bio-protocol

Open Access

Published: Sep 05, 2026

Humanizing Antibodies and Nanobodies From Scratch With HuDiff

Yang Nan[1, #], Hongshuai Sun[1, #], Bo Zhang[1], Yue Yang[3], Jianxin He[2, *] and Qifeng Bai[1, *]

[1]School of Basic Medical Sciences, Lanzhou University, Lanzhou 730000, China
[2]College of Basic Medicine, Gansu University of Chinese Medicine, Lanzhou, China
[3]Key Laboratory of Environmental Ecology and Population Health in Northwest Minority Areas, State Ethnic Affairs Commission, Northwest Minzu University, No.1 Xibeixincun, Chengguan District, Lanzhou, Gansu 730030, China
*For correspondence: baiqf@lzu.edu.cn; hejx4663@gszy.edu.cn
[#]Contributed equally to this work

Abstract

Antibody (Ab) and nanobody (Nb) humanization is essential for reducing immunogenicity in therapeutic applications. HuDiff is an adaptive autoregressive diffusion approach that generates humanized antibodies and nanobodies from scratch using only complementarity-determining region sequences as input, eliminating the need for preexisting human templates. The method follows a two-stage training pipeline: pretraining on human antibody sequences to learn framework region patterns, followed by fine-tuning on target-species sequences. HuDiff-Ab processes paired heavy and light chains for conventional antibodies, while HuDiff-Nb can incorporate a specialized inpainting mode to preserve critical nanobody framework residues. This protocol provides a complete step-by-step guide for implementing HuDiff, covering data preparation, model training, and sequence generation.

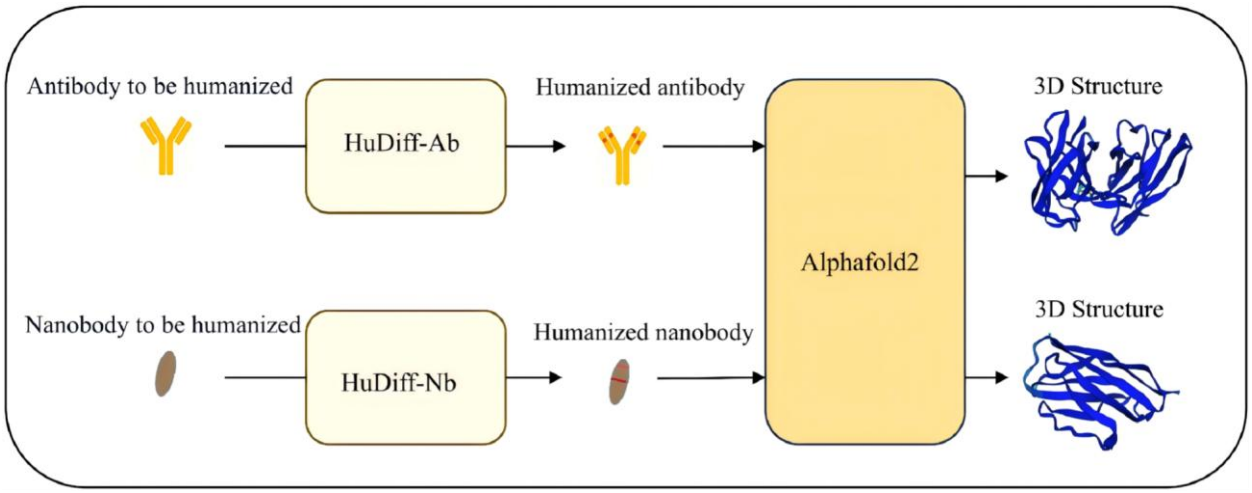
Key features

- Requires only CDR sequences as input and does not require human template selection.
- Uses a two-stage training strategy, with pretraining on human antibody sequences and fine-tuning on target-species sequences guided by humanness scores.
- HuDiff-Ab humanizes paired heavy and light chains simultaneously, whereas HuDiff-Nb provides an inpainting mode to preserve key framework residues.
- Generates multiple diverse humanized candidates for downstream experimental screening.

Keywords: Humanization, Antibody, Nanobody, Deep learning, Protein design, Autoregressive diffusion model

This protocol is used in: Nature Machine Intelligence (2025), DOI: 10.1038/s42256-025-01120-9

Graphical overview



Graphical overview of the humanization and 3D structure prediction pipeline. The antibody and nanobody are humanized using HuDiff-Ab and HuDiff-Nb, respectively. Both resulting humanized constructs are modeled by AlphaFold2 to obtain their 3D structures. Antibodies and nanobodies are processed by their corresponding models to generate humanized antibodies and nanobodies. The humanized portions are the red parts in the middle of the antibody and nanobody diagrams. Then, AlphaFold is used to generate their 3D structures.

Background

Monoclonal antibodies and nanobodies derived from non-human sources, such as rodents and camelids, are widely used in research and therapy due to their high specificity and affinity. However, their clinical application is often limited by immunogenicity, as the human immune system may recognize these foreign proteins and trigger anti-drug antibody responses, such as human anti-mouse antibody (HAMA) reaction [1]. Such responses can reduce drug efficacy and cause adverse effects, highlighting the necessity of reducing immunogenicity through humanization while preserving antigen-binding activity.

Traditional humanization methods, particularly complementarity-determining region (CDR) grafting, involve transplanting the CDRs from a non-human antibody onto a human framework template. This approach is labor-intensive and often requires back-mutations to maintain structural integrity and binding affinity [2]. The success of this strategy depends strongly on the availability of suitable human frameworks and, in many cases, prior structural knowledge. Compared to template-based methods (e.g., CDR grafting), HuDiff [3] eliminates manual template selection and back-mutations. Unlike structure-based approaches (e.g., AbAdapt), HuDiff does not require 3D structures. Compared to sequence-based tools like Sapiens or AbNatiV, HuDiff generates full sequences rather than scoring existing ones, enabling de novo humanization.

Computational approaches have been developed to address these limitations. Structure-based methods rely on three-dimensional models to guide residue selection but are computationally demanding. More recent machine learning tools such as Sapiens, Llamanade, and AbNatiV leverage large-scale sequence data to predict humanness and guide humanization [4]. While these methods improve efficiency, many still depend on preexisting human frameworks or require structural modeling. Nanobodies, the variable domains of camelid heavy-chain antibodies, offer advantages including small size and high stability [5]. However, their humanization presents additional challenges, as certain framework residues critical for stability must be preserved.

HuDiff is a template-free generative approach based on autoregressive diffusion models. Without requiring human templates, it generates humanized antibodies or nanobodies by reconstructing humanized framework regions using only CDR sequences as input [3]. Here, we provide a step-by-step protocol for humanizing antibodies and nanobodies using HuDiff.

Software and datasets

Type	Software/dataset/resource	Version	Date	License	Access
Software	Ubuntu	22.04 LTS	2022	Open-source	Free
Software	Python	3.9	2020	Open-source	Free
Software	PyTorch	1.13.0	2022	BSD-3-Clause	Free
Software	CUDA Toolkit	11.6	2022	NVIDIA End User License Agreement	Free
Software	abnumber	1.0.0	2024	MIT	Free
Software	pandas	1.5.3	2022	BSD-3-Clause	Free
Software	scikit-learn	1.2.0	2022	BSD-3-Clause	Free
Software	HuDiff source code repository	V1	2025	MIT	Free (https://github.com/TencentAI4S/HuDiff) (https://doi.org/10.5281/zenodo.16974296)
Software	AlphaFold2 (ColabFold)	2.3.0	2021	Apache-2.0	Free
Software	pymol-open-source	2.5	—	BSD-3-Clause	Free
Software	AbNatiV	1.0	2024	GPL-3.0	Free (https://gitlab.developers.cam.ac.uk/ch/sormanni/abnativ)
Dataset	HuDiff release dataset	V1	2025-03-15	CC BY 4.0	Free (https://huggingface.co/cloud77/HuDiff/resolve/main/release_data_dir.tar.gz)

Note: AbNatiV installation requires the environment.yml file provided in the GitLab repository.

