AG Neeße - Journal Club

DeepMicro: deep representation learning for disease prediction based on microbiome data

Min Oh & Liqing Zhang

Linh Dang

2026-02-12

Paper

www.nature.com/scientificreports

SCIENTIFIC REPORTS
natureresearch

OPEN

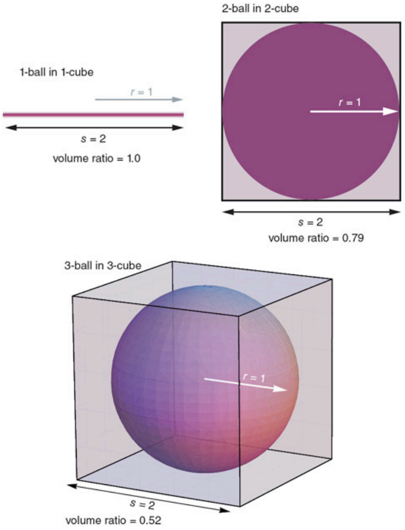
DeepMicro: deep representation learning for disease prediction based on microbiome data

Min Oh & Liqing Zhang*

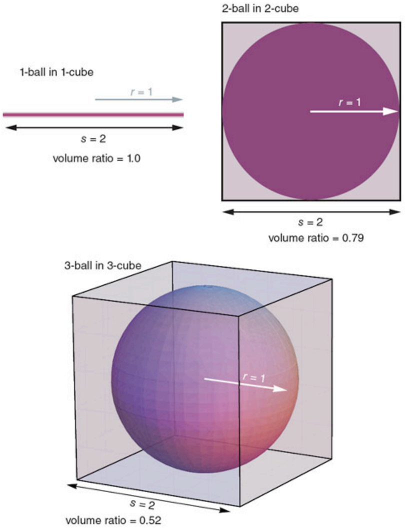
Table of Contents

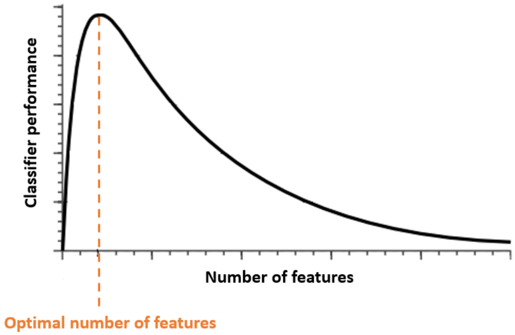
- Problem Statement
- Materials & Methods
- Results
- Discussion

The Curve of Dimensionality

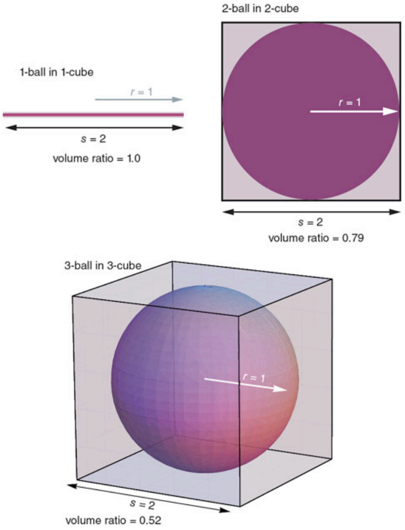


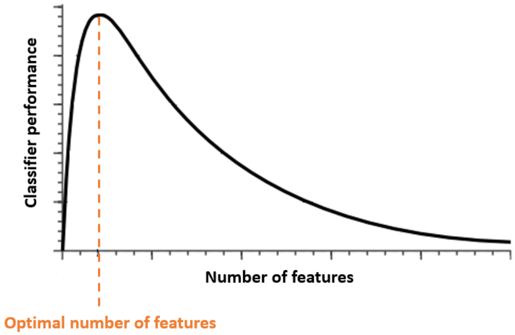
The Curve of Dimensionality





The Curve of Dimensionality





Adding dimensions →

- exponential increase in volume
- data sparity and distance metric less meaningful

High Dimensionality in Microbiome Data

Profile Type	IBD	EW-T2D	C-T2D	Obesity	Cirrhosis	Colorectal
strain-level marker profile	91,756	83,456	119,792	99,568	120,553	108,034
abundance profile	443	381	572	465	542	503

DeepMicro

Current Challenges:

Current Challenges:

- Effective dimensionality reduction, yet preserves the intrinsic structure of the microbiome data.

