Python File & Path	Description
./Deepurify/CallGenesTools/prodigal.py	This file implements functions to run Prodigal on FASTA data. It is capable of running Prodigal using multiple threads and extracting essential information from the Prodigal output.
./Deepurify/CallGenesTools/CallGenesUtils.py	This file implements functions to identify single-copy genes (SCGs) within a FASTA file. It initiates the Prodigal tool to locate open reading frames (ORFs) and subsequently employs Hidden Markov Models (HMM) to identify the SCGs. This procedure can be run with multiple threads.
./Deepurify/DataTools/DataUtils.py	This file implements functions for data processing tasks and for creating a taxonomic tree structured in a JSON format.
./Deepurify/FilterBinsTools/FilterUtils.py	This file implements functions to either eliminate or divide contigs within a FASTA file based on predicted taxonomic lineages and identified single-copy genes (SCGs). This process can be executed efficiently with the utilization of multiple threads.
./Deepurify/LabelContigTools/LabelBinUtils.py	This file implements functions to assign taxonomic lineages to contigs present in FASTA files. This task can be achieved through two distinct searching methods, the greedy search and the top-k search, which aim to identify the most closely related taxonomic lineages for each contig within the taxonomic tree. Furthermore, this process can be expedited by using the power of GPUs and multiple threads.
./Deepurify/Model/Convolutions.py	This file implements the modified EfficientNet architecture.
./Deepurify/Model/Embedding.py	This file implements the positional embedding.
./Deepurify/Model/EncoderModels.py	This file implements the Deepurify model, composed of a GseqFormer for sequence encoding and an LSTM for taxonomic lineage encoding.
./Deepurify/Model/FormerLayers.py	This file implements the modified UniFormer architecture.
./Deepurify/Model/Loss.py	This file implements the focal loss for both binary cross-entropy and cross-entropy loss.
./Deepurify/SelectMAGsTools/SelectionUitls.py	This file implements a depth-first search algorithm to discover the largest possible count of high- and medium-quality Metagenome-Assembled Genomes (MAGs) from a MAG-separated tree.
./Deepurify/SeqProcessTools/SequenceDataset.py	This file implements the dataset class for training Deepurify.
./Deepurify/SeqProcessTools/SequenceUtils.py	This file implements a collection of functions designed to handle DNA sequences and taxonomic lineags. Among its capabilities are functions for embedding DNA sequences into matrices and generating negative taxonomic lineages during the training stage.
./Deepurify/TrainTools/Scheduler.py	This file implements learning rate warmup with cosine decay schedule.
./Deepurify/TrainTools/TrainUtils.py	This file contains the implementation of the training procedure for Deepurify.
./Deepurify/IOUtils.py	This file contains the IO functions for Deepurify, including operations like reading and writing FASTA files.
./Deepurify/clean_func.py	This file implements a 'cleanMAG' function that serves as the central component linking the entire decontamination workflow.
./Deepurify/funcs.py	This file implements a function that connects the whole decontamination workflow, and additionally, it includes functions for running CheckM in parallel.
./Deepurify/cli.py	The implementation of the Bash commands for Deepurify.
./TrainScript.py	The script uses for configuring and training Deepurify.