**Algorithms** CS 5312

# Representation Learning and Data Reduction

*Imdad Ullah Khan*

# Contents



# 1 Introduction

Recall from early lecture a data analyst gets the data in a raw format that may have a number of issues qualitative issues. Raw data is what one gets from sources, it is often very hard to analyze. To make it ready for analytics one needs to perform certain preprocessing on the data to make it ready for analytics. In some cases this preprocessing may be the end goal of the data analytics.

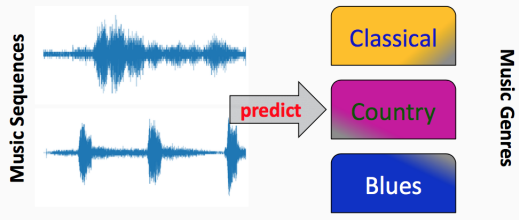
We discussed some basic preporcessing such data cleaning, handling missing values, merging data, data transformation and normalization. In this session we discuss some more detailed data preprocessing tasks. These are major tasks and can be the end goal of data analytics. These tasks typically come under the heading of data reduction, representation learning, feature engineering etc.

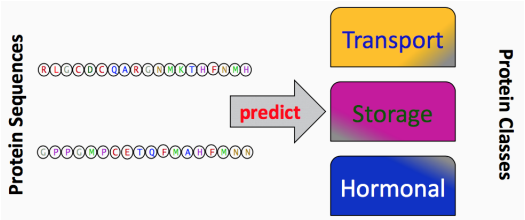
The aim of this session is to make you familiar with these very interesting topics and important concepts, each of which have many applications and is often the main topic of a whole course.

# 2 Representation Learning

Also called feature learning or feature engineering is a set of methods to automatically learn a representation for the dataset that is used for further data analytics.

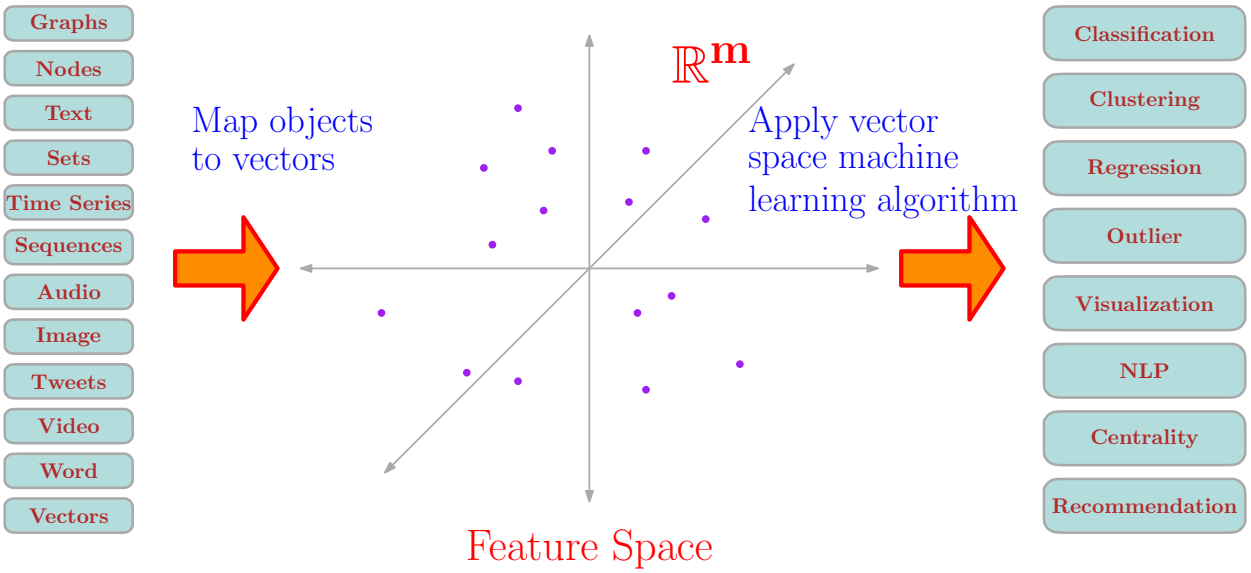
This is required because machine learning algorithm require data in a particular format (such as real vectors), usually that input is mathematically and computationally convenient to process. Real-world data comes in a variety of format (recall our discussion on the 'V' for variety).



