HTSinfer

Release 0.9.0

Zavolan Lab

May 18, 2022

MODULES

1 htsinfer **1**

1.1 htsinfer package 1

2 Indices and tables **23**

Python Module Index **25**

Index **27**

CHAPTER

ONE

HTSINFER

1.1 htsinfer package

HTSinfer project root

1.1.1 Submodules

1.1.2 htsinfer.cli module

Command-line interface client.

htsinfer.cli.**main**() → None
Entry point for CLI executable.

htsinfer.cli.**parse_args**() → argparse.Namespace
Parse CLI arguments.

Returns Parsed CLI arguments.

htsinfer.cli.**setup_logging**(*verbosity: str = 'INFO'*) → None
Configure logging.

Parameters **verbosity** – Level of logging verbosity.

1.1.3 htsinfer.exceptions module

Custom exceptions.

exception htsinfer.exceptions.**FileProblem**
Bases: Exception

Exception raised when file could not be opened or parsed.

exception htsinfer.exceptions.**InconsistentFastqIdentifiers**
Bases: Exception

Exception raised when inconsistent FASTQ sequence identifiers were ecountered.

exception htsinfer.exceptions.**KallistoProblem**
Bases: Exception

Exception raised when running kallisto index and quant commands.

exception htsinfer.exceptions.**MetadataWarning**
Bases: Exception

Exception raised when metadata could not be determined.

exception htsinfer.exceptions.**StarProblem**
Bases: Exception

Exception raised when running STAR index and quant commands.

exception htsinfer.exceptions.**UnknownFastqIdentifier**
Bases: Exception

Exception raised when a FASTQ sequence identifier of unknown format was ecountered.

exception htsinfer.exceptions.**WorkEnvProblem**
Bases: Exception

Exception raised when the work environment could not be set up or cleaned.

1.1.4 htsinfer.get_library_source module

Infer library source from sample data.

class htsinfer.get_library_source.**GetLibSource**(*paths: Tuple[pathlib.Path, Optional[pathlib.Path]]*, *transcripts_file: pathlib.Path*, *out_dir: pathlib.Path = PosixPath('/home/docs/checkouts/readthedocs.org/user_builds/htsinfer/checkou*
tmp_dir: pathlib.Path = PosixPath('/tmp/tmp_htsinfer'), *min_match_pct: float = 2*, *min_freq_ratio: float = 2*)
Bases: object

Determine the source of FASTQ sequencing of a single- or paired-end seguencing library.

Parameters

- **paths** – Tuple of one or two paths for single-end and paired end library files.
- **transcripts_file** – File path to an uncompressed transcripts file in FASTA format. Expected to contain |-separated sequence identifier lines that contain an organism short name and a taxon identifier in the fourth and fifth columns, respectively. Example sequence identifier: *rpl-13|ACYPI006272|ACYPI006272-RA|apisum|7029*
- **out_dir** – Path to directory where output is written to.
- **tmp_dir** – Path to directory where temporary output is written to.
- **min_match_pct** – Minimum percentage of reads that are consistent with a given source in order for it to be considered as the to be considered the library's source.
- **min_freq_ratio** – Minimum frequency ratio between the first and second most frequent source in order for the former to be considered the library's source.

Attrubutes:

paths: Tuple of one or two paths for single-end and paired end library files.

transcripts_file: File path to an uncompressed transcripts file in FASTA format. Expected to contain |-separated sequence identifier lines that contain an organism short name and a taxon identifier in the fourth and fifth columns, respectively. Example sequence identifier: *rpl-13|ACYPI006272|ACYPI006272-RA|apisum|7029*

out_dir: Path to directory where output is written to. tmp_dir: Path to directory where temporary output is written to. min_match_pct: Minimum percentage of reads that are consistent with a

given source in order for it to be considered as the to be considered the library's source.

min_freq_ratio: Minimum frequency ratio between the first and second most frequent source in order for the former to be considered the library's source.

`create_kallisto_index()` → pathlib.Path
Build Kallisto index from FASTA file of target sequences.

Returns Path to Kallisto index.

Raises *KallistoProblem* – Kallisto index could not be created.

`evaluate()` → *htsinfer.models.ResultsSource*
Infer read source.

Returns Source results object.

`get_source`(*fastq: pathlib.Path*, *index: pathlib.Path*) → *htsinfer.models.Source*
Determine source of a single sequencing library file.

Parameters

- **fastq** – Path to FASTQ file.
- **index** – Path to Kallisto index.

Returns Source of library file.

`static get_source_expression`(*kallisto_dir: pathlib.Path*) → pandas.core.frame.DataFrame
Return percentages of total expression per read source.

Parameters **kallisto_dir** – Directory containing Kallisto quantification results.

Returns

Data frame with columns *source_ids* (a tuple of source short name and taxon identifier, e.g., *("hsapiens", 9606)*) and *tpm*, signifying the percentages of total expression per read source. The data frame is sorted by total expression in descending order.

Raises *FileProblem* – Kallisto quantification results could not be processed.

`run_kallisto_quantification`(*fastq: pathlib.Path*, *index: pathlib.Path*) → pathlib.Path
Run Kallisto quantification on individual sequencing library file.

Parameters

- **fastq** – Path to FASTQ file.
- **index** – Path to Kallisto index.

Returns Path to output directory.

Raises *KallistoProblem* – Kallisto quantification failed.

1.1.5 htsinfer.get_library_stats module

Infer read orientation from sample data.

class htsinfer.get_library_stats.**GetLibStats**(*paths: Tuple[pathlib.Path, Optional[pathlib.Path]]*, *tmp_dir: pathlib.Path = PosixPath('/tmp/tmp_htsinfer')*)

Bases: object

Determine library statitics of a single- or paired-end seguencing library.

Parameters

- **paths** – Tuple of one or two paths for single-end and paired end library files.
- **tmp_dir** – Path to directory where temporary output is written to.

paths
Tuple of one or two paths for single-end and paired end library files.

tmp_dir
Path to directory where temporary output is written to.

evaluate() → *htsinfer.models.ResultsStats*
Infer read statistics.

Returns Statistics results object.

static fastq_get_min_max_read_length(*fastq: pathlib.Path*) → Tuple[int, int]
Get number of records in a FASTQ file.

Parameters **fastq** – Path to FASTQ file.

Returns Tuple of minimum and maximum read lengths in input file.

Raises ***FileProblem*** – Could not process FASTQ file.

1.1.6 htsinfer.get_library_type module

Infer mate information from sample data.

class htsinfer.get_library_type.**GetFastqType**(*path: pathlib.Path*)

Bases: object

Determine type (single/paired) information for an individual FASTQ sequencing library.