Current Challenges:

- Effective dimensionality reduction, yet preserves the intrinsic structure of the microbiome data.
- Deep learning algorithm to predict disease states.

DeepMicro

Current Challenges:

- Effective dimensionality reduction, yet preserves the intrinsic structure of the microbiome data.
- Deep learning algorithm to predict disease states.

Goals:

- robust low-dimensional representations from high-dimensional microbiome profiles

Current Challenges:

- Effective dimensionality reduction, yet preserves the intrinsic structure of the microbiome data.
- Deep learning algorithm to predict disease states.

Goals:

- robust low-dimensional representations from high-dimensional microbiome profiles
- Deep learning framework

Datasets

Disease	Dataset Name	# total samples	# of healthy controls	# of patient samples
Inflammatory Bowel Disease	IBD	110	85	25
Type 2 Diabetes	EW-T2D	96	43	53
	C-T2D	344	174	170
Obesity	Obesity	253	89	164
Liver Cirrhosis	Cirrhosis	232	114	118
Colorectal Cancer	Colorectal	121	73	48

Datasets

Disease	Dataset Name	# total samples	# of healthy controls	# of patient samples
Inflammatory Bowel Disease	IBD	110	85	25
Type 2 Diabetes	EW-T2D	96	43	53
	C-T2D	344	174	170
Obesity	Obesity	253	89	164
Liver Cirrhosis	Cirrhosis	232	114	118
Colorectal Cancer	Colorectal	121	73	48

- **Sequencing Method**: whole-genome shotgun metagenomic

Datasets

Disease	Dataset Name	# total samples	# of healthy controls	# of patient samples
Inflammatory Bowel Disease	IBD	110	85	25
Type 2 Diabetes	EW-T2D	96	43	53
	C-T2D	344	174	170
Obesity	Obesity	253	89	164
Liver Cirrhosis	Cirrhosis	232	114	118
Colorectal Cancer	Colorectal	121	73	48

- **Sequencing Method**: whole-genome shotgun metagenomic
- **Tool**: MetaPhlAn2 was used to extract 1) strain-level marker pro-file and 2) species-level relative abundance profile.

Profile Extraction

Profile Extraction

Relative abundance

	Sample_1	Sample_2	...	Sample_N
species_1	$a_{1,1}$	$a_{1,2}$	...	$a_{1,N}$
species_2	$a_{2,1}$	$a_{2,2}$	...	$a_{2,N}$
...				
species_m	$a_{m,1}$	$a_{m,2}$	...	$a_{m,N}$

- $a_{i,j} \in [0, 1]$
- $m \approx 500$

Profile Extraction

Relative abundance

	Sample$_1$	Sample$_2$	...	Sample$_N$
species$_1$	$a_{1,1}$	$a_{1,2}$	...	$a_{1,N}$
species$_2$	$a_{2,1}$	$a_{2,2}$	...	$a_{2,N}$
...				
species$_m$	$a_{m,1}$	$a_{m,2}$	...	$a_{m,N}$

