Differentiable Unsupervised Feature Selection based on a Gated Laplacian

Ofir Lindenbaum, Uri Shaham, Erez Peterfreund, Jonathan Svirsky, Nicolas Casey, Yuval Kluger

November 2021

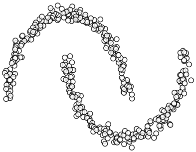
Unsupervised Learning

Find structures in dataset $\tilde{X} \quad \mathrm{R}^{N \times D_c}$ (**no label information**)

Unsupervised Learning

Find structures in dataset $\tilde{X}$ $\quad$ $R^{N \times D_c}$ (**no label information**)

- Clustering



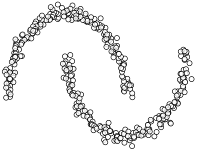
Two-moons

Find subsets
$C_1, ..., C_k$ $\quad$ $\tilde{X}$ so that
points within C_i are
"similar"

Unsupervised Learning

Find structures in dataset $\tilde{X}$ $\quad R^{N \times D_c}$ (**no label information**)

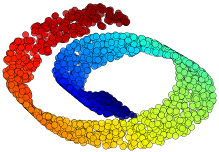
- Clustering



Two-moons

Find subsets
$C_1, ..., C_k \quad \tilde{X}$ so that
points within C_i are
"similar"

- Manifold Learning



Swiss-roll

Embed $\tilde{X}$ into a lower
dimension without loosing
much information

Unsupervised Learning: Nuisance Variables

Consider an **informative** data $\tilde{X}$ concatenated with nuisance variables $X = \tilde{X}|\xi_1|\ldots|\xi_{D_n}{}_{N\times(D_c+D_n)}$, where $\xi_{i,j}$ $N(0,1)$ and $D = D_c + D_n$

Unsupervised Learning: Nuisance Variables

Consider an **informative** data $\tilde{X}$ concatenated with nuisance variables $X = \tilde{X}|\xi_1|...|\xi_{D_n}$ $_{N\times(D_c+D_n)}$, where $\xi_{i,j}$ $N(0,1)$ and $D = D_c + D_n$

- Spectral Clustering (Ng. et al.)

Cluster assignments

Unsupervised Learning: Nuisance Variables

Consider an **informative** data $\tilde{X}$ concatenated with nuisance variables $X = \tilde{X}|\xi_1|...|\xi_{D_n} \quad {}_{N \times (D_c + D_n)}$, where $\xi_{i,j} \quad N(0,1)$ and $D = D_c + D_n$

- Spectral Clustering (Ng. et al.)

Cluster assignments

Nuisance variables

Unsupervised Learning: Nuisance Variables

Consider an **informative** data $\tilde{X}$ concatenated with nuisance variables $X = \left(\tilde{X}|\xi_1|...|\xi_{D_n}\right)_{N\times(D_c+D_n)}$, where $\xi_{i,j} \sim N(0,1)$ and $D = D_c + D_n$

- Spectral Clustering (Ng. et al.)

Cluster assignments

Nuisance variables

Cluster assignments

Unsupervised Learning: Nuisance Variables

Consider an **informative** data $\tilde{X}$ concatenated with nuisance variables $X = \tilde{X}|\xi_1|\ldots|\xi_{D_n}\ {}_{N\times(D_c+D_n)}$, where $\xi_{i,j}\quad N(0,1)$ and $D = D_c + D_n$

- Spectral Clustering (Ng. et al.)

Cluster assignments Nuisance variables Cluster assignments

- ISOMAP (Tenenbaum et al.)

Embedding

Unsupervised Learning: Nuisance Variables

Consider an informative data $\tilde{X}$ concatenated with nuisance variables $X = \tilde{X} | {}_1 | \dots | {}_{D_n}$ $N \times (D_c + D_n)$, where ${}_{i;j}$ $N(0;1)$ and $D = D_c + D_n$

Spectral Clustering (Ng. et al.)

