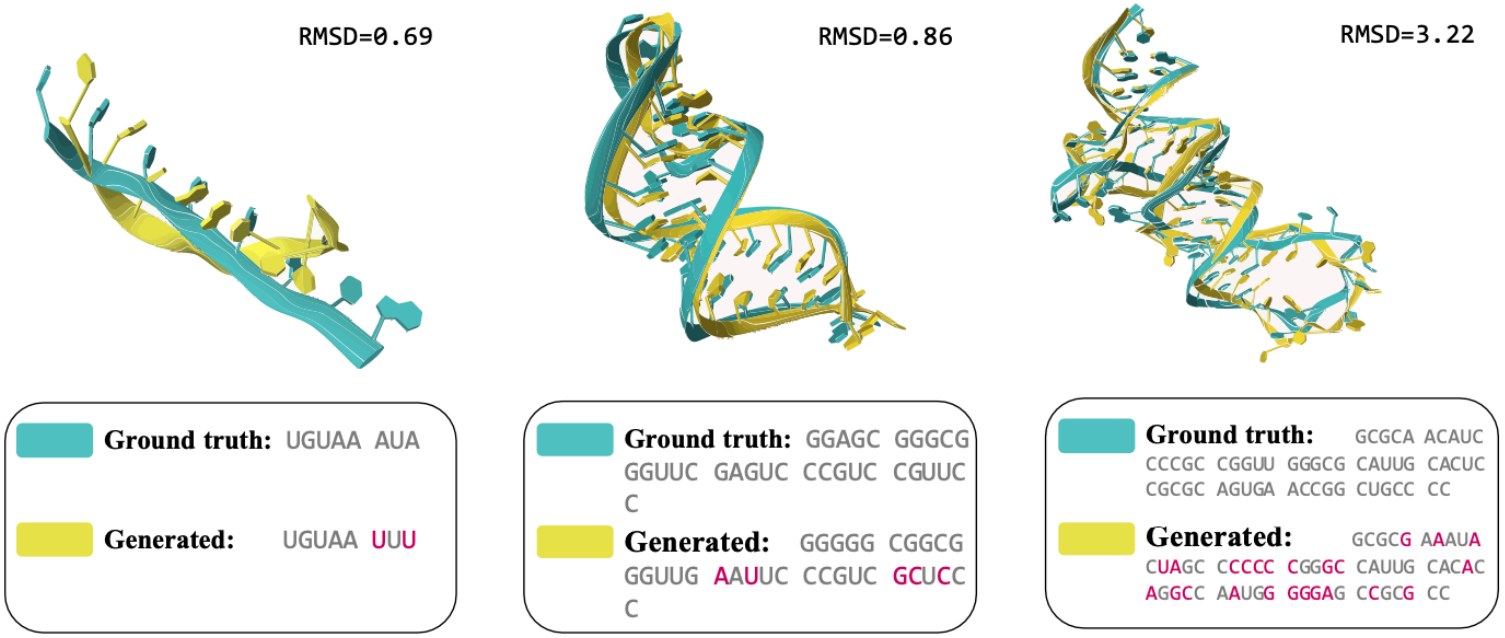
RDesign: Hierarchical Data-efficient Representation Learning for Tertiary Structure-based RNA Design

Introduction

While artificial intelligence has made remarkable strides in revealing the relationship between biological macromolecules' primary sequence and tertiary structure, designing RNA sequences based on specified tertiary structures remains challenging. Though existing approaches in protein design have thoroughly explored structure-to-sequence dependencies in proteins, RNA design still confronts difficulties due to structural complexity and data scarcity.

In this study, we aim to systematically construct a data-driven RNA design pipeline. We crafted a large, well-curated benchmark dataset and designed a comprehensive structural modeling approach to represent the complex RNA tertiary structure. More importantly, we proposed a hierarchical data-efficient representation learning framework that learns structural representations through contrastive learning at both cluster-level and sample-level to fully leverage the limited data. Extensive experiments demonstrate the effectiveness of our proposed method, providing a reliable baseline for future RNA design tasks.



