Comprehensive Evaluation Report on Wastewater Metagenomic Viral-Detection Pipelines

Sponsor: U.S. Centers for Disease Control and Prevention (CDC)

Evaluation Environment: UGA Sapelo2 HPC

Pipeline GitHub repository:

https://github.com/pengsihua2023/WasteWater-Metagenomics-Pipeline-Evaluation

Project completed by: Center for Applied Pathogen Epidemiology, UGA

Group Leader: Dr. Justin Bahl

Prepared by (Report Drafter): Dr. Sihua Peng

Report Compilation Date: February 18th, 2026

Distribution: Internal / For CDC Review

Contents

Comprehensive Evaluation Report on Wastewater Metagenomic Viral-Detection Pipelines

Sponsor: U.S. Centers for Disease Control and Prevention (CDC)
Project: National Wastewater Surveillance System (NWSS) — Evaluation of Metagenomic Viral-Detection Pipelines
Evaluation Environment: UGA Sapelo2 HPC cluster (SLURM scheduler, no sudo/root privileges, Apptainer security-policy constraints)
Integrated Version: v1.0

Abstract

This report evaluates eight deployable wastewater metagenomic viral-detection pipelines. All benchmarking and stress testing were completed on the UGA Sapelo2 HPC cluster, an environment representative of public-health computing infrastructure with strict security constraints (user-space execution and no root privileges).

Key findings

1. **No single tool fits all needs**: Under the high-noise background typical of wastewater, a *tiered strategy* ("screen → confirm → deep evidence") is required rather than relying on one tool to deliver final conclusions.

2. **Deployability is a hard gate**: Five candidate tools (CZID/IDseq, SURPI+, DHO Lab, NAO MGS, TaxTriage) could not be deployed on the target HPC environment due to sudo/root requirements or complex container-build workflows incompatible with Sapelo2 policies.

3. **Assembly is the computational bottleneck**: On large real-world data (289 GB of short reads), assembly steps reached **~768 GB peak RAM**, dominating wall-clock time and resource consumption.

4. **False-positive control is mandatory**: Low-abundance hits from a single tool should **not** be reported as final results without multi-tool confirmation and/or assembly-based evidence.

Recommended operational strategy

Tier	Pipeline	Primary role
Tier 1 (Routine screening)	TaxProfiler (nf-core/taxprofiler)	High-throughput standardized abundance reporting; production entry-point
Tier 2 (Confirmation/verification)	GOTTCHA2, sourmash	High-specificity verification; rapid consistency check
Tier 3 (Deep investigation)	rvdb-viral-metagenome-nf, MLMVD-nf	Novel-virus discovery; strengthened evidence chain

Core recommendation: Establish **TaxProfiler** as the standard daily monitoring entry pipeline. Any *screen-positive* signals should be **mandatory** confirmed by RVDB-based evidence or an assembly-informed workflow such as KrakenMetaReads-nf. Reports should label results explicitly as **screen-positive**, **supportive**, or **confirmed**.

1. Project background and scope

1.1 Background

The CDC NWSS uses metagenomic sequencing to detect and monitor viruses in wastewater. Wastewater samples present several unique challenges:

- **High complexity**: mixtures of human, bacterial, viral, and chemical DNA/RNA. Large host DNA loads, diverse bacteria and phages, and other microbial nucleic acids create extreme background noise, complicating viral signal separation.
- **Fragmented signals**: viral genomes often exist as degraded fragments. Extracted nucleic acids are frequently short and fragmented, reducing the effectiveness of traditional whole-genome alignment strategies.
- **High risk of false positives**: cross-taxon misclassification and database noise. Incompletely characterized microbial sequences and limited reference coverage can lead to incorrect viral calls or incorrect taxonomic assignments.

- **Scale requirement**: high-throughput processing with reproducible outputs. Nationwide monitoring demands pipelines that can handle large sample volumes while producing consistent, auditable results.

This evaluation focuses on the real operational needs of CDC wastewater surveillance, conducting strict HPC deployment checks and performance stress tests for candidate pipelines.

1.2 Pipelines evaluated (deployable)

This evaluation covers the following eight deployable pipelines:

Category	Pipeline	Core method	Source
CDC-recommended (deployed)	nf-core/taxprofiler (MetaTaxProfiler)	Read-level classification + standardized abundance	nf-core standard
CDC-recommended (deployed)	GOTTCHA2	Signature-driven conservative classifier	Standalone CLI tool
CDC-recommended (deployed)	sourmash	MinHash similarity screening	Standalone CLI tool
CDC-recommended (deployed)	CLARK (CLARK/CLARK-S/CLARK-l)	Discriminative k-mer classification	Standalone CLI tool
CDC-recommended (deployed)	nf-core/mag (mag-nf)	Assembly + MAG reconstruction	nf-core standard
Author-developed	rvdb-viral-metagenome-nf	Assembly + protein-homology discovery	Custom Nextflow pipeline
Author-developed	MLMVD-nf	ML-enhanced multi-tool consensus	Custom Nextflow pipeline
Author-developed	KrakenMetaReads-nf	Assemble-first then classify	Custom Nextflow pipeline

Together these pipelines span mainstream approaches for wastewater metagenomic viral detection: fast k-mer methods, assembly-based evidence, protein-level homology for distant viruses, and conservative multi-tool consensus strategies.

