Memory usage analysis of binary clustering algorithm

What is the gain in peak memory usage of the binary clustering algorithm compared to current state-of-the-art clustering methods?

Pavel Verigo[1]

Supervisor(s): Marcel Reinders[1]**, Gerard Bouland**[1]

[1]**EEMCS, Delft University of Technology, The Netherlands**

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 25, 2023

Name of the student: Pavel Verigo
Final project course: CSE3000 Research Project
Thesis committee: Marcel Reinders, Gerard Bouland, Bart Gerritsen

Abstract: The rapid increase in the size of single-cell RNAseq datasets presents significant performance challenges when conducting evaluations and extracting information. We research an alternative input data format that utilizes binarization. Our main focus is an analysis of peak memory usage. An in-depth exploration of the solution's design and implementation is provided, specifically emphasizing the strategies used to minimize memory usage. We analyzed and validated memory usage patterns and asymptotes using memory profiling tools. However, our findings suggest that gains in reducing memory usage on big modern datasets are attributable only to binarized data format rather than workflow interaction with the new format, which we found to be independent of the input format.

1 Introduction

Analyzing single-cell RNA sequencing (scRNA-seq) data has become a key tool for gaining valuable insights into various biological processes. However, the growing size [7] and complexity of scRNA-seq datasets pose significant challenges for data analysis, particularly regarding scalability.

The driving force behind this research is the need for more efficient clustering techniques in analyzing scRNA-seq data. Clustering algorithms are crucial in identifying distinct cell populations within the dataset, providing researchers with insights into cellular heterogeneity and functional diversity. However, existing clustering methods often need more computational efficiency and scalability [10].

In this research, our primary objective is to assess a binary clustering algorithm's peak memory usage gain compared to current state-of-the-art clustering methods for scRNA-seq data analysis. We hypothesize that the binary clustering algorithm proposed by Bouland [3] can be more memory-efficient and computationally faster while maintaining comparable clustering quality, especially considering that scRNA-seq datasets are sparse in their nature [3].

1.1 A Quick Background on Clustering Workflow

Clustering workflow presents a solution to group cells based on the initial gene expression matrix, intending to identify cells with similar genes and draw conclusions on gene function. In figure 1, we illustrate a common research workflow to obtain cluster designations, which commonly consist of the following:

1. Getting gene expression matrix in memory
2. Computing cell-to-cell similarity matrix based on a metric
3. Constructing K nearest neighbors graph based on adding only best K edges for each cell.
4. Running clustering algorithm, on such neighbor graph, Louvain [2] or it modern Leiden [8] alternative can be used to obtain clusters efficiently

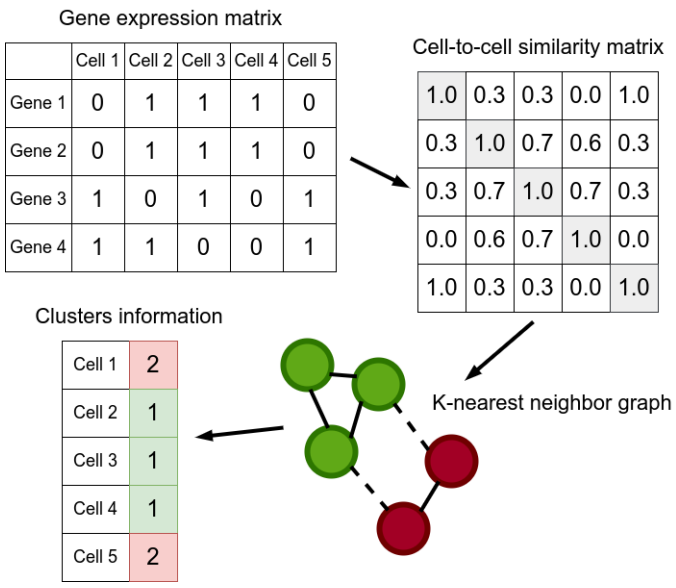

Figure 1: Clustering workflow illustration for five cells, with four genes.

Alternatively, Principal component analysis (PCA) can be used before computing cell to cell similarity matrix. PCA helps to reduce the dimensionality of the original dataset, enabling us to uncover patterns and identify the genes that impact the overall variance within the dataset.

1.2 Research Question and Plan

The research question of this paper is "What is the gain in peak memory usage of the binary clustering algorithm compared to current state-of-the-art clustering methods?". To tackle this, we divided our research into several stages:

- Theoretical analyses on optimal memory usage during clustering workflow. Identify areas that contribute to memory using the most
- Run profiling on our implementation confirming hypothesizes and obtaining the quantitive measurement of memory usage based on input
- We discuss our findings and also compare and explain the difference in memory usage between our solution and currently used toolkits

During the research, we will use such publicly available datasets:

1. Peripheral Blood Mononuclear Cell (PBMC)[1] dataset, from 10X Genomics, a relatively small dataset consisting of 2,700 cells, with 13,712 genes
2. Tabula Muris [4]. Compendium of datasets from different organs of Mus musculus. There are 53,760 cells, with 23,432 genes
3. Tabula Sapiens [5]. Cell atlas from different human organs, due to the enormous size of each dataset, we only used 4,000 Skin cells, with 58,869 genes

4. Entorhinal cortex from individuals with Alzheimer's disease [6]. A single-cell atlas having 13,214 cells, with 10,849 genes

Seurat toolkit[1] will be used as a comparison to our research as it is one of the most popular solutions in the research space.

2 Theoretical Analysis of Memory Usage

Our research aims to compare memory usage based on input parameters for clustering workflow, such as the dataset size. We denote G as the number of genes and C as the number of cells in a gene expression matrix. Additionally, the clustering pipeline takes K as a parameter, setting the number of edges spawned per cell for generating a KNN graph.

We will use bytes as the unit for memory management, given it is the smallest addressable memory unit in modern processor architectures.

2.1 Initial Dataset Memory Usage

We measure memory at the start of loading binarized data into the main memory from the disk. Since each gene expression needs only one bit, we estimate the memory usage is $\frac{1}{8}$ (8 bits per byte) $\times\, C \times G$.

Note that memory that is occupied by the dataset will be used as the main baseline for memory usage in this research.

2.2 Cell-to-Cell Similarity Matrix

To store computed cell-to-cell similarity matrix in memory, given the matrix's symmetry, we need 4 (32-bit float)$\times \frac{1}{2} \times C^2$ bytes of memory. Given our research focus on improving the workflow for big datasets, which may contain $C > 2 \cdot 10^5$ cells, will lead to usage $4 \times \frac{1}{2} \times 4 \cdot 10^{10} = 80$ GiB of RAM usage at least. Such value exceeds the maximum memory of any common research machine. To circumvent this, we will only output the K nearest cells for each cell-based similarity. An efficient implementation will result in lower bound usage of $O(K \times C)$. Concrete values will be discussed in the subsequent section.

None of the similarity metrics do any allocations. Therefore we were not concluding any experiments on the topic of different metrics.

2.3 Graph Construction and Clustering

Based on the nearest K vertices, we construct a KNN graph, which includes weights based on the similarity between cells, and afterward, the selected clustering algorithm will run. As the precise number of bytes per edges/vertices of the graph is challenging to pinpoint due to high implementation details, we introduce a new coefficient Q, representing the number of bytes used in graph construction and clustering per number of edges. Therefore, merging with the output size from the previous section, we define the total number of bytes used as $Q \times K \times C$.

[1] https://satijalab.org/seurat/

2.4 Finalising and Observation

Excluding small and auxiliary allocations, such as the clustering algorithm output, an array of integers size of C. We identify two terms that we are going to use to approximate memory usage:

$$\frac{1}{8} \times C \times G + Q \times K \times C \quad \text{bytes} \tag{1}$$

Because the second term is not dependent on the initial number of genes G, if

$$\frac{G}{8} >> Q \times K \tag{2}$$

holds, total memory usage will be dependent mostly on the input dataset. To make any arguments on particular G when the above is true, we first need to identify Q.

3 Methodology and Implementation

3.1 Memory Profiling

For profiling, we are using a heap profiler called `Massif`[2]. It measures memory usage alongside program execution and does time-based snapshots, recording all memory allocation. In the end, Massif outputs profiling data, which is processed using custom-made Python scripts.

3.2 Implementation

Our main language for implementing our pipeline is `C++20`[3], as advised by our project supervisor. This language integrates seamlessly with tooling provided by the R[4] language, which we built bindings for as a part of the project. We are using the `Rcpp`[5] package, which simplifies the integration. We personalized each step of the pipeline, ensuring a flexible design that enables easy substitution of different distance metrics and clustering methods. We employed the third-party `igraph`[6] library to efficiently implement Louvain and Leiden clustering since developing a graph clustering algorithm was beyond the purview of our research project.