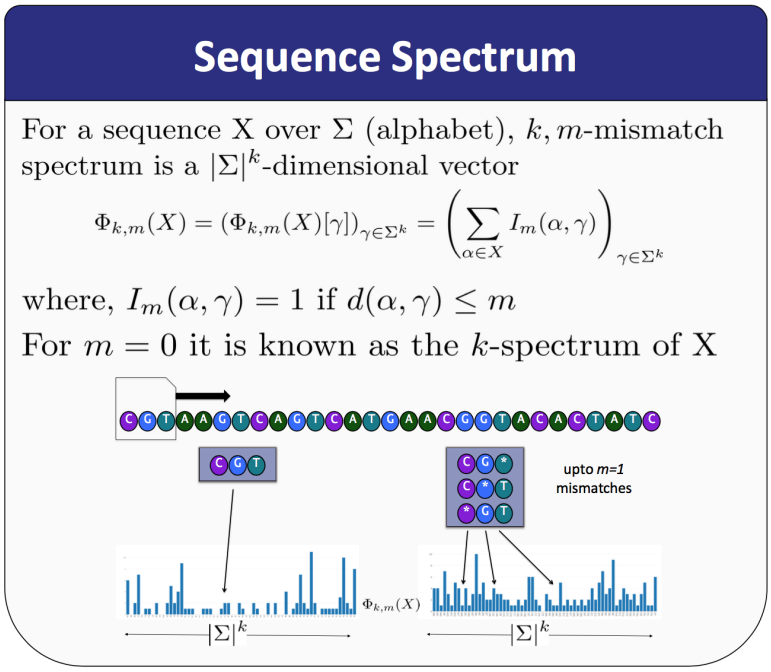

In many application the raw data is not in vector format, hence to apply data analytics applicable on vector spaces one first map data objects into a feature space (i.e. represent each object with a feature vector) and then apply analytics algorithms.

Representation learning can be supervised (such as dictionary learning or deep neural network) or unsupervised (such as principal component analysis, matrix factorization, autoencoders etc.)

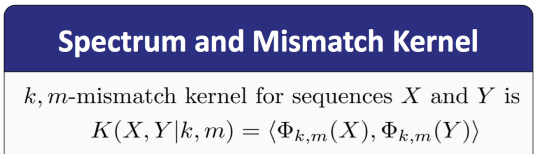


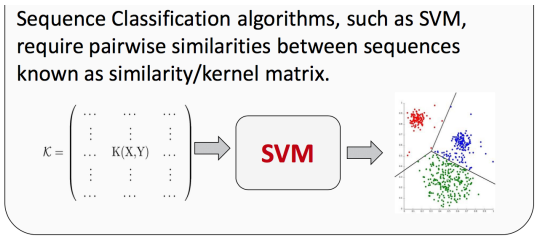

The representation learning approach has yielded great success in capturing complex relationships across various disciplines such as biological networks, social media, node attributes in social networks [12, 33, 8].

In a highly successful method, these feature vectors are based on counts of various substructures in the objects.




A kernel function to estimate the pairwise similarity between objects is then defined which usually is an inner product between the corresponding feature vectors. The kernel matrix (values of the kernel function for all pairs) is then input to machine learning algorithm.




This approach has been used for classifying images [11], sequences [15, 18], and graphs [17, 26, 34]. In the descriptors learning approach, objects are mapped to low dimensional vectors of features extracted from objects with the goal to map similar objects closely in the Euclidean space. Many descriptors have been proposed for EEG and EMG sequences [9, 30] electricity consumptions [6, 7] and graphs [14, 29, 31, 27, 16]. More recent but computationally expensive methods employ deep networks together with domain specific techniques for embedding nodes [13], graphs [32, 20] and texts [25, 24, 23].