(a) 7CGK (b) 6UGI (c) 7D7V

Dataset Details

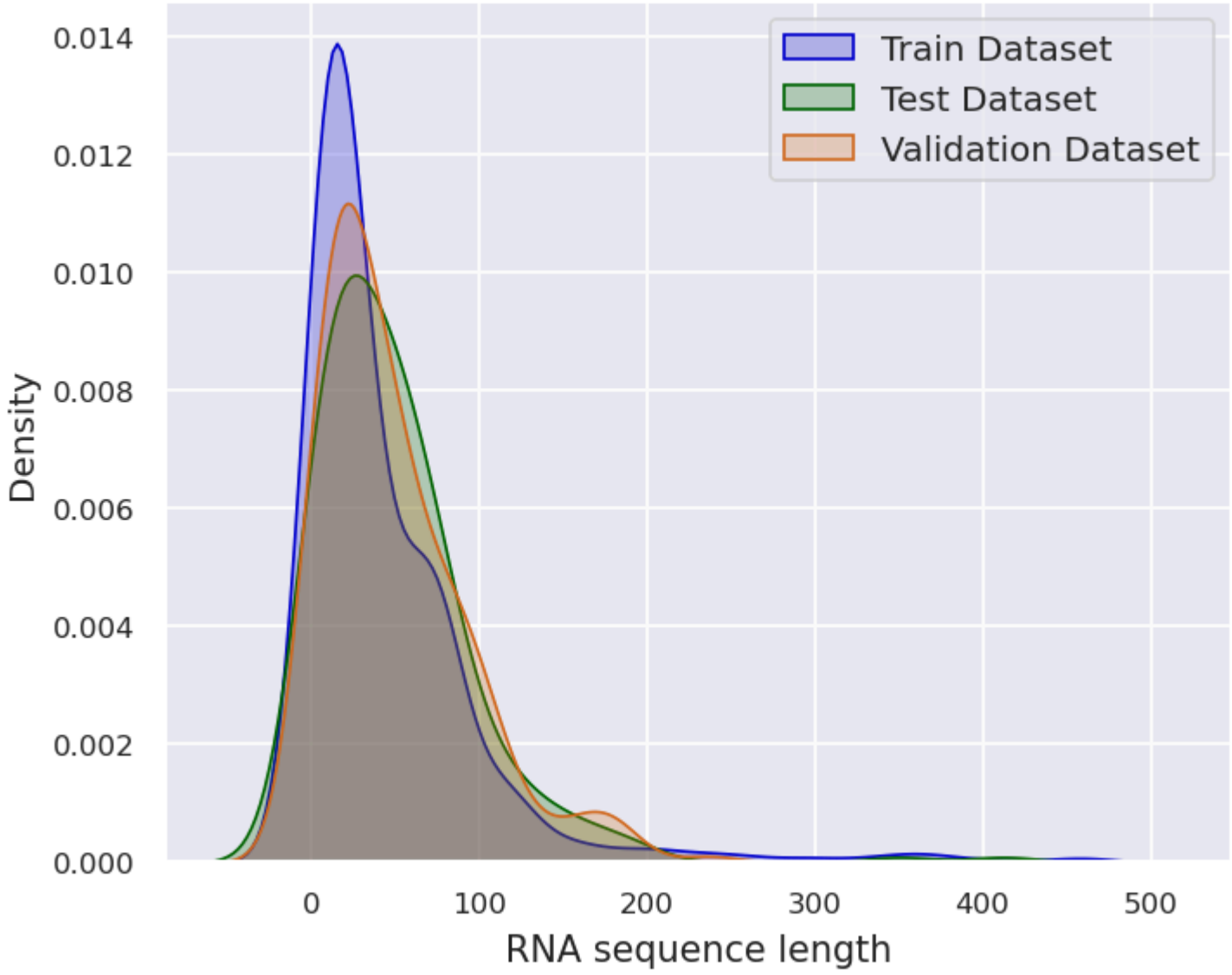
Data Clustering

We used the TM-score to assess structural similarity among RNA structures. To group samples with similar structures, we employed a **graph-based** clustering approach. Each structure is represented as a **node**, and the TM-Score between every pair of structures is computed. If the TM-Score for a pair exceeds 0.45, an **edge** is drawn between them, indicating their high similarity. After processing all pairs in this manner,

clusters are identified as connected components within the graph. From our dataset of 2218 RNA structures, we yielded 987 distinct clusters, which could be validated in the dataset below.

Data Splitting

To **Minimize the information leakage**, we split the collected data based on two principles: (i) **avoiding similar structures in different sets** and (ii) **maintaining similar length distributions across sets**. After obtaining 987 clusters, we added each to the training, validation, and testing datasets, maintaining vigilance to prevent similar structures from appearing in divergent sets. Specifically, we calculated the **average length** of all sequences and sequentially allocated clusters to the respective train/validation/test sets, managing to align each cluster closely with the global average length. Consequently, we maintained a consistent sequence length distribution across all sets, as depicted in the figure below.