Cluster assignments Nuisance variables Cluster assignments

ISOMAP (Tenenbaum et al.)

Embedding Nuisance variables

Unsupervised Learning: Nuisance Variables

Consider an informative data $\tilde{X}$ concatenated with nuisance variables $X = \left[\tilde{X} \,|\, _1 | \dots | _{D_n} \right] \; _{N \quad (D_c + D_n)}$, where $_{i;j} \quad N(0;1)$ and $D = D_c + D_n$

Spectral Clustering (Ng. et al.)

Cluster assignments Nuisance variables Cluster assignments

ISOMAP (Tenenbaum et al.)

Embedding Nuisance variables Embedding

Unsupervised Feature Selection: Motivation

Goal (informal)

Find a subset of informative variables $S \subseteq \{1, \ldots, D\}$ to improve clustering or manifold learning

Unsupervised Feature Selection: Motivation

Goal (informal)

Find a subset of informative variables $S \subseteq \{1, \ldots, D\}$ to improve clustering or manifold learning

When should we use feature selection?

Unsupervised Feature Selection: Motivation

Goal (informal)

Find a subset of informative variables $S \subseteq \{1, \ldots, D\}$ to improve clustering or manifold learning

When should we use feature selection?

- # of variables exceeds the # of measurements ($D > N$)

Unsupervised Feature Selection: Motivation

Goal (informal)

Find a subset of informative variables $S \subseteq \{1, \ldots, D\}$ to improve clustering or manifold learning

When should we use feature selection?

- \# of variables exceeds the \# of measurements ($D > N$)
- Some of the variables are nuisance (i.e. noisy and information-poor

Unsupervised Feature Selection: Motivation

Goal (informal)

Find a subset of informative variables $S \subseteq \{1, \ldots, D\}$ to improve clustering or manifold learning

When should we use feature selection?

- # of variables exceeds the # of measurements ($D \gg N$)
- Some of the variables are nuisance (i.e. noisy and information-poor

Bio-informatics:

N - individuals, D - genes
Cluster traits/conditions

Unsupervised Feature Selection: Motivation

Goal (informal)

Find a subset of informative variables $S \subseteq \{1, \ldots, D\}$ to improve clustering or manifold learning

When should we use feature selection?

- \# of variables exceeds the # of measurements ($D > N$)
- Some of the variables are nuisance (i.e. noisy and information-poor

Bio-informatics:

N - individuals, D - genes
Cluster traits/conditions

)

Unsupervised Feature Selection: Motivation

Goal (informal)

Find a subset of informative variables $S \subset \{1, \ldots, D\}$ to improve clustering or manifold learning

When should we use feature selection?

- # of variables exceeds the # of measurements ($D > N$)
- Some of the variables are nuisance (i.e. noisy and information-poor

Bio-informatics:

N - individuals, D - genes
Cluster traits/conditions

)

The idea: use "smoothness" to identify informative variables

Unsupervised Learning: Kernel Methods

The graph Laplacian[1] is a useful tool for unsupervised learning

[1] Ng et al. (2001), Belkin et al. (2003)

Unsupervised Learning: Kernel Methods

The graph Laplacian[1] is a useful tool for unsupervised learning

Given measurements $\{x_n\}_{n=1}^{N}$

[1] Ng et al. (2001), Belkin et al. (2003)

Unsupervised Learning: Kernel Methods

The graph Laplacian[1] is a useful tool for unsupervised learning

Given measurements $\{x_n\}_{n=1}^{N}$

- Compute Gaussian kernel $K_{i,j} = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right)$

[1] Ng et al. (2001), Belkin et al. (2003)

Unsupervised Learning: Kernel Methods

The graph Laplacian[1] is a useful tool for unsupervised learning

Given measurements $\{x_n\}_{n=1}^{N}$

- Compute Gaussian kernel $K_{i,j} = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right)$
- Compute $L = S - K$, where $S_{i,i} = \sum_j K_{i,j}$

