Y., and Huang, J. A comprehensive antigen-antibody complex database unlocking insights into interaction interface. *Elife*, 14:RP104934, 2025.

A. Model Architecture

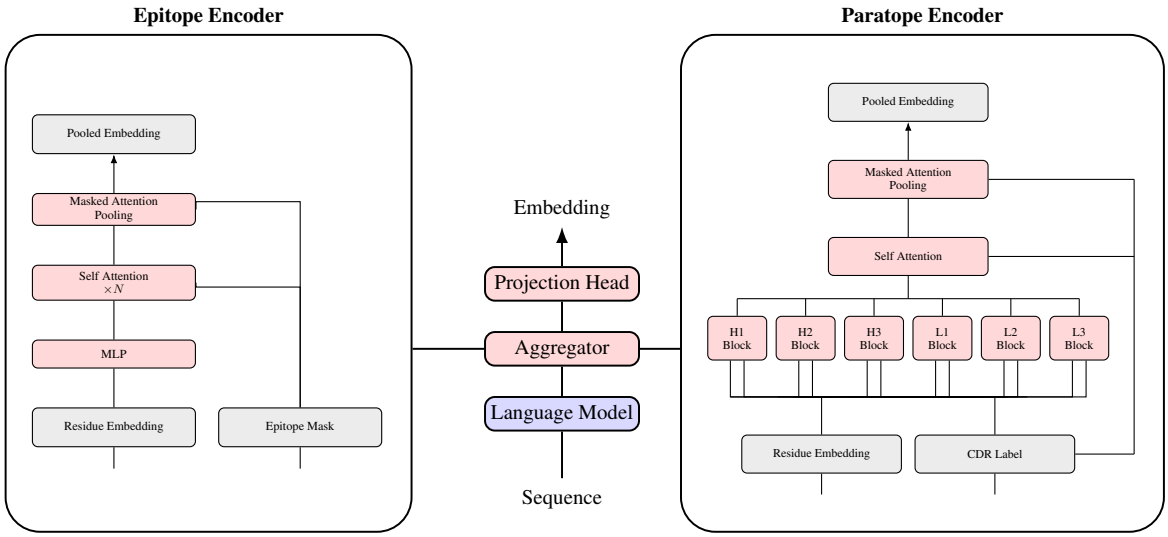

Figure 3. **EpiCLIP architecture.** A frozen antigen and antibody backbone first produce residue-level representations, which are then summarized by candidate-conditioned epitope and paratope aggregation modules and projected into a shared contrastive embedding space. The same latent space supports both binder retrieval and epitope mapping.

Dual Encoder Architecture EpiCLIP uses separate epitope and paratope encoders to map approximate interface candidates into a shared retrieval space. On both sides, a frozen pretrained residue-level backbone first produces contextual residue embeddings, after which a lightweight trainable aggregation module summarizes the candidate region into a single vector and a projection head maps that vector into the shared embedding space. Concretely, for an epitope candidate (x, m) and a paratope candidate (y, c), the model computes

$$H_{\text{epi}} = B_{\text{epi}}(x), \qquad H_{\text{para}} = B_{\text{para}}(y), \tag{10}$$

$$z_{\text{epi}} = A_{\text{epi}}(H_{\text{epi}}, m), \qquad z_{\text{para}} = A_{\text{para}}(H_{\text{para}}, c), \tag{11}$$

$$e = \frac{P_{\text{epi}}(z_{\text{epi}})}{\|P_{\text{epi}}(z_{\text{epi}})\|_2}, \qquad p = \frac{P_{\text{para}}(z_{\text{para}})}{\|P_{\text{para}}(z_{\text{para}})\|_2}, \tag{12}$$

where B_{epi} and B_{para} denote the frozen antigen and antibody backbones, A_{epi} and A_{para} denote the candidate-conditioned aggregation modules, and P_{epi} and P_{para} are projection heads into the shared contrastive space. The architectural differences between the two sides arise entirely in the aggregation modules: epitopes are represented as masked antigen patches, whereas paratopes are represented through the six annotated CDR loops.

The epitope and paratope encoders differ in how they aggregate candidate residues, but both reuse the same masked attention pooling operator as a basic trainable summarization primitive. We therefore define this shared operator first, and then describe how it is composed into the epitope and paratope aggregation modules.