Downloading code, required packages and checkpoints

```
!pip install addict fvcore matplotlib numpy scikit-learn timm tqdm nni fair-esm pandas wan

!pip install torch_geometric
!pip install torch_scatter torch_sparse torch_cluster torch_spline_conv -f https://data.py
!pip install biopython
!pip install tmtools
```

```
!pip install torcheval
!pip install py3Dmol

! git clone https://github.com/A4Bio/RDesign.git
%cd RDesign
!mkdir data
!wget -P data https://github.com/A4Bio/RDesign/releases/download/data/train_data.pt
!wget -P data https://github.com/A4Bio/RDesign/releases/download/data/val_data.pt
!wget -P data https://github.com/A4Bio/RDesign/releases/download/data/test_data.pt
!wget -P data https://github.com/A4Bio/RDesign/releases/download/data/val_data.pt
!wget -P data https://github.com/A4Bio/RDesign/releases/download/data/rfam_data.pt
```

```
Collecting addict
  Downloading addict-2.4.0-py3-none-any.whl.metadata (1.0 kB)
Collecting fvcore
  Downloading fvcore-0.1.5.post20221221.tar.gz (50 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 50.2/50.2 kB 534.8 kB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages (
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (1.26.
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.10/dist-packages
Collecting timm
  Downloading timm-1.0.9-py3-none-any.whl.metadata (42 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 42.4/42.4 kB 2.5 MB/s eta 0:00:00
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (4.66.5
Collecting nni
  Downloading nni-3.0-py3-none-manylinux1_x86_64.whl.metadata (19 kB)
Collecting fair-esm
  Downloading fair_esm-2.0.0-py3-none-any.whl.metadata (37 kB)
Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages (2.1.
Collecting wandb
  Downloading wandb-0.18.2-py3-none-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.met
Collecting biotite
  Downloading biotite-1.0.1-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl
Collecting Bio
  Downloading bio-1.7.1-py3-none-any.whl.metadata (5.7 kB)
Requirement already satisfied: transformers in /usr/local/lib/python3.10/dist-packages
Collecting yacs>=0.1.6 (from fvcore)
  Downloading yacs-0.1.8-py3-none-any.whl.metadata (639 bytes)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.10/dist-packages
Requirement already satisfied: termcolor>=1.1 in /usr/local/lib/python3.10/dist-packag
Requirement already satisfied: Pillow in /usr/local/lib/python3.10/dist-packages (from
Requirement already satisfied: tabulate in /usr/local/lib/python3.10/dist-packages (fr
Collecting iopath>=0.1.7 (from fvcore)
  Downloading iopath-0.1.10.tar.gz (42 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 42.2/42.2 kB 2.4 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-pack
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-pac
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-pac
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packa
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-pack
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist-
Requirement already satisfied: scipy>=1.6.0 in /usr/local/lib/python3.10/dist-packages
Requirement already satisfied: joblib>=1.2.0 in /usr/local/lib/python3.10/dist-package
Requirement already satisfied: threadpoolctl>=3.1.0 in /usr/local/lib/python3.10/dist-
Requirement already satisfied: torch in /usr/local/lib/python3.10/dist-packages (from
Requirement already satisfied: torchvision in /usr/local/lib/python3.10/dist-packages
Requirement already satisfied: huggingface_hub in /usr/local/lib/python3.10/dist-packa
Requirement already satisfied: safetensors in /usr/local/lib/python3.10/dist-packages
```

```
Collecting astor (from nni)
  Downloading astor-0.8.1-py2.py3-none-any.whl.metadata (4.2 kB)
Requirement already satisfied: cloudpickle in /usr/local/lib/python3.10/dist-packages
Collecting colorama (from nni)
  Downloading colorama-0.4.6-py2.py3-none-any.whl.metadata (17 kB)
Collecting filelock<3.12 (from nni)
  Downloading filelock-3.11.0-py3-none-any.whl.metadata (2.5 kB)
Collecting json-tricks>=3.15.5 (from nni)
  Downloading json_tricks-3.17.3-py2.py3-none-any.whl.metadata (16 kB)
```

Load model and datasets

```
import sys
sys.path.append('/content/RDesign/')
import torch
import json
import argparse
from main import Exp
from tqdm import tqdm
from methods.utils import cuda
from sklearn.metrics import precision_recall_
import numpy as np
import os.path as osp
import torch.nn.functional as F
from Bio.PDB import PDBParser, MMCIFParser

from API.rpuzzles_dataset import RPuzzlesDat
import py3Dmol
import os
import requests

svpath = '/content/RDesign/checkpoints/'
config = json.load(open(svpath+'model_param.
config['load_full_data'] = False # @param {t
args = argparse.Namespace(**config)
exp = Exp(args)
exp.method.model.load_state_dict(torch.load(
exp.method.model.eval()

def process_single_pdb(file_path, chain_name
    backbone_atoms = ['P', "O5'", "C5'", "C4
    alphabet_set = 'AUCG'

    file_name = osp.basename(file_path)
    file_extension = file_name.split('.')[-1
    structure_name = file_name.split('.')[0]

    if file_extension == 'pdb':
        parser = PDBParser()
    elif file_extension == 'cif':
        parser = MMCIFParser()
    else:
        raise ValueError("Unsupported file f

    structure = parser.get_structure('', fil
    coords = {
        'P': [], "O5'": [], "C5'": [], "C4'"
    }
```