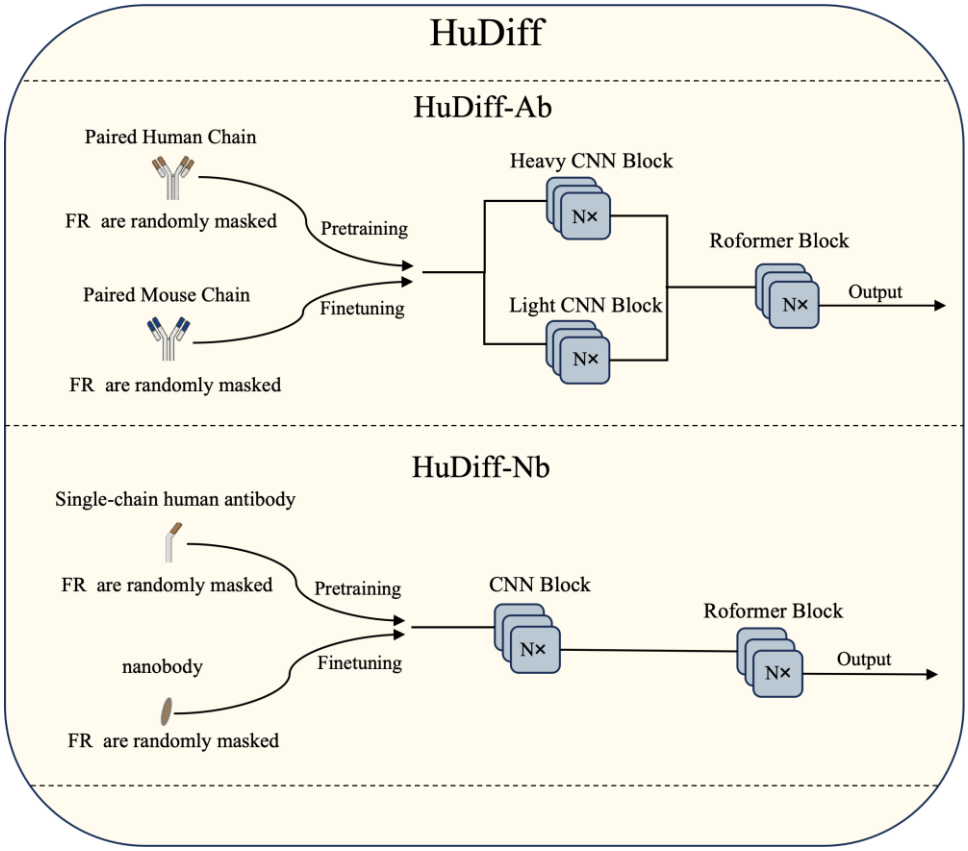


Figure 1. Architecture diagram of HuDiff. HuDiff-Ab architecture: In the pretraining stage, the model receives paired human heavy and light chain sequences with random masking in the framework (FR) regions. These sequences sequentially

pass through dual convolutional neural network (CNN) blocks and RoFormer blocks to output the amino acid probabilities for the masked positions. In the fine-tuning stage, the model processes paired mouse heavy and light chain sequences with random FR masking, similarly utilizing the dual CNN and RoFormer blocks to predict the amino acid probabilities. HuDiff-Nb architecture: In the pretraining stage, the model takes human antibody heavy chain sequences with random FR masking as input. These sequences pass through a single CNN block and a RoFormer block to output the amino acid probabilities for the masked positions. In the fine-tuning stage, the model processes nanobody sequences with random FR masking, employing the same single CNN and RoFormer block architecture to generate the probability outputs. RoFormer: Rotary Transformer. N: number of layers in the neural network.

Procedure

All computational steps should be performed on a workstation equipped with an NVIDIA GPU (e.g., RTX 3060 Ti with 8 GB VRAM). Training requires ≥32 GB RAM; inference requires ≥16 GB RAM. The protocol is written for Ubuntu 22.04 LTS with NVIDIA GPUs. Windows users can use WSL2; macOS users can perform inference on CPU (training not supported). A Docker option is available (start_docker.sh). The entire software environment is managed with Conda to ensure reproducibility. It is strongly recommended to use an integrated development environment (IDE) such as Visual Studio Code (VS Code), PyCharm, or Cursor IDE for code execution and debugging.

GPU and PyTorch/CUDA compatibility:
HuDiff requires PyTorch 1.13.0 with CUDA 11.6. Supported GPUs include any NVIDIA GPU with compute capability ≥7.0 (e.g., V100, T4, RTX 2080/3060/3090, A100). For older GPUs (e.g., GTX 1080, compute capability 6.1), CUDA 11.6 may still work, but performance may degrade. We recommend using a GPU with ≥8 GB VRAM for inference and ≥16 GB VRAM for training. Check your GPU compatibility at NVIDIA CUDA GPUs. If using a different CUDA version, reinstall PyTorch accordingly.

```
#> pip install torch==1.13.0+cu116 --extra-index-url https://download.pytorch.org/whl/cu116
```

Note: "#>" represents a terminal command line where commands are entered.

A. Development, training, and fine-tuning of HuDiff

This section describes the training procedure for the HuDiff model. The model architecture is shown in Figure 1. The training and fine-tuning procedures described are primarily intended for two types of users: those who aim to reproduce the original HuDiff model development process and those who seek to use this workflow as a reference for developing or improving related algorithms.

A1. Installation and environment setup

Critical: All input antibody/nanobody sequences must be IMGT-numbered. Use the abnumber package consistently.

1. Clone the HuDiff repository and navigate into it.

```
#> git clone https://github.com/TencentAI4S/HuDiff.git
#> cd HuDiff
```

2. Create and activate a Conda environment.

```
#> conda env create -f environment.yaml
#> conda activate hudiff
#> conda install -c conda-forge pymol-open-source
```

3. Verify installation. Run a simple test command.

```
#> python -c "import torch; print(torch.__version__)"
```

Script names in this protocol correspond to those in the GitHub repository. If discrepancies arise, refer to the README. If you prefer using Docker, a start_docker.sh script is provided in the repository. Run:

```
#> ./start_docker.sh
```

For more details, see the Docker section in the GitHub README.

A2. Data preparation

1. Download the release data directory. Download the dataset from the HuDiff repository (Hugging Face) using wget or a web browser (https://huggingface.co/cloud77/HuDiff/resolve/main/release_data_dir.tar.gz).

2. Extract and organize the data

```
#> tar -xzf release_data_dir.tar.gz
```

A3. HuDiff-Ab pretraining and fine-tuning

1. Pretraining

a. Execute the pretraining script:

```
#> antibody_scripts/antibody_run.sh
```

Set the config_path, data_path (LMDB file for antibody sequences), and log_path parameters before running the script.

b. Monitor training progress: Training logs are saved to tmp/antibody_pretrain_log/. Checkpoints are saved to tmp/antibody_pretrain_log/checkpoint/.

c. Select the best checkpoint: Use the automatically saved checkpoint with the lowest validation loss. The training script monitors the validation loss and saves the best model automatically. After training completes, locate the best checkpoint (usually saved as best_model.pt) in the checkpoint directory (e.g., tmp/antibody_pretrain_log/checkpoint/). Verify that the checkpoint corresponds to the epoch with the lowest validation loss (check the training logs if needed). Then, copy it to the release directory.

```
#> cp tmp/antibody_pretrain_log/checkpoint/best_model.pt \
release_data_dir/checkpoints/antibody/pretrain_antibody.pt
```

This serves as a critical synchronization point. At this juncture, the algorithm suspends to allow for manual inspection of the intermediate representations before proceeding to the subsequent phase.

2. Finetuning

a. Execute the fine-tuning script:

```
#> antibody_scripts/antibody_finetune.sh
```

Inputs: The best pretrained checkpoint and the fine-tuning dataset for antibodies.

b. Monitor fine-tuning progress: Finetuning logs are saved to tmp/antibody_finetune_log/. Checkpoints are saved to tmp/antibody_finetune_log/checkpoint/.

c. Select and save the best checkpoint: The fine-tuning script (antibody_finetune.sh) is configured to automatically monitor the validation loss and save the checkpoint with the lowest value (best_model.pt). You do not need to manually evaluate checkpoints.

```
#> cp tmp/antibody_finetune_log/checkpoint/best_model.pt \
release_data_dir/checkpoints/antibody/antibody.pt
```

This serves as a critical synchronization point. At this juncture, the algorithm suspends to allow for manual inspection of the intermediate representations before proceeding to the subsequent phase.

A4. HuDiff-Nb pretraining and fine-tuning

1. Pretraining

a. Execute the pretraining script

```
#> nanobody_scripts/nanobody_run.sh
```

Set the config_path, data_path (LMDB file for antibody sequences), and log_path parameters before running the script.

b. Monitor training progress: Training logs are saved to tmp/nanobody_pretrain_log/. Checkpoints are saved to tmp/nanobody_pretrain_log/checkpoint/.

c. Select the best checkpoint: Use the automatically saved checkpoint with the lowest validation loss. The training script monitors the validation loss and saves the best model automatically. After training completes, locate the best checkpoint (usually saved as best_model.pt) in the checkpoint directory (e.g., tmp/nanobody_pretrain_log/checkpoint/). Verify that the checkpoint corresponds to the epoch with the lowest validation loss (check the training logs if needed). Then, copy it to the release directory.

```
#> cp tmp/nanobody_pretrain_log/checkpoint/best_model.pt \
release_data_dir/checkpoints/nanobody/pretrain_nanobody.pt
```

This serves as a critical synchronization point. At this juncture, the algorithm suspends to allow for manual inspection of the intermediate representations before proceeding to the subsequent phase.

2. Finetuning

a. Install AbNatiV. AbNatiV is used as a humanness scorer during fine-tuning. It must be installed separately with its own conda environment to avoid dependency conflicts.

Critical: AbNatiV is only required for HuDiff-Nb fine-tuning, not for antibody humanization.

Clone the AbNatiV repository from GitLab.

```
#> git clone https://gitlab.developers.cam.ac.uk/ch/sormanni/abnativ.git
#> cd abnativ
```

Create a dedicated conda environment using the provided environment.yml.

```
#> conda env create --name abnativ --file environment.yml
#> conda activate abnativ
```

The environment.yml file is included in the AbNatiV GitLab repository and contains all necessary dependencies (Python 3.8, PyTorch, BioPython, etc.).

Verify the installation.

```
#> abnativ --help
```

Note the installation path for later configuration.

```
#> which abnativ
```

Example output: /path/to/anaconda3/envs/abnativ/bin/abnativ.

Configure the path in HuDiff.

Export the AbNatiV path so HuDiff can locate it during fine-tuning.

```
#> export ABNATIV_PATH=/path/to/anaconda3/envs/abnativ/bin/abnativ
```

Alternatively, set the path directly in the nanobody fine-tuning configuration file.
For detailed installation instructions and troubleshooting, refer to https://gitlab.developers.cam.ac.uk/ch/sormanni/abnativ.
After installation, note the path to the abnativ executable (e.g., /path/to/anaconda3/envs/abnativ/bin/abnativ).
Set it in the HuDiff configuration or export ABNATIV_PATH=/path/to/abnativ.
If you encounter any issues, refer to the AbNatiV documentation or the HuDiff troubleshooting section.

b. Execute the fine-tuning script:

```
#> nanobody_scripts/nanofinetune_run.sh
```

Inputs: The best pretrained checkpoint, finetuned nanobody dataset, and inpainting enabled.

c. Monitor fine-tuning progress: Logs are saved to tmp/nanobody_finetune_log/. Checkpoints are saved to tmp/nanobody_finetune_log/checkpoint/.

d. Select and save the best checkpoint: The fine-tuning script (nanofinetune_run.sh) is configured to automatically monitor the validation loss and save the checkpoint with the lowest value (best_model.pt). You do not need to manually evaluate checkpoints.

```
#> cp tmp/nanobody_finetune_log/checkpoint/best_model.pt \
release_data_dir/checkpoints/nanobody/nanobody.pt
```

This serves as a critical synchronization point. At this juncture, the algorithm suspends to allow for manual inspection of the intermediate representations before proceeding to the subsequent phase.
If you have completed the training/fine-tuning steps above, or if you wish to use the pretrained checkpoints directly, proceed to section B for humanization of your target antibodies or nanobodies.

B. Humanization of mouse antibodies and camelid nanobodies using HuDiff

Section B provides step-by-step instructions for humanization using either the checkpoints obtained from section A or the released pretrained checkpoints. All necessary code and pretrained weights are available in the main GitHub repository.

B0. Quick start (inference only)
If you wish to generate humanized sequences without training, skip section A and directly use the released checkpoints. After completing installation (A1) and data download (A2), proceed to B2 (for antibodies) or B3 (for nanobodies). Inference time: ~2 min per sequence on GPU, ~20 min on CPU. If you have already completed A1 and A2, skip B1 and proceed directly to B2 or B3.

B1. Preparation
Before running the humanization scripts, ensure the following steps are completed.
First, clone the repository and set up the environment if not already done:

```
#> git clone https://github.com/TencentAI4S/HuDiff.git
#> cd HuDiff
#> conda env create -f environment.yaml
#> conda activate hudiff
```

Download the release data directory:

```
#> wget https://huggingface.co/cloud77/HuDiff/resolve/main/release_data_dir.tar.gz
#> tar -xzf release_data_dir.tar.gz
```

Place the release_data_dir folder inside the HuDiff/ root directory.
If you have trained your own models in Part A, you may use those checkpoints instead.

B2. Humanization with HuDiff-Ab

Before running, validate your FASTA file with abnumber:

```
#> python -c "from abnumber import Chain; chain=Chain('EVQL...'); print(chain.imgt)"
```

Expected output: The IMGT-numbered sequence with gaps inserted (e.g., 'EVQL...' becomes 'E V Q L ...' with proper spacing according to IMGT numbering). If the sequence is valid and recognized, no error message appears. Example of a valid output:
EVQLVESGGGLVQPGGSLRLSCAASGFTFSSYWMSWVRQAPGKGLEWVANIKQDGSEKYYVDSVKGRFTISRDN
AKNSLYLQMNSLRAEDTAVYYCAR
If the sequence is not IMGT-numberable, an exception is raised (e.g., "NumberingError: Sequence does not match IMGT scheme"). Always ensure your input sequences pass this validation before proceeding.