Masked Attention Pooling Let $H \in \mathbb{R}^{L\times d}$ denote a sequence of residue embeddings, where $H = [h_1, \dots, h_L]$ and each $h_i \in \mathbb{R}^d$, and let $m \in \{0,1\}^L$ be a binary mask indicating which residues belong to the candidate region of interest. The masked attention pooling block $z = \mathrm{MAP}(H, m)$ summarizes the variable-length sequence H into a single fixed-dimensional vector z by learning a scalar importance score for each residue and normalizing these scores only over the masked positions.

We first compute an unnormalized attention score for each residue using a learned scoring network $a : \mathbb{R}^d \to \mathbb{R}$, implemented as a small multilayer perceptron:

$$s_i = a(h_i), \qquad i = 1, \dots, L. \tag{13}$$

These scores are then converted into attention weights with a mask-aware softmax,

$$\alpha_i = \frac{m_i \exp(s_i)}{\sum_{j=1}^{L} m_j \exp(s_j)}, \tag{14}$$

so that residues with $m_i = 0$ receive zero weight and do not contribute to the final representation. The pooled embedding is given by the weighted sum

$$z = \sum_{i=1}^{L} \alpha_i h_i. \tag{15}$$

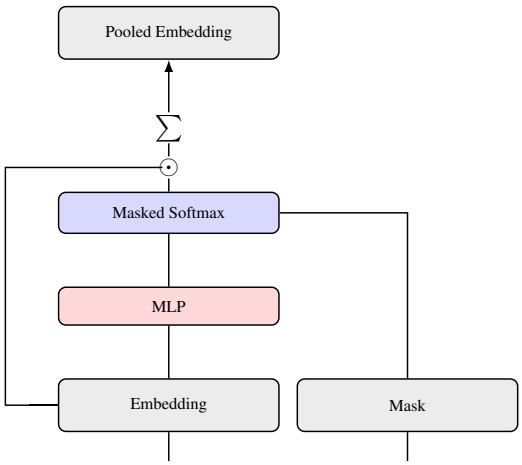

Figure 4. **Masked attention pooling.** Residue embeddings are scored by a small MLP, normalized with a mask-aware softmax over valid positions, and combined into a single pooled embedding. This operator is reused as a trainable summarization primitive in both the epitope and paratope aggregation modules.

Thus, masked attention pooling can be interpreted as a learned soft selection over the residues inside the candidate region, while enforcing hard exclusion of residues outside the mask. In contrast to uniform average pooling, this allows the model to emphasize the most informative residues within the proposed interface while preserving the requirement that the output depend only on the masked subset.

Bottleneck Self-Attention Epitope Aggregator Let $H \in \mathbb{R}^{L\times d_{\mathrm{res}}}$ denote the residue-level backbone embeddings for an antigen of length L, and let $m \in \{0,1\}^L$ denote the candidate epitope mask. A natural way to model interactions within the proposed epitope would be to apply self-attention directly to the backbone embeddings. However, the trainable projections in a standard self-attention block scale quadratically with the hidden dimension, i.e. as $O(d_{\mathrm{res}}^2)$ per block. In our low-data regime, this can make interaction modeling unnecessarily parameter-heavy. To address this, we first project the residue embeddings into a lower-dimensional bottleneck space $d_{\mathrm{b}} \ll d_{\mathrm{res}}$, apply self-attention in that space, and then summarize the resulting masked sequence with masked attention pooling. This reduces the parameter scale of each attention block from $O(d_{\mathrm{res}}^2)$ to $O(d_{\mathrm{b}}^2)$, while still allowing the model to capture interactions among residues within the candidate epitope.

Formally, we first apply a learned linear projection

$$Z^{(0)} = HW_{\mathrm{b}} + \mathbf{1}b_{\mathrm{b}}^{\top}, \tag{16}$$

where $W_{\mathrm{b}} \in \mathbb{R}^{d_{\mathrm{res}}\times d_{\mathrm{b}}}$, $b_{\mathrm{b}} \in \mathbb{R}^{d_{\mathrm{b}}}$, and $Z^{(0)} \in \mathbb{R}^{L\times d_{\mathrm{b}}}$.

