Algorithm 1 Ensemble Voting of User Genome from GTDB to NCBI

Input: Initialize parsing genome GTDB-NCBI-r203 mapping metadata with training genome capacity $\mathcal{N}$

Input: Initialize reading user genome Q and tree T classified with GTDB-Tk

1: Fixed ensemble weights θ, θ' to constant 0.5

2:

3: **for** $genome = 1, N$ **do**

4:     Associating genome accession ID gid with NCBI taxonomy in Dict $\mathcal{D}_{ncbi\_taxa}$

5:     Associating representative genome ID rid with GTDB taxonomy in Dict $\mathcal{D}_{gtdb\_taxa}$

6:     Clustering genome gid into representative genome rid in Dict $\mathcal{D}_{cluster}$

7:     Associating specific $taxon$ with NCBI taxonomy lineage in Dict $\mathcal{D}_{ncbi\_lineage}$

8:     Storing $\mathcal{D}_{ncbi\_taxa}$, $\mathcal{D}_{gtdb\_taxa}$, $\mathcal{D}_{cluster}$ and $\mathcal{D}_{ncbi\_lineage}$

9: **end for**

10:

11: Initializing GTDB cluster with NCBI-type taxonomy in Dict $\mathcal{N}4\mathcal{G}_{cluster}$

12: **for** $rid, gids$ in $\mathcal{D}_{ncbi\_taxa}$.items() **do**

13:     **for** $rank = r_{species}, r_{kindom}$ **do**

14:         Initializing hit list for $\mathcal{D}_{ncbi\_taxa}$ with $\mathcal{L}_{hit}$

15:         **for** ε in $gids$ **do**

16:             **if** ε in $\mathcal{D}_{ncbi\_taxa}$ **then**

17:                 appending $\mathcal{D}_{ncbi\_taxa}[\varepsilon][rank]$ into $\mathcal{L}_{hit}$

18:             **end if**

19:         **end for**

20:         **if** $\mathcal{L}_{hit}$ is *NOT* empty **then**

21:             Counting element occurence in $\mathcal{L}_{hit}$

22:             Applying heapsort to $\mathcal{L}_{hit}$ and finding top 1 element

23:             $tax_{top}, count_{top} = \mathcal{HEAPSORT}(\mathcal{L}_{hit})$

24:             **if** $count_{top} >= \theta * \mathcal{L}_{hit}$.size() && tax_{top} is *NOT* unassigned **then**

25:                 $\mathcal{N}4\mathcal{G}_{cluster}[rid] = \mathcal{D}_{ncbi\_lineage}[tax_{top}]$

26:                 **break**

27:             **end if**

28:         **end if**

29:         **if** $\mathcal{N}4\mathcal{G}_{cluster}[rid]$ is *NOT* unassigned **then**

30:             Reporting representative genome cannot be converted to NCBI taxonomy

31:         **end if**

32:     **end for**

33:     Storing $\mathcal{N}4\mathcal{G}_{cluster}[rid]$

34: **end for**

35:

36: Mapping training genome or user genome from GTDB taxonomy to NCBI taxonomy
37: Initializing final hit list as $\mathcal{L}_{hit_{final}}$
38: **for** $rank = r_{species}, r_{kindom}$ **do**
39:     **if** $\mathcal{D}_{gtdb\_taxa}[rank]$ is *unassigned* **then**
40:         Continue
41:     **end if**
42:     Initializing hit list for $\mathcal{D}_{ncbi\_taxa}$ with $\mathcal{L}_{hit_{genome}}$
43:     Mapping genome accession gid to tree leaf node $\mathcal{NODE}_{map}$
44:     traversing leaf nodes $\mathcal{NODE}_{map}$ to find NCBI decendants with $\mathcal{N}4\mathcal{G}_{cluster}[rid]$
45:     Storing $\mathcal{N}4\mathcal{G}_{cluster_{hit}}$ as list $\mathcal{L}_{cluster_{hit}}$
46:     **for** rid in $\mathcal{L}_{cluster_{hit}}$ **do**
47:         appending $\mathcal{N}4\mathcal{G}_{cluster}[rid][rank]$ into $\mathcal{L}_{hit_{genome}}$
48:         Counting element occurence in $\mathcal{L}_{hit_{genome}}$
49:         Applying heapsort to $\mathcal{L}_{hit_{genome}}$ and finding top 1 element
50:         $tax_{top}, count_{top} = \mathcal{HEAPSORT}(\mathcal{L}_{hit_{genome}})$
51:         **if** $count_{top} >= \theta' * \mathcal{L}_{hit_{genome}}$.size() && tax_{top} is *NOT* unassigned **then**
52:             $\mathcal{L}_{hit_{final}} = \mathcal{D}_{ncbi\_lineage}[tax_{top}]$
53:             **break**
54:         **end if**
55:     **end for**
56:     Storing $\mathcal{L}_{hit_{final}}$
57: **end for**