The antibody humanization script accepts input in two formats: a paired-chain FASTA file containing both heavy and light chain sequences, or separate heavy and light chain sequences provided as strings.
When using the paired-chain FASTA file, the script identifies the heavy and light chains as follows: it first looks for keywords in the FASTA description lines (heavy chain or light chain). If these keywords are not found, it will then assign the first sequence as the heavy chain and the second sequence as the light chain in order. For best results and to avoid ambiguity, include these keywords explicitly in your FASTA headers.
For humanization using a paired-chain FASTA file, prepare a file with both chains, for example my_antibody.fasta. The file must follow the IMGT numbering scheme and contain properly formatted heavy and light chain sequences. Example content:
2B04_H
EVQLVESGGGLVQPGGSLRLSCAASGFTFSSYWMSWVRQAPGKGLEWVANIKQDGSEKYYVDSVKGRFTISRDN
AKNSLYLQMNSLRAEDTAVYYCAR…
2B04_L
DIQMTQSPSSLSASVGDRVTITCRASQSISSYLNWYQQKPGKAPKLLIYAASSLQSGVPSRFSGSGSGTDFTLTISSL
QPEDFATYYCQQSYSTP…
Run the following command, adjusting the file path as needed:

```
#> python antibody_scripts/sample_for_anti_cdr.py \
 --ckpt release_data_dir/checkpoints/antibody/antibody.pt \
 --anti_complex_fasta /path/to/your/antibody.fasta
```

Alternatively, you can supply the heavy and light chain sequences directly using the --heavy_seq and --light_seq arguments:

```
#> python antibody_scripts/sample_for_anti_cdr.py \
 --ckpt release_data_dir/checkpoints/antibody/antibody.pt \
 --heavy_seq \
 "EVQLVESGGGLVQPGGSLRLSCAASGFTFSSYWMSWVRQAPGKGLEWVANIKQDGSEKYYVDSVKGRF \
 TISRDNAKNSLYLQMNSLRAEDTAVYYCAR..." \
 --light_seq \
 "DIQMTQSPSSLSASVGDRVTITCRASQSISSYLNWYQQKPGKAPKLLIYAASSLQSGVPSRFSGSGSGTDF          \
TLTISSLQPEDFATYYCQQSYSTP..."
```

The script generates multiple humanized variants (default 10). Outputs are saved in "antibody_sample_log/sample_humanization_result.csv" (CSV format) with columns: "Specific," "name," "hseq," and "lseq". A sample is shown in Table 1.
The number of generated variants is set by the --sample_number argument in the script (default 10). To increase diversity, increase this value (e.g., --sample_number 20). Higher values produce more candidates for downstream screening. If the script accepts a --batch_size parameter, you may also adjust it, but --sample_number is the primary control.
Important: Replace the placeholder sequences with your actual heavy and light chain sequences. Both chains must be IMGT-numbered and complete. The script generates multiple humanized antibody variants. By default, outputs are saved in the antibody_sample_log directory, which is configurable in the script. The results of antibody humanization are presented in Table 1.

Table 1. Humanized antibody sequences derived from an antibody

	Specific	hseq	lseq
2B04	mouse	QVQLKQSGPGLVAPSQSLSITCTVSG FSLINYAISWVRQPPGKGLEWLGVI WTGGGTNYNSALKSRLSISKDNSKS QVFLKMNSLQTDDTARYYCARKDY YGRYYGMDYWGQGTSVTVS	QAVVTQESALTTSPGETVTLTCRSS TGAVTTSNYANWVQEKPDHLFTG LIGGTNNRAPGVPARFSGSLIGDKA ALTITGAQTEDEAIYFCALWYNNH WVFGGGTKLTVL
2B04-hAb1	humanization	QVQLQESGPGLVKPSETLSLTCSVSG FSLINYAISWIRQAPGKGLEWIGVIW TGGGTNYNSALKSRLTISRDTSKSQA FLRLSSVTAADTAVYYCARKDYYGR YYGMDYWGQGTLVTVS	QAVVTQEPSVSVSPGGTVTLTLRSS TGAVTTSNYANWYQQKPGQPPRG LIGGTNNRAPWVPARFSGSILGGK AALTLSSAQPEDEADYYCALWYN NHWVFGGGTKLTVL
2B04-hAb2	humanization	QVQLQESGPGLVKPSETLSLTCTVSG FSLINYAISWIRQPPGKGLEWLGVIW TGGGTNYNSALKSRLTISVDTSKSQF SLKLSSVTAADTAVYYCARKDYYGR YYGMDYWGQGTLVTVS	QAVVTQEPSVTVSPGGTVTLTCRSS TGAVTTSNYANWFQQKPGQPPRGL IGGTNNRAPGVPDRFSGSFLGGDA ALTLSGLQPEDEADYYCALWYNN HWVFGGGTKLTVL
2B04-hAb3	humanization	QVQLQESGPGLVKPSETLSLTCSVSG FSLINYAISWIRQAPGKGLEWIGVIW TGGGTNYNSALKSRLTISGDTSKSQV FLKLNSVTAADTAVYYCARKDYYG RYYGMDYWGQGTLVTVS	QAVVTQEPSLTVSPGGTVTLTCRSS TGAVTTSNYANWFQQKPGQAPRG LIGGTNNRAPGVPSRFSGSILGGEA ALTLSGVQPEDEADYYCALWYNN HWVFGGGTKLTVL
2B04-hAb4	humanization	QVQLQESGPGLVKPSETLSLTCSVSG FSLINYAISWIRQAPGKGLEWIGVIW TGGGTNYNSALKSRLTISVDTSKSQV FLKLSSVTAADTAVYYCARKDYYGR YYGMDYWGQGTLVTVS	QAVVTQEPSLTVSPGGTVTLTCRSS TGAVTTSNYANWFQQKPGQAPRG LIGGTNNRAPGVPARFSGSILGGRA ALTLSGVQAEDEADYYCALWYNN HWVFGGGTKLTVL
2B04-hAb5	humanization	QVQLQESGPGLVKPSETLSLTCSVSG FSLINYAISWIRQAPGKGLEWLGVIW TGGGTNYNSALKSRLTISRDTSKSQF FLKLSSVTAADTAVYYCARKDYYGR YYGMDYWGQGTLVTVS	QAVVTQEPSLTVSPGGTVTLTCRSS TGAVTTSNYANWFQQRPGQAPRG LIGGTNNRAPGVPARFSGSILGGNP ALTLSGVRLEDEADYYCALWYNN HWVFGGGTKLTVL
2B04-hAb6	humanization	QVQLQESGPGLVKPSETLSLTCSVSG FSLINYAISWIRQPPGKGLEWVGVIW TGGGTNYNSALKSRLTISRDTSKSQV FLKLNSVTAADTAVYYCARKDYYG RYYGMDYWGQGTLVTVS	QAVVTQEPSLTVSPGGTVTLTCRSS TGAVTTSNYANWFQQKPGQPPRGL IGGTNNRAPWVPARFSGSILGGRA ALTLSGQSEDEADYYCALWYNNH WVFGGGTKLTVL
2B04-hAb7	humanization	QVQLQESGPGLVKPSETLSLTCTVSG FSLINYAISWIRQPPGKGLEWIGVIWT GGGTNYNSALKSRLTISVDTSKSQFS LKLSSVTAADTAVYYCARKDYYGRY YGMDYWGQGTLVTVS	QAVVTQEPSLTVSPGGTVTLTCRSS TGAVTTSNYANWFQQKPGQAPRG LIGGTNNRAPGVPARFSGSILGGKA ALTLSGAQVEDEADYYCALWYNN HWVFGGGTKLTVL
2B04-hAb8	humanization	QVQLQESGPGLVKPSETLSLTCSVSG FSLINYAISWIRQAPGKGLEWLGVIW TGGGTNYNSALKSRLTISVDTSKSQA SLKLSSVTAADTAVYYCARKDYYGR YYGMDYWGQGTLVTVS	QAVVTQEPSLTVSPGGTVTLTCRSS TGAVTTSNYANWFQQKPGQAPRG LIGGTNNRAPWVPARFSGSILGGK AALTLSGVQPEDEADYYCALWYN NHWVFGGGTKLTVL
2B04-hAb9	humanization	QVQLQESGPGLVKPSETLSLTCSVSG FSLINYAISWIRQPPGKGLEWLGVIW TGGGTNYNSALKSRLTISVDTSKSQV FLKLNSVTAADTAVYYCARKDYYG RYYGMDYWGQGTLVTVS	QAVVTQEPSLTVSPGGTVTLTCRSS TGAVTTSNYANWFQQKPGQPPRGL IGGTNNRAPGVPARFSGSLLGGKA ALTLSSAQPEDEADYYCALWYNN HWVFGGGTKLTVL
2B04-hAb10	humanization	QVQLQESGPGLVKPSETLRLTCTVSG FSLINYAISWIRQAPGKGLEWLGVIW TGGGTNYNSALKSRLTISLDTSKSQF	QAVVTQEPSLTVSPGGTVTLTCRSS TGAVTTSNYANWFQQKPGQAPRG LIGGTNNRAPGVPARFSGSILGGEA

