Mona Rams and Tim Conrad*

Dictionary learning for transcriptomics data reveals type-specific gene modules in a multi-class setting

https://doi.org/10.1515/itit-2019-0048
Received December 4, 2019; revised February 13, 2020; accepted February 20, 2020

Abstract: Extracting information from large biological datasets is a challenging task, due to the large data size, high-dimensionality, noise, and errors in the data. Gene expression data contains information about which gene products have been formed by a cell, thus representing which genes have been read to activate a particular biological process. Understanding which of these gene products can be related to which processes can for example give insights about how diseases evolve and might give hints about how to fight them.

The *Next Generation RNA-sequencing* method emerged over a decade ago and is nowadays state-of-the-art in the field of gene expression analyses. However, analyzing these large, complex datasets is still a challenging task. Many of the existing methods do not take into account the underlying structure of the data.

In this paper, we present a new approach for RNA-sequencing data analysis based on dictionary learning. Dictionary learning is a sparsity enforcing method that has widely been used in many fields, such as image processing, pattern classification, signal denoising and more. We show how for RNA-sequencing data, the atoms in the dictionary matrix can be interpreted as modules of genes that either capture patterns specific to different types, or else represent modules that are reused across different scenarios. We evaluate our approach on four large datasets with samples from multiple types. A Gene Ontology term analysis, which is a standard tool indicated to help understanding the functions of genes, shows that the found gene-sets are in agreement with the biological context of the sample types. Further, we find that the sparse representations of samples using the dictionary can be used to identify type-specific differences.

Keywords: Information extraction, Transcriptomics, Omics, Big Data, Dictionary Learning, Compressed Sensing

ACM CCS: Applied computing → Life and medical sciences → Computational biology, Information systems → Information retrieval

1 Introduction

Next generation RNA-sequencing (RNA-seq) uses high throughput approaches to enable genome and transcriptome profiling. Emerged over a decade ago, it is increasingly replacing the use of microarrays in the field of gene expression analyses and thereby becoming the method of choice in this discipline. Reasons are technological advances, such as increased genome coverage, reduced sequencing costs and less background noise.

Application of RNA-seq data include for instance feature selection, identification of alternative splicing and detection of differentially expressed pathways. A challenge in the analysis of RNA-seq data is its high dimensionality. This comes along with the curse of dimensionality, as usually a good performance on training data can be achieved well. Reproducibility, however, is often a problem as small changes in the data can lead to different results. Therefore the biological relevance of such results needs to be questioned. To solve this issue, methods enforcing sparsity were introduced. In sparse methods coefficients of variables related to noise or uninformative variables are set to zero. Well known representatives are Lasso [21], elastic net [25], and sparse random forest [9].

Even though gene expression data is highly structured [1, 2, 18, 20], many applications act independent of this underlying structure. In this paper, we carry out an analysis of RNA-seq data with a variant of dictionary learning (DiL).

DiL is a constrained matrix factorisation approach that compresses a dataset, preserving most variation in the lower dimensional space. It is especially interesting in the analysis of big data as it allows to analyse the data in

***Corresponding author: Tim Conrad,** Freie Universität Berlin, Institute for Mathematics, Arnimallee 6, 14195 Berlin, Germany; and Zuse Institute Berlin, Takustraße 7, 14195 Berlin, Germany; and Berlin Institute for the Foundations of Learning and Data, Berlin, Germany, e-mail: conrad@math.fu-berlin.de
Mona Rams, Freie Universität Berlin, Institute for Mathematics, Arnimallee 6, 14195 Berlin, Germany, e-mail: mrams@math.fu-berlin.de

a compressed space. The runtime of data analysis once it is compressed is significantly smaller than an analysis of the data in the original space. At the same time, unlike other data compression methods, DiL allows to interpret the meaning of the new dimension. Further, the variant we present in this paper requires significantly less runtime than the original DiL setting (precise runtime benefits depends on the size of dataset and DiL parameters).

DiL has been widely used in many fields, such as image processing, pattern classification, and signal denoising [4, 23, 6]. The basic idea in DiL is, that a dataset can be well constructed from a linear combination of few modules (atoms) of some "basis" (dictionary) given that the data possesses some structure that can be represented sparsely. To our knowledge, its applications to Omics data, however, is rare.

When applying DiL to RNA-seq data from multiple sample types, the atoms can be described as the basic elements of the data (e. g. specific genes sets that are mutually activated/deactivated), that capture patterns of the analysed data. Each sample can then be reconstructed (with small errors) by a linear combination of these atoms. In interpretation this means, that the dictionary atoms entail the main building blocks, or modules of the analysed data and the sparse coding can be interpreted as a data compression.

Dictionary learning (DiL) and compressed sensing (CS) are data compression methods for which the problem of sparsity per sample is solved, meaning that the number of selected modules per sample is always $\leq s$ (where s is a defined integer, describing the level of sparsity). They have found wide application in image compression. However, for the application to biological data we found only few examples. In 2002 Prat et al. [18] applied CS to microarray data to recover a support set of genes for each gene in a genome. In 2014 Emad et al. [7] applied CS to Omics data to discover directed interactions in gene networks. In 2016 Khormuji and Khormuji [11] used a novel sparse coding algorithm for classification of tumors based on gene expression data. In 2017 Clearly et al. [3] presented a method based on compressed sensing, to generate a high dimensional transcriptomic profile from sequencing a small, random selection of genes. In their review on low rank representation and its application in bioinformatics You et al. [24] concluded that researchers need to make full use of the structure of problems. In her thesis from 2019 [12], Koletou applied nested dictionary learning on different types of Omics data from prostate cancer patients. She developed nested dictionary learning, describing an iterative application of dictionary learning: In the first step, the dictionary is learned from the data and further dictionaries are learned from the dictionary in the pre-descending step. Reason for this nested approach is the aim of a drastic dimensional decrease as the first dictionary from the data still has 10 % of the initial dimension (in the number of genes).

In this study we evaluate whether a variant of DiL can be used to separate gene expression data of different types and whether the atoms of the dictionary are biologically reasonable. Further, we analyse whether dictionary learning can be used to classify RNA-seq data with respect to type, e. g. tissue type, in the lower dimension. In particular we evaluate the use of dictionary learning for analysis of RNA-seq data from multiple different types and the influence of parameters in this analysis. Thereto, we focus on the following topics:

1. How do the parameters for dictionary learning influence results? (e. g. number of atoms and sparsity)
2. How well can the found gene-sets separate the types?
3. Does the biological evaluation of the dictionary matrix (using GO-terms) agree with the types?

We start by analysing simulated datasets which are designed to contain (expression) patterns similar to biological datasets. Based on results from the simulation study we can decrease the parameter search space for real world datasets.

When analysing real world datasets we find that sample clustering of the sparse codings results in high overlap of the true type-groups. Further, a GO-term analysis of the identified modules is in agreement with the biological context of the sample types.

The analysed dataset in focus is the GTEx RNA-seq with more than 50k reads and 26 tissues with 96 samples per tissue. Further, results for two tissue RNA-seq datasets from GEO (GSE120795) and Expression Atlas (E-Mtab-2836), one single cell dataset from GEO (GSE112004), and one simulated dataset are evaluated.

2 Identification of gene modules from RNA-seq data using a variant of dictionary learning

2.1 Dictionary learning

The key idea in dictionary learning (DiL) is that a variety of data can be well constructed from a linear combination of few modules (atoms) of some "basis" (dictionary) given that the data possesses a sparse structure.

Given an input dataset $X = [x_1, ..., x_p], x_i \in \mathbb{R}^d$, and a positive integer $m \in \mathbb{Z}_{>0}$, the number of columns the dictionary will have, and $s \in \mathbb{Z}_{>0}$, the sparsity level per sample, in the dictionary learning problem we wish to find a matrix $D \in \mathbb{R}^{d \times m}$ and s-sparse vectors $r_1, ..., r_p \in \mathbb{R}^m$, so called sparse codings, such that $Dr_i \approx x_i \; \forall i$. D is called the dictionary and its columns are referred to as atoms. The vectors r_i, with at most s non-zero entries, contain the coefficients for the columns of D to linearly represent x_i. To make the choices of D and r_i unique, the constraint $\|d_i\|_2 = 1$ is added. The sparsity level, s, often satisfies $s \ll m$. This can be formulated as:

$$\underset{\mathbf{D},R}{\operatorname{argmin}} \sum_{i=1}^{P} \|x_i - \mathbf{D}r_i\|_2^2 + \lambda \|r_i\|_0 \quad s.t. \; \|d_i\|_2 = 1, \tag{1}$$

where $\mathbf{D} \in \mathbb{R}^{d \times m} : D = [d_1, ..., d_m]$ and $R = [r_1, ..., r_p]$, $r_i \in \mathbb{R}^m$. In the standard setting the dictionary is an overcomplete system, meaning $m > d$.