config['load_full_data']: ☐

```
    for model in structure:
        if chain_name is None:
            chain = list(model.get_chains())
        else:
            chain = model[chain_name]

        seq = ''
        coords_dict = {atom_name: [np.nan, n

        for residue in chain:
            if residue.id[0] == " ":
                seq += residue.get_resname()

            for atom in residue:
                if atom.name in backbone_ato
                    coords_dict[atom.name] =

            list(map(lambda atom_name: coord

        for atom_name in backbone_atoms:
            assert len(seq) == len(coords[at

        bad_chars = set(seq).difference(alph
        if len(bad_chars) != 0:
            print('Found bad characters in s

        break

    data = {
        'seq': seq,
        'coords': coords,
        'chain_name': chain.id,
        'name': structure_name
    }

    return data


def featurize_HC(batch):
    """ Pack and pad batch into torch tensor
    alphabet = 'AUCG'
    B = len(batch)
    lengths = np.array([len(b['seq']) for b
    L_max = max([len(b['seq']) for b in batc
    X = np.zeros([B, L_max, 6, 3])
    S = np.zeros([B, L_max], dtype=np.int32)
    clus = np.zeros([B], dtype=np.int32)
    ss_pos = np.zeros([B, L_max], dtype=np.i

    ss_pair = []
    names = []

    for i, b in enumerate(batch):
        x = np.stack([b['coords'][c] for c i

        l = len(b['seq'])
        x_pad = np.pad(x, [[0, L_max-l], [0,
        X[i,:,:,:] = x_pad
```

```
        indices = np.asarray([alphabet.index
        S[i, :l] = indices
        names.append(b['name'])

        clus[i] = i

    mask = np.isfinite(np.sum(X,(2,3))).asty
    numbers = np.sum(mask, axis=1).astype(in
    S_new = np.zeros_like(S)
    X_new = np.zeros_like(X)+np.nan
    for i, n in enumerate(numbers):
        X_new[i,:n,::] = X[i][mask[i]==1]
        S_new[i,:n] = S[i][mask[i]==1]

    X = X_new
    S = S_new
    isnan = np.isnan(X)
    mask = np.isfinite(np.sum(X,(2,3))).asty
    X[isnan] = 0.
    # Conversion
    S = torch.from_numpy(S).to(dtype=torch.l
    X = torch.from_numpy(X).to(dtype=torch.f
    mask = torch.from_numpy(mask).to(dtype=t
    clus = torch.from_numpy(clus).to(dtype=t
    return X, S, mask, lengths, clus, names

def eval_sequence(exp,data):
  alphabet = 'AUCG'
  S_preds, S_trues, name_lst, rec_lst = [],
  S_preds_lst, S_trues_lst = [], []
  for idx, sample in enumerate(data):
      sample = featurize_HC([sample])
      X, S, mask, lengths, clus, names = sam
      X, S, mask = cuda((X, S, mask), device=
      logits, gt_S = exp.method.model.sample
      log_probs = F.log_softmax(logits, dim=
      S_pred = torch.argmax(log_probs, dim=1

      S_preds += S_pred.cpu().numpy().tolist
      S_trues += gt_S.cpu().numpy().tolist()

      S_preds_lst.append(''.join([alphabet[a
      S_trues_lst.append(''.join([alphabet[a
      name_lst.extend(names)

      cmp = S_pred.eq(gt_S)
      recovery_ = cmp.float().mean().cpu().n
      rec_lst.append(recovery_)

  _, _, f1, _ = precision_recall_fscore_supp

  return name_lst, f1, rec_lst, S_preds_lst,

def highlight_differences(pred_seq, true_seq
    highlighted_pred = []
    highlighted_true = []

    for pred_char, true_char in zip(pred_seq
```

```
        if pred_char == true_char:
            highlighted_pred.append(pred_cha
            highlighted_true.append(true_cha
        else:
            # ANSI escape sequences for red 
            highlighted_pred.append(f'\033[9
            highlighted_true.append(f'\033[9

    return ''.join(highlighted_pred), ''.joi


def load_processed_data(single_data, pdb_fil
    if single_data == "True":
        if chain_name == "":
            chain_name = None
        processed_data = [process_single_pdb
    elif dataset == 'test':
        processed_data = exp.test_loader.dat
    elif dataset == 'Rfam':
        rfam_dataset = RPuzzlesDataset('/con
        processed_data = rfam_dataset
    else:
        raise ValueError("Invalid input: Ple
    return processed_data

def visualize_pdb(pdb_file):
    with open(pdb_file, 'r') as f:
        true_pdb = f.read()
    view = py3Dmol.view(width=400, height=30
    view.addModel(true_pdb, 'pdb')
    view.setStyle({'model': 0}, {"cartoon":
    view.zoomTo()
    view.show()

def print_results(single_data, name_lst, f1, 
    if single_data == "True":
        print('Name:', name_lst[0])
        print('F1_Score:', np.mean(f1), 'Rec

        highlighted_pred_seq, highlighted_tr
        print('Predicted Sequence:\n' + high
        print('True Sequence:\n' + highlight
    else:
        print('F1_Score:', np.mean(f1), 'Rec

def inference(single_data='True', pdb_file=N
    processed_data = load_processed_data(sin

    if single_data == "True" and pdb_file:
        visualize_pdb(pdb_file)

    name_lst, f1, rec_lst, S_preds_lst, S_tr
    print_results(single_data, name_lst, f1, 


def download_structure(pdb_id, save_dir='./'
    """Download PDB or CIF file, trying PDB
    pdb_url = f'https://files.rcsb.org/downl
    cif_url = f'https://files.rcsb.org/downl
```

```
    pdb_file = os.path.join(save_dir, f'{pdb_
    cif_file = os.path.join(save_dir, f'{pdb_

    # Try downloading PDB file first
    response = requests.get(pdb_url)
    if response.status_code == 200:
        with open(pdb_file, 'wb') as f:
            f.write(response.content)
        print(f'{pdb_id}.pdb downloaded succ
        return pdb_file, f'{pdb_id}.pdb'
    else:
        print(f'PDB file for {pdb_id} not fo

    # If PDB file not found, try downloading
    response = requests.get(cif_url)
    if response.status_code == 200:
        with open(cif_file, 'wb') as f:
            f.write(response.content)
        print(f'{pdb_id}.cif downloaded succ
        return cif_file, f'{pdb_id}.cif'
    else:
        raise Exception(f'Failed to download
```

```
---------------------------------------------------------------------------
OSError                                   Traceback (most recent call last)
<ipython-input-2-f10af47d2ffb> in <cell line: 6>()
      4 import json
      5 import argparse
----> 6 from main import Exp
      7 from tqdm import tqdm
      8 from methods.utils import cuda

                                  8 frames
/usr/lib/python3.10/ctypes/__init__.py in __init__(self, name, mode, handle,
use_errno, use_last_error, winmode)
    372
    373         if handle is None:
--> 374             self._handle = _dlopen(self._name, mode)
    375         else:
    376             self._handle = handle

OSError: /usr/local/lib/python3.10/dist-packages/torch_scatter/_version_cuda.so:
undefined symbol: _ZN5torch3jit17parseSchemaOrNameERKSs
```