1.3 Tools excluded (deployment failures)

The following five tools could not complete deployment evaluation due to incompatibility with the target HPC security model:

Tool	Failure reason	Details
CZID (IDseq)	Cloud-native architecture; not HPC-oriented	Designed to run primarily in AWS/cloud backends; not deployable on an isolated HPC without cloud resources
SURPI+	Requires sudo; incompatible container-build workflow	Needs root privileges (e.g., for Docker builds and kernel-level operations), conflicting with Sapelo2 Apptainer security policies
DHO Lab	Complex system dependencies; needs sudo	Requires system-level libraries/modules needing root installation
NAO MGS	Complex system dependencies; needs sudo	Similar system-level dependency chain requiring root
TaxTriage	Complex dependencies; needs sudo	Installation scripts assume administrator privileges and are not designed for unprivileged user environments

1.4 Datasets

Two datasets of different scales were used to characterize pipeline behavior:

Run ID	Dataset type	Short-read input	Short-read size	Long-read input	Long-read size	Relative scale
#1	Simulated dataset (CDC benchmark)	llnl_66ce4dde_R1/R2.fastq.gz	1.36 GB	llnl_66d1047e.fastq.gz	264 MB	1×
#2	Real wastewater sample	SRR35987572_1/2.fastq.gz	289 GB	—	—	212.5×

Run #1 (simulated): - Synthetic community with known viral composition *(metadata not provided to the evaluation team at runtime)*

- Intended to provide ground truth for sensitivity calculations *(ground-truth file not available during Round 1)*
- Suitable for baseline benchmarking
- Small enough for rapid iteration

Run #2 (real wastewater): - Real sample with high complexity and true background noise

- Large host/bacterial content typical of production scenarios
- 289 GB short reads, representative of production-scale throughput
- Used to identify upper bounds of runtime and memory requirements

1.5 Key assumptions

1. Conclusions are drawn for the **UGA Sapelo2 HPC environment** (no sudo/root; Apptainer constraints) and generalize to public-health HPC sites with similar security policies.

2. Performance statements are based on **empirical measurements** from this project rather than idealized laboratory benchmarks. Hardware, database versions, and settings may shift absolute values in other deployments.

3. **Database versions strongly influence results**; this evaluation assumes the then-latest available versions at the time of testing.

4. **Evidence hierarchy**: assembled **contig** evidence is stronger than read counts; **protein-homology** evidence is stronger than nucleotide k-mer matching when targeting distant viruses.

2. Evaluation dimensions and methods

2.1 Evaluation framework

We used a seven-dimension framework, each with explicit criteria and evidence-capture points:

Dimension	Definition	Decision criteria
Detection performance	Sensitivity for known viruses; low-abundance behavior; false-positive control; novel-virus potential	Empirical detection on benchmark data; method-based assessment (k-mer, assembly, protein homology, ML)
Reproducibility & portability	Consistency across environments	Containerization (Apptainer/Docker); version pinning; workflow engine (Nextflow); database version tracking
Computational efficiency	Resource consumption & scalability	Wall time; peak memory; CPU utilization; parallelization
Maintainability	Ease of updates & debugging	Code structure; logging completeness; community support
Interpretability & auditability	Trustworthiness of results	Evidence-chain completeness (reads/contigs/proteins); confidence scoring; standardized outputs
Usability	Learning curve	Documentation completeness; reasonable defaults; entry barrier
Compliance & risk	Licensing & interpretation risks	Database provenance; platform dependence risks

Detection-performance notes: - *Sensitivity for known viruses*: ideally computed using ground-truth labels from benchmark datasets
- *Low-abundance performance*: ability to detect weak viral signals
- *False-positive control*: specificity in negative/noise-heavy conditions
- *Novel/distant viruses*: ability to detect viruses not well represented in nucleotide databases

2.2 Evidence-tiering framework

To avoid contradictory "highest sensitivity" vs "highest specificity" conclusions, we distinguish:

- **Default output**: a pipeline's standard output (often higher sensitivity but more noise).

- **Consensus/threshold output**: output after multi-tool cross-checking, host removal, and coverage/assembly thresholds (often higher specificity but reduced sensitivity).

This distinction is essential for public-health reporting, because false positives can trigger unnecessary responses, while false negatives can miss emerging threats.

2.3 Evidence levels

Level	Definition	Minimum requirement
Screen-positive	Single-tool hit above threshold	One tool + threshold filtering
Supportive	Multi-tool agreement or orthogonal support	≥2 independent tools **OR** assembly evidence
Confirmed	Complete multi-evidence chain	≥2 tools + contig (>5 kb) + coverage support

Screen-positive: - Detected by one tool
- Above abundance threshold (e.g., RPM ≥ 10)
- Not cross-validated
- Must be marked "pending verification" and should not drive public-health action alone

Supportive: - Reported by ≥2 independent tools, **or** one tool plus assembly evidence
- Can serve as supportive evidence, but still requires careful interpretation

Confirmed: - Confirmed by multiple tools

- Assembled contig >5 kb

- Reasonable coverage distribution (not a single hotspot artifact)

- Suitable as a confirmed call for reporting

2.4 Pipeline strategy categories

The eight pipelines fall into four methodological categories:

Strategy type	Representative pipelines	Evidence type	Notes
Read-level classification	TaxProfiler, CLARK, GOTTCHA2	Read-level k-mer/signature evidence	Fast, but evidence is short-fragment based
Assemble-then-classify	KrakenMetaReads-nf, nf-core/mag	Contig-level classification evidence	Stronger evidence; may lose low-abundance targets
Protein-homology evidence chain	rvdb-viral-metagenome-nf	DIAMOND vs RVDB protein evidence	Detects distant viruses; compute-intensive
Viral-feature + multi-tool consensus	MLMVD-nf	VirSorter2 + DeepVirFinder + viralFlye voting	Highest specificity; may miss true positives

