OptiK User Manual

Overview

OptiK is a command-line tool for unsupervised k-mer size optimization in genome analysis. It computes k-mer frequency matrices across a specified range of k values, reduces dimensionality via truncated SVD, performs clustering, evaluates clustering quality using internal metrics, and identifies the optimal k-mer size based on a rank-based consensus of these metrics.

Installation

1. Clone the repository:

   ```
   git clone https://github.com/yourusername/OptiK.git
   cd OptiK
   ```

2. Create a virtual environment and install dependencies:

   ```
   python3 -m venv venv
   source venv/bin/activate
   pip install -r requirements.txt
   ```

   Alternatively, use Conda:

   ```
   conda env create -f environment.yml
   conda activate optik
   ```

Usage

Basic command:

```
python3 optik_v2.py -i <input_dir> -o <output_dir>
```

Required Arguments

- -i, --input: Directory containing genome FASTA files
- -o, --output: Directory where results will be stored

Optional Arguments

- --k-range K_MIN K_MAX: Range of k-mer sizes to evaluate (default: 3 8)
- --clusterer: Clustering algorithm (kmeans or agglomerative; default: kmeans)
- --plot-umap: Generate UMAP projections for visualization
- --random-seed: Seed for reproducibility (default: 42)
- --svd-components: Number of components for truncated SVD (default: auto)

- --n-clusters MIN MAX: Range of cluster numbers to test (default: 3 8)
- --threads: Number of threads for parallel processing (default: 1)

Output Files

- metrics.csv: Clustering scores for each k and cluster count
- best_k.txt: Selected optimal k-mer size
- cluster_assignments_kX.csv: Cluster assignments for optimal k
- umap_kX.png: UMAP plot if visualization is enabled
- svd_kX.npy, matrix_kX.npy: Intermediate data matrices

How OptiK Chooses the Optimal k

For each k:

1. Compute average Silhouette, Calinski-Harabasz, and Davies-Bouldin scores across cluster counts
2. Rank each k within each metric
3. Compute cumulative rank score
4. Select k with lowest total rank
5. Break ties by favoring higher Silhouette and lower Davies-Bouldin scores

Reproducibility

- Set --random-seed for deterministic output
- All configuration parameters and outputs are saved to the output directory

Example

```
python3 optik_v2.py \
  -i ./data/HpGP/ \
  -o ./results/ \
  --k-range 3 8 \
  --clusterer kmeans \
  --plot-umap \
  --random-seed 42
```

Notes

- Input FASTA files should be high-quality assembled genomes.
- For metagenomic or draft assemblies, preprocessing and normalization may be required.

- For best performance on large datasets, use lower k_max or activate parallel threads.

License

Citation

Gutierrez Escobar AJ. OptiK: An Entropy-Driven Framework for Optimal k-mer Size Selection in Comparative Genomics. bioRxiv (2024).