Parameters **path** – File path to read library.

path
File path to read library.

seq_ids
List of sequence identifier prefixes of the provided read library, i.e., the fragments up until the mate information, if available, as defined by a named capture group `prefix` in a regular expression to extract mate information.

seq_id_format
The sequence identifier format of the read library, as identified by inspecting the first read and matching one of the available regular expressions for the different identifier formats.

result
The current best guess for the type of the provided library.

Examples

```
>>> lib_type = GetFastqType(
...     path="tests/files/first_mate.fastq"
... ).evaluate()
<OutcomesType.first_mate: 'first_mate'>
```

evaluate() → None
Decide library type.

Raises **NoMetadataDetermined** – Type information could not be determined.

class htsinfer.get_library_type.**GetLibType**(*path_1: pathlib.Path*, *path_2: Optional[pathlib.Path] = None*)

Bases: object

Determine type (single/paired) information for a single or a pair of FASTQ sequencing libraries.

Args: path_1: Path to single-end library or first mate file. path_2: Path to second mate file.

Attributes: path_1: Path to single-end library or first mate file. path_2: Path to second mate file. results: Results container for storing library type information for

the provided files, as well as the mate relationship between the two files, if applicable.

Examples:

```
>>> GetLibType(
...     path_1="tests/files/first_mate.fastq"
... ).evaluate()
ResultsType(file_1=<OutcomesType.single: 'single'>, file_2=<OutcomesTyp
```

e.not_available: 'not_available'>, relationship=<OutcomesTypeRelationship.not_a vailable: 'not_available'>)

```
>>> GetLibType(
...     path_1="tests/files/first_mate.fastq",
...     path_2="../tests/test_files/second_mate.fastq",
... ).evaluate()
ResultsType(file_1=<OutcomesType.first_mate: 'first_mate'>, file_2=<Out
```

comesType.second_mate: 'second_mate'>, relationship=<OutcomesTypeRelationship.s plit_mates: 'split_mates'>)

('first_mate', 'second_mate', 'split_mates')

evaluate() → None
Decide type information and mate relationship.

1.1.7 htsinfer.get_read_layout module

Infer adapter sequences present in reads.

class htsinfer.get_read_layout.**GetAdapter3**(*path: pathlib.Path*, *adapter_file: pathlib.Path*, *out_dir: pathlib.Path = PosixPath('/home/docs/checkouts/readthedocs.org/user_builds/htsinfer/checkouts/lat*
min_match_pct: float = 2, *min_freq_ratio: float = 2*)

Bases: object

Determine 3' adapter sequence for an individual FASTQ library.

Parameters

- **path** – File path to read library.
- **adapter_file** – Path to text file containing 3' adapter sequences (one sequence per line) to scan for.
- **out_dir** – Path to directory where output is written to.
- **min_match_pct** – Minimum percentage of reads that contain a given adapter Minimum percentage of reads that contain a given adapter sequence in order for it to be considered as the library's 3'-end adapter.
- **min_freq_ratio** – Minimum frequency ratio between the first and second most frequent adapter in order for the former to be considered as the library's 3'-end adapter.

path
File path to read library.

adapter_file
Path to text file containing 3' adapter sequences (one sequence per line) to scan for.

out_dir
Path to directory where output is written to.

min_match_pct
Minimum percentage of reads that contain a given adapter Minimum percentage of reads that contain a given adapter sequence in order for it to be considered as the library's 3'-end adapter.

min_freq_ratio
Minimum frequency ratio between the first and second most frequent adapter in order for the former to be considered as the library's 3'-end adapter.

adapters
List of adapter sequences.

trie
Trie data structure of adapter sequences.

adapter_counts
Dictionary of adapter sequences and corresponding count percentages.

result
The most frequent adapter sequence in FASTQ file.

Examples

```
>>> GetAdapter3(
...     path_1="tests/files/sra_sample_2.fastq",
...     adapter_file="data/adapter_fragments.txt",
... ).evaluate()
<"AAAAAAAAAAAAAA">
```

evaluate() → None
Search for adapter sequences and validate result confidence constraints.

class htsinfer.get_read_layout.**GetReadLayout**(*path_1: pathlib.Path*, *path_2: Optional[pathlib.Path] = None*, *adapter_file: pathlib.Path = PosixPath('/home/docs/checkouts/readthedocs.org/user_builds/htsinfer/checkouts/*, *out_dir: pathlib.Path = PosixPath('/home/docs/checkouts/readthedocs.org/user_builds/htsinfer/checkouts/*, *min_match_pct: float = 2*, *min_freq_ratio: float = 2*)

Bases: `object`

Determine the adapter sequence present in the FASTQ sequencing libraries.

Parameters

- **path_1** – Path to single-end library or first mate file.
- **path_2** – Path to second mate file.
- **adapter_file** – Path to text file containing 3' adapter sequences (one sequence per line) to scan for.
- **out_dir** – Path to directory where output is written to.
- **min_match_pct** – Minimum percentage of reads that contain a given adapter sequence in order for it to be considered as the library's 3'-end adapter.
- **min_freq_ratio** – Minimum frequency ratio between the first and second most frequent adapter in order for the former to be considered as the library's 3'-end adapter.

path_1
Path to single-end library or first mate file.

path_2
Path to second mate file.

adapter_file
Path to text file containing 3' adapter sequences (one sequence per line) to scan for.

out_dir
Path to directory where output is written to.

min_match_pct
Minimum percentage of reads that contain a given adapter sequence in order for it to be considered as the library's 3'-end adapter.

min_freq_ratio
Minimum frequency ratio between the first and second most frequent adapter in order for the former to be considered as the library's 3'-end adapter.

results
Results container for storing adapter sequence information for the provided files.

Examples

```
>>> GetReadLayout(
...     path_1="tests/files/sra_sample_2.fastq",
...     adapter_file="data/adapter_fragments.txt",
... ).evaluate()
ResultsLayout(
    file_1=<Layout().adapt_3: "AAAAAAAAAAAAAAA">,
    file_2=<Layout().adapt_3: None>,
)
```

(continues on next page)

(continued from previous page)

```
>>> GetReadLayout(
...     path_1="tests/files/sra_sample_1.fastq",
...     path_2="tests/files/sra_sample_2.fastq",
...     adapter_file="data/adapter_fragments.txt",
...     min_match_pct=2,
...     min_freq_ratio=1,
... ).evaluate()
ResultsLayout(
    file_1=<Layout().adapt_3: "AAAAAAAAAAAAAAA">,
    file_2=<Layout().adapt_3: "AAAAAAAAAAAAAAA">,
)
```

evaluate() → None
Decide adapter sequence.

1.1.8 htsinfer.get_read_orientation module

Infer read orientation from sample data.

class htsinfer.get_read_orientation.**GetOrientation**(*paths: Tuple[pathlib.Path, Optional[pathlib.Path]]*, *library_type:* htsinfer.models.ResultsType, *library_source:* htsinfer.models.ResultsSource, *transcripts_file: pathlib.Path*, *tmp_dir: pathlib.Path = PosixPath('/tmp/tmp_htsinfer')*, *threads_star: int = 1*, *min_mapped_reads: int = 20*, *min_fraction: float = 0.75*)

Bases: object

Determine library strandedness and relative read orientation of a single- or paired-end seguencing library.

Parameters

- **paths** – Tuple of one or two paths for single-end and paired end library files.
- **library_type** – ResultsType object with library type and mate relationship.
- **library_source** – ResultsSource object with source information on each library file.
- **transcripts_file** – File path to an uncompressed transcripts file in FASTA format.
- **tmp_dir** – Path to directory where temporary output is written to.
- **threads_star** – Number of threads to run STAR with.
- **source** – Source (organism, tissue, etc.) of the sequencing library.
- **min_mapped_reads** – Minimum number of mapped reads for deeming the read orientation result reliable.
- **min_fraction** – Minimum fraction of mapped reads required to be consistent with a given read orientation state in order for that orientation to be reported. Must be above 0.5.
- **mate_relationship** – Type/mate relationship between the provided files.

paths
Tuple of one or two paths for single-end and paired end library files.

library_type
ResultsType object with library type and mate relationship.

library_source
ResultsSource object with source information on each library file.

transcripts_file
File path to an uncompressed transcripts file in FASTA format.

tmp_dir
Path to directory where temporary output is written to.

threads_star
Number of threads to run STAR with.

source
Source (organism, tissue, etc.) of the sequencing library.

min_mapped_reads
Minimum number of mapped reads for deeming the read orientation result reliable.

min_fraction
Minimum fraction of mapped reads required to be consistent with a given read orientation state in order for that orientation to be reported. Must be above 0.5.

mate_relationship
Type/mate relationship between the provided files.

create_star_index(*fasta: pathlib.Path*, *index_string_size: int = 5*) → pathlib.Path
Prepare STAR index.

Parameters

- **fasta** – Path to FASTA file of sequence records to create index from.
- **index_string_size** – Size of SA pre-indexing string, in nucleotides.

Returns Path to directory containing STAR index.

Raises ***StarProblem*** – STAR index could not be created.

evaluate() → *htsinfer.models.ResultsOrientation*
Infer read orientation.

Returns Orientation results object.

static generate_star_alignments(*commands: Dict[pathlib.Path, List[str]]*) → None
Align reads to index with STAR.

Parameters **commands** – Dictionary of output paths and corresponding STAR commands.

Raises ***StarProblem*** – Generating alignments failed.

static get_fasta_size(*fasta: pathlib.Path*) → int
Get size of FASTA file in total nucleotides.

Parameters **fasta** – Path to FASTA file.

Returns Total number of nucleotides of all records.

Raises ***FileProblem*** – Could not open FASTA file for reading.

static get_frequencies(**items: Any*) → Dict[Any, float]
Get frequencies of arguments as fractions of the number of all arguments.

Parameters ***items** – Items to get frequencies for.

Returns Dictionary of arguments and their frequencies.

static get_star_index_string_size(*ref_size: int*) → int
Get length of STAR SA pre-indexing string.

Cf. https://github.com/alexdobin/STAR/blob/51b64d4fafb7586459b8a61303e40beceeead8c0/doc/STARmanual.pdf

Parameters **ref_size** – Size of genome/transcriptome reference in nucleotides.

Returns Size (in nucleotides) of SA pre-indexing string.

prepare_star_alignment_commands(*index_dir: pathlib.Path*) → Dict[pathlib.Path, List[str]]
Prepare STAR alignment commands.

Parameters **index_dir** – Path to directory containing STAR index.

Returns Dictionary of output paths and corresponding STAR commands.

process_alignments(*star_dirs: List[pathlib.Path]*) → *htsinfer.models.ResultsOrientation*
Determine read orientation of one or two single-ended or one paired-end sequencing library.

Parameters **star_dirs** – List of one or two paths to STAR output directories.

Returns Read orientation state of library or libraries.

process_paired(*sam: pathlib.Path*) → *htsinfer.models.ResultsOrientation*
Determine read orientation of a paired-ended sequencing library.

Parameters **sam** – Path to SAM file.

Returns

Read orientation state of each mate and orientation state relationship of library.

process_single(*sam: pathlib.Path*) → *htsinfer.models.StatesOrientation*
Determine read orientation of a single-ended sequencing library.

Parameters **sam** – Path to SAM file.

Returns Read orientation state of library.

subset_transcripts_by_organism() → pathlib.Path
Filter FASTA file of transcripts by current sources.

The filtered file contains records from the indicated sources. Typically, this is one source. However, for if two input files were supplied that are originating from different sources (i.e., not from a valid paired-ended library), it may be from two different sources. If no source is supplied (because it could not be inferred), no filtering is done.

Returns Path to filtered FASTA file.

Raises ***FileProblem*** – Could not open input/output FASTA file for reading/writing.

static sum_dicts(**dicts: Dict[Any, float]*) → Dict[Any, float]
Sum of dictionaries with numeric values.

Parameters ***dicts** – Dictionaries to sum up.

Returns Dictionary with union of keys of input dictionaries and all values added up.

1.1.9 htsinfer.htsinfer module

Main module.

class htsinfer.htsinfer.**HtsInfer**(*path_1: pathlib.Path*, *path_2: Optional[pathlib.Path] = None*, *out_dir: pathlib.Path = PosixPath('/home/docs/checkouts/readthedocs.org/user_builds/htsinfer/checkouts/latest/docs/api/r*, *tmp_dir: pathlib.Path = PosixPath('/tmp/tmp_htsinfer')*, *cleanup_regime: htsinfer.models.CleanupRegimes = CleanupRegimes.DEFAULT*, *records: int = 0*, *threads: int = 1*, *transcripts_file: pathlib.Path = PosixPath('/home/docs/checkouts/readthedocs.org/user_builds/htsinfer/checkouts/latest/data/trans*, *read_layout_adapter_file: pathlib.Path = PosixPath('/home/docs/checkouts/readthedocs.org/user_builds/htsinfer/checkouts/latest/data/adap*, *read_layout_min_match_pct: float = 2*, *read_layout_min_freq_ratio: float = 2*, *lib_source_min_match_pct: float = 2*, *lib_source_min_freq_ratio: float = 2*, *read_orientation_min_mapped_reads: int = 20*, *read_orientation_min_fraction: float = 0.75*)

Bases: `object`

Determine sequencing library metadata.