We then apply a stack of N_{epi} self-attention blocks in the bottleneck space:

$$Z^{(\ell)} = T_\ell\Big(Z^{(\ell-1)}, m\Big), \qquad \ell = 1, \dots, N_{\mathrm{epi}}, \tag{17}$$

where each T_ℓ denotes a transformer-style self-attention block operating under the mask m. In particular, residues with $m_i = 0$ are excluded from attention, so that contextualization occurs only within the proposed epitope.

Finally, the output of the attention stack is summarized using the masked attention pooling operator defined above:

$$e_{\mathrm{epi}} = \mathrm{MAP}\Big(Z^{(N_{\mathrm{epi}})}, m\Big). \tag{18}$$

The resulting vector $e_{\mathrm{epi}} \in \mathbb{R}^{d_{\mathrm{b}}}$ is the aggregated epitope representation.

Hierarchical Paratope Aggregator Let $H^{(\mathrm{H})} \in \mathbb{R}^{L_{\mathrm{H}} \times d_{\mathrm{res}}}$ and $H^{(\mathrm{L})} \in \mathbb{R}^{L_{\mathrm{L}} \times d_{\mathrm{res}}}$ denote the heavy- and light-chain residue embeddings produced by the backbone, and let the six CDR loops be denoted by $\{\mathrm{H1}, \mathrm{H2}, \mathrm{H3}, \mathrm{L1}, \mathrm{L2}, \mathrm{L3}\}$. The key inductive bias of the paratope aggregator is that the antibody binding surface is largely determined by interactions among these six loops. Rather than applying self-attention over the full heavy-light residue axis, which would scale quadratically in the total sequence length, we first summarize each loop independently and then model interactions only among the resulting six loop embeddings. This yields a fixed six-token bottleneck, reducing cross-loop interaction modeling from $O((L_{\mathrm{H}} + L_{\mathrm{L}})^2)$ to $O(1)$, i.e. $6^2 = 36$.

For each loop $\ell \in \{\mathrm{H1}, \mathrm{H2}, \mathrm{H3}, \mathrm{L1}, \mathrm{L2}, \mathrm{L3}\}$, let m_ℓ denote the corresponding binary mask over the appropriate chain. We compute a loop-level embedding by applying the same masked attention pooling operator to each loop:

$$u_\ell = \begin{cases} \mathrm{MAP}\big(H^{(\mathrm{H})}, m_\ell\big), & \ell \in \{\mathrm{H1}, \mathrm{H2}, \mathrm{H3}\}, \\ \mathrm{MAP}\big(H^{(\mathrm{L})}, m_\ell\big), & \ell \in \{\mathrm{L1}, \mathrm{L2}, \mathrm{L3}\}. \end{cases} \tag{19}$$

Here the same pooling module is reused across all six loops, so the model learns a shared within-loop summarization rule.

We then stack the six loop embeddings into a short sequence

$$U^{(0)} = \begin{bmatrix} u_{\mathrm{H1}} \\ u_{\mathrm{H2}} \\ u_{\mathrm{H3}} \\ u_{\mathrm{L1}} \\ u_{\mathrm{L2}} \\ u_{\mathrm{L3}} \end{bmatrix} \in \mathbb{R}^{6 \times d_{\mathrm{loop}}}, \tag{20}$$

together with a loop-presence mask $q \in \{0, 1\}^6$, where $q_\ell = 1$ if loop ℓ contains at least one residue and $q_\ell = 0$ otherwise.

To model dependencies among loops, we apply a small self-attention stack over this six-token sequence:

$$U^{(k)} = S_k\Big(U^{(k-1)}, q\Big), \qquad k = 1, \ldots, N_{\mathrm{para}}, \tag{21}$$

where each S_k denotes a mask-aware self-attention block operating over the loop axis. Finally, the contextualized loop sequence is summarized with masked attention pooling:

$$e_{\mathrm{para}} = \mathrm{MAP}\Big(U^{(N_{\mathrm{para}})}, q\Big). \tag{22}$$

The resulting vector $e_{\mathrm{para}} \in \mathbb{R}^{d_{\mathrm{loop}}}$ is the final paratope representation. In this way, the hierarchical aggregator separates *within-loop summarization* from *across-loop interaction modeling*, yielding a compact architecture that is both biologically structured and computationally efficient.

Implementation Choices Table 1 summarizes the default model choices used for the EpiCLIP experiments.

Table 1. **Architecture choices for EpiCLIP.** Model-defining settings for the main experimental configuration. Training hyperparameters are reported separately Appendix D

Component	Setting
Antigen backbone	ESM-2 (650M) (Lin et al., 2023), frozen
Antibody backbone	IgBert (Kenlay et al., 2024), frozen
Epitope aggregator	Bottleneck self-attention
Epitope bottleneck dimension	256
Epitope attention layers	1
Paratope aggregator	Hierarchical (self-attention-then-pool)
Paratope loop dimension	256
Shared projection dimension	128

B. Synthetic Epitope Generation

Synthetic epitopes are used for EpiCLIP training and inference. To prevent leakage, in our experiments all synthetic epitope generation stages operate on antigen structures predicted from sequence alone, so the folding model, residue prior, and geometric sampler are applied without access to antibody-conditioned holo complex geometry.

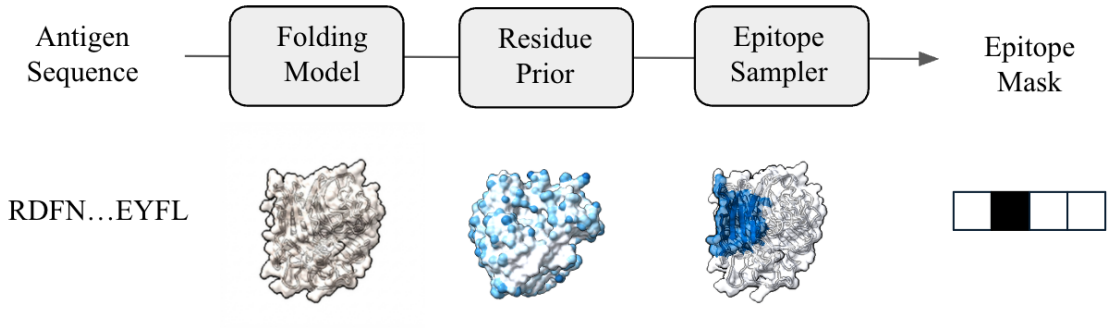

Figure 5. **Synthetic epitope generation pipeline**. Starting from antigen sequence alone, we first predict an antigen structure, then compute an antibody-agnostic residue-level prior, and finally construct candidate epitope masks with the geometric sampler. Because all stages operate on predicted antigen structures rather than resolved holo complexes, candidate generation avoids leakage from antibody-conditioned complex geometry.

B.1. Factorization of the Mask Distribution

Our goal is to generate candidate epitope masks m for an antigen x from a distribution over plausible surface regions, $m \sim P(m \mid x)$. Following Eq. 6, we factor this distribution through a latent residue-level score vector $s \in \mathbb{R}^{L_x}$:

$$P(m \mid x) = \int P(m \mid s, x)\, P(s \mid x)\, ds.$$

Here, $P(s \mid x)$ is a residue-level prior over plausible interaction sites, and $P(m \mid s, x)$ is a geometric distribution over coherent residue patches conditioned on that prior. In our experiments, the predictor is deterministic: an RSA-based function produces a single score vector $\hat{s}(x)$. This corresponds to a point-mass prior

$$P(s \mid x) = \delta\big(s - \hat{s}(x)\big), \tag{23}$$

under which the marginal mask distribution reduces to

$$P(m \mid x) = \int P(m \mid s, x)\, \delta\big(s - \hat{s}(x)\big)\, ds = P(m \mid \hat{s}(x), x). \tag{24}$$

Thus, candidate generation in our setting is fully determined by a conditional geometric sampler driven by the predicted residue-level prior. For a candidate mask m, we define the unnormalized patch-level score

$$q(m; \hat{s}) = \mathrm{median}(\{\hat{s}_j : m_j = 1\})\,, \tag{25}$$

and use $q(m; \hat{s})$ as an unnormalized proxy for $P(m \mid \hat{s}(x), x)$ when ranking candidate patches produced by the sampler.