[1] Ng et al. (2001), Belkin et al. (2003)

Unsupervised Learning: Kernel Methods

The graph Laplacian[1] is a useful tool for unsupervised learning

Given measurements $\{x_n\}_{n=1}^{N}$

- Compute Gaussian kernel $K_{i;j} = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right)$
- Compute $L = S - K$, where $S_{i;i} = \sum_j K_{i;j}$
- Compute eigen-pairs $\Lambda = \{\lambda_0, \lambda_1, \ldots, \lambda_{N-1}\}$ and $\{\phi_n\}_{n=0}^{N-1}$

[1] Ng et al. (2001), Belkin et al. (2003)

Unsupervised Learning: Kernel Methods

The graph Laplacian[1] is a useful tool for unsupervised learning

Given measurements $\{x_n\}_{n=1}^{N}$

- Compute Gaussian kernel $K_{i,j} = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right)$
- Compute $L = S - K$, where $S_{i,i} = \sum_j K_{i,j}$
- Compute eigen-pairs $\Lambda = \lambda_0 \le \lambda_1 \le \dots \le \lambda_{N-1}$ and $\{\phi_n\}_{n=0}^{N-1}$

0 1 2 3 4

[1] Ng et al. (2001), Belkin et al. (2003)

Unsupervised Feature Selection

"Low-frequency" features could be identi ed using the Laplacian-Score[2]

[2] Niyogi et al. (2006)

Unsupervised Feature Selection

"Low-frequency" features could be identified using the Laplacian-Score[2]

Normalize each feature $f_d = \frac{f_d}{\|f_d\|_2}$, where $f_d = (x_1^{(d)}, \ldots, x_N^{(d)})^\top$

[2] Niyogi et al. (2006)

Unsupervised Feature Selection

"Low-frequency" features could be identified using the Laplacian-Score[2]

- Normalize each feature $\tilde{f}_d = \frac{f_d}{\|f_d\|_2}$, where $f_d = (x_1^{(d)}, \dots, x_N^{(d)})^\top$
- Compute the Rayleigh quotient

$$\text{Laplacian-Score}(d) \triangleq \tilde{f}_d^\top L \tilde{f}_d$$

[2] Niyogi et al. (2006)

Unsupervised Feature Selection

"Low-frequency" features could be identi ed using the Laplacian-Score[2]

- Normalize each feature $f_d = \frac{f_d}{\|f_d\|_2}$, where $f_d = (x_1^{(d)}, \ldots, x_N^{(d)})^\top$
- Compute the Rayleigh quotient

$$\text{Laplacian-Score}(d) \triangleq f_d^\top L f_d = \sum_{n=0}^{N-1} {}_n \langle {}_n, f_d \rangle^2,$$

where $L = \sum_{n=0}^{N-1} {}_n \, {}_n \, {}_n^\top$ is the eigen-decomposition of L

[2] Niyogi et al. (2006)

Unsupervised Feature Selection

"Low-frequency" features could be identified using the Laplacian-Score[2]

- Normalize each feature $\tilde{f}_d = \frac{f_d}{\|f_d\|_2}$, where $f_d = (x_1^{(d)}, \ldots, x_N^{(d)})^\top$
- Compute the Rayleigh quotient

$$\text{Laplacian-Score}(d) \triangleq \tilde{f}_d^\top L \tilde{f}_d = \sum_{n=0}^{N-1} \lambda_n \langle \xi_n, \tilde{f}_d \rangle^2,$$

where $L = \sum_{n=0}^{N-1} \lambda_n \xi_n \xi_n^\top$ is the eigen-decomposition of L

Feature is correlated with low frequency eigenvectors

---

[2] Niyogi et al. (2006)

Unsupervised Feature Selection

"Low-frequency" features could be identified using the Laplacian-Score[2]