The optimization problem in equation (1) is non-convex, which means that only sub-optimal solutions can be found in polynomial time. To improve the computational feasibility and efficiency, (1) is usually reformulated using the L1-norm:

$$\underset{\mathbf{D}, r_i \in \mathbb{R}^n}{\operatorname{argmin}} \sum_{i=1}^{P} \|x_i - \mathbf{D}r_i\|_2^2 + \lambda \|r_i\|_1 \quad s.t. \; \|d_i\|_2 = 1. \tag{2}$$

2.2 DiL for RNA-seq data

When applying DiL to RNA-seq data, according to the standard DiL setting the orientation of the data X would be $X \in \mathbb{R}^{p \times g}$, where g is the number of genes and p is the number of samples. As typically $g >> p$, the dictionary is overcomplete only for this data orientation (compare Figure 1).

If $X \in \mathbb{R}^{p \times g}$, then $D \in \mathbb{R}^{p \times m}$. This, however, would mean that the dictionary does not entail information about the genes. In fact, this information would be in the sparse vectors, while at the same time each samples would be represented by many (up to m) sparse gene vectors which would only slightly fulfil the aim of dimension reduction.

We suggest to apply DiL to X, for $X \in \mathbb{R}^{g \times p}$. If $X \in \mathbb{R}^{g \times p}$, then $D \in \mathbb{R}^{g \times m}$. Only in this setting the dictionary entails information about the genes and each sample is represented by few (up to s) atoms. This does, however, mean that the dictionary is not overcomplete as $D = X \in \mathbb{R}^{g \times p}$ would be an optimal solution to (2) already. A further increase of m would not lead to better solutions. Therefore, we neglect the overcompleteness while still using the formulation in (2). This means that we are already enforcing sparsity by the selection of m, as this will describe the number of dimensions we are compressing the data to and $m < p$ in our variant of DiL.

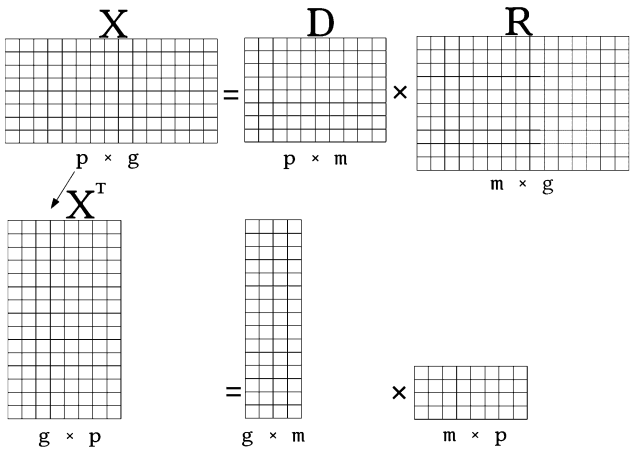


Figure 1: Visualisation of the two optional orientations of the data X for DiL and the resulting orientations of the matrices D, the dictionary, and R, the sparse codings. At the top $X \in \mathbb{R}^{p \times g}$ and D is overcomplete. At the bottom $X \in \mathbb{R}^{p \times g}$ (p, number of samples; g, number of genes) and D is not overcomplete. The overcomplete dictionary requires a long computational time, whereas runtime for our method is a lot shorter.

2.3 Implementation and complexity analysis

We use *DictionaryLearnig* [15] for small datasets (up to a size of 100 MB) and *MiniBatchDictionaryLearnig* [16] for large datasets. The sparse codings (r_i) are computed using orthogonal matching pursuit (OMP) [17] for which parallel and GPU versions are available [8]. A detailed complexity analysis of OMP can be found in [19]. To summarize, the overall algorithmic complexity is:

$$\mathcal{O}\left(p * (2gm + s^2m + 7sm + s^3 + 4sg) + 5gm^2\right)$$

Recall that p is the number of samples (varying between different datasets), and g is the number of genes (varying between different species).

However, for the case of using an overcomplete dictionary the runtime would be:

$$\mathcal{O}\left(g * (2pm + s^2m + 7sm + s^3 + 4sp) + 5pm^2\right)$$

As in RNA-seq experiments typically $g >> p$, this would require a highly increased runtime.

Table 1: Overview of the four real world datasets analysed in this study. Dataset D4 is a single cell dataset. Datasets D1 and D4 contain larger number of samples than D2 and D3.

ID	Database	Types	Samples per type	Reads/Genes	Type class
D1	GTEX	26	92	55,091	Tissues
D2	GEO	8	8	53,679	Tissues
D3	Expression Atlas	8	8	49,914	Tissues
D4	GEO	9	32	11,781	B-cell type & experiment time

3 Results

Recall, that the dictionary atoms can be described as the basic elements of the data. When applied to RNA-seq data we interpret the atoms as specific genes sets that are mutually activated/deactivated in a large number of samples. The sparse codings entail information of how to reconstruct each sample (with small errors) by a linear combination of a subset of the atoms.

In this section we evaluate applications of our DiL based approach to different datasets to assess its relevance for RNA-seq data analysis. In general, if our approach is suitable for RNA-seq data, for one thing, main differences in the data should be preserved in the compressed representation, hence the sparse codings. To analyse if this is the case, we apply our approach to datasets with samples from different types. This allows to evaluate whether the respective types are represented differently in the sparse codings. Types can be anything referring to different classes of samples such as phenotype, stimulation in an experiment or donor. At the same time, the compression should be done in a biologically reasonable way, which we assess by an evaluation of the dictionary atoms from the real world datasets.

We apply our approach to simulated and real world datasets. The simulated datasets are constructed to have five different sample types. Five different simulation strategies are performed, which vary in construction of groups and noise. The four real world datasets each contain data from multiple types and with multiple samples per type. One of the datasets stems from a single cell experiment, the others are bulk experiments. Table 1 shows an overview of the four real world datasets. Details on simulated and real world data are given in section 5.

For each dataset we compute dictionaries of different sizes and reconstructions for different sparsity levels. We evaluate how the parameters for dictionary learning (e. g. number of atoms and sparsity) influence the results, how well the found gene-sets separate the types, and whether the biological evaluation of the genes corresponding to the atoms agrees with the types.

Results from the dictionary learning are (1) the dictionary matrix and (2) a sparse coding vector for each sample. To evaluate the sparse coding, we use clustering and analyse whether the clusters have a high agreement with the sample types. For the RNA-seq data, the dictionary is evaluated by a Gene Ontology (GO) term analysis [5] (of the genes corresponding to the high values in the dictionary matrix). Gene Ontology assigns genes to a) biochemical activities, b) biological goals and c) cellular structures thereby helping to understand functions of the genes and their products. We use the GO term analysis to evaluate whether these genes can be associated with the sample type.

3.1 Evaluation of results

We apply DiL to datasets from multiple types and with multiple samples per type. The respective sparse codings should reveal type specific characteristics. To evaluate whether this is the case, the sparse codings are clustered and the resulting clusters are compared with the true type-groups. The clustering is performed with k-means [14] (implementation from *Python's sklearn*) with $k = n$ (where n is the number of types of the dataset). To evaluate the clustering we compute the adjusted rand index (ARI) and the adjusted mutual information (AMI) of the k-means clusters and the true type-groups.

The ARI [10] is based on the Rand index, which is defined as the number of pairs of elements that are either in the same group or in different groups in two partitions divided by the total number of pairs of elements. A problem with the Rand index, which is corrected in the ARI, is that the expected value of the Rand index between two random partitions is not constant. The maximum ARI is 1 and its expected value is 0 for random clusters.

The mutual information measures how much knowing one clustering reduces uncertainty about the other clustering. Similarly as for the Rand Index and the ARI, the AMI [22] is an adjustment of the mutual information to account for chance.

Additionally, the reconstruction error is measured as the Euclidean distance of the data matrix and the matrix resulting from $D * R$ (compare (1), (2)).

3.2 How do the parameters for dictionary learning influence results?

The dictionary learning requires two parameters: The number of atoms of the dictionary, m, and the sparsity, s. In this section multiple parameter combinations are tested to evaluate how they influence the results.

3.2.1 Simulated data

For the simulated datasets the algorithm is run for dictionaries with $m = [1, 2, 3, 4, 5, 10, 20]$. Recall, that the simulated data is constructed to have $n = 5$ types and five different simulation strategies are performed, which vary in construction of groups and noise. As our method is new a new method these parameters are selected based on a first grid search over a wide parameter range. In the selected range ($1 \leq m \leq 20$) we see the most relevant changes in ARI, AMI and reconstruction error. Results are visualised in Figure 2. Construction of the simulated data is explained in the Data section.