B.2. RSA Predictor

For the residue-level predictor $P(s \mid x)$, we use relative solvent accessibility (RSA) (Tien et al., 2013) as a simple antibody-agnostic prior over plausible interaction residues. RSA measures how exposed each antigen residue is to solvent, providing a weak structural signal for whether that residue lies on the protein surface and is therefore available to participate in binding. In our setting, the predictor outputs a scalar score s_i for each residue i, which is then used only to bias candidate generation toward exposed surface regions. We deliberately chose RSA rather than a stronger learned epitope predictor, since the goal of $P(s \mid x)$ is not to solve epitope mapping directly, but to provide a weak biophysical prior for sampling plausible candidate patches. This reduces the risk of task leakage while still constraining the combinatorial search space to regions that could realistically support protein-protein interactions.

B.3. Geometric Sampler

At a high level, the geometric sampler constructs a finite set of spatially coherent candidate masks $M(x) = \{m_1, \ldots, m_K\}$ on the antigen structure, conditioned on the residue-level RSA scores $\hat{s}(x)$. More formally, under the deterministic predictor described above, synthetic epitope generation reduces to constructing candidate masks under the conditional distribution $P(m \mid \hat{s}(x), x)$. Operationally, we approximate this distribution with a scored finite proposal pool $\mathcal{M}_{\text{prop}}$ of geometric patches centered at high-priority residues, where each proposal is assigned an unnormalized score $q(m; \hat{s})$. The final candidate set $M(x)$ is then obtained by selecting from this pool, either through diversity-aware clustering or direct score-based ranking.

Proposal Pool The first stage of the geometric sampler constructs a proposal pool $\mathcal{M}_{\text{prop}}$ by centering local patches at high-scoring residues under the predictor prior. Starting from the residue-level score vector $\hat{s}(x)$, we identify hotspot residues whose scores exceed a threshold τ, and around each hotspot we form a spherical patch on the antigen structure. Each patch is assigned a patch-level score $q(m; \hat{s})$, yielding a scored finite proposal pool of candidate epitopes.

Algorithm 1 Geometric proposal pool generation

Require: Antigen C_α coordinates $X = (x_1, \ldots, x_{L_x}) \in \mathbb{R}^{L_x \times 3}$, residue scores $\hat{s} \in \mathbb{R}^{L_x}$, hotspot threshold τ, patch radius r, optional patch cap k
Ensure: Scored proposal pool $\mathcal{M}_{\text{prop}}$

```
H ← {i ∈ {1, ..., L_x} : ŝ_i ≥ τ}                          // hotspot anchors
M_prop ← ∅
for all i ∈ H do
    m^(i) ← Ball(x_i; X, r, k)                              // local spherical patch
    q(m^(i); ŝ) ← median({ŝ_j : m_j^(i) = 1})               // patch score
    M_prop ← M_prop ∪ {m^(i)}
end for
return SortByScore(M_prop, q(·; ŝ))                         // descending patch score
```

In Algorithm 1, each residue is represented by a single 3D point in the coordinate matrix X, such that the $\text{Ball}(x_i; X, r, k)$ operator returns the binary mask of residues whose C_α coordinates lie within Euclidean distance r of anchor residue i, optionally truncated to the k nearest residues when a patch-size cap is used.

Candidate Selection The second stage of the geometric sampler converts the scored proposal pool $\mathcal{M}_{\text{prop}}$ into a final candidate set $M(x)$. When diversity selection is enabled, highly overlapping proposal masks are clustered together and only one representative is retained from each cluster. When diversity selection is disabled, candidates are returned directly in descending order of their patch scores. In both cases, the goal is to produce a compact candidate set that preserves high-scoring antigen regions while avoiding unnecessary redundancy.

In Algorithm 2, proposal masks are compared using Dice distance,

$$d_{\text{Dice}}(m, m') = 1 - \frac{2|m \cap m'|}{|m| + |m'|}, \tag{26}$$

and clustered by complete-linkage hierarchical agglomerative clustering. We use Dice here because candidate selection is a symmetric redundancy-reduction problem between proposal masks, with neither side playing a privileged role. The operator $\text{HAC}(\mathcal{M}_{\text{prop}}, d_{\text{Dice}}, \cdot)$ takes as input the proposal pool, the pairwise Dice distance, and either a target number of clusters K or a distance threshold t_{div}, and returns the corresponding cluster family $\mathcal{C}$. Thus, $M(x)$ is a selected subset of the scored proposal pool $\mathcal{M}_{\text{prop}}$, obtained either by diversity-aware clustering or by direct score-based ranking.