- Normalize each feature $\tilde{f}_d = \frac{f_d}{\|f_d\|_2}$, where $f_d = (x_1^{(d)}, \ldots, x_N^{(d)})^\top$
- Compute the Rayleigh quotient

$$\text{Laplacian-Score}(d) \triangleq \tilde{f}_d^\top L \tilde{f}_d = \sum_{n=0}^{N-1} \lambda_n \langle \xi_n, \tilde{f}_d \rangle^2,$$

where $L = \sum_{n=0}^{N-1} \lambda_n \xi_n \xi_n^\top$ is the eigen-decomposition of L

Feature is correlated with low frequency eigenvectors $\Rightarrow$ Weighted by small eigenvalues

[2] Niyogi et al. (2006)

Unsupervised Feature Selection

"Low-frequency" features could be identified using the Laplacian-Score[2]

- Normalize each feature $\tilde{f}_d = \frac{f_d}{\|f_d\|_2}$, where $f_d = (x_1^{(d)}, \ldots, x_N^{(d)})^\top$
- Compute the Rayleigh quotient

$$\text{Laplacian-Score}(d) \triangleq \tilde{f}_d^\top L \tilde{f}_d = \sum_{n=0}^{N-1} \lambda_n \langle \phi_n, \tilde{f}_d \rangle^2,$$

where $L = \sum_{n=0}^{N-1} \lambda_n \phi_n \phi_n^\top$ is the eigen-decomposition of L

Feature is correlated with low frequency eigenvectors	$\Rightarrow$	Weighted by small eigenvalues	$\Rightarrow$	Small Laplacian-Score

[2] Niyogi et al. (2006)

Unsupervised Feature Selection

"Low-frequency" features could be identi ed using the Laplacian-Score[2]

- Normalize each feature $f_d = \frac{f_d}{\|f_d\|_2}$, where $f_d = (x_1^{(d)}, \ldots, x_N^{(d)})^{\top}$
- Compute the Rayleigh quotient

$$\text{Laplacian-Score}(d) \stackrel{4}{=} f_d^{\top} L f_d = \sum_{n=0}^{N-1} {}_n \langle {}_n, f_d \rangle^2,$$

where $L = \sum_{n=0}^{N-1} {}_n \, {}_n \, {}_n^{\top}$ is the eigen-decomposition of L

Feature is correlated with low frequency eigenvectors	)	Weighted by small eigenvalues	)	Small Laplacian-Score

Problem

When many nuisance variable are added they dominate the Laplacian

[2] Niyogi et al. (2006)

Unsupervised Feature Selection

"Low-frequency" features could be identified using the Laplacian-Score[2]

- Normalize each feature $\tilde{f}_d = \frac{f_d}{\|f_d\|_2}$, where $f_d = (x_1^{(d)}, \ldots, x_N^{(d)})^\top$
- Compute the Rayleigh quotient

$$\text{Laplacian-Score}(d) \triangleq \tilde{f}_d^\top L \tilde{f}_d = \sum_{n=0}^{N-1} \lambda_n \langle \psi_n, \tilde{f}_d \rangle^2,$$

where $L = \sum_{n=0}^{N-1} \lambda_n \psi_n \psi_n^\top$ is the eigen-decomposition of L

Feature is correlated with low frequency eigenvectors	⇒	Weighted by small eigenvalues	⇒	Small Laplacian-Score

Problem

When many nuisance variable are added they dominate the Laplacian

- e.g. (# nuisance variables) (cluster separation)^4 → Laplacian "breaks"

[2] Niyogi et al. (2006)

Di erentiable Unsupervised Feature Selection (DUFS)

The idea: "clean" the Laplacian by gating nuisance features

Di erentiable Unsupervised Feature Selection (DUFS)

The idea: "clean" the Laplacian by gating nuisance features

Filter features using stochastic gates

Di erentiable Unsupervised Feature Selection (DUFS)