For all simulated datasets the clustering is in entire agreement with the types for a dictionary with 5 atoms no matter how sparsely the data is represented. Further, all clustering scores are 1 for settings without noise (A, B, C) whenever the dictionary has 4 or more atoms. For simulation setting C, the median clustering scores are 1 for a all dictionaries (1, 2, 5, and 10 atoms). Setting E is the only one for which the clustering scores are not 1 for a dictionary with 4 atoms. In the settings with noise (D, E), the clustering scores decrease when the number of atoms exceeds 5, whereas in the settings without noise clustering scores remain 1 for any number of atoms ≥ 4. An explanation could be that the additional dictionary atoms in settings D and E are trying to explain the noise which is independent from the data types and thus leads to a worse clustering result.

The median reconstruction error is minimal among all evaluated dictionaries for $m \geq 5$. In all settings without noise the minimal median reconstruction error is zero, whereas in the ones with noise there appears to be information that cannot be explained by the dictionary for $m \leq 20$ and the reconstruction error remains > 0.

Note that the ordering of "high genes", which are sorted for each group to appear in blocks in our visualisations (Figure 8), does not influence results.

Summarising the simulation results, it shows that the selection of the number of dictionary atoms has a big influence on the results, especially for noisy data. In this small example it seems to be a good choice to set the number of dictionary atoms equal to the number of groups in the dataset.

3.2.2 Real world data

Experiments for the real world datasets, are run for dictionaries with $m \in [1, 2, ..., 2n]$, where n is the number of types in each dataset and $s \in [1, 2, ..., m]$ for each dictionary. The choice of values of m is based on the results from the simulated data (where $m = n$ showed to be a good choice and we would like to analyse the surrounding parameter space as well).

As behaviour of ARI and AMI are similar in our experiments (see Figure 3 for an example for dataset D1), we focus on the ARI in the detailed parameter evaluation and conclusions can be transferred to AMI.

For all RNA-seq datasets, we find that in general the ARI increases for increasing number of atoms up to a certain value where m is close to n (see Figure 4). Considering the sparsity, for many dictionaries the ARI first increases when s is increased and stays close to some ARI for larger values of s. ARIs for D2 are worse than for the other three datasets. Dataset D2, however, showed to have many samples with a strikingly high number of zero values. Discarding samples with zero-entries per sample of more than 75 %, yields significantly better results with a maximum ARI of 0.80. For dataset D1, we perform evaluations for both $k = 26$ ($n = 26$ with tissue labels as general types) and $k = 48$ ($n = 48$ with tissue labels as detailed types) as true labels.

The maximum ARI is: 0.77/0.73 for D1 (for general types/detailed types), 0.56 for D2, 1.00 for D3, 0.7 for D4. The maximum AMI is: 0.88/0.85 (for general types/detailed types) for D1, 0.68 for D2, 1.00 for D3, 0.8 for D4.

3.3 How well can the found gene-sets separate the types?

To find the best parameter combination for the biological evaluation, we suggest to perform a grid search with $m \in [0.5n, 1.5n]$ and for each dictionary $s = [1, 2, ..., m]$ and select the parameters corresponding to the best ARI. In all experiments we conducted the best ARI of a wide range of parameters lay in this interval.

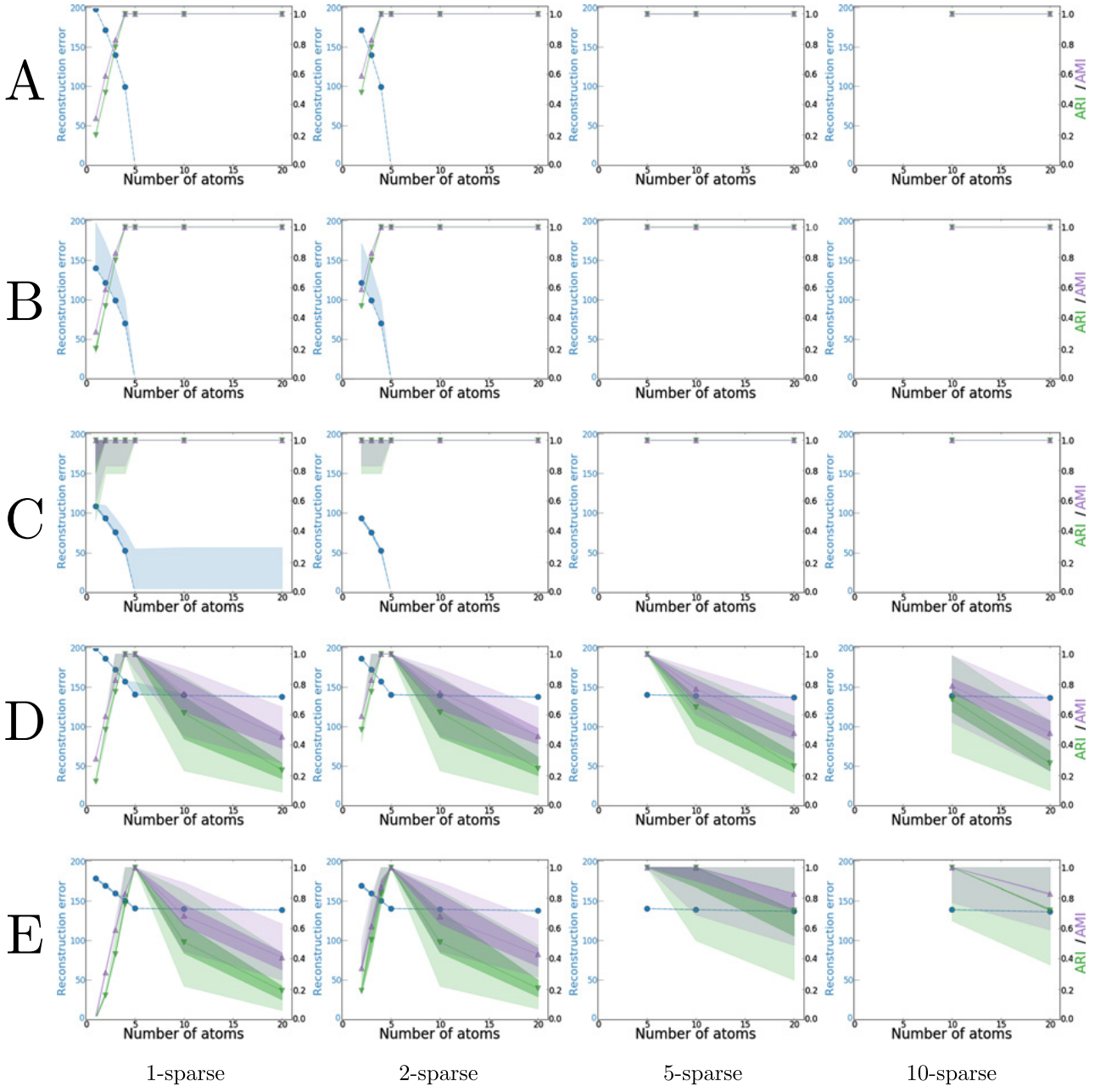


Figure 2: The plots show evaluations for dictionary learning of the simulated data with several parameters. The data is simulated to consist of five groups (precise simulation setting is given by capital letters, compare section 5.1). Shown are the reconstruction error (blue), and ARI (green), respectively AMI (purple) for the k-means clusters of the sparse codings. The x-axis displays the number of atoms of the dictionaries. The lines and points display the median value in 100 simulations, the shadows display the 25th and 75th percentile. In each column results for 1, 2, 5, respectively 10-sparse coding are visualised (see bottom for respective sparsity). In all settings without noise (A, B, C) the minimal median reconstruction error is zero. Further, in all settings the clustering scores are 1 for a dictionary with 5 atoms no matter how sparsely the data is represented. In the settings with noise (D, E), the clustering scores decrease when the number of atoms exceeds 5, whereas in the settings without noise the clustering scores stay 1 for any number of atoms ≥ 4.

We evaluate to what extent the resulting clusters overlap with the type groups and which types are grouped, respectively split.