B.4. Training and Inference Sampling

Both training and inference begin from the same scored proposal pool $\mathcal{M}_{\text{prop}}$, and differ only in how those proposals are filtered and selected. During training, each candidate mask $m \in \mathcal{M}_{\text{prop}}$ is compared to the ground-truth epitope E using

Algorithm 2 Candidate selection from the proposal pool

Require: Scored proposal pool $\mathcal{M}_{\text{prop}}$, diversity flag `select_diverse`, target number of clusters K or diversity threshold t_{div}
Ensure: Final candidate set $M(x)$

```
if select_diverse = false then
    return SortByScore(M_prop, q(·; ŝ))                    // optionally truncate to top-K
end if
if K is specified then
    C ← HAC(M_prop, d_Dice, K)                             // cut into K clusters
else
    C ← HAC(M_prop, d_Dice, t_div)                         // cut by Dice threshold
end if
M(x) ← ∅
for all C ∈ C do
    m*_C ← arg max_{m∈C} q(m; ŝ)                           // highest-scoring representative
    M(x) ← M(x) ∪ {m*_C}
end for
return SortByScore(M(x), q(·; ŝ))                          // descending patch score
```

the Tversky overlap.

$$\mathrm{Tv}_{\alpha,\beta}(m, E) = \frac{|m \cap E|}{|m \cap E| + \alpha|m \setminus E| + \beta|E \setminus m|}. \tag{27}$$

We use Tversky overlap here rather than Dice because labeling is an asymmetric comparison between a candidate patch and the ground-truth epitope. For epitope mapping, it is generally more important that a candidate cover the full binding region than that it match the exact epitope boundary, since a coarse but covering patch remains biologically informative whereas a patch that omits true binding residues can lose essential interaction context. This asymmetry also matches the encoder design: residues outside the candidate mask are excluded from the model, while moderately oversized candidates can still be refined downstream. This induces three bins,

$$\mathcal{P} = \{m \in \mathcal{M}_{\text{prop}} : \mathrm{Tv}_{\alpha,\beta}(m, E) \geq \tau_+\}, \tag{28}$$

$$\mathcal{N} = \{m \in \mathcal{M}_{\text{prop}} : \mathrm{Tv}_{\alpha,\beta}(m, E) \leq \tau_-\}, \tag{29}$$

$$\mathcal{U} = \mathcal{M}_{\text{prop}} \setminus (\mathcal{P} \cup \mathcal{N}), \tag{30}$$

corresponding to positive, negative, and intermediate-overlap candidates, respectively. Positive training examples are drawn from $\mathcal{P}$, while intra-antigen negatives are drawn from $\mathcal{N}$; intermediate-overlap candidates in $\mathcal{U}$ are discarded to avoid ambiguous supervision. When multiple candidates in a bin are highly redundant, candidate selection is applied within that bin before sampling so that the selected patches cover distinct antigen regions. At inference time, no ground-truth epitope is available, so this overlap-based binning is omitted entirely: the final candidate set $M(x)$ is obtained directly from the scored proposal pool $\mathcal{M}_{\text{prop}}$ using Algorithm 2, and retrieval or epitope mapping is performed over those candidates. In this way, training and inference operate on the same family of approximate epitope regions, differing only in whether ground-truth overlap is available for labeling.

C. Data and Preprocessing

The training and evaluation data were derived from the Antibody-Antigen Complex Database (AACDB) (Zhou et al., 2025). We retain only entries with a single antigen chain and paired heavy- and light-chain antibody sequences. Antibody variable regions are extracted and AHo-numbered with AntPack (Parkinson & Wang, 2024), and converted into the six-loop paratope annotation used by EpiCLIP. Ground-truth antigen epitope masks and heavy/light paratope masks are derived from resolved heavy-atom contacts, with a distance threshold of 5.5Å. For synthetic epitope generation, antigen structures are predicted from sequence alone with ESMFold (Lin et al., 2023), and all downstream candidate-generation steps operate on these predicted antigen structures rather than resolved holo complexes. To obtain leakage-controlled train/validation/test splits, we cluster antigens with MMseqs2 (Steinegger & Söding, 2017) at 40% sequence identity and assign entire antigen clusters,

rather than individual complexes, to the three splits. Cluster assignment is optimized to match the target 50/25/25 split while also requiring each split to contain at least 10% of the assigned antigen clusters. This yields 3332 curated complexes across 542 antigen clusters, with no single antigen cluster accounting for more than 20% of any split.