3. Integrated findings (thematic)

3.1 Detection performance overview

3.1.1 Sensitivity for known viruses

Pipeline	Sensitivity (qualitative)	Core method	Notes
MetaTaxProfiler	★★★★★ (highest)	Kraken2 k-mer classification	Highest detection rate for RefSeq-covered viruses
rvdb-viral	★★★★☆ (high)	Assembly + DIAMOND protein homology	Protein layer can recover some k-mer misses
GOTTCHA2	★★★☆☆ (medium)	Unique-signature mapping	Conservative; may miss variants
sourmash	★★★☆☆ (medium)	MinHash similarity	Fast screening; not strictly quantitative
CLARK	★★★☆☆ (medium)	Discriminative k-mer	Strong for in-database targets
KrakenMetaReads-nf	★★★☆☆ (medium)	Assemble-then-classify	Assembly may lose low-abundance targets
MLMVD-nf	★★☆☆☆ (low)	Conservative consensus	Designed for strict filtering; trades sensitivity
nf-core/mag	★★☆☆☆ (low)	Not detection-first	Primarily for MAG reconstruction

Conclusion: MetaTaxProfiler has the highest sensitivity for known viruses, followed by rvdb-viral.

3.1.2 Novel/distant virus discovery

Pipeline	Discovery potential	Core advantage	Best use
rvdb-viral	★★★★★ (highest)	DIAMOND + RVDB protein homology	Detects distant viruses at ~40–50% similarity
nf-core/mag	★★★★☆ (high)	Strong assembly; large DNA viruses	NCLDV / large DNA virus reconstruction
MLMVD-nf	★★★☆☆ (medium)	DeepVirFinder + VirSorter2	ML detection beyond direct homology
MetaTaxProfiler	★★★☆☆ (medium)	Kraken2 DB coverage	Depends on database inclusion
KrakenMetaReads-nf	★★★☆☆ (medium)	Contig-based classification	Depends on assembly quality
sourmash	★★☆☆☆ (low–medium)	k=31 MinHash	Moderate-complexity targets
GOTTCHA2	★☆☆☆☆ (low)	Conservative signatures	Highly similar targets only
CLARK	★☆☆☆☆ (low)	Strict k-mer	In-database targets only

Conclusion: rvdb-viral is decisively strongest for distant/environmental viruses; protein-homology detection can reach targets that nucleotide k-mer tools typically miss.

3.1.3 Specificity and false-positive control

Pipeline	Default specificity	After consensus/threshold	Key control mechanism
GOTTCHA2	★★★★★ (highest)	High	Unique signatures avoid shared regions
MLMVD-nf	★★★★☆ (high)	Built-in multi-tool consensus	3-tool > 2-tool > 1-tool tiering
CLARK	★★★★☆ (high)	High	Discriminative k-mers
nf-core/mag	★★★★☆ (high)	High	MAG QC (CheckM/GUNC)
sourmash	★★★★☆ (med–high)	Medium	Containment thresholds
rvdb-viral	★★★☆☆ (medium)	High	Dual-track + protein validation
KrakenMetaReads-nf	★★★☆☆ (medium)	Med–high	Assembly evidence
MetaTaxProfiler	★★★☆☆ (medium)	High	Threshold + host removal + multi-evidence

Key point: Under default settings, *all* tools can generate false positives; practical use requires thresholds, host removal, and cross-tool confirmation.

3.2 Computational requirements

3.2.1 Empirical resource consumption

Run	Data scale	Total runtime	Peak RAM	Primary bottleneck
#1	1.36 GB short reads	≤48 hours	256 GB	Overall pipeline limits
#2	289 GB short reads	2–12 days	**768 GB**	Metagenome assemblers

Key observations:

1. **Assembly dominates resources**: On 289 GB data, metaSPAdes consumed most memory and wall time; de Bruijn graph construction scales roughly with input volume.

2. **Read-level classifiers are more stable**: MetaTaxProfiler, CLARK, and sourmash typically stay around ~100–200 GB memory since they avoid assembly.

3. **Nonlinear scaling**: From 1× to 212× data size, assembly-based workflows scale superlinearly, meaning runtime grows faster than data volume.

3.2.2 Typical pipeline-level requirements

Pipeline	Typical time (1×)	Peak RAM	Resource intensity	Main time driver
sourmash	2–36 h	256 GB	★★★★★ (lowest)	Sketching
CLARK	2–24 h	58–156 GB	★★★★☆	DB loading
GOTTCHA2	8–27 h	256 GB	★★★☆☆	Signature alignment
MetaTaxProfiler	12–32 h	256 GB	★★★☆☆	Kraken2 classification
KrakenMetaReads-nf	24–264 h	768 GB	★★☆☆☆	Assembly + classification
rvdb-viral	48–288 h	512 GB	★☆☆☆☆	Assembly + DIAMOND
MLMVD-nf	48–288 h	256 GB	★☆☆☆☆	Multi-tool execution
nf-core/mag	5–264 h	256 GB	★★☆☆☆	Assembly + binning

3.3 Engineering maturity

3.3.1 Deployability matrix

Pipeline	Containerization	Nextflow support	No-sudo execution	Overall maturity
MetaTaxProfiler	☑ Apptainer (complete)	☑ nf-core	☑	★★★★★
nf-core/mag	☑ Apptainer (complete)	☑ nf-core	☑	★★★★★
KrakenMetaReads-nf	☑ Hybrid	☑ DSL2	☑	★★★★☆
rvdb-viral	⚠ Conda-first	☑ DSL2	☑	★★★★☆
sourmash	⚠ Optional	☑ DSL2	☑	★★★☆☆
MLMVD-nf	⚠ Hybrid	☑ DSL2	☑	★★★☆☆
GOTTCHA2	✗ Conda only	✗ scripts	☑	★★☆☆☆
CLARK	✗ host install	✗ scripts	☑	★★☆☆☆

Key observations: 1. **nf-core pipelines are strongest operationally**: MetaTaxProfiler and nf-core/mag provide robust logs, resume/checkpointing, and batch processing. 2. **Containerization is critical for reproducibility**: Fully containerized Apptainer pipelines are far more portable across sites. 3. **Standalone scripts increase maintenance cost**: GOTTCHA2 and CLARK require manual environment management and are harder to migrate.