Figure 5 shows a visualisation of the sparse codings of all samples in each dataset. This visual representation already reveals that the different types have different sparse coding values. This is especially the case for dataset D2, where each type is represented by one atom. For the other datasets these differences are not as clear, but still visible. This is in agreement with the ARI, which

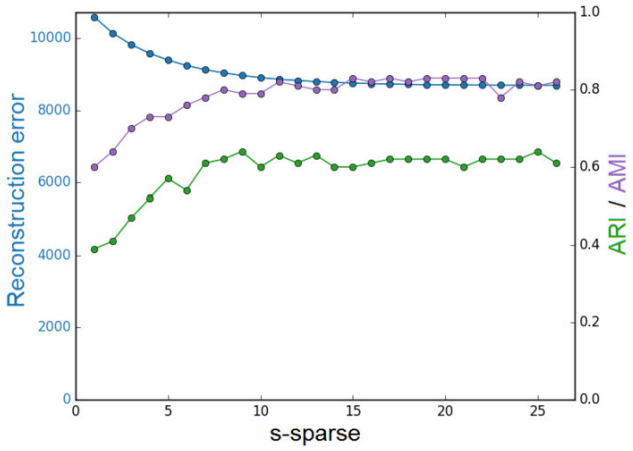


Figure 3: Visualisation of ARI, AMI, and reconstruction error for dataset D1 for the dictionary with 26 atoms (as many atoms as types) and sparsities $s \in [1, 2, ..., 26]$. The AMI is always higher than ARI (about 0.2, but the difference varies slightly). For smaller values of s until around $s = 10$ for increasing s all measurements are improving (reconstruction error decreases, ARI and AMI increase). For higher values of s, ARI and AMI stay close to 0.6 and the reconstruction error decreases in smaller steps.

is 1 for dataset D2 and between 0.56 and 0.77 for the other datasets.

3.3.1 D1: GTEx

For dataset D1 the maximal ARI for the general types of 0.77 is reached for $m = 42, s = 39$ (see Figure 4). Further clustering scores for $m = 42, s = 39$ are: $ARI_{detailed} = 0.66, AMI_{general} = 0.87, AMI_{detailed} = 0.81$. In the resulting clusters (compare Figure 6),

- 13 of 26 clusters have samples of one tissue type only and
- in 18/24/25 clusters more than 90 % of samples are from one/two/three tissue type(s).

When the detailed labels are chosen as the tissue types with subtypes,

- 10 of 26 clusters have samples of one tissue type only and
- in 13/21/24 clusters more than 90 % of samples are from one/two/three tissue type(s).

Considering the detailed tissue types, some are separated as for example tissues belonging to either cerebrum or cerebellum (in clusters 15, respectively 21). This, however, is a reasonable separation and we are happy to find that our algorithm detects differences in these groups. Note, that this decreases the ARI when the general type labels are used – as all brain tissues have the same type label – even though this is actually a correct finding and it shows that the methods is capable of detecting subgroups.

For the detailed types, the maximum ARI of 0.73 is reached for 39/32, 42/26, 47/34 (m/s).

3.3.2 D2: GEO *GSE120795*

For dataset D2 the maximal ARI of 0.56 is reached for $m = 7, s = 2$ (AMI = 0.67). Five clusters of the sparse codings are comprised of one tissue only, whereas clusters 2,4, and 7 are a mix of multiple tissues (compare Figure 10). We do not have detailed information for the brain tissue explaining which samples are grey/white matter, but it is conceivable that the separation of the brain samples into two clusters could be explained by that.

Pie charts with an additional preprocessing step regarding the number of zero-entries can be found in the Appendix.

3.3.3 D3: Expression Atlas *E-MTAB-2836*

For dataset D3 the maximal ARI is 1.00 (reached for $m = 7$, $s = 1$), which means that the clusters of the sparse codings are in entire agreement with the types (compare Figure 11).

3.3.4 D4: GEO *GSE112004*

For dataset D4 the maximal ARI of 0.7 is reached for $m = 8, s \in [4, 5, 6, 7, 8]; m = 9, s = 8; m = 11, s = 3$ (AMI = 0.8).

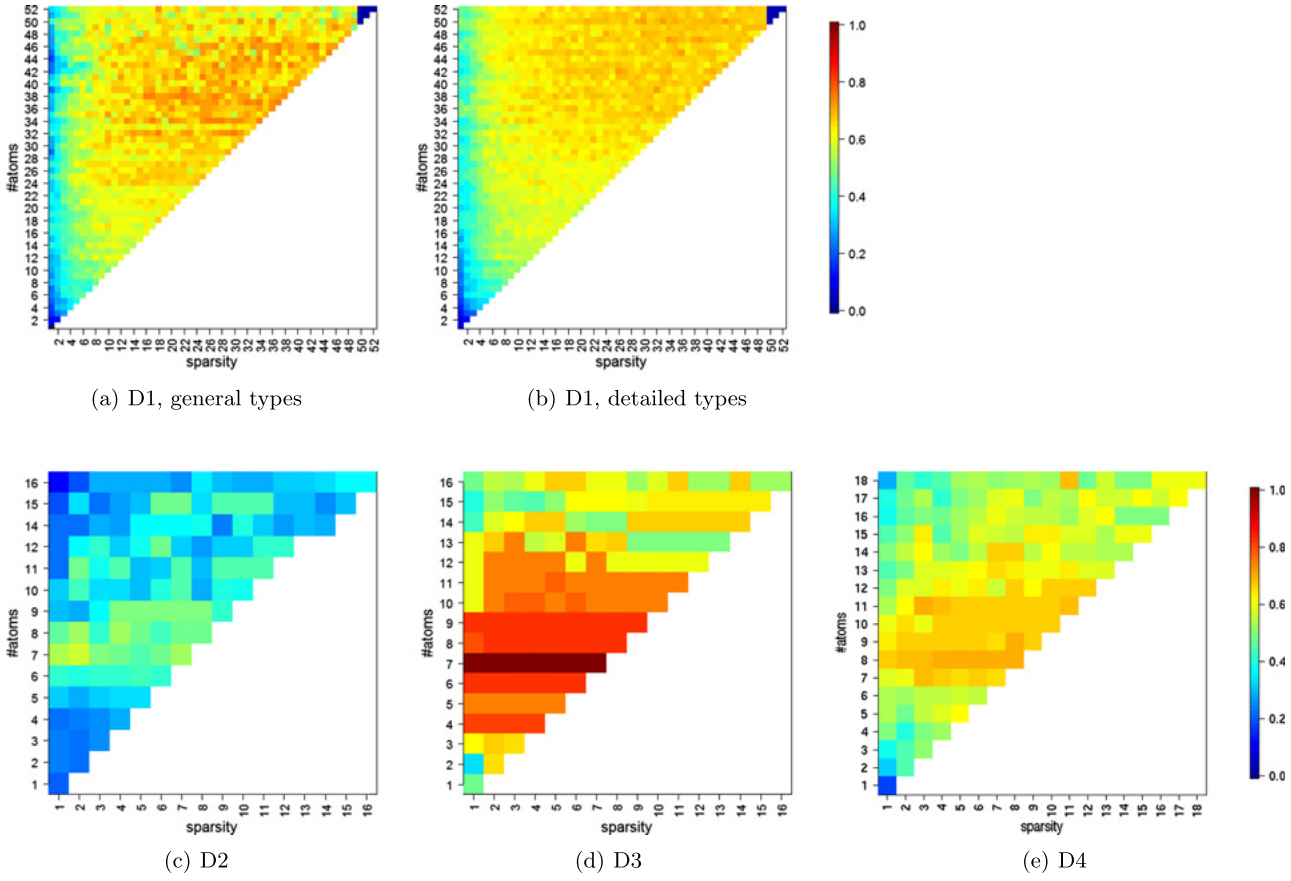
The three treatment types are separated, wrong assignments appear only for times within one treatment class (compare Figure 12).

3.3.5 Summary of parameter evaluation for the real world datasets

It shows that there is a wide range of parameters for which the variation of ARI is small. E. g. setting $m = s = n$ results in an ARI that is larger than $0.8 * ARI_{best}$ for all datasets. A small grid search around $m = s = n$, e. g. $m \in$ all integer values $\in [0.5n, 1.5n]$, $s \in$ all integer values $\in [0.5m, 1.5m]$, improves results further.

Restriction to positive dictionary entries

In experiments with an additional constraint allowing for positive dictionary entries only, similar maximum ARIs



(a) D1, general types (b) D1, detailed types

(c) D2 (d) D3 (e) D4

Figure 4: Visualisation of the ARIs for the four real world datasets for all parameters evaluated (number of atoms, $m \in [1, 2, ..., 2n]$, where n is the number of types in each dataset and $s \in [1, 2, ...m]$ for each dictionary). Subject to certain exceptions, 1.) the ARI increases for increasing number of atoms until it reaches a maximum and then decreases for further increase of m (compare values from bottom to top), 2.) for increase of s, the ARI first increases and then stays relatively constant (compare values from left to right). ARIs for D2 are significantly worse than the for the other three datasets. The parameters sets of the highest ARIs for D1 vary depending on the types the clusters are compared with (detailed/general). As expected the optimal range for the detailed types (larger n) lies in the range of higher number of atoms, respectively sparsities.

(±0.01) are reached, however, the number of atoms is often higher: 47 atoms for D1 and 8 atoms for D2, D3, and D4.