Parameters

- **path_1** – Path to single-end library or first mate file.
- **path_2** – Path to second mate file.
- **out_dir** – Path to directory where output is written to.
- **tmp_dir** – Path to directory where temporary output is written to.
- **cleanup_regime** – Which data to keep after run concludes; one of *CleanupRegimes*.
- **records** – Number of input file records to process; set to *0* to process all records.
- **threads** – Number of threads to run STAR with.
- **transcripts_file** – File path to transcripts FASTA file.
- **read_layout_adapter_file** – Path to text file containing 3' adapter sequences to scan for (one sequence per line).
- **read_layout_min_match_pct** – Minimum percentage of reads that contain a given adapter in order for it to be considered as the library's 3'-end adapter.
- **read_layout_min_freq_ratio** – Minimum frequency ratio between the first and second most frequent adapter in order for the former to be considered as the library's 3'-end adapter.
- **lib_source_min_match_pct** – Minimum percentage of reads that are consistent with a given source in order for it to be considered as the to be considered the library's source.
- **lib_source_min_freq_ratio** – Minimum frequency ratio between the first and second most frequent source in order for the former to be considered the library's source.
- **read_orientation_min_mapped_reads** – Minimum number of mapped reads for deeming the read orientation result reliable.
- **read_orientation_min_fraction** – Minimum fraction of mapped reads required to be consistent with a given read orientation state in order for that orientation to be reported. Must be above 0.5.

path_1

Path to single-end library or first mate file.

path_2
Path to second mate file.

out_dir
Path to directory where output is written to.

run_id
Random string identifier for HTSinfer run.

tmp_dir
Path to directory where temporary output is written to.

cleanup_regime
Which data to keep after run concludes; one of *CleanupRegimes*.

records
Number of input file records to process.

threads
Number of threads to run STAR with.

transcripts_file
File path to transcripts FASTA file.

read_layout_adapter_file
Path to text file containing 3' adapter sequences to scan for (one sequence per line).

read_layout_min_match_pct
Minimum percentage of reads that contain a given adapter in order for it to be considered as the library's 3'-end adapter.

read_layout_min_freq_ratio
Minimum frequency ratio between the first and second most frequent adapter in order for the former to be considered as the library's 3'-end adapter.

lib_source_min_match_pct
Minimum percentage of reads that are consistent with a given source in order for it to be considered as the to be considered the library's source.

lib_source_min_freq_ratio
Minimum frequency ratio between the first and second most frequent source in order for the former to be considered the library's source.

read_orientation_min_mapped_reads
Minimum number of mapped reads for deeming the read orientation result reliable.

read_orientation_min_fraction
Minimum fraction of mapped reads required to be consistent with a given read orientation state in order for that orientation to be reported. Must be above 0.5.

path_1_processed
Path to processed *path_1* file.

path_2_processed
Path to processed *path_2* file.

transcripts_file_processed
Path to processed *transcripts_file* file.

state
State of the run; one of *RunStates*.

results
Results container for storing determined library metadata.

clean_up()
Clean up work environment.

evaluate()
Determine library metadata.

get_library_source() → *htsinfer.models.ResultsSource*
Determine library source.

Returns Library source results.

get_library_stats()
Determine library statistics.

get_library_type()
Determine library type.

get_read_layout()
Determine read layout.

get_read_orientation()
Determine read orientation.

prepare_env()
Set up work environment.

print()
Print results to STDOUT.

process_inputs()
Process and validate inputs.

1.1.10 htsinfer.models module

Data models.

class htsinfer.models.**CleanupRegimes**(*value*)
Bases: enum.Enum

Enumerator of cleanup regimes.

DEFAULT = 'default'

KEEP_ALL = 'keep_all'

KEEP_NONE = 'keep_none'

KEEP_RESULTS = 'keep_results'

class htsinfer.models.**Layout**(*, *adapt_3: str = None*)
Bases: pydantic.main.BaseModel

Read layout of a single sequencing file.

Parameters **adapt_3** – Adapter sequence ligated to 3'-end of sequence.

adapt_3
Adapter sequence ligated to 3'-end of sequence.

Type Optional[str]

adapt_3: Optional[str]

class htsinfer.models.**LogLevels**(*value*)

Bases: enum.Enum

Log level enumerator.

CRITICAL = 50

DEBUG = 10

ERROR = 40

INFO = 20

WARN = 30

WARNING = 30

class htsinfer.models.**ReadLength**(*, *min: int = None*, *max: int = None*)

Bases: pydantic.main.BaseModel

Read length of a sequencing file.

Parameters

- **min** – Minimum read length.
- **max** – Maximum read length.

min

Minimum read length.

Type Optional[int]

max

Maximum read length.

Type Optional[int]

max: Optional[int]

min: Optional[int]

class htsinfer.models.**Results**(*, *library_stats: htsinfer.models.ResultsStats = ResultsStats(file_1=Stats(read_length=ReadLength(min=None, max=None)), file_2=Stats(read_length=ReadLength(min=None, max=None)))*, *library_type: htsinfer.models.ResultsType = ResultsType(file_1=<StatesType.not_available: None>, file_2=<StatesType.not_available: None>, relationship=<StatesTypeRelationship.not_available: None>)*, *library_source: htsinfer.models.ResultsSource = ResultsSource(file_1=Source(short_name=None, taxon_id=None), file_2=Source(short_name=None, taxon_id=None))*, *read_orientation: htsinfer.models.ResultsOrientation = ResultsOrientation(file_1=<StatesOrientation.not_available: None>, file_2=<StatesOrientation.not_available: None>, relationship=<StatesOrientationRelationship.not_available: None>)*, *read_layout: htsinfer.models.ResultsLayout = ResultsLayout(file_1=Layout(adapt_3=None), file_2=Layout(adapt_3=None))*)

Bases: pydantic.main.BaseModel

Container class for aggregating results from the different inference functionalities.

Parameters

- **library_type** – Library type inference results.
- **library_source** – Library source inference results.
- **orientation** – Read orientation inference results.
- **read_layout** – Read layout inference results.
- **type** – Library type inference results.
- **source** – Library source inference results.
- **read_orientation** – Read orientation inference results.
- **read_layout** – Read layout inference results.

library_source: *htsinfer.models.ResultsSource*

library_stats: *htsinfer.models.ResultsStats*

library_type: *htsinfer.models.ResultsType*

read_layout: *htsinfer.models.ResultsLayout*

read_orientation: *htsinfer.models.ResultsOrientation*

class htsinfer.models.**ResultsLayout**(*, *file_1:* htsinfer.models.Layout = *Layout(adapt_3=None)*, *file_2:* htsinfer.models.Layout = *Layout(adapt_3=None)*)

Bases: pydantic.main.BaseModel

Container class for read layout of a sequencing library.