3.4 Risks and compliance

3.4.1 Technical risks

Risk	Affected pipelines	Severity	Mitigation
Database coverage bias	All read-level classifiers	High	Use rvdb-viral discovery mode; combine multiple databases
Assembly dropouts	rvdb-viral, MLMVD-nf, KrakenMetaReads	Medium	Run read-level classifier in parallel to capture low-abundance signals
Database drift	All DB-dependent tools	Medium	Quarterly DB updates; snapshot & audit versions
Accumulated false positives	All read-level classifiers	High	Enforce thresholds + multi-tool confirmation
Container incompatibility	CLARK, GOTTCHA2	Low	Provide Conda fallback environments

3.4.2 Operational risks

Risk	Description	Mitigation
Deployment failure	Five CDC candidate tools cannot run on the HPC target	Prefer containerized pipelines; verify compatibility before adoption
Resource exhaustion	Large-scale assembly can exceed memory	Tiered execution: screen all samples; assemble only flagged samples
Interpretation inconsistency	Different analysts may grade evidence differently	Publish CDC internal interpretation SOP; use fixed decision rules

3.5 Long-term scalability

3.5.1 Tiered execution strategy

Scenario	Tier	Resource need	Typical pipelines	Sample share
Routine monitoring	Tier 1 full screening	Low (~100–200 GB)	MetaTaxProfiler	95%
Flagged-sample confirmation	Tier 2 verification	Medium (~256 GB)	GOTTCHA2, sourmash	4%
Deep investigation	Tier 3 assembly/discovery	High (~512–768 GB)	rvdb-viral, MLMVD-nf	0.5%
Genome reconstruction	Tier 4 MAG	Highest (≥768 GB)	nf-core/mag	0.5%

3.5.2 Long-read support

Pipeline	Short-read	Long-read	Long-read optimization
MetaTaxProfiler	☑	☑	NanoPlot/Porechop
rvdb-viral	☑	☑	metaFlye + viralFlye dual-track
MLMVD-nf	☑	☑	3-tool consensus
KrakenMetaReads-nf	☑	☑	viralFlye circular/linear distinction
GOTTCHA2	☑	☑	Minimap2 alignment
nf-core/mag	☑	☑	Flye container
sourmash	☑	☑	Direct sketching
CLARK	☑	⚠ CLARK-S	Spaced k-mer

3.6 Application mapping

Use case	Recommended pipeline(s)	Confidence level	Output type
Routine wastewater monitoring	MetaTaxProfiler → GOTTCHA2	Screen-positive → Supportive	RPM abundance tables
Outbreak tracking	MetaTaxProfiler → rvdb-viral	Supportive → Confirmed	Contigs + protein evidence
Novel virus discovery	rvdb-viral → MLMVD-nf	Evidence chain	High-confidence candidate set
Quality control	sourmash	Screening	MinHash signatures
Ecological studies	nf-core/mag	MAG evidence	Reconstructed genomes
Resource-limited scenario	sourmash / CLARK-l	Screening only	Fast classification

4. Pipeline profiles (key features)

4.1 TaxProfiler (MetaTaxProfiler / nf-core/taxprofiler)

Role: production-grade high-throughput screening (“front door”)

Core workflow: - Short reads: FastQC/fastp → Kraken2 → (optional Bracken) → RPM calculation
- Long reads: NanoPlot/Porechop → Kraken2 → RPM calculation

Strengths: - Highest engineering maturity (nf-core standard) - Standardized tables for cross-batch comparisons - Strong batch mode; supports -resume - Supports Illumina + Nanopore/PacBio

Limitations: - Primarily read-level evidence; limited assembly/protein chain - Default output requires thresholds to control false positives - Novel virus discovery depends on database coverage

Recommended: routine monitoring; weekly reports; batch screening

4.2 rvdb-viral-metagenome-nf

Role: deep discovery + protein-homology evidence chain

Core workflow: - Short reads: MEGAHIT + metaSPAdes assembly → Prodigal → DIAMOND vs RVDB → taxonomy enrichment
- Long reads: metaFlye + viralFlye dual-track → Prodigal → DIAMOND → dual-track consensus

Strengths: - Best distant/environmental virus discovery (protein homology rather than k-mer) - Suitable for NCLDV/large DNA viruses - Dual-track comparisons enable confidence stratification - End-to-end auditable evidence chain

Limitations: - Highest compute demands (SPAdes may require ~512 GB RAM) - Longest runtime - More complex environment configuration

Recommended: incident response; difficult-sample adjudication; novel virus exploration

4.3 MLMVD-nf (ML-enhanced viral discovery)

Role: strict consensus engine for high-confidence candidates

Core workflow: - Short reads: MEGAHIT + metaSPAdes → VirSorter2 + DeepVirFinder
- Long reads: metaFlye → VirSorter2 + DeepVirFinder + viralFlye → 3-level confidence tiers

Strengths: - Built-in multi-tool consensus yields very high specificity - Explicit three-level confidence labeling - DeepVirFinder supports detection beyond direct sequence homology - viralFlye provides Pfam domain validation