3.3 Binary Array

We implement a common data structure called binary array to store binarized gene expression of cells. This structure efficiently stores bits in data blocks, the size of which depends on the system's specific CPU architecture. For instance, in a 64-bit machine, we use 64 bits blocks, as depicted in Figure 2.

3.4 Dataset and Hardware Independence of Memory Usage

Our implementation is not affected by the dataset content. If the input dataset contains only ones, the memory usage will not be changed. Therefore, we are not that concerned that random sampling reduces dataset sparsity. This comes from

[2] https://valgrind.org/docs/manual/ms-manual.html
[3] https://en.cppreference.com/w/cpp/20
[4] https://www.r-project.org/
[5] https://www.rcpp.org/
[6] https://igraph.org/

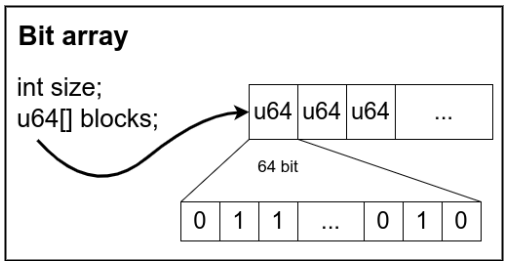

Figure 2: Diagram of the bit array structure. Block size equal to 64.

the fact that any data of the same size will use the same memory usage, and computing neighborhood graphs and clustering is independent of dataset values in terms of memory usage. If we use any cutoff on edge weight before adding to a KNN graph, we must be more careful with sampling.

We must note that memory usage is independent of the machine used. We identify that memory access speed is dependent, but such analysis is out of this research's scope.

4 Profiling Experiment

In order to evaluate our hypotheses, we conducted multiple profiling scenarios.

4.1 Estimating Q

To estimate Q, we ran the clustering workflow on a PBMC dataset sampled with varying numbers of cells. We only count the amount of memory used without taking into account the memory used by the dataset, so the resulting memory should be equal to $Q \times K \times C$ bytes. Furthermore, the workflow was executed with different K values to demonstrate the independence of Q from it. Figure 3 depicts resulting linear approximations on 15 data points.

K	ρ, Pearson Coefficient	Slope (on MiB scale)
5	0.99	854.40
10	0.99	1534.03
15	0.99	2310.75

Table 1: Linear Approximation Results

Next, to approximate Q, we run linear approximation on slope values from Table 1. Figure 4 reveals linear approximation, which results in $Q = 145.63$ MiB per cell, with Pearson coefficient $\rho = 0.9992$.

4.2 More Genes, More Memory used

One of our memory usage estimation conclusions from Formula 1 is that datasets with a high number of genes will benefit from binarized data greatly because most memory allocated during the construction of the KNN graph and clustering could be neglected.

We run our simple workflow on the Tabula Sapiens (skin only) and Entorhinal cortex cells dataset. And computed memory usage after the dataset loaded and peak memory usage, Table 2.

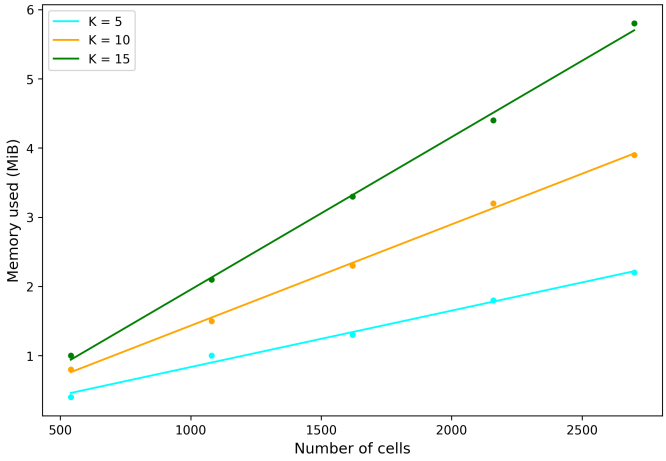

Figure 3: Peak memory usage of workflow excluding memory used by dataset, X-axis represent number cells in the dataset.