Parameters

- **file_1** – Adapter sequence present in first file.
- **file_2** – Adapter sequence present in second file.

file_1

Adapter sequence present in first file.

Type *htsinfer.models.Layout*

file_2

Adapter sequence present in second file.

Type *htsinfer.models.Layout*

file_1: *htsinfer.models.Layout*

file_2: *htsinfer.models.Layout*

class htsinfer.models.**ResultsOrientation**(*, *file_1:* htsinfer.models.StatesOrientation = *StatesOrientation.not_available*, *file_2:* htsinfer.models.StatesOrientation = *StatesOrientation.not_available*, *relationship:* htsinfer.models.StatesOrientationRelationship = *StatesOrientationRelationship.not_available*)

Bases: pydantic.main.BaseModel

Container class for aggregating library orientation.

Args: file_1: Read orientation of first file. file_2: Read orientation of second file. relationship: Orientation type relationship between the provided files.

file_1

Read orientation of first file.

Type *htsinfer.models.StatesOrientation*

file_2
Read orientation of second file.

Type *htsinfer.models.StatesOrientation*

file_1: ***htsinfer.models.StatesOrientation***

file_2: ***htsinfer.models.StatesOrientation***

relationship: ***htsinfer.models.StatesOrientationRelationship***

class htsinfer.models.**ResultsSource**(**, file_1:* htsinfer.models.Source = *Source(short_name=None, taxon_id=None)*, *file_2:* htsinfer.models.Source = *Source(short_name=None, taxon_id=None)*)

Bases: pydantic.main.BaseModel

Container class for aggregating library source.

Parameters

- **file_1** – Library source of the first file.
- **file_2** – Library source of the second file.

file_1
Library source of the first file.

Type *htsinfer.models.Source*

file_2
Library source of the second file.

Type *htsinfer.models.Source*

file_1: ***htsinfer.models.Source***

file_2: ***htsinfer.models.Source***

class htsinfer.models.**ResultsStats**(**, file_1:* htsinfer.models.Stats = *Stats(read_length=ReadLength(min=None, max=None))*, *file_2:* htsinfer.models.Stats = *Stats(read_length=ReadLength(min=None, max=None))*)

Bases: pydantic.main.BaseModel

Container class for aggregating library statistics information.

Parameters

- **file_1** – Library statistics for the first file.
- **file_2** – Library statistics for the second file.

file_1
Library statistics for the first file.

Type *htsinfer.models.Stats*

file_2
Library statistics for the second file.

Type *htsinfer.models.Stats*

file_1: ***htsinfer.models.Stats***

file_2: ***htsinfer.models.Stats***

class htsinfer.models.**ResultsType**(*, *file_1:* htsinfer.models.StatesType = *StatesType.not_available*, *file_2:* htsinfer.models.StatesType = *StatesType.not_available*, *relationship:* htsinfer.models.StatesTypeRelationship = *StatesTypeRelationship.not_available*)

Bases: pydantic.main.BaseModel

Container class for aggregating library type and mate relationship information.

Parameters

- **file_1** – Library type of the first file.
- **file_2** – Library type of the second file.
- **relationship** – Type/mate relationship between the provided files.

file_1
Library type of the first file.

Type *htsinfer.models.StatesType*

file_2
Library type of the second file.

Type *htsinfer.models.StatesType*

relationship
Type/mate relationship between the provided files.

Type *htsinfer.models.StatesTypeRelationship*

file_1: ***htsinfer.models.StatesType***

file_2: ***htsinfer.models.StatesType***

relationship: ***htsinfer.models.StatesTypeRelationship***

class htsinfer.models.**RunStates**(*value*)

Bases: enum.IntEnum

Enumerator of run states and exit codes.

ERROR = 2

OKAY = 0

WARNING = 1

class htsinfer.models.**SeqIdFormats**(*value*)

Bases: enum.Enum

An enumeration.

class htsinfer.models.**Source**(*, *short_name: str = None*, *taxon_id: int = None*)

Bases: pydantic.main.BaseModel

Library source of an individual sequencing file.

Parameters

- **short_name** – Library source short name, e.g., “hsapiens”.
- **taxon_id** – Library source taxon identifer, e.g., *9606*.

short_name
Library source short name, e.g., “hsapiens”.

Type Optional[str]

taxon_id
Library source taxon identifer, e.g., *9606*.

Type Optional[int]

short_name: Optional[str]

taxon_id: Optional[int]

class htsinfer.models.**StatesOrientation**(*value*)
Bases: enum.Enum

Enumerator of read orientation types for individual library files. Cf. https://salmon.readthedocs.io/en/latest/library_type.html

not_available
Orientation type information is not available for a given file, either because no file was provided, the file could not be parsed, an orientation type has not yet been assigned.

stranded_forward
Reads are stranded and come from the forward strand.

stranded_reverse
Reads are stranded and come from the reverse strand.

unstranded
Reads are unstranded.

not_available = None

stranded_forward = 'SF'

stranded_reverse = 'SR'

unstranded = 'U'

class htsinfer.models.**StatesOrientationRelationship**(*value*)
Bases: enum.Enum

Enumerator of read orientation type relationships for paired-ended libraries. Cf. https://salmon.readthedocs.io/en/latest/library_type.html

inward_stranded_forward
Mates are oriented toward each other, the library is stranded, and first mates come from the forward strand.

inward_stranded_reverse
Mates are oriented toward each other, the library is stranded, and first mates come from the reverse strand.

inward_unstranded
Mates are oriented toward each other and the library is unstranded.

not_available
Orientation type relationship information is not available, likely because only a single file was provided or because the orientation type relationship has not been or could not be evaluated.

inward_stranded_forward = 'ISF'

inward_stranded_reverse = 'ISR'

inward_unstranded = 'IU'

not_available = None

class htsinfer.models.**StatesType**(*value*)
Bases: enum.Enum

Possible outcomes of determining the sequencing library type of an individual FASTQ file.

file_problem
There was a problem with opening or parsing the file.

first_mate
All of the sequence identifiers of the processed file counts indicate that the library represents the first mate of a paired-end library.

mixed_mates
All of the sequence identifiers of the processed file include mate information. However, the file includes at least one record for either mate, indicating that the library represents a mixed mate library.