# 3 Data Compression

Data compression is the process of encoding information with fewer bits than the original representation and often deals with large volumes of data. Given a point set $X = \{x_1, x_2, \ldots, x_n\}$, the

objective is to find a compression scheme $f : X \mapsto X'$ (encoder) and a decompressor $g : X' \mapsto X$ (decoder) such that $\sum_{i=1}^{n} \|x_i - g(f(x_i))\|^p$, called the called $\ell_p$ reconstruction error, is minimized.

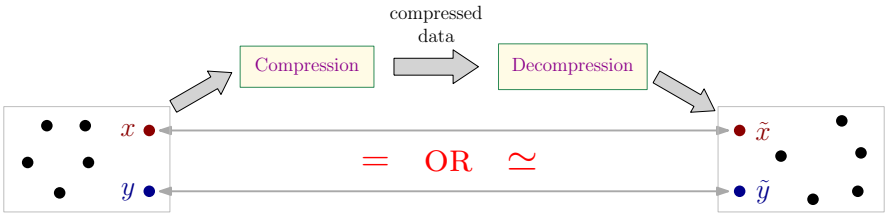
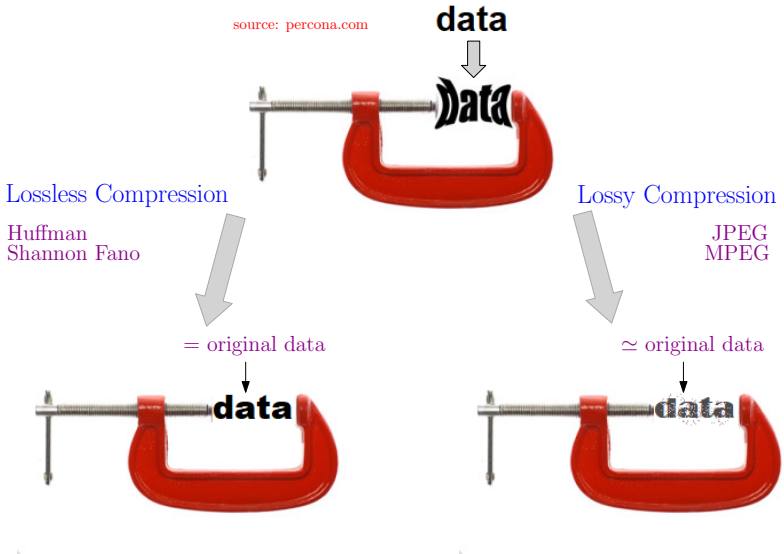



Figure 1

Note that $g$ is not necessarily $f^{-1}$. If $g = f^{-1}$, compression is called *lossless* otherwise it is *lossy*. In lossless compression, the original data is recovered completely as it is after decompression, i.e. $X = X'$ but in lossy compression, the recovered data maybe somewhat 'distorted', i.e. $X \simeq X'$ as shown in Figure **??**




# 4 Graph Summarization Compression

Graphs on millions of nodes and billion of edges are increasingly common

The huge magnitudes of graphs (order of a graph is the number of nodes in it and size of a graph is the number of edges in it) pose serious computational challenges for graph Analysis and

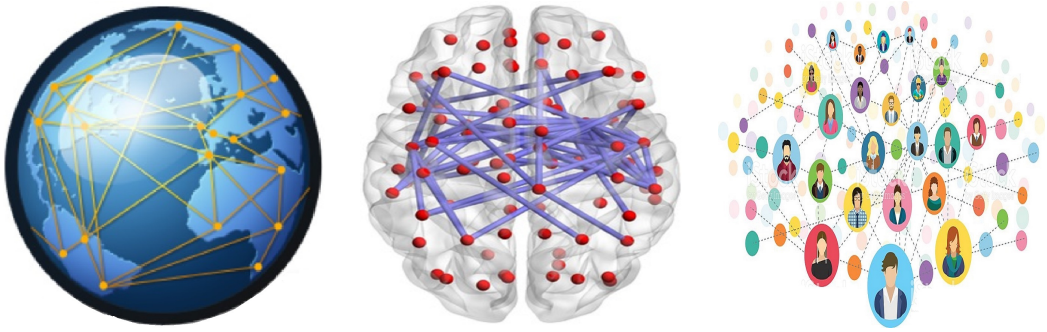
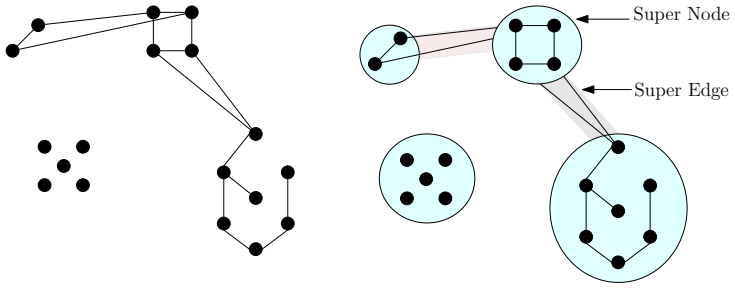