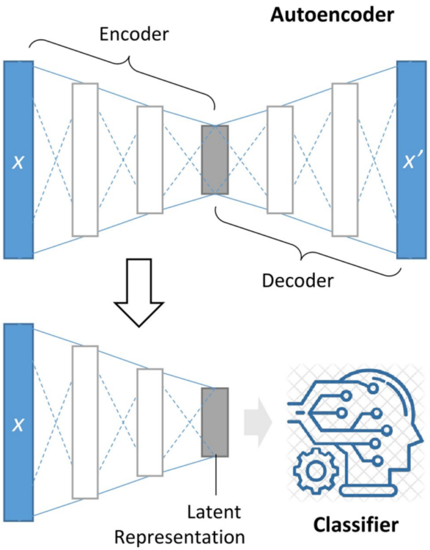
- $a_{i,j} \in [0, 1]$
- $m \approx 500$

Strain-level marker

	Sample$_1$	Sample$_2$	...	Sample$_N$
Marker$_1$	$b_{1,1}$	$b_{1,2}$	...	$b_{1,N}$
Marker$_2$	$b_{2,1}$	$b_{2,2}$	...	$b_{2,N}$
...				
Marker$_M$	$b_{M,1}$	$b_{M,2}$	...	$b_{M,N}$

- $b_{i,j} \in \{0, 1\}$
- $M \approx 100,000$

Deep representation learning



Key ideas:

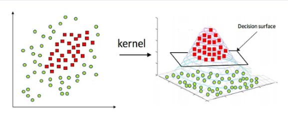
- Input: x, encoder function: $f_\phi(.)$, decoder function: $f'_\theta(.)$.
- $f(.)$ and $f^{'}(.)$ belong to one of the autoencoder framework: ***SAE***, ***DAE***, ***VAE***, ***CAE***
- Objective funtion:
  $\arg\min_{\phi,\theta} L(x, x^{'}) = \|x - x^{'}\| = \|x - f_\phi(f'_\theta(x))\|$
- Low-dimensional representation of x is $f_\theta(x)$.
  - could be used as features for other classifier such as *Random Forest*, *SVM*, or deep learning method itself.

Classifiers used in this study (1)

Classifiers used in this study (1)

support vector machine (SVM)

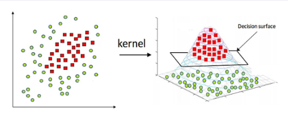
- radial basis function (RBF) kernel
- linear kernel function kernel



Classifiers used in this study (1)

support vector machine (SVM)

- radial basis function (RBF) kernel
- linear kernel function kernel



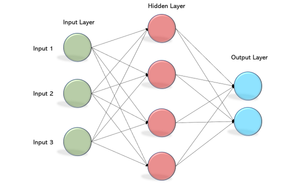
Random Forest (RF)

- Various number of tress - Impurity: Gini, information gain
- 100 combinations of hyper-parameters of RF.

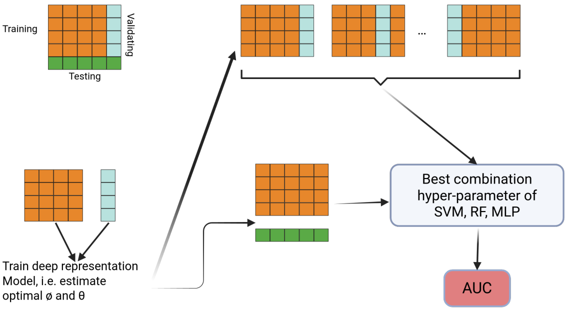
Classifiers used in this study (2)

Multi-Layer Perceptron (MLP)

- 1 input layer - up to 3 hidden layers - 1 output layer
- Various units in the first hidden layer - various dropout rate
- 120 hyper-parameter combinations of MLP



Experimental Setting



Takeaway notes

- Training / testing set ratio: 80/20
- Only training set:
  - 80 / 20 : for training and validating → optimal deep representation.
- For each classifier (SVM, RF, MLP), the best combination hyper-parameters is chosen by 5-folds cross validation
- Evaluation: AUC

Four Groups in the Assessment

Four Groups in the Assessment

DeepMicro

- Autoencoders: SAE, DAE, CAE, VAE
- Classifier: SVM, RF, MLP

Four Groups in the Assessment

DeepMicro

- Autoencoders: SAE, DAE, CAE, VAE
- Classifier: SVM, RF, MLP

MetAML

- Built-in classifier: SVM and RF
- Each with best hyper-parameters.