not_available
Library type information is not available for a given file, either because no file was provided, the file could not be parsed, a library type has not yet been assigned, the processed file contains records with sequence identifiers of an unknown format, of different formats or that are inconsistent in that they indicate the library represents both a single-ended and paired-ended library at the same time.

second_mate
All of the sequence identifiers of the processed file indicate that the library represents the second mate of a paired-end library.

single
All of the sequence identifiers of the processed file indicate that the library represents a single-end library.

first_mate = 'first_mate'

mixed_mates = 'mixed_mates'

not_available = None

second_mate = 'second_mate'

single = 'single'

class htsinfer.models.**StatesTypeRelationship**(*value*)
Bases: enum.Enum

Possible outcomes of determining the sequencing library type/mate relationship between two FASTQ files.

not_available
Mate relationship information is not available, likely because only a single file was provided or because the mate relationship has not yet been evaluated.

not_mates
The library type information of the files is not compatible, either because not a pair of first and second mate files was provided, or because the files do not compatible sequence identifiers.

split_mates
One of the provided files represents the first and the the other the second mates of a paired-end library.

not_available = None

not_mates = 'not_mates'

split_mates = 'split_mates'

class htsinfer.models.**Stats**(*, *read_length:* htsinfer.models.ReadLength = *ReadLength(min=None, max=None)*)
Bases: pydantic.main.BaseModel

Library statistics of an individual sequencing file.

Parameters **read_length** – Tuple of minimum and maximum length of reads in library.

read_length
Tuple of minimum and maximum length of reads in library.

Type *htsinfer.models.ReadLength*

read_length: *htsinfer.models.ReadLength*

1.1.11 htsinfer.subset_fastq module

FASTQ subsetting, extraction and validation.

class htsinfer.subset_fastq.**SubsetFastq**(*path: pathlib.Path*, *out_dir: pathlib.Path = PosixPath('/home/docs/checkouts/readthedocs.org/user_builds/htsinfer/checkouts/latest/*
records: int = 0)

Bases: object

Subset, uncompress and validate a FASTQ file.

Parameters

- **path** – Path to FASTQ file.
- **out_dir** – Path to directory where output is written to.
- **records** – Number of input file records to process; set to *0* to process all records.

path
Path to FASTQ file.

out_dir
Path to directory where output is written to.

records
Number of input file records to process.

out_path
Path for uncompressed, filtered *path* file.

n_processed
Total number of processed records.

Raises *FileProblem* – The input file could not be parsed or the output file could not be written.

process()
Uncompress, subset and validate files.

1.1.12 htsinfer.utils module

Utilities used across multiple HTSinfer modules.

htsinfer.utils.**convert_dict_to_df**(*dic: Dict*, *col_headers: Optional[Tuple[str, str]] = None*, *sort: bool = False*, *sort_by: int = 0*, *sort_ascending: bool = True*) → pandas.core.frame.DataFrame

Convert dictionary to two-column data frame.

Parameters

- **dic** – Dictionary to convert.
- **col_headers** – List of column headers. Length MUST match number of dictionary keys/data frame columns.

- **sort** – Whether the resulting data frame is supposed to be sorted.
- **sort_by** – Column index used for sorting. Ignored if *sort* is *False*.
- **sort_ascending** – Whether the data frame is supposed to be sorted in ascending order. Ignored if *sort* is *False*.

Returns Data frame prepared from dictionary.

Raises **ValueError** – Raised if number of provided column headers does not match the number of data frame columns.

htsinfer.utils.**validate_top_score**(*vector: List[float]*, *min_value: float = 2*, *min_ratio: float = 2*, *accept_zero: bool = True*, *rev_sorted: bool = True*) → bool

Validates whether (1) the maximum value of a numeric list is equal to or higher than a specified minimum value AND (2) that the ratio of the first and second highest values of the list is higher than a specified minimum ratio.

If the passed list/vector does NOT contain at least two items, the function returns *False*.

Parameters

- **vector** – List of numbers.
- **min_value** – Minimum value required in first row of *column_index* for validation to pass.
- **min_ratio** – Minimum ratio of first and second rows of *column_index* required for validation to pass.
- **accept_zero** – Whether to accept a top score (i.e., return *True*) if the second highest value in the provided list is zero. If not set to *True*, *False* is returned in these cases.
- **rev_sorted** – Whether the list of numbers is sorted in descencing numeric order.

Returns Whether data frame *data* satisfies the *min_value* and *min_ratio* constraints for value in column *column_index*.

Raises **ValueError** – Raised if one of the list items can not be interpreted as a number.

CHAPTER

TWO

INDICES AND TABLES

- genindex
- modindex

PYTHON MODULE INDEX

h

htsinfer, 1
htsinfer.cli, 1
htsinfer.exceptions, 1
htsinfer.get_library_source, 2
htsinfer.get_library_stats, 4
htsinfer.get_library_type, 4
htsinfer.get_read_layout, 5
htsinfer.get_read_orientation, 8
htsinfer.htsinfer, 11
htsinfer.models, 13
htsinfer.subset_fastq, 20
htsinfer.utils, 20

INDEX

A

adapt_3 (*htsinfer.models.Layout attribute*), 13
adapter_counts (*htsinfer.get_read_layout.GetAdapter3 attribute*), 6
adapter_file (*htsinfer.get_read_layout.GetAdapter3 attribute*), 6
adapter_file (*htsinfer.get_read_layout.GetReadLayout attribute*), 7
adapters (*htsinfer.get_read_layout.GetAdapter3 attribute*), 6

C

clean_up() (*htsinfer.htsinfer.HtsInfer method*), 13
cleanup_regime (*htsinfer.htsinfer.HtsInfer attribute*), 12
CleanupRegimes (*class in htsinfer.models*), 13
convert_dict_to_df() (*in module htsinfer.utils*), 20
create_kallisto_index() (*htsinfer.get_library_source.GetLibSource method*), 3
create_star_index() (*htsinfer.get_read_orientation.GetOrientation method*), 9
CRITICAL (*htsinfer.models.LogLevels attribute*), 14

D

DEBUG (*htsinfer.models.LogLevels attribute*), 14
DEFAULT (*htsinfer.models.CleanupRegimes attribute*), 13