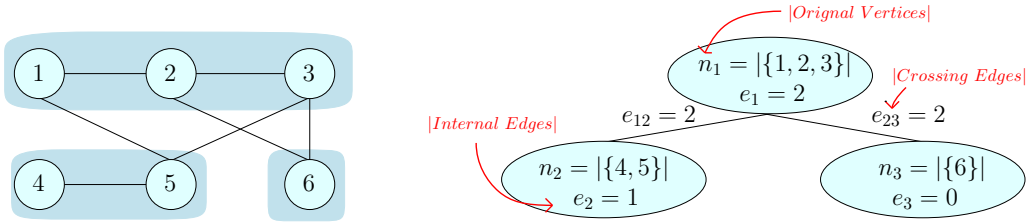
Figure 2 a) World Wide Web – b) Biological Network — c) Social Network

Processing, graph visualization, storage and retrieval and transfer over a network (communication bandwidth)

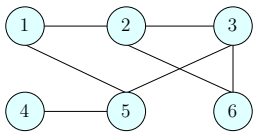
A very successful approach to overcome these challenges is graph summarization or graph compression. Where we perform all processing and network analytics instead on a compact version of the graph. The summary of a graph removes certain details yet preserve the essential structure of the graph. More formally, a summary is a partition of vertices of the graph represented by a *supergraph*. Each *super node* represents a subset of vertices of the original graph and Each *super edge* between a pair of supernodes represents edges between the subsets corresponding to the vertices in the supernodes.

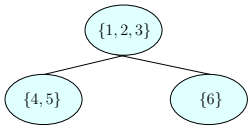



The summary is a graph with weights both on superedges and supernodes.




Given the smaller order and size of the summary graph it is easier to process and visualization. One can perform graph analytics such as get approximate answers to graph structure queries ($(u, v) \sim E$) from the summary only. As pointed above, the summary sometimes is the desired output (with applications in aggregate statistics and privacy preservation).




$$A = \begin{array}{c} \\ \mathbf{1} \\ \mathbf{2} \\ \mathbf{3} \\ \mathbf{4} \\ \mathbf{5} \\ \mathbf{6} \end{array} \begin{array}{c} \begin{array}{cccccc} \mathbf{1} & \mathbf{2} & \mathbf{3} & \mathbf{4} & \mathbf{5} & \mathbf{6} \end{array} \\ \begin{bmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \qquad \bar{A} = \begin{array}{c} \\ \mathbf{1} \\ \mathbf{2} \\ \mathbf{3} \\ \mathbf{4} \\ \mathbf{5} \\ \mathbf{6} \end{array} \begin{array}{c} \begin{array}{cccccc} \mathbf{1} & \mathbf{2} & \mathbf{3} & \mathbf{4} & \mathbf{5} & \mathbf{6} \end{array} \\ \begin{bmatrix} 0 & \frac{2}{3} & \frac{2}{3} & \frac{1}{3} & \frac{1}{3} & \frac{2}{3} \\ \frac{2}{3} & 0 & \frac{2}{3} & \frac{1}{3} & \frac{1}{3} & \frac{2}{3} \\ \frac{2}{3} & \frac{2}{3} & 0 & \frac{1}{3} & \frac{1}{3} & \frac{2}{3} \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & 0 & 1 & 0 \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & 1 & 0 & 0 \\ \frac{2}{3} & \frac{2}{3} & \frac{2}{3} & 0 & 0 & 0 \end{bmatrix} \end{array}$$

**Goodness of a Summary** is measured by the Reconstruction Error: the $\ell_p$ norm of error matrix

$$\left\|A - \bar{A}\right\|_p = RE_p(G|S) = \left(\sum_{i=1}^{n}\sum_{j=1}^{n} |A(i,j) - \bar{A}(i,j)|^p\right)^{1/p}$$

where $\overline{A}$, the expected adjacency matrix is defined as: $\bar{A}(u,v)$ is the probability of an edge between $u$ and $v$

$$\bar{A}(u,v) = \begin{cases} 0 & \text{if } u = v \\ \dfrac{e_i}{\binom{n_i}{2}} & \text{if } u, v \in V_i \\ \dfrac{e_{ij}}{n_i n_j} & \text{if } u \in V_i, v \in V_j \end{cases}$$

This matrix difference can be naively computed in $O(n^2)$

For algorithms to compute the best summary efficiently see [19, 21, 22, 10]. For applications of graph summarization in network immunization and estimating spectral radius of a graph see [1, 3, 5, 4, 28, 2]

# 5 Low Distortion Embedding

Given two metric spaces $(X, d)$ and $(Y, d')$ and a real $\alpha > 0$, the low distortion embedding problem is to find an embedding function $f : X \mapsto Y$ such that

$$\forall\ x_i, x_j \in X \quad \frac{1}{\alpha} d(x_i, x_j) \ \leq\ d'(f(x_i), f(x_j)) \ \leq\ d(x_i, x_j)$$

.

Points in $X$ are embedded into $Y$, almost preserving pairwise distances, because the space $Y$ may be easy to work with or the distance metric $d'$ may be computationally nicer. Examples

include Graph vertices with shortest paths distances embedded to $(\mathbb{R}^k, \ell_2)$, as shown in Figure 3 or sequences with edit distance embedded into Euclidean space.

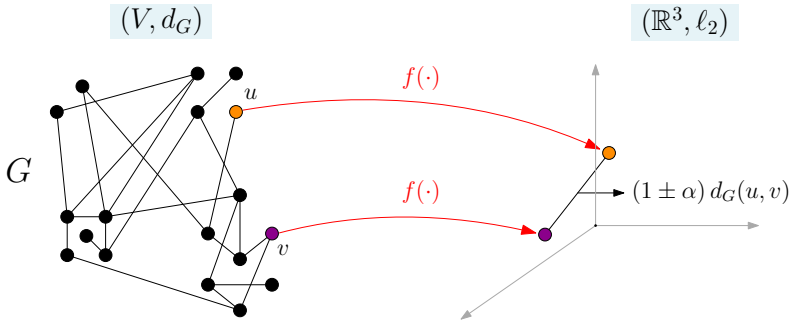



Figure 3

# 6 Multi-dimensional Scaling

Given $X = \{x_1, \ldots, x_n\}$ and pairwise distance matrix $D = \{d_{ij}\}$, the multi-dimensional scaling problem is to find a $k$-dimensional representation $\{x'_1, x'_2, \ldots, x'_n\}$ for points in $X$ such that $\forall\ x_i, x_j \in X \quad d(x'_i, x'_j) \sim D(i, j)$. Figure 4 shows an example for $k = 2$.

source: statisticshowto.com

| CITY | Clearwater | Daytona Beach | Ft. Lauderdale | Gainesville | Jacksonville |
|---|---|---|---|---|---|
| Clearwater | 0 | 159 | 247 | 131 | 197 |
| Daytona Beach | 159 | 0 | 230 | 97 | 89 |
| Ft. Lauderdale | 247 | 230 | 0 | 309 | 317 |
| Gainesville | 131 | 97 | 309 | 0 | 68 |
| Jacksonville | 197 | 89 | 317 | 68 | 0 |

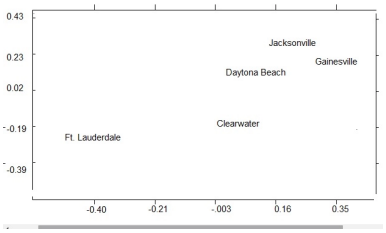



Figure 4

Many methods depending on whether or not the given and required distance measure is metric or Euclidean. A distance measure is metric if it satisfies the triangle inequality, i.e. $d(x, z) \leq d(x, y) + d(y, z)$. Squared Euclidean distances are not metric.

# 7 Dimensionality Reduction

**Problem 1.** *Given a point set $\mathcal{X} = \{x_1, \ldots, x_n\} \subset \mathbb{R}^m$, Find a dimensionality reduction function $f : \mathbb{R}^m \mapsto \mathbb{R}^k$, $k \ll m$ such that $d(f(x_i), f(x_j)) \simeq d(x_i, x_j)$, i.e.*

$$\forall\ x_i, x_j \in \mathcal{X} \quad (1 - \epsilon) d(x_i, x_j) \ \leq\ d(f(x_i), f(x_j)) \ \leq\ (1 + \epsilon) d(x, y)$$

Figure 5 shows two points in a high dimensional space reduced to a 5 dimensional space.

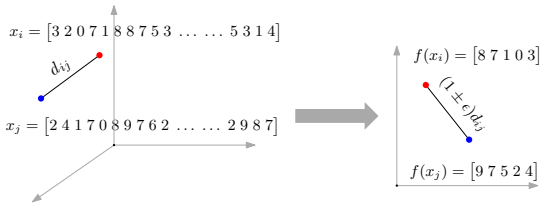



Figure 5

If $f$ is a linear function, it is called linear diensionality reduction and $f$ can be represented by a linear transformation $A$, i.e. $f(\mathcal{X}) = A\mathcal{X}$, $\mathcal{X}$: the $n \times m$ data matrix with each $x_i \in \mathcal{X}$ as a row.

Note that dimensionality reduction is a special case of low distortion embedding since the distance measure $d$ is the same in both domain and co-domain, i.e. the given and required distance measure is the same. Dimensionality reduction is, however, different from data compression since it does not require that $x \simeq f(x)$, but only that $d\big(f(x_i), f(x_j)\big) \simeq d(x_i, x_j)$ as shown in Figure 6.

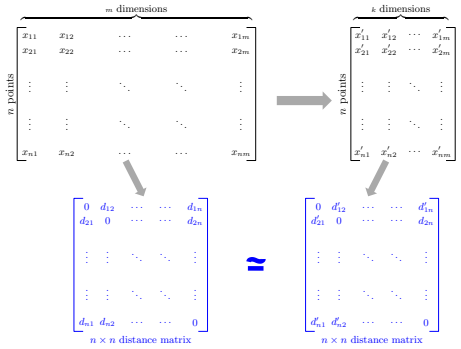



Figure 6

While the specific method of dimensionality reduction depends on the objective, all dimensionality reduction methods can be divided into two broad categories (as shown in Figure 7)

1. Feature Selection: Select a few variables that are the *most relevant* and discard the rest
2. Feature Extraction - Create new features from data, which are usually are linear or non-linear combination of old ones. The objective is least reconstruction error or maximum inter-class separation.

## 7.1 Feature Selection

An immediate idea (that does not work) for dimensionality reduction is to select a fixed subset of coordinates. Simply consider the first $k$ coordinates for each vector and ignore the rest. In the worst case all the information could be in the remaining coordinates (suppose all vectors are

almost 0 in the all of the first $k$ coordinates). Similarly considering the last $k$ coordinates or any fixed subset of coordinates for that matter would not work since all meaningful information about at least some of the classes of points could remain in the non-selected features.

One may suggest that, in order to not let the adversary know what coordinates you would choose, choose a random subset of $k$ coordinates and ignore the remaining. This may not work either as, again, all meaningful information about at least some of the classes of points could remain in the non-sampled features. For example, suppose that vectors have non zero entries only in the first $k$ coordinates. Then, a randomly chosen subset will select the mostly zero coordinates and distort most pairs of points.

Both the above methods are *data oblivious*. A *data dependent* approach would be to eliminate/select feature based on a goodness measure or (ir)relevance score of the data. For example:

- Feature variance - eliminate coordinate with close to 0 variance
- Eliminate one in a pair with close to $\pm 1$ correlation
- Eliminate features "independent" of class variable ($\rho$ or $\chi^2$)
- For each feature find training accuracy of classifier based on that feature only - eliminate those with low accuracy
- Score based on normalized mutual information, information gain, conditional entropy
- We discussed a domain specific criterion of eliminating feature - stop word removal for text analysis

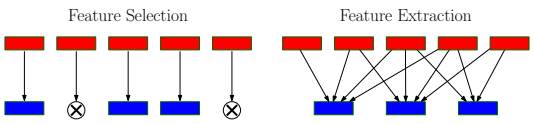



Figure 7

## 7.2 Feature Extraction

The objective of feature extraction is to create new (fewer) features using some (linear or non-linear) combinations of the old ones such that reconstruction error is minimized or inter-class separation is maximized. A form of feature extraction is *feature aggregation* which aggregates groups of coordinates. For example, means of $k$ groups of $m/k$ coordinates. One can construct examples where this would not work as well. It depends on how groups are made and a deterministic strategy can be countered by adversary. Randomized groups may also have problems.

Therefore, there is need for more sophisticated feature extraction methods.Before we discuss such a dimensionality reduction technique, i.e. let's recall the key linear algebra concept of projection.

# References


[1] S. Abbas, J. Tariq, A. Zaman, and I. Khan. Sampling based efficient algorithm to estimate the spectral radius of large graphs. In *IEEE International Conference on Distributed Computing Systems Workshops, ICDCSW*, pages 175–180, 2017.

[2] M. Ahmad, S. Ali, J. Tariq, I. Khan, M. Shabbir, and A. Zaman. Combinatorial trace method for network immunization. *Information Sciences*, 519:215 – 228, 2020.

[3] M. Ahmad, J. Tariq, M. Farhan, M. Shabbir, and I. Khan. Spectral methods for immunization of large networks. In *14th Australasian Data Mining Conference (AusDM)*, pages 137–145, 2016.

[4] M. Ahmad, J. Tariq, M. Shabbir, and I. Khan. Spectral methods for immunization of large networks. *Australasian Journal of Information Systems*, 21, 2017.

[5] Muhammad Ahmad, Juvaria Tariq, Muhammad Farhan Mudassir Shabbir, and Imdadullah Khan. Who should receive the vaccine? In *Australasian Data Mining Conference, AusDM 2016*, pages 137–145, 2016.

[6] S. Ali, H. Mansoor, N. Arshad, and I. Khan. Short term load forecasting using smart meter data. In *International Conference on Future Energy Systems*, pages 419–421, 2019.

[7] S. Ali, H. Mansoor, I. Khan, N. Arshad, Muhammad A. Khan, and S. Faizullah. Short-term load forecasting using ami data. *arXiv preprint arXiv:1912.12479*, 2020.

[8] Sarwan Ali, Muhammad Haroon Shakeel, Imdadullah Khan, Safiullah Faizullah, and Muhammad Asad Khan. Predicting attributes of nodes using network structure. *ACM Transactions on Intelligent Systems and Technology*, 12(2), 2021.

[9] M. Atzori, A. Gijsberts, C. Castellini, B. Caputo, AM Hager, S. Elsig, G. Giatsidis, F. Bassetto, and H. Müller. Electromyography data for non-invasive naturally-controlled robotic hand prostheses. *Scientific data*, 1(1):1–13, 2014.

[10] M. Beg, M. Ahmad, A. Zaman, and I. Khan. Scalable approximation algorithm forgraph summarization. In *Advances in Knowledge Discovery and Data Mining (PAKDD)*, pages 502–514, 2018.

[11] L. Bo, X. Ren, and D. Fox. Kernel descriptors for visual recognition. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 244–252, 2010.

[12] P. Cui, X. Wang, J. Pei, and W. Zhu. A survey on network embedding. *IEEE Transactions on Knowledge and Data Engineering*, 31(5):833–852, 2018.

[13] A. Duran and M. Niepert. Learning graph representations with embedding propagation. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 5119–5130, 2017.

[14] A. Dutta and H. Sahbi. Stochastic graphlet embedding. *Trans. Neural Netw. Learning Syst.*, 30(8):2369–2382, 2019.