Check the dataset

```
import _pickle as cPickle
test_data = cPickle.load(open('/content/RDesign/data/test_data.pt', 'rb'))
print(test_data[0].keys())
```

Model Inference

Test model performance on our proposed test dataset

```
single_data = False # @param {type:"string"}
pdb_file = '/content/RDesign/example.pdb'  #
dataset = 'test'  # @param {type:"string"}
chain_name = ""  # @param {type:"string"}

inference(single_data=single_data, pdb_file=
```

single_data: False

pdb_file: /content/RDesign/exam

dataset: test

chain_name: Insert text here

For single-chain RNA, we could just use the default setting

```
pdb_file = '/content/RDesign/example.pdb'  #
dataset = 'test'  # @param {type:"string"}
chain_name = ""  # @param {type:"string"}

inference(pdb_file=pdb_file, dataset=dataset
```

pdb_file: /content/RDesign/exam

dataset: test

chain_name: Insert text here

For Protein-RNA complex, we need to specify which chain we need

```
pdb_id = '6TPH' # @param {type:"string"}
pdb_file_path, pdb_file_name = download_stru
pdb_file = pdb_file_path  # Use the download
dataset = 'test'  # @param {type:"string"}
chain_name = "B"  # @param {type:"string"}
inference( pdb_file=pdb_file, dataset=datase
```

pdb_id: 6TPH

dataset: test

chain_name: B

Double-click (or enter) to edit