3.4 Does the biological evaluation of the dictionary matrix agree with the types?

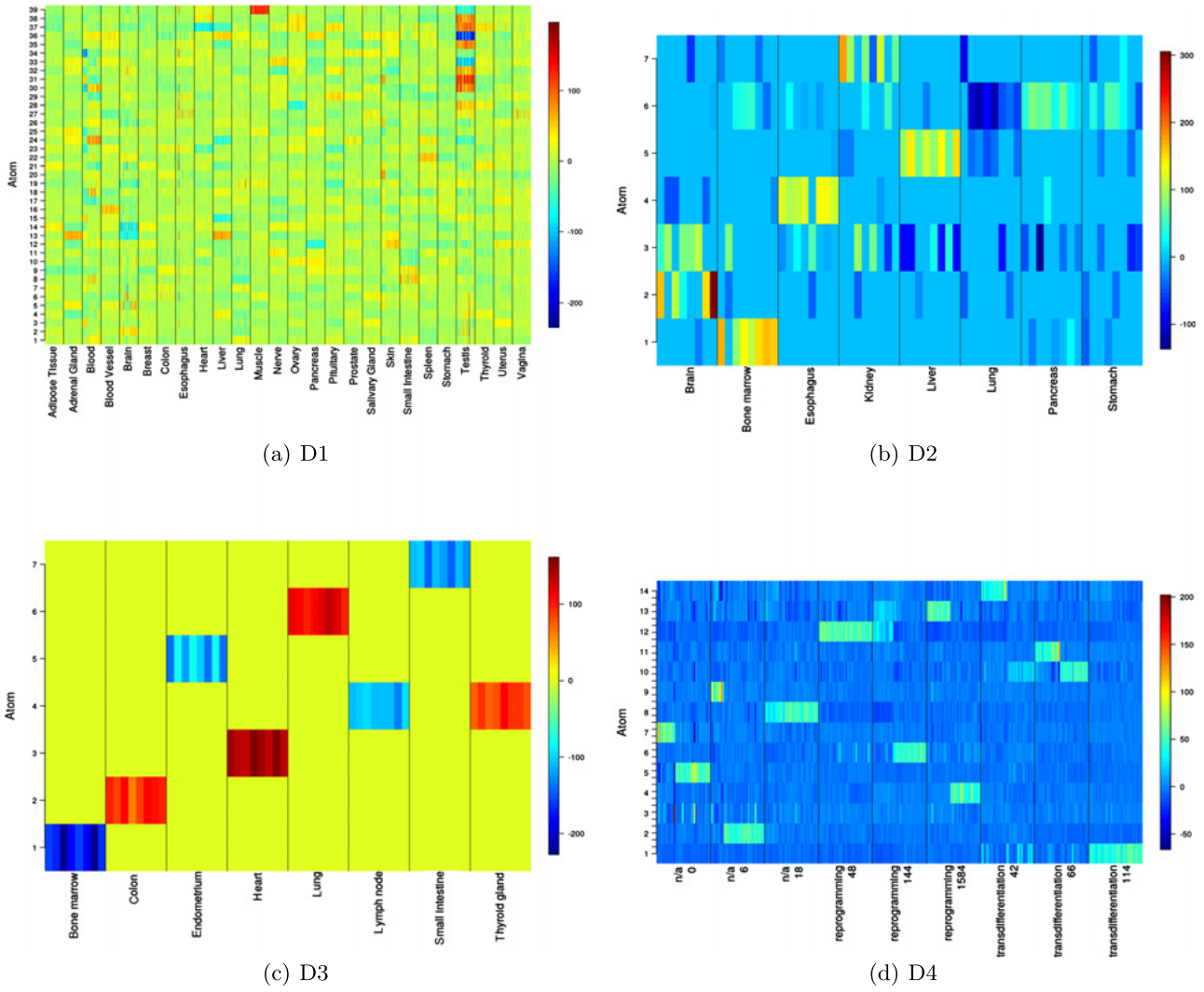
Let v_{pos} represent the sorted absolute values of D and v_{neg} the sorted absolute negative values of D. To select a set of genes for each group, only the largest absolute values of v_{pos} and v_{neg} are evaluated (compare [13]): All values in the dictionary matrix that are in the interval $]1st - percentile, 99th - percentile[$ are set to zero. The resulting dictionary will be referred to as *2%Dictionary*. Note that this most likely leads to atoms that have different amount of non-zero values.

To assign a subset of atoms to each type, we compute a value measuring the atom-preference among all samples for each type:

- From all samples' sparse codings, we compute the mean value for each atom.
- The resulting values are divided by the absolute sum of all atom selection values.

For the biological analysis, for each type, we evaluate genes corresponding to those atoms with top three absolute atom-preferences. A GO-term analysis of these genes is performed. Only GO-terms with a p-value $\leq 10^{-5}$ are considered. Each atom is analysed separately and the genes with positive, respectively negative values are analysed separately as well.

In the *2%Dictionary*, the number of non-zero entries per atom varies between 127 and 1841 (out of 55,091) for



(a) D1 (b) D2

(c) D3 (d) D4

Figure 5: Shown are the values of the sparse codings (values colour coded, see legend). The x-axis represents the samples ordered by type and the atoms are shown on the y-axis. The black lines indicate the border between each two types. This means that each coloured horizontal stripe reflects the values of the sparse coding for one atom for one sample. This way, multiple dimensions can be visualised – and therefore compared among different types. What has been confirmed by the ARIs – that for each type (most of the) sample types have a similar and unique representation – is confirmed by this visualisation.

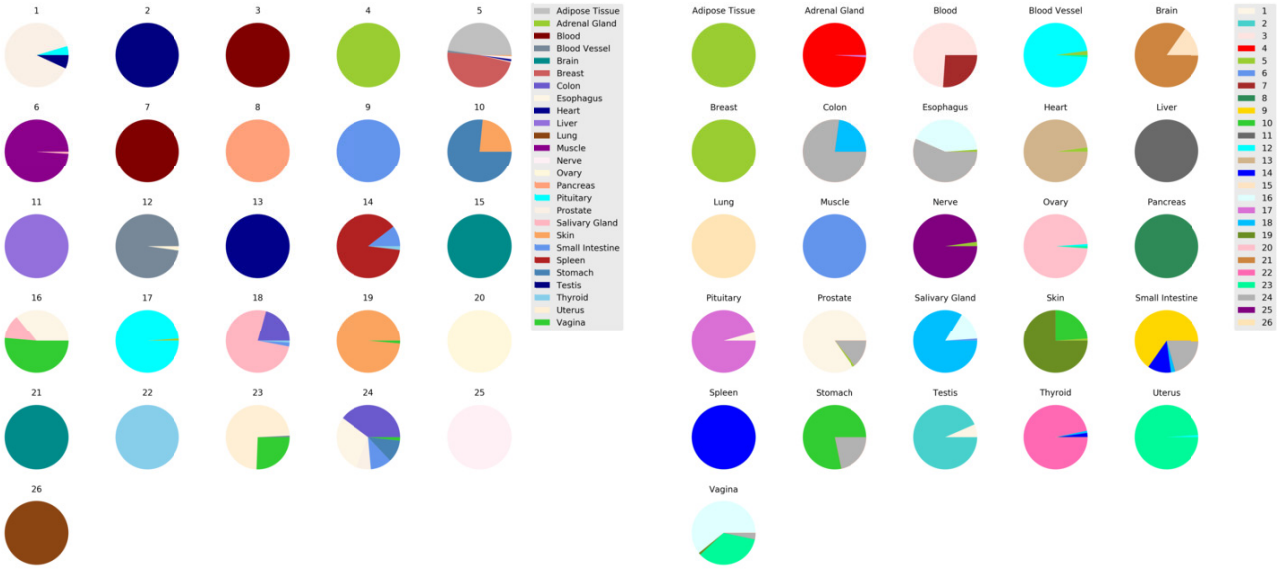
dataset D1; Between 560 and 1601 (out of 53,679) for D2; Between 131 and 1331 (out of 49,914) for D3; Between 152 and 329 (out of 11,781) for D4. For many atoms the GO-terms entail terms that can be associated with the corresponding tissue (see Figure 7).

4 Use case runtime evaluation

The experiments were carried out on a machine with Intel(R) Core(TM) i5-8500 processors and 16 GB RAM. The runtime scales with a) dataset size and b) m, the number of atoms. Depending on the parameter m the runtime for the smaller datasets varies between 0.4 and 1.4 seconds for $m = 1$, respectively $m = 20$; For the larger dataset D1 the runtime varies between 3.15 and 46.45 seconds for the same parameters (see Table 2).

5 Data

Evaluations of our algorithm are performed on different datasets: (1) Simulated data and (2) four real world dataset from varying sources. Details on the respective datasets are given in the following paragraphs.