E

ERROR (*htsinfer.models.LogLevels attribute*), 14
ERROR (*htsinfer.models.RunStates attribute*), 17
evaluate() (*htsinfer.get_library_source.GetLibSource method*), 3
evaluate() (*htsinfer.get_library_stats.GetLibStats method*), 4
evaluate() (*htsinfer.get_library_type.GetFastqType method*), 5
evaluate() (*htsinfer.get_library_type.GetLibType method*), 5
evaluate() (*htsinfer.get_read_layout.GetAdapter3 method*), 6
evaluate() (*htsinfer.get_read_layout.GetReadLayout method*), 8
evaluate() (*htsinfer.get_read_orientation.GetOrientation method*), 9
evaluate() (*htsinfer.htsinfer.HtsInfer method*), 13

F

fastq_get_min_max_read_length() (*htsinfer.get_library_stats.GetLibStats static method*), 4
file_1 (*htsinfer.models.ResultsLayout attribute*), 15
file_1 (*htsinfer.models.ResultsOrientation attribute*), 15, 16
file_1 (*htsinfer.models.ResultsSource attribute*), 16
file_1 (*htsinfer.models.ResultsStats attribute*), 16
file_1 (*htsinfer.models.ResultsType attribute*), 17
file_2 (*htsinfer.models.ResultsLayout attribute*), 15
file_2 (*htsinfer.models.ResultsOrientation attribute*), 16
file_2 (*htsinfer.models.ResultsSource attribute*), 16
file_2 (*htsinfer.models.ResultsStats attribute*), 16
file_2 (*htsinfer.models.ResultsType attribute*), 17
file_problem (*htsinfer.models.StatesType attribute*), 19
FileProblem, 1
first_mate (*htsinfer.models.StatesType attribute*), 19

G

generate_star_alignments() (*htsinfer.get_read_orientation.GetOrientation static method*), 9
get_fasta_size() (*htsinfer.get_read_orientation.GetOrientation static method*), 9
get_frequencies() (*htsinfer.get_read_orientation.GetOrientation static method*), 9
get_library_source() (*htsinfer.htsinfer.HtsInfer method*), 13
get_library_stats() (*htsinfer.htsinfer.HtsInfer method*), 13
get_library_type() (*htsinfer.htsinfer.HtsInfer method*), 13

get_read_layout() (*htsinfer.htsinfer.HtsInfer method*), 13
get_read_orientation() (*htsinfer.htsinfer.HtsInfer method*), 13
get_source() (*htsinfer.get_library_source.GetLibSource method*), 3
get_source_expression() (*htsinfer.get_library_source.GetLibSource static method*), 3
get_star_index_string_size() (*htsinfer.get_read_orientation.GetOrientation static method*), 10
GetAdapter3 (*class in htsinfer.get_read_layout*), 5
GetFastqType (*class in htsinfer.get_library_type*), 4
GetLibSource (*class in htsinfer.get_library_source*), 2
GetLibStats (*class in htsinfer.get_library_stats*), 4
GetLibType (*class in htsinfer.get_library_type*), 5
GetOrientation (*class in htsinfer.get_read_orientation*), 8
GetReadLayout (*class in htsinfer.get_read_layout*), 6

H

htsinfer
    module, 1
HtsInfer (*class in htsinfer.htsinfer*), 11
htsinfer.cli
    module, 1
htsinfer.exceptions
    module, 1
htsinfer.get_library_source
    module, 2
htsinfer.get_library_stats
    module, 4
htsinfer.get_library_type
    module, 4
htsinfer.get_read_layout
    module, 5
htsinfer.get_read_orientation
    module, 8
htsinfer.htsinfer
    module, 11
htsinfer.models
    module, 13
htsinfer.subset_fastq
    module, 20
htsinfer.utils
    module, 20

I

InconsistentFastqIdentifiers, 1
INFO (*htsinfer.models.LogLevels attribute*), 14
inward_stranded_forward (*htsinfer.models.StatesOrientationRelationship attribute*), 18
inward_stranded_reverse (*htsinfer.models.StatesOrientationRelationship attribute*), 18
inward_unstranded (*htsinfer.models.StatesOrientationRelationship attribute*), 18

K

KallistoProblem, 1
KEEP_ALL (*htsinfer.models.CleanupRegimes attribute*), 13
KEEP_NONE (*htsinfer.models.CleanupRegimes attribute*), 13
KEEP_RESULTS (*htsinfer.models.CleanupRegimes attribute*), 13

L

Layout (*class in htsinfer.models*), 13
lib_source_min_freq_ratio (*htsinfer.htsinfer.HtsInfer attribute*), 12
lib_source_min_match_pct (*htsinfer.htsinfer.HtsInfer attribute*), 12
library_source (*htsinfer.get_read_orientation.GetOrientation attribute*), 9
library_source (*htsinfer.models.Results attribute*), 15
library_stats (*htsinfer.models.Results attribute*), 15
library_type (*htsinfer.get_read_orientation.GetOrientation attribute*), 8
library_type (*htsinfer.models.Results attribute*), 15
LogLevels (*class in htsinfer.models*), 14

M

main() (*in module htsinfer.cli*), 1
mate_relationship (*htsinfer.get_read_orientation.GetOrientation attribute*), 9
max (*htsinfer.models.ReadLength attribute*), 14
MetadataWarning, 1
min (*htsinfer.models.ReadLength attribute*), 14
min_fraction (*htsinfer.get_read_orientation.GetOrientation attribute*), 9
min_freq_ratio (*htsinfer.get_read_layout.GetAdapter3 attribute*), 6
min_freq_ratio (*htsinfer.get_read_layout.GetReadLayout attribute*), 7
min_mapped_reads (*htsinfer.get_read_orientation.GetOrientation attribute*), 9
min_match_pct (*htsinfer.get_read_layout.GetAdapter3 attribute*), 6

min_match_pct (*htsinfer.get_read_layout.GetReadLayout attribute*), 7
mixed_mates (*htsinfer.models.StatesType attribute*), 19
module
    htsinfer, 1
    htsinfer.cli, 1
    htsinfer.exceptions, 1
    htsinfer.get_library_source, 2
    htsinfer.get_library_stats, 4
    htsinfer.get_library_type, 4
    htsinfer.get_read_layout, 5
    htsinfer.get_read_orientation, 8
    htsinfer.htsinfer, 11
    htsinfer.models, 13
    htsinfer.subset_fastq, 20
    htsinfer.utils, 20

N

n_processed (*htsinfer.subset_fastq.SubsetFastq attribute*), 20
not_available (*htsinfer.models.StatesOrientation attribute*), 18
not_available (*htsinfer.models.StatesOrientationRelationship attribute*), 18
not_available (*htsinfer.models.StatesType attribute*), 19
not_available (*htsinfer.models.StatesTypeRelationship attribute*), 19
not_mates (*htsinfer.models.StatesTypeRelationship attribute*), 19

O

OKAY (*htsinfer.models.RunStates attribute*), 17
out_dir (*htsinfer.get_read_layout.GetAdapter3 attribute*), 6
out_dir (*htsinfer.get_read_layout.GetReadLayout attribute*), 7
out_dir (*htsinfer.htsinfer.HtsInfer attribute*), 12
out_dir (*htsinfer.subset_fastq.SubsetFastq attribute*), 20
out_path (*htsinfer.subset_fastq.SubsetFastq attribute*), 20

P

parse_args() (*in module htsinfer.cli*), 1
path (*htsinfer.get_library_type.GetFastqType attribute*), 4
path (*htsinfer.get_read_layout.GetAdapter3 attribute*), 6
path (*htsinfer.subset_fastq.SubsetFastq attribute*), 20
path_1 (*htsinfer.get_read_layout.GetReadLayout attribute*), 7
path_1 (*htsinfer.htsinfer.HtsInfer attribute*), 11
path_1_processed (*htsinfer.htsinfer.HtsInfer attribute*), 12
path_2 (*htsinfer.get_read_layout.GetReadLayout attribute*), 7
path_2 (*htsinfer.htsinfer.HtsInfer attribute*), 11
path_2_processed (*htsinfer.htsinfer.HtsInfer attribute*), 12
paths (*htsinfer.get_library_stats.GetLibStats attribute*), 4
paths (*htsinfer.get_read_orientation.GetOrientation attribute*), 8
prepare_env() (*htsinfer.htsinfer.HtsInfer method*), 13
prepare_star_alignment_commands() (*htsinfer.get_read_orientation.GetOrientation method*), 10
print() (*htsinfer.htsinfer.HtsInfer method*), 13
process() (*htsinfer.subset_fastq.SubsetFastq method*), 20
process_alignments() (*htsinfer.get_read_orientation.GetOrientation method*), 10
process_inputs() (*htsinfer.htsinfer.HtsInfer method*), 13
process_paired() (*htsinfer.get_read_orientation.GetOrientation method*), 10
process_single() (*htsinfer.get_read_orientation.GetOrientation method*), 10

R

read_layout (*htsinfer.models.Results attribute*), 15
read_layout_adapter_file (*htsinfer.htsinfer.HtsInfer attribute*), 12
read_layout_min_freq_ratio (*htsinfer.htsinfer.HtsInfer attribute*), 12
read_layout_min_match_pct (*htsinfer.htsinfer.HtsInfer attribute*), 12
read_length (*htsinfer.models.Stats attribute*), 19, 20
read_orientation (*htsinfer.models.Results attribute*), 15
read_orientation_min_fraction (*htsinfer.htsinfer.HtsInfer attribute*), 12
read_orientation_min_mapped_reads (*htsinfer.htsinfer.HtsInfer attribute*), 12
ReadLength (*class in htsinfer.models*), 14
records (*htsinfer.htsinfer.HtsInfer attribute*), 12
records (*htsinfer.subset_fastq.SubsetFastq attribute*), 20
relationship (*htsinfer.models.ResultsOrientation attribute*), 16
relationship (*htsinfer.models.ResultsType attribute*), 17
result (*htsinfer.get_library_type.GetFastqType attribute*), 4

result (*htsinfer.get_read_layout.GetAdapter3 attribute*), 6
Results (*class in htsinfer.models*), 14
results (*htsinfer.get_read_layout.GetReadLayout attribute*), 7
results (*htsinfer.htsinfer.HtsInfer attribute*), 12
ResultsLayout (*class in htsinfer.models*), 15
ResultsOrientation (*class in htsinfer.models*), 15
ResultsSource (*class in htsinfer.models*), 16
ResultsStats (*class in htsinfer.models*), 16
ResultsType (*class in htsinfer.models*), 16
run_id (*htsinfer.htsinfer.HtsInfer attribute*), 12
run_kallisto_quantification() (*htsinfer.get_library_source.GetLibSource method*), 3
RunStates (*class in htsinfer.models*), 17

S

second_mate (*htsinfer.models.StatesType attribute*), 19
seq_id_format (*htsinfer.get_library_type.GetFastqType attribute*), 4
seq_ids (*htsinfer.get_library_type.GetFastqType attribute*), 4
SeqIdFormats (*class in htsinfer.models*), 17
setup_logging() (*in module htsinfer.cli*), 1
short_name (*htsinfer.models.Source attribute*), 17, 18
single (*htsinfer.models.StatesType attribute*), 19
Source (*class in htsinfer.models*), 17
source (*htsinfer.get_read_orientation.GetOrientation attribute*), 9
split_mates (*htsinfer.models.StatesTypeRelationship attribute*), 19
StarProblem, 2
state (*htsinfer.htsinfer.HtsInfer attribute*), 12
StatesOrientation (*class in htsinfer.models*), 18
StatesOrientationRelationship (*class in htsinfer.models*), 18
StatesType (*class in htsinfer.models*), 18
StatesTypeRelationship (*class in htsinfer.models*), 19
Stats (*class in htsinfer.models*), 19
stranded_forward (*htsinfer.models.StatesOrientation attribute*), 18
stranded_reverse (*htsinfer.models.StatesOrientation attribute*), 18
subset_transcripts_by_organism() (*htsinfer.get_read_orientation.GetOrientation method*), 10
SubsetFastq (*class in htsinfer.subset_fastq*), 20
sum_dicts() (*htsinfer.get_read_orientation.GetOrientation static method*), 10

T

taxon_id (*htsinfer.models.Source attribute*), 17, 18
threads (*htsinfer.htsinfer.HtsInfer attribute*), 12
threads_star (*htsinfer.get_read_orientation.GetOrientation attribute*), 9
tmp_dir (*htsinfer.get_library_stats.GetLibStats attribute*), 4
tmp_dir (*htsinfer.get_read_orientation.GetOrientation attribute*), 9
tmp_dir (*htsinfer.htsinfer.HtsInfer attribute*), 12
transcripts_file (*htsinfer.get_read_orientation.GetOrientation attribute*), 9
transcripts_file (*htsinfer.htsinfer.HtsInfer attribute*), 12
transcripts_file_processed (*htsinfer.htsinfer.HtsInfer attribute*), 12
trie (*htsinfer.get_read_layout.GetAdapter3 attribute*), 6

U

UnknownFastqIdentifier, 2
unstranded (*htsinfer.models.StatesOrientation attribute*), 18

V

validate_top_score() (*in module htsinfer.utils*), 21

W

WARN (*htsinfer.models.LogLevels attribute*), 14
WARNING (*htsinfer.models.LogLevels attribute*), 14
WARNING (*htsinfer.models.RunStates attribute*), 17
WorkEnvProblem, 2