Limitations: - Lowest sensitivity (intentionally conservative) - Complex dependency chain (multiple Conda environments) - Long runtime

Recommended: generating high-confidence candidates; prioritizing wet-lab validation targets

4.4 KrakenMetaReads-nf

Role: assemble-first, context-aware classification

Core workflow: - Short reads: fastp → MEGAHIT/SPAdes → Bowtie2 re-mapping → Kraken2 classification → RPM/RPKM
- Long reads: metaFlye/viralFlye → Minimap2 re-mapping → Kraken2 classification

Strengths: - Contig-level classification reduces read-level ambiguity - Supports short/long reads - Dual-assembler cross-check - Includes virus consensus outputs

Limitations: - Assembly can lose low-abundance viruses - Higher resource consumption than read-only methods

Recommended: confirmation requiring contig evidence; adjudicating flagged samples

4.5 GOTTCHA2

Role: conservative signature-driven confirmation tool

Core method: Minimap2 mapping to GOTTCHA2 species-level signature database

Strengths: - Highest specificity (unique signatures avoid shared regions) - Moderate resource demand; fast execution - Strong confirmatory utility

Limitations: - Sensitivity may be lower (conservative design) - Limited novel/distant virus capability

Recommended: confirmatory re-check for TaxProfiler positives

4.6 sourmash

Role: extremely fast MinHash similarity screening

Core method: sourmash sketch → sourmash gather vs viral databases

Strengths: - Fastest runtime - Low memory footprint - Useful containment checks

Limitations: - Default mode does not track abundance - Not strictly quantitative - Limited novel-virus capability

Recommended: large-scale pre-screening; rapid sample comparisons; QC

4.7 CLARK / CLARK-S / CLARK-l

Role: fast discriminative k-mer classifier

Core method: k-mer decomposition → indexed lookup → taxonomic assignment

Strengths: - Very high speed - Multiple modes for different resource profiles - CLARK-l low-memory mode (≈4 GB)

Limitations: - Weak container support - Limited out-of-database detection - Short evidence chain

Recommended: rapid initial screening for known targets *(not recommended as the sole basis for CDC conclusions)*

4.8 nf-core/mag (mag-nf)

Role: genome-reconstruction oriented MAG evidence

Core workflow: assembly → (optional binning) → MAG QC (CheckM/GUNC)

Strengths: - Best genome reconstruction capability - MAG QC provides strong quality evidence - nf-core engineering standard

Limitations: - Not detection-first - High resource demand - Viral detection is not the primary design goal

Recommended: ecological studies; genomic context; deep structural analysis

5. Mapping tables

5.1 Pipeline-to-scenario mapping

Scenario	Recommended pipeline(s)	Confidence	Notes
Routine high-throughput screening	TaxProfiler	Screen-positive only	Standardized entry
High-specificity confirmation	GOTTCHA2	Confirm-positive	Signature verification
Novel virus discovery	rvdb-viral	Evidence chain required	Protein homology
Strict candidate set	MLMVD-nf	High confidence	Multi-tool consensus
Genome reconstruction	nf-core/mag	MAG evidence	Deep analysis
Contig classification	KrakenMetaReads-nf	Assembly evidence	Context-aware
Fast k-mer classification	CLARK	In-database targets	Supplementary
Fast containment check	sourmash	Screening only	Lightweight

5.2 Evaluation-weight matrix (summary scores)

Pipeline	Detection	Reproducibility	Efficiency	Maintainability	Interpretability	Usability	Overall
TaxProfiler	4	5	3	5	4	5	4.3
GOTTCHA2	4	4	5	4	4	5	4.3
KrakenMetaReads	4	5	2	5	4	4	4.0
nf-core/mag	3	5	2	5	5	3	3.8
MLMVD-nf	4	3	4	4	5	3	3.8
rvdb-viral	5	4	2	4	5	3	3.7
sourmash	3	4	3	4	3	4	3.5
CLARK	3	3	4	3	3	4	3.3

5.3 Deployment status

Pipeline	Deployable on Sapelo2	Container support	HPC-fit	Recommendation
TaxProfiler	✓	Apptainer (complete)	✓	**Primary**
GOTTCHA2	✓	Conda	✓	**Primary**
rvdb-viral	✓	Conda (default)	✓	**Primary**
nf-core/mag	✓	Apptainer (complete)	✓	Deep analysis
MLMVD-nf	✓	Apptainer + Conda	✓	Secondary
KrakenMetaReads-nf	✓	Apptainer + Conda	✓	Secondary
sourmash	✓	Apptainer/Conda	✓	Supplementary
CLARK	✓	Host install	✓	Supplementary

5.4 Glossary

Term	Definition
Sensitivity	Ability to avoid false negatives; higher sensitivity detects more true targets
Specificity	Ability to avoid false positives; higher specificity yields fewer false calls
Evidence chain	Multiple independent lines of evidence supporting a call
Contig	Assembled contiguous sequence fragment
MAG	Metagenome-assembled genome
RPM	Reads per million (normalized abundance)
RPKM	Reads per kilobase per million (length-normalized abundance)
NCLDV	Nucleocytoplasmic large DNA viruses
HPC	High-performance computing
Apptainer	Secure container runtime (formerly Singularity)

5.5 Database resources

Pipeline	Database	Update frequency	Source
MetaTaxProfiler	Kraken2/Bracken	Monthly	NCBI RefSeq
rvdb-viral	RVDB (DIAMOND)	2–3×/year	Pasteur Institute
GOTTCHA2	GOTTCHA2 signatures	Periodic	GitHub
CLARK	RefSeq	Manual	NCBI
sourmash	NCBI viruses (k=31)	Annual	NCBI
MLMVD-nf	VirSorter2 + DeepVirFinder	Tool/model dependent	Model-driven