D. Training Methodology

Training Configuration Model-defining settings for EpiCLIP are summarized in Table 1. We train EpiCLIP with the bidirectional InfoNCE objective described in Subsection 4.3, using a batch size of 8 antibody-antigen complexes. This batch size is defined at the complex level; the total number of embedded objects per batch is larger, since each complex contributes one paratope candidate together with a variable number of synthetic epitope candidates as described below. Optimization uses AdamW with learning rate 10^{-4} and weight decay 10^{-2} for model parameters, while the learnable loss scalars are optimized separately with learning rate 10^{-5} and zero weight decay. We initialize the InfoNCE temperature to 0.1 and weight the paratope-to-epitope and epitope-to-paratope directions equally with $\lambda = 0.5$. Training uses a warmup-cosine learning-rate schedule with 5% warmup, runs for at most 15 epochs, and applies gradient clipping with norm 5.0. We save checkpoints by both validation loss and validation blind-binder performance, and use the latter directly for model selection. EpiCLIP was implemented in PyTorch and PyTorch Lightning and trained on a single NVIDIA L40S GPU, with training completing in less than 10 hours.

Synthetic Supervision Configuration For both training and inference, synthetic epitope candidates are generated using the geometric proposal family described above in Subsection B.3 with patch radius $r = 15.0$Å. During training, we use a hotspot threshold of 0.1 so that negative candidates are drawn from structurally plausible surface regions rather than arbitrary low-score patches. Candidate masks are then binned against the ground-truth epitope using the Tversky overlap from Eq. 27 with $\alpha = 0$ and $\beta = 1$, together with positive and negative thresholds of $\tau_+ = 0.7$ and $\tau_- = 0.5$, respectively. Under this choice, the Tversky score reduces to epitope coverage, i.e. the fraction of the ground-truth epitope captured by the candidate mask, matching the asymmetric motivation described in Subsection B.4. From this binned proposal pool, we select one positive synthetic epitope as the highest-overlap candidate and eight intra-antigen negatives using diversity-aware selection from the low-overlap bin.

Baseline Training Baseline models are trained with the same bidirectional InfoNCE objective as EpiCLIP, using the same frozen residue-level backbones, projection heads, and training hyperparameters. For the sequence-only CLIP baseline, the antigen and antibody are each represented as a single full-sequence candidate, which can be viewed as using all-ones masks over the corresponding inputs. For the exact-interface CLIP baseline, the antigen and antibody are represented by their resolved epitope and paratope interface annotations. In both cases, training operates on one antigen candidate and one antibody candidate per complex, so these baselines do not use synthetic epitope sampling or intra-antigen decoys. On both the antigen and antibody sides, we use simple mean pooling rather than the EpiCLIP aggregation modules. This choice is partly architectural, since the hierarchical paratope aggregator assumes CDR-structured inputs and is therefore not applicable to arbitrary full-sequence or interface masks, and partly comparative, since mean pooling more closely matches the design choices used by CALM (Lee et al., 2026).

E. Evaluation Methodology

All evaluation is performed on the held-out test split described in Section C, using the deployment tasks introduced in Section 1 and formalized in Section 3. We evaluate each model directly on binder retrieval and epitope mapping, using the same candidate-based inference setting that EpiCLIP is designed to support at deployment.