[15] M. Farhan, J. Tariq, A. Zaman, M. Shabbir, and I. Khan. Efficient approximation algorithms for strings kernel based sequence classification. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 6935–6945, 2017.

[16] Z.R Hassan, M. Shabbir, I. Khan, and W. Abbas. Estimating descriptors for large graphs. In *Advances in Knowledge Discovery and Data Mining (PAKDD)*, pages 779–791, 2020.

[17] R. Kondor and H. Pan. The multiscale laplacian graph kernel. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 2982–2990, 2016.

[18] P. Kuksa, I. Khan, and V. Pavlovic. Generalized similarity kernels for efficient sequence classification. In *SIAM Intern. Conf. on Data Mining (SDM)*, pages 873–882, 2012.

[19] Kristen LeFevre and Evimaria Terzi. GraSS: Graph Structure Summarization. In *International Conference on Data Mining, SDM*, pages 454–465, 2010.

[20] C. Morris, M. Ritzert, M. Fey, W. Hamilton, J. Lenssen, G. Rattan, and M. Grohe. Weisfeiler and leman go neural: Higher-order graph neural networks. In *AAAI Conf. on Artificial Intelligence*, pages 4602–4609, 2019.

[21] Saket Navlakha, Rajeev Rastogi, and Nisheeth Shrivastava. Graph summarization with bounded error. In *International Conference on Management of Data, SIGMOD*, pages 419–432, 2008.

[22] Matteo Riondato, David García-Soriano, and Francesco Bonchi. Graph Summarization with Quality Guarantees. In *International Conference on Data Mining, ICDM*, pages 947–952, 2014.

[23] M. H. Shakeel, S. Faizullah, T. Alghamidi, and I. Khan. Language independent sentiment analysis. In *International Conference on Advances in the Emerging Computing Technologies (AECT)*, pages 1–5, 2020.

[24] M. H. Shakeel, A. Karim, and I. Khan. A multi-cascaded model with data augmentation for enhanced paraphrase detection in short texts. *Information Processing & Management*, 57(3):1–19, 2020.

[25] M.H. Shakeel., A. Karim, and I. Khan. A multi-cascaded deep model for bilingual sms classification. In *International Conference on Neural Information Processing*, pages 287–298, 2019.

[26] N. Shervashidze, P. Schweitzer, E. Van Leeuwen, K. Mehlhorn, and K. Borgwardt. Weisfeiler-lehman graph kernels. *Journal of Machine Learning Research (JMLR)*, 12:2539–2561, 2011.

[27] K. Shin, M. Hammoud, E. Lee, J. Oh, and C. Faloutsos. Tri-fly: Distributed estimation of global and local triangle counts in graph streams. In *PAKDD*, pages 651–663, 2018.

[28] J. Tariq, M. Ahmad, I. Khan, and M. Shabbir. Scalable approximation algorithm for network immunization. In *21st Pacific Asia Conference on Information Systems, PACIS*, page 200, 2017.

[29] A. Tsitsulin, D. Mottin, P. Karras, A. Bronstein, and E. Müller. Netlsd: Hearing the shape of a graph. In *Intern. Conf. on Knowledge Discovery and Data Mining (SIGKDD)*, pages 2347–2356, 2018.

[30] A. Ullah, S. Ali, I. Khan, M.A. Khan, and S. Faizullah. Effect of analysis window and feature selection on classification of hand movements using emg signal. In *Proceedings of SAI Intelligent Systems Conference*, pages 400–415, 2020.

[31] S. Verma and Z. Zhang. Hunt for the unique, stable, sparse and fast feature learning on graphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 88–98, 2017.

[32] K. Xu, W. Hu, J. Leskovec, and S. Jegelka. How powerful are graph neural networks? In *Intern. Conf. on Learning Representations (ICLR)*, 2019.

[33] D. Zhang, J. Yin, X. Zhu, and C. Zhang. Network representation learning: A survey. *IEEE transactions on Big Data*, 6(1):3–28, 2020.

[34] D. Zhu, X. Dai, K. Yang, J. Chen, and Y. He. Pcane: preserving context attributes for network embedding. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, pages 156–168, 2019.