5.6 Pipeline decision tree

Need high-throughput standardized monitoring?
├─ Yes → TaxProfiler (Tier-1 entry)
│ └─ Critical positives? → confirm with GOTTCHA2
│
Need novel/distant virus discovery?
├─ Yes → rvdb-viral (deep discovery)
│ └─ Need strict high-confidence candidates? → MLMVD-nf (3-tool consensus)
│
Need genome-level evidence?
├─ Yes → nf-core/mag (MAG reconstruction)
│
Need rapid pre-screen?
└─ Yes → sourmash (lightweight & fast)

6. Core conclusions

6.1 Five unanimous consensus points

Across nine independent internal evaluation notes, all analyses converged on five points:

#	Consensus statement	Evidence basis
1	**No single tool meets all needs**	All reports recommend a tiered strategy
2	**TaxProfiler is the primary screening tool**	Highest engineering maturity; nf-core standardization
3	**Deployability is the key gate**	Five CDC candidates failed HPC deployment
4	**Assembly is the main bottleneck**	Peak RAM increased from ~256 GB to ~768 GB at 212× scale
5	**False-positive control needs multi-evidence**	Single-tool hits must be labeled "screen-positive"

6.2 Strategic recommendations

1. **Adopt a tiered defense strategy**: stop searching for a single "best tool"; implement a graded response system (screen → confirm → deep dive).
2. **Standardize on TaxProfiler as the entry pipeline**: run it for all incoming samples to produce standardized abundance tables.
3. **Publish an evidence-tier SOP**: enforce screen-positive, supportive, confirmed labeling.
4. **Reserve high-memory resources**: allocate ≥768 GB nodes only to a small subset (<5%) of high-priority samples.
5. **Govern database versions**: quarterly updates with monthly snapshots and audit tracking.

7. Limitations

1. **Single real-world dataset at scale**: Run #2 included only one real wastewater sample and may not represent all wastewater types.
2. **Database time dependence**: results may change with database updates; routine re-evaluation is required.

3. **Limited ground truth**: lack of wet-lab–validated positives limits the certainty of sensitivity estimates.

4. **Operator variability not assessed**: analyst-to-analyst consistency using the same tools was not evaluated.

8. Technical deep-dive

8.1 Methodological comparison

8.1.1 k-mer classification

k-mer methods are widely used; they decompose reads into fixed-length k-mers and match them to a database index.

Principle: - Decompose reads into k-mers (e.g., k=31)
- Build k-mer → taxon index
- Count k-mer coverage per taxon
- Estimate relative abundance from coverage

Representative tools: Kraken2, CLARK

Pros: - Fast
- Often sensitive for in-database targets
- Mature tooling ecosystem

Cons: - Strongly dependent on database coverage
- Short k-mers can yield false positives in conserved regions
- Poor for distant viruses

8.1.2 Signature mapping

Uses longer unique signatures to improve specificity.

Principle: - Extract unique signatures from reference genomes

- Map reads to signatures using aligners

- Determine taxonomy from mapped signatures

Representative tool: GOTTCHA2

Pros: - Very high specificity

- Highly interpretable evidence

Cons: - Lower sensitivity (can miss variants/distant strains)

- Database construction can be complex

8.1.3 MinHash similarity

A probabilistic approximation to estimate set similarity quickly.

Principle: - Convert sequences into MinHash signatures

- Use Jaccard similarity estimates

- Use containment to estimate how much of a target is present

Representative tool: sourmash

Pros: - Extremely fast

- Memory-efficient signatures

- Useful for large-scale screening

Cons: - Not strictly quantitative by default

- Less suited for precise abundance estimation

8.1.4 Assemble-then-classify

Assemble reads before classification to improve context and reduce ambiguity.

Principle: - Assemble metagenomes using MEGAHIT or SPAdes

- Classify contigs against reference databases

- Compute abundance using contig coverage

Representative pipeline: KrakenMetaReads-nf

Pros: - Stronger evidence (contigs > reads)

- Potential to recover new genomic fragments

Cons: - Assembly is resource intensive

- Low-abundance targets may be lost

- Assembly errors can introduce artifacts

8.1.5 Protein homology

Uses predicted proteins and fast protein aligners to detect distant viruses.

Principle: - Predict genes with Prodigal

- Align proteins using DIAMOND

- Assign taxonomy based on protein similarity

Representative pipeline: rvdb-viral-metagenome-nf

Pros: - Better for distant viruses (protein similarity is more conserved)

- Complete evidence chain (contig → gene → protein)

Cons: - Very compute intensive

- Depends on gene-prediction quality

8.1.6 ML + consensus

Combines multiple tools and ML signals to improve specificity.

Principle: - Run multiple viral detectors (VirSorter2, DeepVirFinder, etc.)

- Vote or weight results

- Output consensus with confidence tiers

Representative pipeline: MLMVD-nf

Pros: - Highest specificity

- Clear confidence-tier outputs

Cons: - Sensitivity may decrease

- Complex dependencies

- Longer runtimes

8.2 Database dependence

8.2.1 Major databases

Database	Content	Update frequency	Pros/cons
NCBI RefSeq	Viral genomes	Monthly	Broad coverage; limited environmental viruses
Kraken2 standard	RefSeq viruses	Monthly	Fast; RefSeq-dependent
Kraken2 PlusPFP	Expanded library	Quarterly	Broader but larger
RVDB	Viral proteins	2–3×/year	Good for distant detection
GOTTCHA2 signatures	Species-level signatures	Periodic	High specificity; limited breadth

8.2.2 Database selection suggestions

- **Routine monitoring**: Kraken2 standard viral DB (monthly) + Bracken for abundance correction

- **Novel virus discovery**: add RVDB protein DB to capture distant viruses

- **High-confidence confirmation**: use GOTTCHA2 signatures as second-pass validation

8.3 Host-removal strategy

Host contamination is a major confounder in wastewater metagenomics; effective host removal is a prerequisite for quality calls.