	SLKLSSVTAVDTAVYYCARKDYYGR YYGMDYWGQGTLVTVS	ALTLSGVQAEDEADYYCALWYNN HWVFGGGTKLTVL

For nanobody humanization, the workflow is similar but uses a dedicated script with inpainting mode.

B3. Humanization with HuDiff-Nb

Nanobody humanization uses a dedicated script that supports an inpainting mode to preserve critical framework residues, such as hydrophilic residues in FR2. Input is a single FASTA file containing the nanobody sequence.

Prepare a FASTA file, for example, my_nanobody.fasta, with one entry:

3-2A2-4

QVQLQESGGGLVQAGGSLRLSCAASGRTFSSYAMGWFRQAPGKEREFVAAISWSGGRTYYADSVKGRFTISRDN AKNTVYLQMNSLKPEDTAVYYCAAD…

Critical: The --inpaint_sample True argument is essential for preserving nanobody stability and expression. Always enable inpainting mode for nanobody humanization.

Run the humanization command:

```
#> python nanobody_scripts/sample_for_nano_cdr.py \
 --ckpt release_data_dir/checkpoints/nanobody/nanobody.pt \
 --nano_complex_fasta /path/to/your/nanobody.fasta \
 --model finetune_vh \
 --inpaint_sample True
```

The argument "--model finetune_vh" specifies that the model fine-tuned on VHH (nanobody) sequences should be used. The --inpaint_sample argument activates the inpainting mode, which preserves key framework residues during humanization. This is strongly recommended for nanobodies to retain stability and expression.

If the generated variants show low humanness scores or limited diversity, the number of generated nanobody variants is controlled by the --sample_number argument (default 10). To increase diversity, set a higher value, e.g., --sample_number 20. If the script supports --batch_size, you may also adjust it, but the primary parameter is --sample_number. Refer to the script's help (python nanobody_scripts/sample_for_nano_cdr.py –help) for available arguments.

Output files are saved in the data/fasta_file directory by default, containing humanized nanobody sequences. The results of nanobody humanization are presented in Table 2.

Table 2. Humanized nanobody sequences derived from a nanobody

	Specific	hseq
3-2A2-4	camel	QVQLQESGGGLVQPGESLRLSCAASGSISTLNVMGWYRQAPGKQRE LVAQITLDGSPEYADSVKGRFTITKDGAQSTLYLQMNNLKPEDTAVYF CKLENGGFFYYWGQGTQVTVSTHHHHHH
3-2A2-4-hNb1	humanization	EVQLVESGGGLVQPGGSLRLSCAASGSISTLNVMSWYRQAPGKQRE LVAQITLDGSPEYADSVKGRFTISRDNSKNTLYLQMNSLRPEDTAFYY CKLENGGFFYYWGQGTQVTVSS
3-2A2-4-hNb2	humanization	EVQLVESGGGLVQPGGSLRLSCAASGSISTLNVMSWYRQAPGKQRE LVAQITLDGSPEYADSVKGRFTISRDNAKNTLYLQMNSLRPEDTAVYY CKLENGGFFYYWGQGTQVTVSS
3-2A2-4-hNb3	humanization	EVQLVESGGGLVQPGGSLRLSCAASGSISTLNVMNWYRQAPGKQRE LVAQITLDGSPEYADSVKGRFTISRDNAKNTLYLQMNSLRPEDTAVYY CKLENGGFFYYWGQGTQVTVSS
3-2A2-4-hNb4	humanization	EVQLVESGGGLVQPGGSLRLSCAASGSISTLNVMGWYRQAPGKQRE LVAQITLDGSPEYADSVKGRFTISRDNAKNTLYLQMNSLKPEDTAVYY CKLENGGFFYYWGQGTQVTVSS

3-2A2-4-hNb5	humanization	EVQLVESGGGLVQPGGSLRLSCAASGSISTLNVMDWYRQAPGKQRE LVAQITLDGSPEYADSVKGRFTISRDNAKNTLYLQMNSLRPEDTAVYY CKLENGGFFYYWGQGTQVTVSS
3-2A2-4-hNb6	humanization	EVQLVESGGGLVQPGGSLRLSCAASGSISTLNVMNWYRQAPGKQRE LVAQITLDGSPEYADSVKGRFTISRDNAKNTLYLQMNSLKPEDTAVYY CKLENGGFFYYWGQGTQVTVSS
3-2A2-4-hNb7	humanization	EVQLVESGGGLVQPGGSLRLSCAASGSISTLNVMNWYRQAPGKQRE LVAQITLDGSPEYADSVKGRFTISRDNAKNTLYLQMNSLRQEDTAVY YCKLENGGFFYYWGQGTQVTVSS
3-2A2-4-hNb8	humanization	EVQLVESGGGLVQPGGSLRLSCAASGSISTLNVMSWYRQAPGKQRE LVAQITLDGSPEAADSVKGRFTISRDNAKNTLYLQMNSLRPEDTAVY YCKLENGGFFYYWGQGTQVTVSS
3-2A2-4-hNb9	humanization	EVQLVESGGGLVQPGGSLRLSCAASGSISTLNVMGWYRQAPGKQRE LVAQITLDGSPEYAASVKGRFTISRDNAKNTLYLQMNSLKPEDTAVYY CKLENGGFFYYWGQGTQVTVSS
3-2A2-4-hNb10	humanization	EVQLVESGGGLVQPGGSLRLSCAASGSISTLNVMSWYRQAPGKQRE LVAQITLDGSPEYADSVKGRFTISRDNAKNTLYLQMNSLRPEDTAFYY CKLENGGFFYYWGQGTQVTVSS
3-2A2-4-hNb11	humanization	EVQLVESGGGLVQPGGSLRLSCAASGSISTLNVMSWYRQAPGKQRE LVAQITLDGSPEYSDSVKGRFTISRDNAKNTLYLQMNSLRPEDTAVYY CKLENGGFFYYWGQGTQVTVSS
3-2A2-4-hNb12	humanization	EVQLVESGGGLVQPGGSLRLSCAASGSISTLNVMNWYRQAPGKQRE LVAQITLDGSPEYADSGKGRFTISRDNAKNTLYLQMNSLRPEDTAVYY CKLENGGFFYYWGQGTQVTVSS
3-2A2-4-hNb13	humanization	EVQLVESGGGLVQPGGSLRLSCAASGSISTLNVMSWYRQAAGKQRE LVAQITLDGSPEYSDSVKGRFTISRDNAKNTLYLQMNSLRPEDTAVYY CKLENGGFFYYWGQGTQVTVSS
3-2A2-4-hNb14	humanization	EVQLVESGGGLVQPGGSLRLSCAASGSISTLNVMRWYRQAPGKQRE LVAQITLDGSPEYADSVKGRFTISRDNAKNTLYLQMNSLRPEDTAVYY CKLENGGFFYYWGQGTQVTVSS

Based on a comprehensive analysis, three sequences—the antibodies 2B04-hAb4, 2B04-hAb7, and 2B04-hAb9, along with the nanobodies 3-2A2-4-hNb3, 3-2A2-4-hNb10, and 3-2A2-4-hNb11—were selected from the humanization results for subsequent structural analysis.