Four Groups in the Assessment

DeepMicro

- Autoencoders: SAE, DAE, CAE, VAE
- Classifier: SVM, RF, MLP

PCA-based

- Principal components explaining 99%
- Classifier: RF, SVM, MLP

MetAML

- Built-in classifier: SVM and RF
- Each with best hyper-parameters.

Four Groups in the Assessment

DeepMicro

- Autoencoders: SAE, DAE, CAE, VAE
- Classifier: SVM, RF, MLP

MetAML

- Built-in classifier: SVM and RF
- Each with best hyper-parameters.

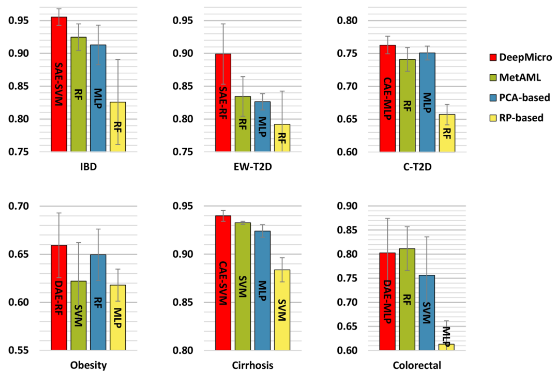
PCA-based

- Principal components explaining 99%
- Classifier: RF, SVM, MLP

Gaussian Random Projection (RP)-based

- Another high dimensional reduction method
- components to be automatically adjusted according to Johnson-Lindenstrauss lemma
- Classifier: RF, SVM, MLP

Results

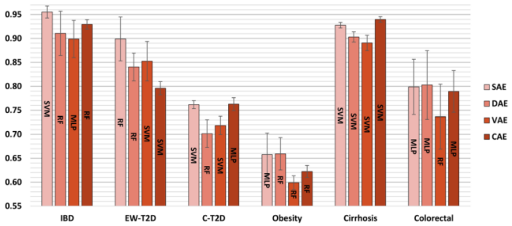


Notes

For the strain-level marker profile:

- DeepMicro outperforms others on 5/6 datasets
- The marker profile generally perform better than the abundance profile (not shown here).

Autoencoder Assessment



Notes

- No specific autoencoder dominates others.
- For abundance profile, CAE with RF outperforms others.

Result Notice

Result Notice

MLP on original profile (without representation learning)

- perform better than MetAML in three datasets (EW-T2D, C-T2D, and Obesity)
- on abundance profile: worse than traditional methods.

Result Notice

MLP on original profile (without representation learning)

- perform better than MetAML in three datasets (EW-T2D, C-T2D, and Obesity)
- on abundance profile: worse than traditional methods.

DeepMicro

- Running time 8x - 30x faster than other basis approaches.

Discussion

Discussion

- Dimensional reduction by PCA: slightly better results only on 2/6 datasets $\rightarrow$
    - Essensial information was dropped
    - Noise was remained.

Discussion

- Dimensional reduction by PCA: slightly better results only on 2/6 datasets $\rightarrow$
  - Essensial information was dropped
  - Noise was remained.
- Autoencoders:
  - keep essential information in a condensed way
  - highly depends on properties of datasets.

Discussion

- Dimensional reduction by PCA: slightly better results only on 2/6 datasets →
  - Essensial information was dropped
  - Noise was remained.
- Autoencoders:
  - keep essential information in a condensed way
  - highly depends on properties of datasets.
- Adding healthy controls generally results in better performance, **But** here the performance was slightly dropped whey including healthy samples in the training phase.
  - Explanation: changes in negative samples rarely contributes to classification of positive samples.
  - Adding heathy samples before traing-testing split, results in better performance.
  - In general, adding negative samples create more balanced dataset, leading to better and roburst performance.