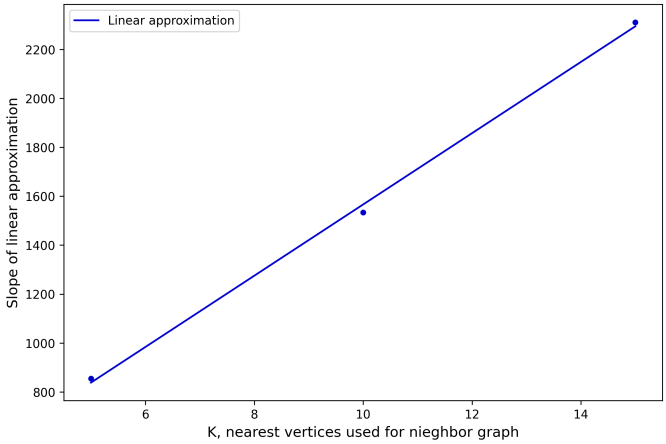

Figure 4: Memory used per cell in the clustering part of the workflow. X-axis represents K number of nearest cells to which edges will be connected in the neighbor graph.

G	Dataset loaded	Peak memory
10850	17.8 MiB	32.4 MiB
58870	28.3 MiB	32.7 MiB

Table 2: Memory usage when datasets are loaded and peak usage after clustering is completed. Alzheimer's and Skin datasets are shown in the first and second rows, respectively.

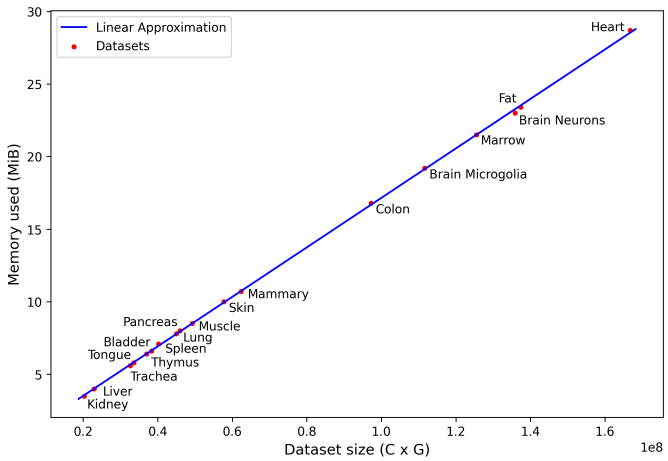

Figure 5: Peak memory usage of different datasets from Tabula Muris. X-axis represents the dataset size.

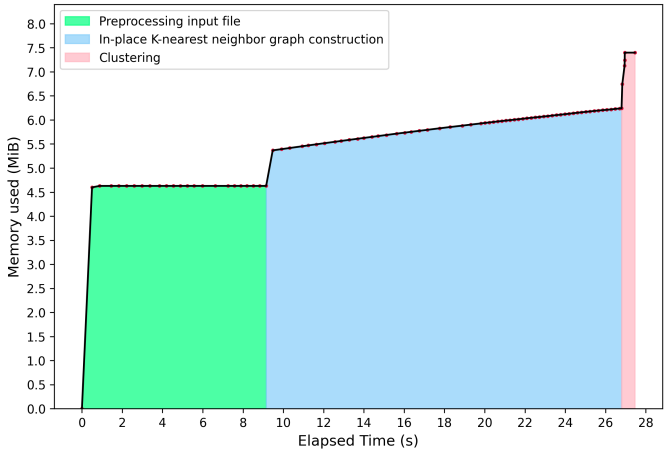

Figure 6: Memory usage of our implementation on binarized PBMC dataset. X-axis represents the time elapsed from the start.

4.3 Linearity with constant number of genes

In figure 5, we plotted peak memory usage relative to dataset size $C \times G$, on Tabula Muris dataset collection, where each dataset contains 23,000 genes. We run linear approximation returning slope coefficient of ≈ 0.178 with Pearson coefficient $\rho = 0.99$. Note that the slope value is close to $\frac{1}{8} = 0.125$, concluding that gene-expression counts use the most memory.

4.4 Comparing with Seurat

We executed our implementation and Seurat on the PBMC dataset. Figures 6 and 7 display the memory usage graphs based on time elapsed from the start. Note that for Seurat, we subtracted memory allocated for the Seurat binary when it is loaded into R runtime.