Algorithm 2 Rebuilding NCBI-Tree Denovo with Custom Taxonomy

Input: Custom taxonomy table with capacity $\mathcal{C}$ with column taxa as List $\mathcal{L}_{taxa7}$
Output: NCBI-TREE nodes.dmp and names.dmp for custom taxonomy

    Initializing NCBI-Tree as Directed Acyclic Graph(DAG) with Dict $\mathcal{D}_{dag}$
2: Initializing vertices and edges in DAG with List $\mathcal{L}_{vertices}$ and $\mathcal{L}_{edges}$
    Initializing taxa node ever seen with Dict $\mathcal{D}_{taxa_{seen}}$
4: Initializing taxa node ID with Dict $\mathcal{D}_{taxid_{node}}$

6: **for** $genome = 1, \mathcal{C}$ **do**
    for $\varepsilon, taxon$ in enumerate($\mathcal{L}_{taxa7}$) **do**
8:      Adding Vertex $taxon$ into $\mathcal{D}_{dag}$ when not in $\mathcal{D}_{dag}$, or nothing to do
       if $taxon\ NOT$ in $\mathcal{D}_{dag}$ **then**
10:       $\mathcal{D}_{dag}[taxon] = []$
        Updating Taxa Node ID with occurence order of incrementing by 1
12:       $\mathcal{D}_{taxid_{node}}[taxon] = \text{len}(\mathcal{D}_{taxid_{node}}) + 1$
      end if
14:

      Updating Edges from root to leaf(RTL) in Linked List
16:     **if** $\varepsilon = 0$ **then**
       Adding Edge $taxon$ from root of $\mathcal{D}_{dag}$
18:       $\mathcal{D}_{dag}[root] = [taxon]$
     else
20:      Adding Edge $taxon$ to $taxon_{before}$ denoted with $\mathcal{L}_{taxa7}[\varepsilon - 1]$ of $\mathcal{D}_{dag}$
       $\mathcal{D}_{dag}[taxon_{before}] = [taxon]$
22:     **end if**
    end for
24: **end for**

26: Recursive DFS to generate node.dmp and names.dmp
    Initializing nodes and names as List $\mathcal{L}_{name}$ and $\mathcal{L}_{node}$
28: DFS($taxon$, $\mathcal{L}_{name}$, $\mathcal{L}_{node}$)
    for $taxon_{child}$ in $\mathcal{D}_{dag}[taxon]$ **do**
30: **if** $taxon_{child}$ in $\mathcal{D}_{taxa_{seen}}$ **then**
     Continue
32: **end if**
    $\mathcal{D}_{taxa_{seen}}[taxon_{child}] = flag$
34:

    Appending nodes with $taxid, parent_id, rank, ...$ format
36: $\mathcal{L}_{node}.\text{append}(\mathcal{D}_{taxid_{node}}[taxon_{child}], \mathcal{D}_{taxid_{node}}[taxon], \mathcal{D}_{rank}[taxon_{child}], ...)$

    Appending names with $taxid, name, 'scientific name'$ format
38: $\mathcal{L}_{name}.\text{append}(\mathcal{D}_{taxid_{node}}[taxon_{child}], taxon_{child}, \text{'scientific name'})$
40: DFS($taxon_{child}$, $\mathcal{L}_{name}$, $\mathcal{L}_{node}$)
    end for
42: