
Sketch

Release 0.14.2

Daniel Baker

Aug 07, 2023

CONTENTS::

1	Getting Started	1
1.1	Installation (Python)	1
1.2	Installation (C++)	1
1.3	Features	1
2	Modules	3
3	Additional utilities: sketch_ds.util	5
3.1	Computing optimal a and b	6
3.2	Indices and tables	6

GETTING STARTED

Sketch has a range of sketch data structures implemented. All of them are available in C++, but only a subset of functionality has been exposed to Python. This documentation is primarily for the Python interface.

1.1 Installation (Python)

One-line installation:

```
git clone --recursive https://github.com/dnbaker/sketch && cd sketch/python && python3 ↪setup.py build_ext -j4 && python3 setup.py install
```

At this point, you will simply import sketch from python:

```
import sketch_ds
HLL = sketch_ds.hll.hll
h = HLL(10)
for i in range(100000): h.addh(i)
print("Estimated cardinality: %f" % h.report())
```

1.2 Installation (C++)

You don't. Use sketch as a header-only library, but clone recursively.

1.3 Features

1. Sketch structure bindings:

1. Bloom Filter
2. HyperLogLog
3. B-bit minhash
4. Set Sketch

2. Distance calculation functions

3. Miscellaneous

1. fastmod/fastdiv for integer reductions

2. ngram hashing
3. fast hamming space distance calculations

MODULES

There are separate modules for each sketch structure for which there are bindings.

- `sketch_ds.hll`, providing HyperLogLog and comparison, and serialization functions
- `sketch_ds.bf`, providing Bloom Filters and comparison, and serialization functions
- `sketch_ds.bbmh`, providing b-bit minhash implementation + comparison, and serialization functions
- `sketch_ds.setsketch`, providing set sketch + comparison, and serialization functions

For each of these, the module provides construction - either taking parameters or a path to a file. Each of these can be written to and read from a file with `.write()` and a constructor.

They can be compared with each other with member functions, or you can calculate comparison matrices via `sketch_ds.util.jaccard_matrix`, `sketch_ds.util.containment_matrix`, `sketch_ds.util.union_size_matrix`, `sketch_ds.util.intersection_matrix`, all of which are in the `util` module.

Additionally, there are utilities for pairwise distance calculation in the *util* module.

ADDITIONAL UTILITIES: SKETCH_DS.UTIL

- **fastdiv/fastmod:**
 - Python bindings for fastdiv/fastmod; See <https://arxiv.org/abs/1902.01961>
 - fastdiv_ and fastmod_ are in-place modifications, while the un-suffixed returns a new array
- **count_eq**
 - Compute # of equal registers between two 1-d numpy arrays.
- **ccount_eq**
 - Compute row-pair-wise equal register counts between two 2-d numpy arrays.
- **pcount_eq**
 - Compute row-wise upper triangular distance matrix for equal register counts for 1 2-d numpy array.
- **shsisz**
 - Computes intersection size between two sorted hash sets.
- **hash**
 - hashes strings
- **hash_ngrams**
 - takes a list of strings, and then computes

usage

```
def hash\_ngrams(toks, n=3, seed=0):  
:param toks: list of strings  
:param n:    n- for n-grams, default = 3  
:param seed: Set seed for hashing; default = 0  
:returns: np.ndarray, with dtype = np.uint64
```

3.1 Computing optimal a and b

For lossy compression via quantization, `_optimal_ab_` computes the parameter values for best using hash space.

3.2 Indices and tables

- `genindex`
- `modindex`
- `search`