After generating sequences, validate their structures as described in section B4.

B4. Structural validation using AlphaFold2

To assess the foldability and structural integrity of humanized sequences generated by HuDiff, we recommend performing structure prediction using AlphaFold2 (AF2) via the ColabFold platform. This approach does not require dedicated local GPU resources and is suitable for batch prediction of up to approximately 10 sequences.

1. Open the ColabFold notebook:
https://colab.research.google.com/github/sokrypton/ColabFold/blob/main/AlphaFold2.ipynb
2. Prepare the input sequence(s) in the query_sequence cell:
For nanobodies: Enter the single-chain sequence directly.
For antibodies: Enter the heavy and light chains in the format H:<heavy chain sequence>:L:<light chain sequence>.

3. Run all cells (*Runtime → Run all*). The prediction process typically takes ~30 min, depending on sequence length and Colab resource availability.
4. Upon completion, download the resulting ZIP file, which contains predicted structures (PDB files), confidence scores (pLDDT) in Figures 2 and 3, and ranking information (ranking_debug.json).
5. Evaluate prediction quality:
For nanobodies: Framework regions should exhibit high confidence (pLDDT > 90), while CDR loops with pLDDT > 70 are generally acceptable.

For antibodies: In addition to per-residue pLDDT, an interface predicted TM-score (ipTM) > 0.6 indicates stable heavy–light chain pairing.
This validation step ensures that the humanized sequences maintain proper folding and structural compatibility prior to experimental testing [6]. Typical pLDDT distributions for humanized antibodies and nanobodies are shown in Figures 2 and 3, respectively.

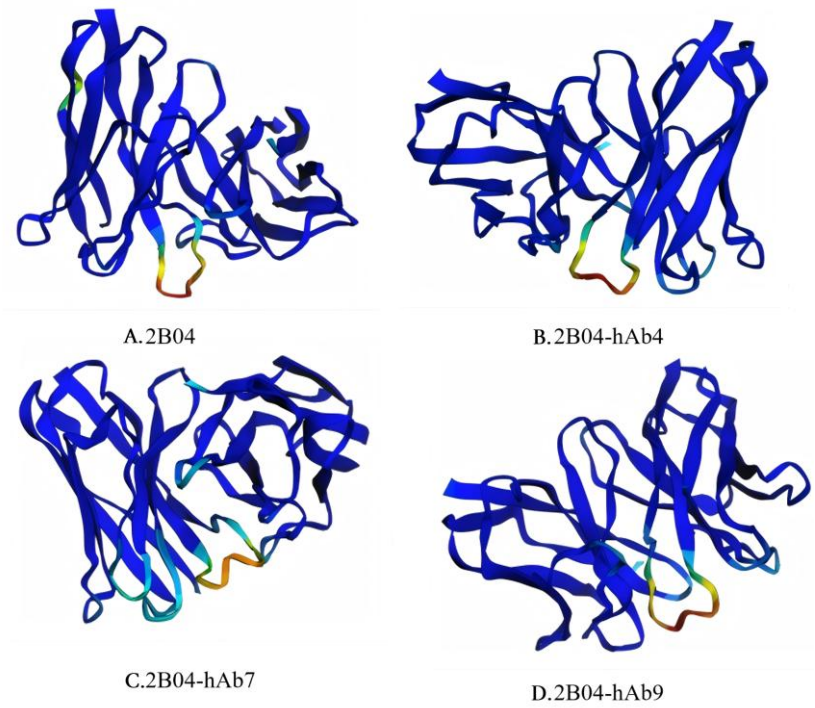


Figure 2. AlphaFold2 prediction pLDDT for antibody and humanized antibody. Colors indicate pLDDT confidence levels: blue >90 (very high), cyan 70–90 (good), and yellow <70 (low). For antibody humanization, aim for framework pLDDT ≥90.

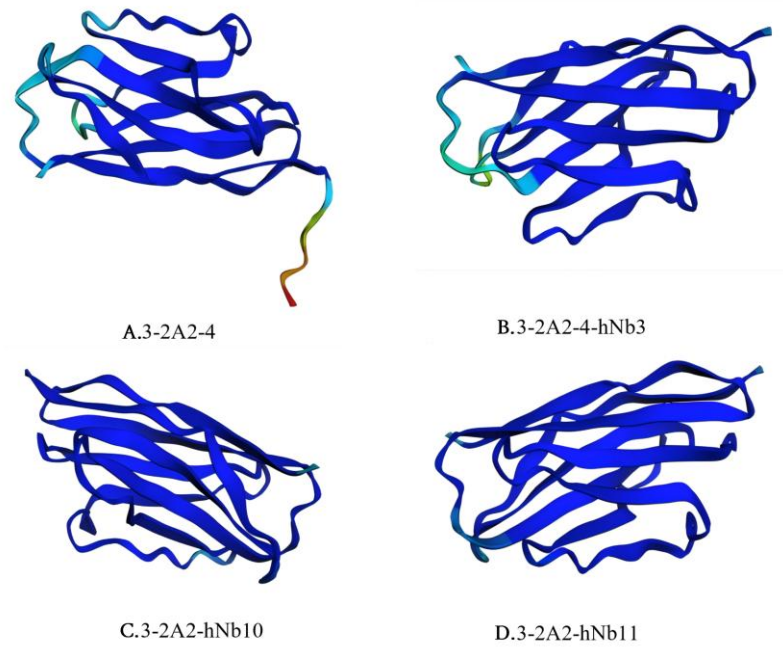


Figure 3. AlphaFold2 prediction pLDDT for nanobody and humanized nanobody. Colors indicate pLDDT confidence levels: blue >90 (very high), cyan 70–90 (good), and yellow <70 (low). For nanobody humanization, aim for framework pLDDT ≥90.

For typical results and their interpretation, see the next section.

Data analysis

Result interpretation

As a representative example of antibody humanization, we applied HuDiff-Ab to the mouse antibody 2B04. The humanness scores (T20, OASIs) of the 10 generated humanized antibodies are shown in Table 3. All variants achieved H T20 > 76, confirming that HuDiff-Ab produces humanized antibodies with humanness levels comparable to experimentally validated antibodies. As shown in Table 3 and Figure 4, the selected variants (2B04-hAb4, 2B04-hAb7, 2B04-hAb9) as well as nanobody variants (3-2A2-4-hNb3, hNb10, hNb11) maintain high humanness scores and structural integrity.

Table 3. Humanness scores of humanized 2B04 antibody variants

Candidate	H T20	L T20	H Z-score	L Z-score	OASIs identity	H OASIs %	L OASIs %
2B04-hAb1	76.34457	72.95457	-0.221	-1.983	0.622642	0.144387	0.086089
2B04-hAb2	80.67237	73.22737	-0.305	-1.670	0.683962	0.389378	0.097297
2B04-hAb3	76.38667	75.72737	-0.310	-1.790	0.665094	0.239027	0.118158
2B04-hAb4	78.10927	75.77277	-0.294	-1.705	0.683962	0.266991	0.153970
2B04-hAb5	77.35297	73.13647	-0.230	-2.129	0.627358	0.333847	0.043000
2B04-hAb6	76.59667	75.09097	-0.244	-2.107	0.635071	0.239027	0.064000
2B04-hAb7	81.42867	76.54000	-0.286	-1.804	0.716981	0.368658	0.210020
2B04-hAb8	78.19337	78.31827	-0.250	-1.953	0.669811	0.185784	0.153970
2B04-hAb9	77.35297	76.54557	-0.368	-2.083	0.698113	0.266991	0.210020
2B04-hAb10	77.26897	75.72737	-0.249	-1.700	0.702830	0.413595	0.128356

We applied HuDiff-Nb to the camelid nanobody 3-2A2-4. Fourteen humanized variants were generated with the inpainting mode enabled to preserve key framework residues. Their humanness scores, evaluated by Frame T20 and AbNatiV (VH/VHH scores, FR-VH, FR-VHH), are summarized in Table 4. All variants show high Frame T20 (>82) and favorable AbNatiV scores, confirming successful humanization while maintaining nanobody-specific framework characteristics.

Table 4. Humanness scores of humanized 3-2A2-4 nanobody variants

Candidate	Frame T20	VH Score	VHH Score	FR-VH	FR-VHH
3-2A2-4-hNb 1	82.64377	0.790327	0.865273	0.905032	0.958485
3-2A2-4-hNb 2	83.39087	0.792862	0.868983	0.908197	0.984039
3-2A2-4-hNb 3	83.50577	0.789046	0.878073	0.908718	0.984557
3-2A2-4-hNb 4	82.35637	0.776672	0.898226	0.884115	0.986167
3-2A2-4-hNb 5	83.50577	0.788471	0.864566	0.906707	0.983701
3-2A2-4-hNb 6	82.35637	0.780503	0.880919	0.890425	0.986167
3-2A2-4-hNb 7	83.39087	0.772448	0.846063	0.889641	0.959778
3-2A2-4-hNb 8	83.50577	0.780126	0.867323	0.891295	0.960514
3-2A2-4-hNb 9	83.50577	0.763323	0.889360	0.866406	0.978205
3-2A2-4-hNb 10	83.50577	0.800210	0.867411	0.909922	0.960631
3-2A2-4-hNb 11	83.50577	0.789639	0.886810	0.906100	0.984989
3-2A2-4-hNb 12	83.50577	0.782453	0.860455	0.907712	0.961226
3-2A2-4-hNb 13	82.35637	0.770191	0.868767	0.895690	0.961101
3-2A2-4-hNb 14	83.50577	0.781371	0.878078	0.907726	0.984557

In step B4, the AlphaFold2-generated PDB files were visualized in PyMOL to clearly reveal the sites of amino acid mutations between the original and humanized sequences (Figures 4 and 5).

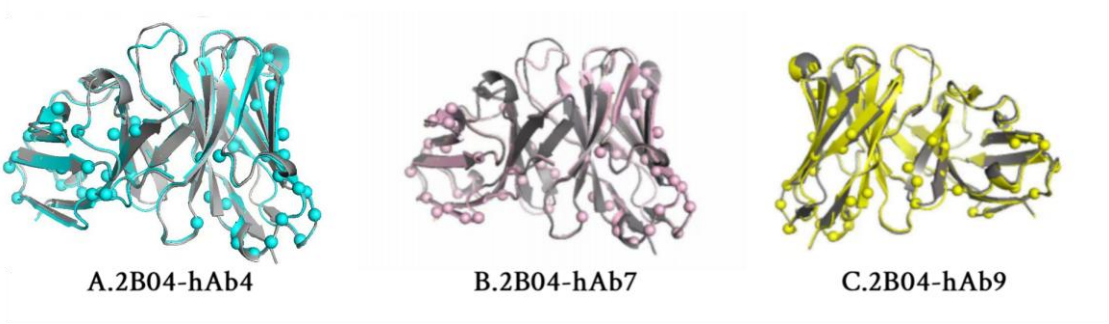


Figure 4. 3D perspective illustrations presented for the positions of mutations in each humanized antibody. These three figures all show the antibodies after mutation; the positions of the mutated amino acids are the spherical positions shown in the figures.

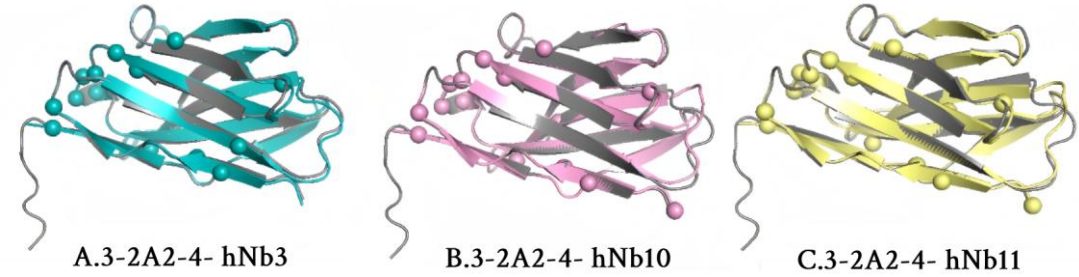


Figure 5. 3D perspective illustrations presented for the positions of mutations in each humanized nanobody. These three figures all show the nanobodies after mutation; the positions of the mutated amino acids are the spherical positions shown in the figures.

Validation of protocol

This protocol has been validated in the original study [3] through both in silico benchmarks and wet-lab experiments.

General notes and troubleshooting

General notes

The following notes and troubleshooting tips complement the step-by-step instructions in the Procedure.
1. IMGT (International ImMunoGeneTics information system) numbering is mandatory: All input antibody/nanobody sequences must be aligned and numbered according to the IMGT scheme. Use the abnumber package consistently for IMGT numbering to avoid incorrect model inputs and invalid sequence outputs.
2. CDR extraction: Ensure CDR boundaries follow the IMGT definition (CDR1, CDR2, CDR3) when preparing input CDR sequences; incorrect CDR boundaries will lead to loss of antigen-binding functionality in humanized variants.

Validation example:
Wrong example: EVQLVESGGGLVQ... (continuous string without spaces): This will cause abnumber to throw a NumberingError.
Correct format: EVQLVESGGGLVQ...YRC A (processed by IMGT, containing specific hyphens or spaces, representing specific numbering positions).
Solution: Use the abnumber toolkit for preprocessing. Run the following code in the command line to verify:

```
#> python -c "from abnumber import Chain; chain = Chain('YOUR_SEQUENCE'); print(chain.imgt)"
```

If the output contains NumberingError, please re-align from the raw sequence using IMGT.
3. Sampling diversity: Generate 10–20 humanized variants per target antibody/nanobody for downstream screening. Select candidates based on multiple metrics (humanness scores, structural validation, sequence diversity) to maximize the chance of obtaining functional variants.
4. Pretrained vs. custom models: The provided pretrained HuDiff checkpoints are sufficient for humanization of mouse antibodies and camelid nanobodies (the most common non-human therapeutic antibody sources). Custom finetuning is only required for working with exotic species (e.g., cattle, sheep, swine) not covered by the pretrained models.
5. Hardware requirements: Model training (pretraining/finetuning) requires an NVIDIA GPU with ≥16 GB VRAM to avoid CUDA out-of-memory errors. Humanization inference (sequence generation) can run on a CPU, though the speed is significantly slower (≈10× slower than GPU inference).
6. AbNatiV dependency: AbNatiV is only required for HuDiffNb finetuning (nanobody humanization); it is not needed for HuDiff-Ab training or humanization of conventional antibodies.
7. Structural visualization: Use pymol-open-source or PyMOL Commercial for 3D structural visualization of AlphaFold2-predicted models; focus on framework residue conservation and CDR loop folding for variant selection.
8. FASTA format pitfalls and sequence alignment misconceptions:
a. Line breaks and encoding: Please ensure your FASTA file uses Unix/Linux line breaks (LF) instead of Windows line breaks (CRLF). Although most parsers are compatible with CRLF, it may cause sequence truncation issues in some Linux environments.
b. Sequence completeness: Do not truncate sequences. Input sequences must contain the complete variable domain framework (from the start of Framework 1 to the end of Framework 4). Truncated sequences (e.g., containing only CDR regions) cannot be correctly numbered by abnumber, leading to inference failure.
c. Special characters: Sequences must not contain numbers, spaces, or special symbols (such as ?). If there are ambiguous positions in the sequence, please use "X" from the standard amino acid single-letter abbreviations, but note that this may affect the accuracy of IMGT numbering.

9. Runtime estimation: The following benchmark data is based on an NVIDIA RTX 3060 Ti (8GB VRAM):
a. Inference: Generating humanized variants for a single antibody sequence (including sampling 10 variants) takes approximately 2 min (GPU) or 20 min (CPU).
b. Fine-tuning: The fine-tuning process for a specific species typically takes 2–4 h, depending on the dataset size (usually 10k–50k sequences).
c. Pretraining: Pretraining from scratch (e.g., reproducing the original results) is a computationally intensive process and is expected to take 3–5 days (using a NVIDIA RTX 3060 GPU cluster).

Troubleshooting

Problem 1: CUDA out-of-memory (OOM) error during model training/finetuning.
Possible cause: Batch size set too large for the available GPU VRAM; the default batch size is optimized for GPUs with ≥8 GB VRAM.
Solution: Reduce the batch size parameter in the training/finetuning script (e.g., from 32 to 16 or 8); close other GPU-intensive applications to free up VRAM; use a GPU with larger VRAM (≥16 GB).

Problem 2: Invalid or truncated sequence output from the humanization script.
Possible cause: Input sequences are not IMGT-numbered; incorrect file path for the FASTA input; corrupted pretrained checkpoint files.
Solution: Recheck IMGT numbering and re-number input sequences with the abnumber package (IMGT scheme); verify the FASTA file path in the script (ensure no typos); re-download the pretrained checkpoint files from the HuDiff repository and verify file integrity.

Problem 3: Model training/finetuning fails with a "file not found" error.
Possible cause: Incorrect file path configuration in the script; the release_data_dir folder is not placed in the HuDiff root directory; missing dataset files (LMDB) in the release data directory.
Solution: Verify all file paths (config_path, data_path, log_path) in the training/finetuning script; ensure the release_data_dir folder is in the HuDiff root directory; re-download and extract the release data directory to restore missing dataset files.

Problem 4: HuDiffNb finetuning fails to detect the AbNatiV executable.
Possible cause: AbNatiV is installed in a different Conda environment; incorrect AbNatiV executable path in the HuDiff configuration; missing AbNatiV model weights.
Solution: Record the full path to the AbNatiV executable and provide it in the HuDiffNb finetuning configuration; re-install AbNatiV according to the official documentation; download the AbNatiV model weights and place them in the designated directory.

Problem 5: Humanized variants show low humanness scores (T20 < 70) or poor structural validation (pLDDT < 70 in framework regions).
Possible cause: Inpainting mode disabled for nanobodies; insufficient finetuning epochs; low-quality input sequences (e.g., incomplete IMGT numbering).
Solution: Enable inpainting mode (--inpaint_sample True) for nanobody humanization; increase the number of finetuning epochs (total_epoch parameter) to improve model convergence; revalidate and correct input sequence IMGT numbering.
For more issues and solutions, please check the GitHub Issues page: https://github.com/TencentAI4S/HuDiff/issues

Acknowledgments

This work was supported by the Fundamental Research Funds for the Central Universities (31920250061). This work was supported by Key Laboratory of Advanced Computing, Gansu Province (Gansu Computing Center) (Grant Number: 25GSXJJS04). We also thank the Supercomputing Center of Lanzhou University and Gansu Computation Center for supporting this paper. This protocol was used in [3].

Author contributions

Conceptualization, Yang Nan, Hongshuai Sun, Jianxin He, and Qifeng Bai; Investigation, Yang Nan, Hongshuai Sun, Bo Zhang, and Yue Yang; Writing—Original Draft, Yang Nan and Hongshuai Sun; Writing—Review & Editing, Yang Nan, Hongshuai Sun, Bo Zhang, Yue Yang, Jianxin He, and Qifeng Bai; Funding acquisition, Yue Yang, Jianxin He, and Qifeng Bai; Supervision, Jianxin He and Qifeng Bai.

Competing interests

There are no conflicts of interest or competing interests.

Ethical considerations

This protocol involves only computational protein design and in silico analysis; no human or animal subjects, biological samples, or experimental procedures involving living organisms were used. All antibody/nanobody sequences used in the protocol are from public domain datasets with open access.

Received: April 09, 2026; Accepted: July 02, 2026; Available online: August 24, 2026; Published: September 05, 2026

References

1. Tjandra, J. J., Ramadi, L. and McKenzie, I. F. C. (1990). Development of human anti-murine antibody (HAMA) response in patients. *Immunol Cell Biol.* 68(6): 367–376. https://doi.org/10.1038/icb.1990.50

2. Wang, Y., Chen, Y. L., Xu, H., Rana, G. E., Tan, X., He, M., Jing, Q., Wang, Q., Wang, G., Xie, Z., et al. (2024). Comparison of “framework Shuffling” and “CDR Grafting” in humanization of a PD-1 murine antibody. *Front Immunol.* 15: e1395854. https://doi.org/10.3389/fimmu.2024.1395854
3. Ma, J., Wu, F., Xu, T., Xu, S., Liu, W., Yan, L., Qu, M., Yang, X., Bai, Q., Xiao, J., et al. (2025). An adaptive autoregressive diffusion approach to design active humanized antibodies and nanobodies. *Nat Mach Intell.* 7(10): 1698–1712. https://doi.org/10.1038/s42256-025-01120-9
4. Ramon, A., Ali, M., Atkinson, M., Saturnino, A., Didi, K., Visentin, C., Ricagno, S., Xu, X., Greenig, M., Sormanni, P., et al. (2024). Assessing antibody and nanobody nativeness for hit selection and humanization with AbNatiV. *Nat Mach Intell.* 6(1): 74–91. https://doi.org/10.1038/s42256-023-00778-3
5. Jovčevska, I. and Muyldermans, S. (2020). The Therapeutic Potential of Nanobodies. *BioDrugs.* 34(1): 11–26. https://doi.org/10.1007/s40259-019-00392-z
6. Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., et al. (2021). Highly accurate protein structure prediction with AlphaFold. *Nature.* 596(7873): 583–589. https://doi.org/10.1038/s41586-021-03819-2