(a) Pie chart by cluster

(b) Pie chart by tissue

Figure 6: Shown are the clustering results of the sparse codings for dataset D1 for the maximal ARI measured (dictionary with 42 atoms and a 39-sparse coding). (a) shows the tissues in the clusters. Most clusters consist of one tissue type (in high extent). Interestingly, some general tissue types are separated into few cluster, such as Brain in cluster 15 and 21. These cluster separate the cerebellum from the cerebrum. (b) shows the clusters for each tissue for the same sparse coding: In (a) we can see for example, that cluster 6 consists of one main type and a few Salivary Gland samples; In (b) we can see that all Muscle tissue samples appear in only one cluster (cluster 6).

Table 2: Wall-clock time of our approach for the four datasets for two selected parameters values for the number of atoms, $m \in [1, 2]$. For dataset D1, function *MiniBatchDictionaryLearnig* was applied due to the large datasize (marked with a '*'). For the other datasets the function *DictionaryLearnig* was used. For the smaller datasets < 100 MB the runtime is < 2 second for $m = 20$. For D1, the runtime for $m = 20$ is still < 1 minute.

Dataset	Wall-clock time [s] for $m = 1$	Wall-clock time [s] for $m = 20$
D1	3.2*	46.5*
D2	0.4	1.4
D3	0.4	1.2
D4	0.4	0.7

5.1 Simulated data

The construction of simulated data is inspired by biological data. We simulate a matrix of 50 samples and 100 reads. The data is simulated to have 5 groups of 10 samples each. In the simulation, for each group we differentiate between a subset of *background genes* and up to 20 *high genes*. The subset-size and the meaning of "high"/"low" is differently defined in each simulation setting. The values for *background genes* and *high genes* are simulated separately. In two of the five settings a subset of the background values are high in all samples to simulate *housekeeping genes*. For a detailed explanation of the simulation settings see Table 3 and Figure 8.

For each setting but the first (because the first does not require a random drawing) we simulate 100 datasets.

5.2 Real world data

We analyse four RNA-seq datasets (see Table 4), with samples from multiple types (see Table 1). The number of samples per type varies within each dataset. To obtain datasets with the same number of samples per type, for each type, samples are randomly selected such that the number of samples per tissue type is the same for all tissues and maximal given the data.

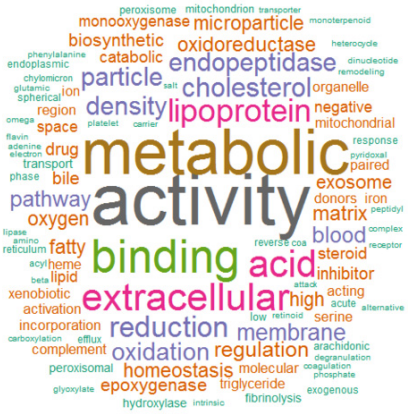
5.2.1 D1: GTEX

The GTEX data contains TPM-normalised counts for 11,688 samples and 56,202 reads.

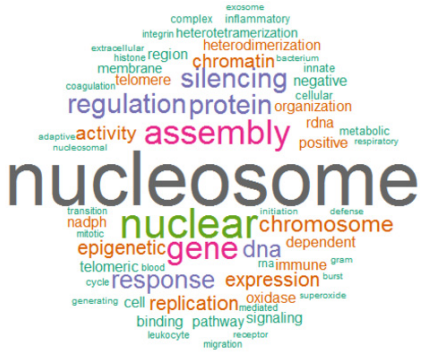
Each sample is labelled with a detailed tissue description (e. g. *Adipose – subcutaneous*, *Adipose – Visceral*, *Brain – Amygdala*,...), as well as a more general one (e. g. *Adipose – Subcutaneous* and *Adipose – Visceral* as type

(a) D1: Muscle

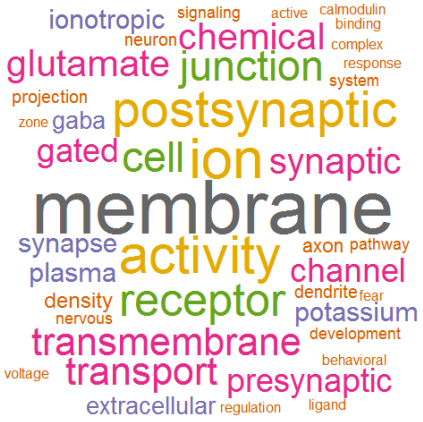
(b) D1: Heart



(c) D1: Liver

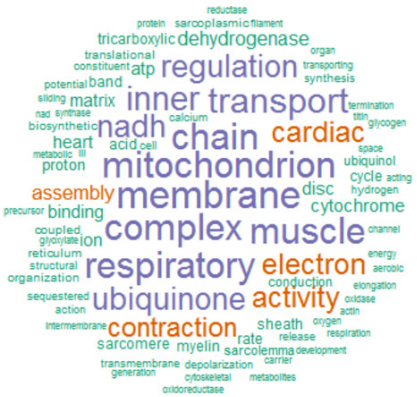


(d) D2: Bone marrow

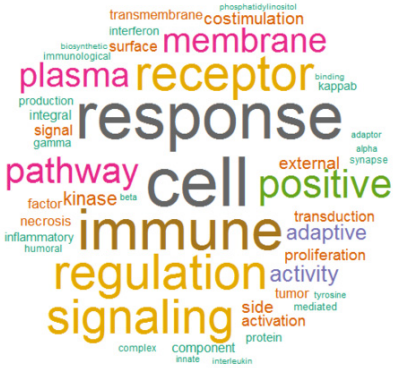


(e) D2: Brain

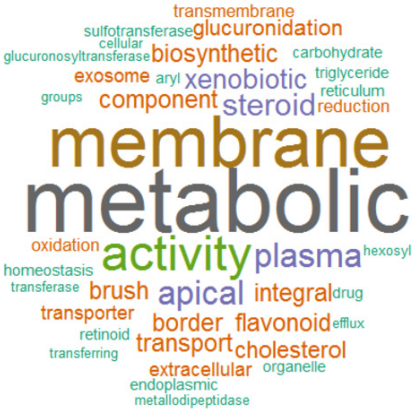
(f) D2: Liver



(g) D3: Heart



(h) D3: Lymph node



(i) D3: Small intestine

Figure 7: Wordclouds of the GO-terms with p-value $\leq 10{-}5$ for a selection of 3 atoms for each tissue dataset. Dataset and tissue name with highest atom-preference for the respective atom is given in the subtitles. Only tissues for which the respective atom is added are shown (referring to positive dictionary entries and sparse coding values, or both negative). For many atoms the GO-terms entail terms that can be associated with the corresponding tissue.

Table 3: Overview of the simulation settings in the 5 simulated dataset. Three gene classes are simulated: *Background entries*, *Housekeepin genes*, and *Group entries*. Among the different simulation settings, for each class the percentage of these gene classes among all genes as well as the particular values are adjusted.

Setting	Background entries		*Housekeeping genes*		Group entries	
	Amount	Value	Amount	Value	Amount	Value
A	All	0	None	–	All	1
B	All	0	None	–	50 %	1
C	60 %	0	40 %	1	50 %	1
D	All	$\mathcal{N}(0, 0.2)$	None	–	50 %	$\mathcal{N}(0.7, 0.2)$
E	60 %	$\mathcal{N}(0, 0.2)$	40 %	$\mathcal{N}(0.7, 0.2)$	50 %	$\mathcal{N}(0.7, 0.2)$

Table 4: Meta data for the four real world datasets analysed in this study. Datasets D1 and D4 contain larger number of samples than D2 and D3. D1, D2, and D3 contain larger number of genes than D4.

ID	Database	Database ID	Total samples	Reads/Genes	Size [MB] raw (preprocessed)
D1	GTEX	GTEX	2,392	55,091	2700 (572)
D2	GEO	GSE120795	64	53,679	24 (34)
D3	Expression Atlas	E-MTAB-2836	64	49,914	37 (5)
D4	GEO	GSE112004	288	11,781	28 (260)

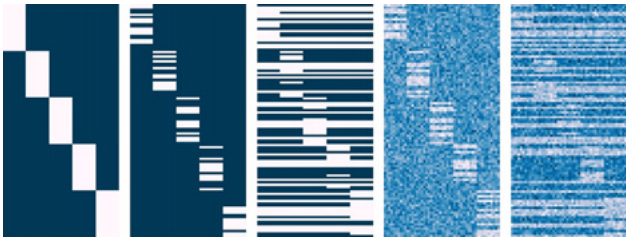


Figure 8: The simulated data is constructed to consist of 5 types. For each type a subset of all group specific entries is high. (a) In the simplest setting, which the others are based on, all group specific entries are 1 and the other values are 0. (b–e) In the other simulations group specific entities are drawn randomly; (d–e) Noise is added; (c), (e) A subset of genes is high in all groups.

Adipose). We evaluate the dictionary learning for the general grouping and use the detailed description in the result analysis only. Data is present for 53 detailed tissue types with 5–564 samples per type, respectively 31 grouped tissue types with 7–1671 samples per type.

The minimum number of samples per tissue is set to 50.

