

```

1  import numpy as np
2  import pandas as pd
3  import scipy
4  import scipy.spatial
5  import pickle
6
7  def extractData(data):
8      #bare bones PDB format reader (For AlphaFold retrieved structures)
9      #returns a distance matrix between all atoms
10     # the coordinates of all atoms and
11     # the amino acids corresponding to each atom
12     aas = {'ALA': 'A',
13           'ARG': 'R',
14           'ASN': 'N',
15           'ASP': 'D',
16           'ASX': 'B',
17           'CYS': 'C',
18           'GLU': 'E',
19           'GLN': 'Q',
20           'GLX': 'Z',
21           'GLY': 'G',
22           'HIS': 'H',
23           'ILE': 'I',
24           'LEU': 'L',
25           'LYS': 'K',
26           'MET': 'M',
27           'PHE': 'F',
28           'PRO': 'P',
29           'SER': 'S',
30           'THR': 'T',
31           'TRP': 'W',
32           'TYR': 'Y',
33           'VAL': 'V'}
34     #creates distance matrix, atom coordinates and atoms -> amino acids
35     atoms_aas=[]
36     coords=[]
37     for line in data:
38         ATOM = line[:4]
39         if ATOM=="ATOM":
40             #ATOM, aid, atm, aa, lixo, aaid, x,y,z, s1, s2, atm2=fs
41             aid = line[6:11].strip()
42             x = float(line[30:38].strip())
43             y = float(line[38:46].strip())
44             z = float(line[46:54].strip())
45             aa = line[17:20].strip()
46             aaid = line[22:26].strip()
47             aid=int(aid)
48             atoms_aas.append(aas[aa]+aaid)
49             coords.append([x,y,z])
50
51     coords=np.array(coords)
52     atoms_aas=np.array(atoms_aas)
53     DM = scipy.spatial.distance_matrix(coords, coords)
54     return DM, coords, atoms_aas
55
56 def process_fp(fp):
57     #returns processed aminoacids without ids from aas, but distinguishing different
58     #occurrences of the same AA
59     L=sorted(list(fp))
60     idx=0
61     prev=L[0][0]
62     L2=[prev+str(idx)]
63     for i in L[1:]:
64         if i[0]==prev: idx+=1
65         else: idx=0
66         prev=i[0]
67         L2.append(prev+str(idx))
68     return frozenset(L2)
69
70 def compute_fps(DM, coords, atoms_aas, max_dist=5):
71     #computes the general structures for the fingerprints
72     fps=[]

```

```

72     N=coords.shape[0]
73     for i in range(N) :
74         fp=set(atoms_aas[DM[i]<max_dist])
75         if len(fp)>2:
76             fps.append(process_fp(fp))
77     return set(fps)
78
79 def getFPs(fname, max_dist=5):
80     #reads a file, extracts the data and computes the structures for the fingerprints
81     data=open(fname, "rt").readlines()
82     dm, crds, aaas = extractData(data)
83     fps=compute_fps(dm, crds, aaas, max_dist)
84
85     return fps
86
87 def h(fp_set, n=16381):
88     #hash and modulus function
89     fp = [0]*n
90     hs = [hash(h) % n for h in fp_set]
91     hs = set(hs)
92     for item in hs:
93         fp[item] = 1
94     return fp
95
96 # get protein ids retrieved from uniprot
97 ids = []
98
99
100 #this is where everything starts. Reads a list of uniprot ids from a file
101 with open(r'ids.txt', 'r') as fp:
102     for line in fp:
103         x = line[:-1]
104         ids.append(x)
105
106
107 #for each id, open the corresponding PDB structure from AlphaFold,
108 # creates the hash maps (fingerprints)
109 # writes the fingerprints in Python's pickle format
110
111 for i in ids:
112     dist = 5
113     fname="pdb_files/{i}.pdb".format(i)
114     fps= getFPs(fname, dist)
115     fps = h(fps)
116     pickle.dump(fps, open("fp_dist{}_ki/{i}_D{}.pickle".format(dist,i,dist), "wb"))
117
118

```