Retrieval Protocols Binder retrieval is evaluated by ranking antibodies from the held-out antibody library against a target antigen. For the sequence-only CLIP baseline, retrieval is performed using full antigen and antibody sequence representations. For the exact-interface CLIP baseline, guided retrieval is performed using ground-truth epitope E and ground-truth paratope P annotations, since this model is trained directly on exact interface masks. For EpiCLIP, antibody candidates are always represented by their CDR-defined paratope candidates, while antigen candidates are drawn from the same synthetic proposal family used during training. In the guided setting, the target epitope m_{target} is chosen as the synthetic candidate with highest Tversky overlap to the ground-truth epitope E, using the same (α, β) parameters as in training. In the blind setting, EpiCLIP scores each antibody by its best-matching candidate region in the antigen candidate set $M(x)$, as in Eq. 5. At inference, we retain up to 64 synthetic candidates per antigen for binder retrieval. Thus, each

model is evaluated on the type of input it was trained to consume, avoiding unnecessary out-of-distribution inference settings while preserving a fair comparison of retrieval behavior. This also highlights a key distinction between model families: EpiCLIP is the only method that natively supports both guided and blind binder retrieval over local approximate antigen regions.

Epitope Mapping Epitope mapping is evaluated only in the unguided setting, since the goal is precisely to localize the binding region on the antigen. For a fixed antigen-antibody pair, the retrieval set consists of synthetic epitope candidates from the same proposal family used elsewhere in EpiCLIP, as described in Section B; in our experiments, we retain up to 256 diverse candidate patches per antigen so that the candidate set covers distinct regions of the antigen surface. The true epitope E is not included directly in this retrieval set, because at deployment the exact binding site is unknown and epitope mapping must operate over approximate candidate regions rather than resolved interfaces. The antibody-side query differs by model family: the sequence-only CLIP baseline uses the full antibody sequence representation, while the exact-interface CLIP baseline uses the exact paratope representation, and EpiCLIP uses the CDR-defined paratope candidate. The correct target epitope is defined as the synthetic candidate m_{target} with highest Tversky overlap to the ground-truth epitope E, using the same overlap definition as in guided binder retrieval. Candidate patches are then ranked by their CLIP score with the antibody-side query, and performance is measured by how highly m_{target} is ranked relative to the other synthetic candidates on the same antigen.

Metrics All reported metrics are computed from the ranked retrieval lists produced by the evaluation protocols above. For a given query, let the rank of the correct target be r and let N denote the size of the corresponding retrieval set. Coverage is defined as the fraction of the ranked list examined, $c = k/N$, where k is a prefix length, and success rate at coverage c is the fraction of queries for which the correct target appears within the top-k positions. The main-text curves in Figure 2 therefore plot success rate as a function of coverage, summarizing retrieval quality across the full ranking rather than at a single cutoff. In the appendix tables, we additionally report percentile rank together with a set of top-of-list metrics: mean reciprocal rank (MRR) and recall at fixed cutoffs (Tables 2 and 3). Percentile rank normalizes the target rank by the size of the retrieval set and therefore reflects aggregate ranking quality across tasks with different candidate-set sizes. By contrast, MRR and $R@K$ place greater weight on whether the correct target appears near the top of the list, making them especially useful for practical screening settings where only the highest-ranked candidates are likely to be inspected. Together, these metrics provide both a global view of ranking quality and a more top-heavy view of early retrieval performance.

F. Extended Results

	Percentile Rank		Top-of-List Metrics		
Method	Mean	Median	MRR	R@1	R@10
CLIP (seq), unguided	0.621	0.671	0.032	0.001	<u>0.077</u>
CLIP (interface), guided	<u>0.693</u>	**0.766**	0.036	0.006	**0.079**
EpiCLIP, guided	0.686	0.740	<u>0.039</u>	<u>0.010</u>	0.066
EpiCLIP, unguided	**0.696**	<u>0.749</u>	**0.039**	**0.012**	0.054

Table 2. **Binder retrieval.** Extended retrieval metrics. In each column, the best result is shown in **bold** and the second-best is <u>underlined</u>.

	Percentile Rank		Top-of-List Metrics		
Method	Mean	Median	MRR	R@1	R@10
CLIP (seq)	0.516	0.522	0.049	0.014	0.090
CLIP (interface)	<u>0.524</u>	<u>0.529</u>	<u>0.054</u>	<u>0.018</u>	<u>0.091</u>
EpiCLIP	**0.656**	**0.695**	**0.120**	**0.052**	**0.241**

Table 3. **Epitope mapping.** Extended retrieval metrics. In each column, the best result is shown in **bold** and the second-best is <u>underlined</u>.