Outlier removal and selection of the same number of samples per type results in a $2{,}392 \times 55{,}091$ matrix ($2{,}392 = 26types * 92samples\_each$).

5.2.2 D2: GEO *GSE120795*

The GEO dataset *GSE120795* contains FPKM-normalised counts for 166 samples and 58,233 reads. Samples stem from 25 tissues with 1–15 samples per tissue.

The minimum number of samples per tissue is set to 8.

Outlier removal and selection of the same number of samples per type results in a $64 \times 53{,}679$ ($64 = 8types * 8samples\_each$) matrix.

Note, that some samples have zero-values for most of the reads and only some high values. These samples could easily be detected by a limitation to the number of zero-values per sample (e. g. no more than 75 % zero-entries).

5.2.3 D3: Expression Atlas *E-MTAB-2836*

The Expression Atlas dataset *E-MTAB-2836* contains counts for 200 samples and 65,217 reads. Samples stem from 32 tissues with 3–13 samples per tissue.

The minimum number of samples per tissue is set to 8.

Outlier removal and selection of the same number of samples per type results in a $64 \times 49{,}914$ ($64 = 8types * 8samples\_each$) matrix.

5.2.4 D4: GEO *GSE112004*

The GEO dataset *GSE112004* is a single cell dataset and contains read counts for 3,648 samples and 11,841 genes (already mapped). Samples stem from mice CD19+ B cells: untreated, treated to trans-differentiate to macrophages, and treated to trans-differentiate to induced pluripotent stem cells (3 types). For each type, measurements at three time points are available (considered as 9 types in total).

The minimum number of samples per tissue is set to 32.

Outlier removal and selection of the same number of samples per type results in a $288 \times 11{,}781$ ($288 = 9types * 32samples\_each$) matrix.

5.2.5 Normalisation

In [3] Clearly et al. suggest to normalise the data by a removal of genes for which the sum of counts is $> 99.5th - percentil$ to "avoid performance statistics that are skewed by few genes with extremely high expression". We adapt this normalisation and perform the same normalisation for the samples. Additionally, to avoid for a bias of different experiments, each sample is normalised by division through the sum of all counts for this sample. Subsequently, to provide numerical stability and allow greater interpretability of the dictionary entries, variables are centred to zero and scaled to have a standard deviation of one.

6 Discussion and conclusion

In this study, we present our results for a new version of dictionary learning. We apply the method to simulated and real world data. We evaluate both, the relation of sample labels and clusters of the sparse codings, as well as the biological relevance of the gene-sets represented by the dictionary atoms. Clustering the sparse codings of the real world datasets showed that the dictionary keeps relevant properties of the data. Further, the biological relevance of the gene-sets detected by the DiL approach was confirmed by a GO-term analysis, which revealed the large potential of dictionary learning in RNA-seq analysis to extract type specific gene-sets. We performed a large study on four datasets, showing that results are not unique to one dataset.

In contrast to [3] the focus of our analysis is not to find a dictionary such that a high-dimensional transcriptomic profile can be generated from sequencing a small, random selection of genes. Rather, we aim at 1.) detecting relevant genes-sets and 2.) thereby confirming that the dictionary keeps relevant properties of the data – this in mind, the sparse codings can be used for further analysis.

In contrast to [12] we perform only one step of dictionary learning which is a lot faster than the standard dictionary learning and again several times faster than the nested dictionary learning presented in [12].

We have only evaluated clusterings with the number of clusters, k, equal to the number of types given in the data, n. In some cases, we found that one type was split whereas others were found in one cluster, which results in an $ARI < 1$. We found that separation of some types was in agreement with subtypes in some cases. Assuming that differences between these subtypes is bigger than among other general types, this leads to those general types being grouped in one cluster. We have to keep in mind, that by setting $k = n$, we rely on the correctness of the data annotation. Evaluating higher values of k might resolve this issue, such that the types that appear in one clustered for $k = n$ will then be in separate clusters.

The DiL method requires two parameters, the number of atoms of the dictionary, m, and the level of sparsity, s. Our experiments revealed that for every dataset evaluated there was a large range of parameters for which the variation of ARI, respectively AMI is small. E. g. setting $m = s = n$, hence only evaluating one parameter setting, results in an ARI that is larger than $0.8 * ARI_{best}$. A small grid search around $m = s = n$ improves results further.

This paper is focused on the method and its analysis on multiple RNA-seq datasets as well as a detailed analysis of the results. We have only performed very limited normalisation of the data. A more extensive normalisation might improve results even further. This becomes visible in the analysis of dataset D2, where one simple additional normalisation step improved the ARI from 0.56 to 0.8.

By thresholding we filtered the dictionary matrix to receive the $2\%Dictionary$ and used this to select the gene-sets. This could be improved by adding a constraint in the dictionary learning process, such that the dictionary itself will be sparse and thresholding would no longer be required.

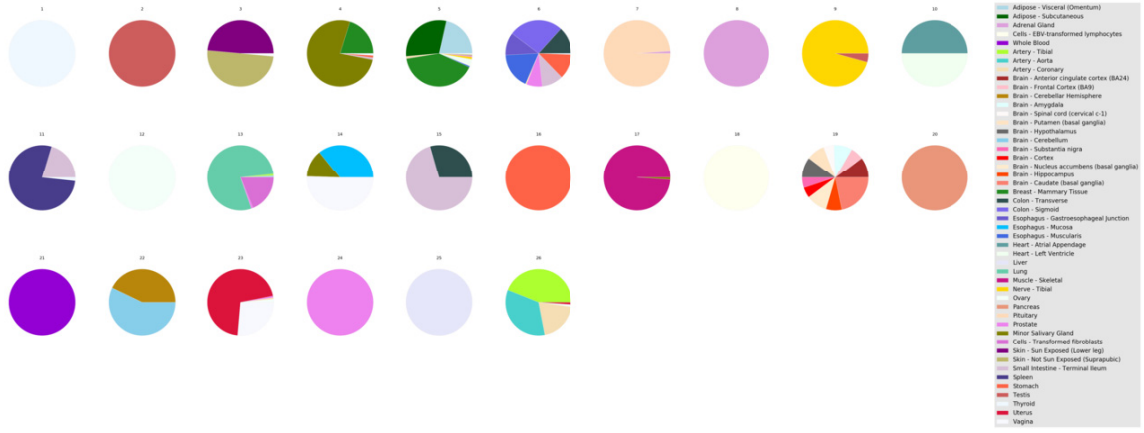
Other related methods, such as NMF and SVD do not enforce sparsity per sample. DiL, however, fulfils this criterion. With the results of this paper in mind, showing that the dictionary keeps relevant properties of the data, one could use the dictionary for the representation of states (e. g. of a single cell), and study its e. g. time-dependent behaviour in the dictionary space. Application of dictionary learning on dataset D4 shows that results for single cell data are similarly good as for pooled cells.

In conclusion, our study demonstrates that our dictionary learning based approach is able to detect biologically relevant modules of genes specific to various types, as well as to represent RNA-seq data in lower dimension.

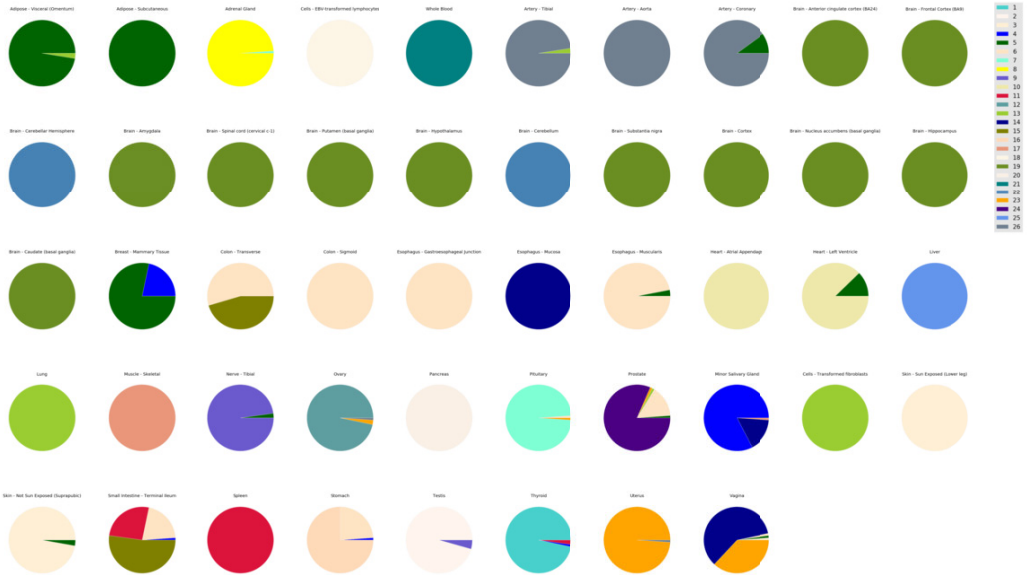
Funding: This study was supported by the Federal Ministry of Education and Research of Germany, grant 3FO18501 (Forschungscampus MODAL) and grant 05M16KEA (IBOSS) to TC and MR. The funders did not have any additional role in the study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Appendix

The following figures show results of the analysis of datasets D1–D4.