8.3.1 Short-read host removal

Recommended tool: Bowtie2

Example:

bowtie2 -x host_index -1 R1.fastq.gz -2 R2.fastq.gz --un-conc-gz unpaired_%.fastq.gz -S /dev/null

Key points: - Use the latest human reference genome

- Consider adding common lab contaminants (e.g., *E. coli*)

- Record the host-removal rate for QC

8.3.2 Long-read host removal

Recommended tool: Minimap2

Example:

minimap2 -ax map-ont host_genome.mmi long_reads.fastq.gz | samtools view -f 4 -b > unaligned.bam

9. Implementation guide

9.1 Deployment checklist

9.1.1 Environment readiness

Item	Notes	Status
Apptainer installed	Verify version ≥ 3.0	☐
Nextflow installed	Recommend ≥ 23.04	☐
Storage	≥ 500 GB for data & DB	☐
Memory	Routine ≥ 256 GB; deep analysis ≥ 768 GB	☐

9.1.2 Database readiness

Item	Notes	Status
Kraken2 DB	Download & verify integrity	☐
Bracken DB	Version-matched to Kraken2	☐
Host index	Bowtie2-formatted human genome	☐
Version logging	Record all DB versions	☐

9.1.3 Test run

Item	Notes	Status
Simulated-data test	Validate pipeline on small dataset	☐
Resume test	Validate -resume behavior	☐
Output format	Confirm outputs meet expectations	☐
Resource monitoring	Record peak memory and time	☐

9.2 Standard operating procedures

9.2.1 Routine screening SOP

1. Raw QC (FastQC)
↓
2. Quality filtering (fastp)
↓
3. Host removal (Bowtie2)
↓
4. TaxProfiler analysis
↓
5. Abundance thresholding (RPM ≥ 10)
↓
6. Output screen-positive list

9.2.2 Confirmation SOP

1. Receive screen-positive list
↓
2. GOTTCHA2 verification
↓
3. sourmash containment check
↓
4. Integrate evidence
↓
5. Output supportive/confirmed results

9.2.3 Deep-analysis SOP

1. Assess sample priority

↓

2. Reserve high-memory nodes

↓

3. Assembly + classification

↓

4. Evaluate contig evidence

↓

5. If needed, run RVDB verification

9.3 Quality-control indicators

9.3.1 Sample-level QC

Metric	Expected	Action if abnormal
Raw reads	>10M	Increase sequencing depth
Host removal rate	>90%	Check contamination
Q30 fraction	>80%	Resequence
Contig N50	>1 kb (virus-enriched)	Reassess assembly quality

9.3.2 Batch-level QC

Metric	Expected	Action if abnormal
Negative control	No viral calls	Investigate contamination
Positive control	Correct detection	Validate workflow
Batch variability	CV < 20%	Apply standardization

10. Decision-support matrix

10.1 Scenario-to-tool quick map

Scenario	First choice	Backup	Notes
Routine monitoring (high throughput)	TaxProfiler	sourmash	Standardized abundance tables
Confirm key targets	GOTTCHA2	sourmash	Signature confirmation
Novel-virus monitoring	rvdb-viral	MLMVD-nf	High resource need
Genome reconstruction	nf-core/mag	KrakenMetaReads-nf	High resource need
Rapid pre-screen	sourmash	CLARK-l	Resource-limited
Quality control	sourmash gather	—	Containment check

10.2 Resource planning guide

Data scale	Suggested strategy	Est. memory	Est. time
<10 GB	TaxProfiler	64 GB	2–4 h
10–50 GB	TaxProfiler	128 GB	4–8 h
50–100 GB	TaxProfiler	256 GB	8–24 h
100–300 GB	Tiered strategy	256 GB	1–3 days
>300 GB	Screening only	512 GB	3–7 days

11. Case studies

11.1 Routine monitoring case

Scenario: CDC processes weekly wastewater samples from 10 states to monitor viral trends.

Sample profile: - 100 samples/week

- ~5 GB short reads per sample

- Targets: SARS-CoV-2, influenza viruses, norovirus, etc.

Tiered execution:

Tier 1 (full screening)

- Pipeline: TaxProfiler (nf-core/taxprofiler)

- Batch mode; automated reporting

- ~128 GB RAM per sample
- Outputs: RPM tables; MultiQC reports

Tier 2 (confirmation)
- Trigger: target detected by TaxProfiler
- Pipeline: GOTTCHA2
- ~256 GB RAM per sample

Tier 3 (deep analysis)
- Trigger: anomalous signal or novel pathogen alert
- Pipeline: rvdb-viral or nf-core/mag
- ~512–768 GB RAM per sample

Weekly resource estimate: - Tier 1: 100 samples × 4 h × 10-way concurrency ≈ 400 core-hours
- Tier 2: 10 samples × 8 h ≈ 80 core-hours
- Tier 3: 1–2 samples × 48 h ≈ 100 core-hours

11.2 Emerging outbreak response case

Scenario: A state reports an unexplained GI illness outbreak and needs rapid pathogen assessment.