Unfortunately, because Seurat does PCA computation and stores a lot more information that we may not need in the simplest workflow resulting in using around 950 MiB amount of memory, having a simple workflow that uses small libraries

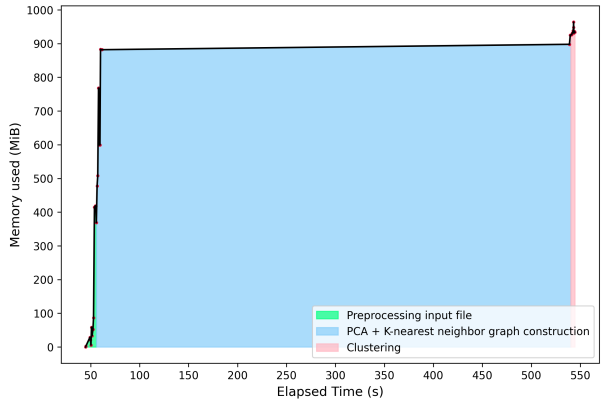

Figure 7: Memory usage of Seurat toolkit on PBMC dataset. X-axis represents the time elapsed from the start.

may be a better solution. Note that the Seurat toolkit, is also a combination of libraries under one API.

5 Discussion and Future Work

5.1 Drawing conclusion on G

Plugin in $Q \approx 145.6$ to Formula 2, and taking into account common that usually inbound $10 \leq K \leq 20$, we may state if dataset have 10^5 genes this results to:

$$\frac{10^5}{8} = 12500 >> 2912 = 145.6 \cdot 20$$

Thus, for big datasets, memory occupied by the dataset being loaded into memory contributes most to peak memory usage.

5.2 Reducing the Q Constant

After running the profiler on the clustering workflow, it provided information about possible optimization of our implementation to decrease the Q constant. By thoroughly examining all allocations in our code responsible for most allocated memory and the `igraph` library, we concluded that it is possible to reduce Q under 100. Below we present some ideas that could help us to achieve this:

1. Directly calculating graph edges into the `igraph` data structure instead of storing them in an intermediate format and then transforming and reallocating them into a new format.
2. Upon examination, we noticed that the `igraph` library performs unnecessary reallocations, such as allocating new edges and weights lists, even when our passed version is not needed elsewhere. Providing an alternative implementation will reduce constant. Unfortunately, this is mostly responsible for $Q > 100$.
3. Storing binarized input in matrix format instead of an array of cells. While it may be tempting to store each cell in its data structure with attached auxiliary data, this would result in duplicate information (such as storing the number of genes for each cell) and would fragment memory because every cell expression would be stored

in a separate memory region instead of a continuous array.

5.3 R Language Integration Considerations

The main reason behind this research is to provide researchers with a better clustering workflow experience. It is assumed that interface to our researcher is to be `R` language. Our current implementation needs to be independent of the `igraph` library, allowing researchers to use other graph analysis libraries, which may have better performance characteristics. We propose that the library providing analysis on binarized input should efficiently load compressed and uncompressed datasets from various representations and, after computing similarity between cells, return only edges and weights as output, from which the KNN graph can be constructed.

One topic of discussion arising from using `R` could be the observation that common `R` scripts for analyzing datasets are often not memory-aware. If a researcher loads memory twice into different variables, all performance analysis becomes unimportant. We propose to provide library users instructions for their workflow to be memory efficient.

5.4 Alternative Memory Layout for Sparse Datasets

We propose another approach to gene expression data storage for future research work. First, you need to identify a group of genes with a very low rate of occurrence per cell and store their indices in a compressed way, while other genes are stored uncompressed in binary format. This idea requires extensive benchmarking and profiling work and thorough analysis of hypotheses across a range of datasets. Such a solution is a mixture of compressed data representation and binarized one. The advantage of our current approach that was tested in this research is the dataset content does not influence that memory usage.

6 Responsible Research

In this study, we utilized pre-existing datasets, the authenticity of which was not questioned. Because we centered our research around memory profiling of the clustering algorithm, we assume that any biases in the given data are irrelevant to our results.

Addressing the ethical aspects of our research, we were not involved in the production of the dataset, nor did we make any personal or biological conclusions based on the algorithm results, thereby limiting our research from having ethical implications.