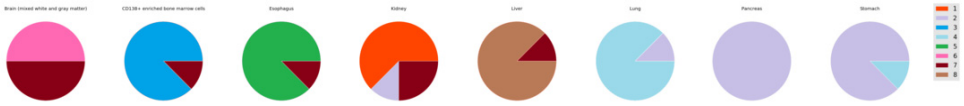
(a) Pie chart by cluster: True labels as detailed tissue types



(b) Pie chart by tissue: True labels as detailed tissue types

Figure 9: Shown are results for dataset D1 for the maximal ARI measured (dictionary with 37 atoms and a 32-sparse coding). (a) Shown are the detailed tissue types in the clusters. Many clusters gather subtypes of one tissue (compare Figure 6), which is what we aim for since the number of clusters is set equal to the number of general tissue types. Interestingly, some general tissue types are separated into few cluster, such as brain in cluster 11 and 19. These cluster separate the cerebellum from the cerebrum. (c, d) Shown are the clusters for each tissue.

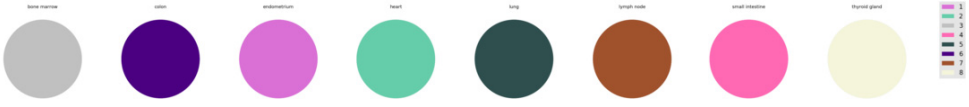
(a) Pie chart by cluster



(b) Pie chart by tissue

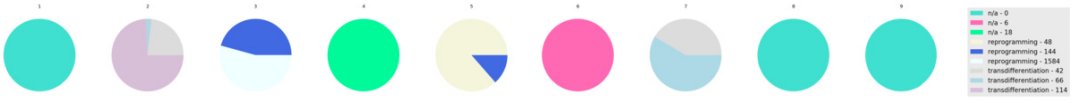
Figure 10: Shown are results for dataset D2 for the dictionary with 7 atoms and a 1-sparse coding. (a) Shown are the tissues in the clusters. Five clusters consist of one tissue type only. (b) Shown are the clusters for each tissue. Interestingly brain tissues are separated into two clusters which might represent a separation of grey and white matter (no data available).

(a) Pie chart by cluster



(b) Pie chart by tissue

Figure 11: Shown are results for dataset D3 for the dictionary with 7 atoms and a 1-sparse coding. (a) Shown are the tissues in the clusters. (b) Shown are the clusters for each tissue. As the ARI for this data is 1, all pies are of one colour only.



(a) Pie chart by cluster

(b) Pie chart by type

Figure 12: Shown are results for dataset D4 for the dictionary with 8 atoms and a 4-sparse coding. The type labels describe the treatment type and time (hours). (a) Shown are the types in the clusters. Five clusters consist of one tissue type only. (b) Shown are the clusters for each tissue. The three treatment types are separated, wrong assignments appear only for times within one treatment class.

References

1. Orly Alter, Patrick O. Brown, and David Botstein. Singular value decomposition for genome-wide expression data processing and modeling. *Proceedings of the National Academy of Sciences*, 97(18):10101–10106, 2000.
2. Sven Bergmann, Jan Ihmels, and Naama Barkai. Iterative signature algorithm for the analysis of large-scale gene expression data. *Physical review E*, 67(3):031902, 2003.
3. Brian Cleary, Le Cong, Anthea Cheung, Eric S. Lander, and Aviv Regev. Efficient generation of transcriptomic profiles by random composite measurements. *Cell*, 171(6):1424–1436, 2017.
4. Ronald R. Coifman and David L. Donoho. Translation-invariant de-noising. In *Wavelets and statistics*, pages 125–150.

Springer, 1995.
5. Gene Ontology Consortium. Gene ontology consortium: Going forward. Nucleic acids res. 43:D1049–d1056, 2015.
6. Michael Elad and Michal Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image processing*, 15(12):3736–3745, 2006.
7. Amin Emad and Olgica Milenkovic. Caspian: A causal compressive sensing algorithm for discovering directed interactions in gene networks. *PloS one*, 9(3):e90781, 2014.
8. Y. Fang, L. Chen, J. Wu, and B. Huang. Gpu implementation of orthogonal matching pursuit for compressive sensing. In *2011 IEEE 17th International Conference on Parallel and Distributed Systems*, pages 1044–1047, Dec. 2011.
9. Lei Huang, Yan Jin, Yaozong Gao, Kim-Han Thung, Dinggang Shen, et al. Alzheimer's Disease Neuroimaging Initiative. Longitudinal clinical score prediction in Alzheimer's disease with soft-split sparse regression based random forest. *Neurobiology of aging*, 46:180–191, 2016.
10. Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of classification*, 2(1):193–218, 1985.
11. Morteza Kolali Khormuji and Mehrnoosh Bazrafkan. A novel sparse coding algorithm for classification of tumors based on gene expression data. *Medical & biological engineering & computing*, 54(6):869–876, 2016.
12. Elina Koletou. *Prostate cancer patient stratification with MINING: Molecular Signatures via Nested Dictionary Learning*. PhD thesis, ETH Zurich, 2019.
13. Jin-Xing Liu, Yong Xu, Chun-Hou Zheng, Heng Kong, and Zhi-Hui Lai. Rpca-based tumor classification using gene expression data. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, 12(4):964–970, 2015.
14. Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982.
15. Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. Online dictionary learning for sparse coding. In *Proceedings of the 26th annual international conference on machine learning*, pages 689–696. ACM, 2009.
16. Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. Online dictionary learning for sparse coding. In *Proceedings of the 26th annual international conference on machine learning*, pages 689–696. ACM, 2009.
17. Yagyensh Chandra Pati, Ramin Rezaiifar, and Perinkulam Sambamurthy Krishnaprasad. Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. In *Signals, Systems and Computers, 1993. 1993 Conference Record of The Twenty-Seventh Asilomar Conference on*, pages 40–44. IEEE, 1993.
18. Yosef Prat, Menachem Fromer, Nathan Linial, and Michal Linial. Recovering key biological constituents through sparse representation of gene expression. *Bioinformatics*, 27(5):655–661, 2011.
19. Ron Rubinstein, Michael Zibulevsky, and Michael Elad. *Efficient implementation of the k-svd algorithm using batch orthogonal matching pursuit*. Technical report, Computer Science Department, Technion, 2008.
20. Eran Segal, Michael Shapira, Aviv Regev, Dana Pe'er, David Botstein, Daphne Koller, and Nir Friedman. Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. *Nature genetics*, 34(2):166, 2003.
21. Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.
22. Nguyen Xuan Vinh, Julien Epps, and James Bailey. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *Journal of Machine Learning Research*, 11(Oct.):2837–2854, 2010.
23. Meng Yang, Lei Zhang, Xiangchu Feng, and David Zhang. Fisher discrimination dictionary learning for sparse representation. In *2011 International Conference on Computer Vision*, pages 543–550. IEEE, 2011.
24. Yuan You, Hongmin Cai, and Jiazhou Chen. Low rank representation and its application in bioinformatics. *Current Bioinformatics*, 13(5):508–517, 2018.
25. Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the royal statistical society: series B (statistical methodology)*, 67(2):301–320, 2005.

Bionotes

MSc. Mona Rams
Freie Universität Berlin, Institute for Mathematics, Arnimallee 6, 14195 Berlin, Germany
mrams@math.fu-berlin.de

Mona Rams received her M. Sc. in Bioinformatics, at the Freie Universität Berlin, Germany, in 2016 with a focus on network analysis and machine learning. She is currently a Ph. D. student at the Medical Bioinformatics Group at Freie Universität Berlin. Her research interests include analysis of transcriptomics data with a focus to machine learning approaches for biomarkers identification.

Prof. Dr. Tim Conrad
Freie Universität Berlin, Institute for Mathematics, Arnimallee 6, 14195 Berlin, Germany
Zuse Institute Berlin, Takustraße 7, 14195 Berlin, Germany
Berlin Institute for the Foundations of Learning and Data, Berlin, Germany
conrad@math.fu-berlin.de

Prof. Dr. Tim Conrad is head of the Medical Bioinformatics group at Freie Universität Berlin, group leader at the Zuse Institute Berlin (ZIB) and principal investigator at the Berlin Institute for the Foundations of Learning and Data (BIFOLD). Tim studied Bioinformatics and Computer Science in Berlin and Melbourne. His research interests include machine learning for bio-medical data, network-based data integration and big data analysis.