Response:

Day 1: urgent sequencing and initial analysis within ~48 h; run multiple tools in parallel
- TaxProfiler: broad rapid screen
- rvdb-viral: deep discovery
- MLMVD-nf: high-confidence candidates

Day 2: integrate evidence; identify consensus hits; exclude likely false positives

Day 3–4: deep validation: assembly evidence; contig-level support; coordinate reference lab validation

Day 5: issue report with evidence-chain details and confidence tiers

11.3 Long-term national program case

Scenario: build a nationwide wastewater surveillance network for continuous trend tracking.

Infrastructure: - Central storage ≥100 TB; local DB mirrors at regional centers

- Compute: 500+ cores routine; high-memory nodes (50+) for assembly; GPUs for ML tools
- Staffing: 2–3 bioinformaticians; 1–2 analysts; 2 IT support
- QA: monthly internal audits; quarterly cross-site validation; annual external audit

12. Technical specifications (reference)

12.1 Recommended hardware

12.1.1 Routine-analysis nodes

Component	Minimum	Recommended
CPU	32 cores	64 cores
Memory	128 GB	512 GB *(original text had a "TB" typo; interpreted as GB)*
Storage	10 TB	20 TB
Network	10 Gbps	25 Gbps

12.1.2 High-memory nodes

Component	Minimum	Recommended
CPU	64 cores	128 cores
Memory	512 GB	2 TB
Storage	20 TB	40 TB
Network	10 Gbps	25 Gbps

12.1.3 Storage system

Type	Use	Capacity
Fast storage	Working directory	100 TB
Large storage	Raw data	500 TB
Archive storage	Historical data	2 PB

12.2 Software environment

12.2.1 Base software

Software	Version	Notes
Linux	RHEL 8+	or equivalent
Apptainer	≥3.0	container runtime
Nextflow	≥23.04	workflow engine
Java	≥11	Nextflow dependency
Python	≥3.8	scripting

12.2.2 Analysis tools

Tool	Version	Use
FastQC	≥0.12	QC
fastp	≥0.23	read filtering
Bowtie2	≥2.5	host removal
Kraken2	≥2.1	classification
Bracken	≥2.8	abundance estimation
MEGAHIT	≥1.2	assembly
metaSPAdes	≥3.15	assembly
DIAMOND	≥2.0	protein alignment

13. Next steps (already initiated)

Round 1 produced initial comparisons, but a key limitation is that the simulated data used in that round lacked accessible, verifiable **ground-truth annotations**, preventing standardized metrics (Sensitivity/Recall, Precision, Specificity, F1-score) for each of the eight pipelines. As a result, cross-pipeline comparisons remain largely qualitative and lack reproducible quantitative support. To address this, we have identified two “gold-standard” datasets and will conduct Rounds 2 and 3 accordingly.

13.1 Round 2: CAMI datasets (Critical Assessment of Metagenome Interpretation)

CAMI is a community blind-assessment framework widely used to benchmark assembly, binning, and profiling tools. We plan to use a high-complexity **CAMI II** dataset. A key advantage is explicit **ground truth** (e.g., read-level provenance labels), enabling quantitative

evaluation of viral detection/annotation outcomes.
Planned metrics for all 8 pipelines: Sensitivity/Recall, Precision, Specificity, F1-score.

13.2 Round 3: ZymoBIOMICS Mock Community (real sequencing)

Round 3 will use real sequencing data corresponding to the **ZymoBIOMICS Viral DNA/RNA Standard D6333**. The standard contains 11 viruses (8 RNA and 3 DNA viruses; examples include SARS-CoV-2, influenza viruses, Zika virus, and bacteriophage MS2), enabling evaluation of detection and false-positive control under real sequencing conditions.
Planned metrics for all 8 pipelines: Sensitivity/Recall, Precision, Specificity, F1-score.

13.3 Manuscript preparation and publication

After completing Round 1 (initial qualitative evaluation), Round 2 (simulation with ground truth), and Round 3 (external validation with real sequencing standards), we will integrate and cross-validate results across the three rounds to form a complete method-evaluation evidence chain and prepare a manuscript for submission to a peer-reviewed journal.

14. References and tool documentation

14.1 Key references

1. Wood DE, Salzberg SL. Kraken: ultrafast metagenomic sequence classification using exact alignments. *Genome Biol.* 2014;15(3):R46.
2. Lu J, Breitwieser FP, Thielen P, Salzberg SL. Bracken: estimating species abundance in metagenomics data. *PeerJ Comput Sci.* 2017;3:e104.
3. Ounit R, Wanamaker S, Close TJ, Lonardi S. CLARK: fast and accurate classification of metagenomic and genomic sequences using discriminative k-mers. *BMC Genomics.* 2015;16:236.

4. Roux S, Emerson JB, Eloe-Fadrosh EA, Sullivan MB. Benchmarking viromics: an in silico evaluation of metagenome-enabled estimates of viral community composition and diversity. *PeerJ.* 2017;5:e3957.
5. Guo J, Li H, Wu L, et al. RVDB: a curated database of recurrent viral integrations in the host genome. *Database (Oxford).* 2019;baz142.

14.2 Tool documentation

Tool	Documentation
nf-core/taxprofiler	https://nf-co.re/taxprofiler
nf-core/mag	https://nf-co.re/mag
sourmash	https://sourmash.readthedocs.io
GOTTCHA2	https://github.com/fulcrumcontent/GOTTCHA2
Kraken2	https://github.com/DerrickWood/kraken2

Integrated Version: v1.0

Original evaluation window: January–February 2026

Primary evaluation platform: UGA Sapelo2 HPC cluster