For transparency and reproducibility, we documented memory profiling methods for clustering workflow under test. All code and data utilized in this study have been made publicly available in a GitHub repository [9].

This research paper utilized LLM, named ChatGPT, as a tool for rewording and suggesting refined English text, each output of which was reviewed. The instructions given to ChatGPT were in the form of "identify + note writing errors + tips in 'text'". ChatGPT was not employed for conducting any analytical tasks or decision-making processes related to this research.

7 Conclusions

In conclusion, binarized scRNA-seq data substantially reduces memory in the clustering pipeline. However, the reduction is primarily caused by the initial dataset compression format. We identified that the binarized format does not have implications for the later stages of the clustering workflow. From our calculations, for a dataset where the gene count exceeds 10^5, the binarized dataset will occupy the most allocated memory during the execution of the memory-aware implementation of workflow. If a researcher wants to compute PCA or return cell-to-cell distance matrix, the memory consumption would be impractical for datasets with more than $2 \cdot 10^5$ cells. This observation factor attributes enormous peak memory reduction compared to the current solutions.

References

[1] 10x Genomics. 3k PBMCs from a Healthy Donor. Online Resource: https://support.10xgenomics.com/single-cell-gene-expression/datasets/1.1.0/pbmc3k, May 2016.

[2] Vincent D. Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, oct 2008.

[3] Gerard A. Bouland, Ahmed Mahfouz, and Marcel J. T. Reinders. Consequences and opportunities arising due to sparser single-cell rna-seq datasets. *Genome Biology*, 24:86, April 2023.

[4] The Tabula Muris Consortium*. Single-cell transcriptomics of 20 mouse organs creates a tabula muris. *Nature*, 562(7727):367–372, October 2018.

[5] The Tabula Sapiens Consortium*. The tabula sapiens: A multiple-organ, single-cell transcriptomic atlas of humans. *Science*, 376(6594):eabl4896, 2022.

[6] Alexandra Grubman, Gabriel Chew, John F. Ouyang, and et al. A single-cell atlas of entorhinal cortex from individuals with alzheimer's disease reveals cell-type-specific gene expression regulation. *Nature Neuroscience*, 22:2087–2097, December 2019.

[7] Lisa Sikkema, Ciro Ramírez-Suástegui, Daniel C. Strobl, Gillett, and et al. An integrated cell atlas of the lung in health and disease. *Nature Medicine*, 29(6):1563–1577, 2023.

[8] V. A. Traag, L. Waltman, and N. J. van Eck. From Louvain to Leiden: guaranteeing well-connected communities. *Scientific Reports*, 9(1):5233, 2019.

[9] Pavel Verigo. binaryclustering-rp. https://github.com/pavelverigo/binaryclustering-rp, 2023. GitHub repository.

[10] Lijia Yu, Yue Cao, Jean Y. H. Yang, and Pengyi Yang. Benchmarking clustering algorithms on estimating the number of cell types from single-cell rna-sequencing data. *Genome Biology*, 23(1):49, 2022.

A Q estimation data points

Table 3: $K = 5$

Number of Cells	Memory Used in clustering (MiB)
540	0.4
1080	1.0
1620	1.3
2160	1.8
2700	2.2

Table 4: $K = 10$

Number of Cells	Memory Used in clustering (MiB)
540	0.8
1080	1.5
1620	2.3
2160	3.2
2700	3.9

Table 5: $K = 15$

Number of Cells	Memory Used in clustering (MiB)
540	1.0
1080	2.1
1620	3.3
2160	4.4
2700	5.8

B Tabula Muris Peak memory usage

Number of Cells	Number of Genes	Peak Memory (MiB)	Dataset Name
1432	23433	5.8	Tongue
1580	23433	6.4	Thymus
5355	23433	21.5	Marrow
2663	23433	10.7	Mammary
1961	23433	8.0	Pancreas
981	23433	4.0	Liver
865	23433	3.5	Kidney
1718	23433	7.1	Spleen
2102	23433	8.5	Muscle
4149	23433	16.8	Colon
1923	23433	7.8	Lung
7115	23433	28.7	Heart
5799	23433	23.0	Brain Neurons
4762	23433	19.2	Brain Microgolia
5862	23433	23.4	Fat
2464	23433	10.0	Skin
1638	23433	6.6	Bladder
1391	23433	5.6	Trachea