The idea: "clean" the Laplacian by gating nuisance features

Filter features using stochastic gates

De ne the gated measurement matrix

$$X = \left[f_1 z_1 \quad f_2 z_2 \quad \dots \quad f_D z_D \right]$$

Di erentiable Unsupervised Feature Selection (DUFS)

The idea: "clean" the Laplacian by gating nuisance features

Filter features using stochastic gates

De ne the gated measurement matrix
$X = f_1 \; z_1 \quad f_2 \; z_2 \quad \ldots \quad f_D \; z_D$

Compute the gated di usion operator $P_x = S_x^{1} K_x$
K_x and S_x are the Kernel and degree matrices

Differentiable Unsupervised Feature Selection (DUFS)

The idea: "clean" the Laplacian by gating nuisance features

Filter features using stochastic gates

Define the gated measurement matrix
$\tilde{X} = \left[f_1 \odot z_1 \quad f_2 \odot z_2 \quad \dots \quad f_D \odot z_D \right]$

Compute the gated diffusion operator $P_{\tilde{x}} = S_{\tilde{x}}^{-1} K_{\tilde{x}}$
$K_{\tilde{x}}$ and $S_{\tilde{x}}$ are the Kernel and degree matrices

Identify smooth features by minimizing

$$L(\ ) := \underbrace{\mathrm{Tr}\left(\tilde{X}^T P_{\tilde{x}} \tilde{X}\right)}_{\text{Smoothness}} + \underbrace{\ \mathbb{E}_z \|z\|_0}_{\text{Regularization}}$$

Gaussian STochastic Gates (STG)

The idea: use a truncated Gaussian to relax the Bernoulli distribution

Gaussian STochastic Gates (STG)

The idea: use a truncated Gaussian to relax the Bernoulli distribution

- Draw from a Gaussian $N(0; 0{:}5)$, shift by $= 0{:}5$

Gaussian STochastic Gates (STG)

The idea: use a truncated Gaussian to relax the Bernoulli distribution

- Draw from a Gaussian $\sim N(0, 0.5)$, shift by $\mu = 0.5$

Gaussian STochastic Gates (STG)

The idea: use a truncated Gaussian to relax the Bernoulli distribution

Truncate into $[0;1]$

Gaussian STochastic Gates (STG)

The idea: use a truncated Gaussian to relax the Bernoulli distribution

De ne the STochastic Gate (STG), denoted by z

Gaussian STochastic Gates (STG)

The idea: use a truncated Gaussian to relax the Bernoulli distribution

Each gate is controlled by a trainable parameter

Gaussian STochastic Gates (STG)

The idea: use a truncated Gaussian to relax the Bernoulli distribution

Each gate is controlled by a trainable parameter

Learn model and gate parameters by minimizing

$$b = \arg\min_{2\,\mathbb{R}^D} \; \mathrm{Tr}\left(X^T P_x X\right) + \mathbb{E}_z\|z\|_0$$

$$+ \sum_{d=1}^{D} \left(\frac{d}{0.5} \right)$$

Gaussian STochastic Gates (STG)

$\boxed{\text{The idea: use a truncated Gaussian to relax the Bernoulli distribution}}$

Each gate is controlled by a trainable parameter

Learn model and gate parameters by minimizing

$$b = \arg\min_{2\,R^D} \mathrm{Tr}\; X^T P_x X \;+\; E_z \|z\|_0$$

$$+\; \sum_{d=1}^{D} \left(\frac{d}{0{:}5} \right)$$

Gaussian STochastic Gates (STG)

$\boxed{\text{The idea: use a truncated Gaussian to relax the Bernoulli distribution}}$

Each gate is controlled by a trainable parameter

Learn model and gate parameters by minimizing

$$b = \arg\min_{2\,R^D} \; \mathrm{Tr}\; X^{T} P_{x} X \;+\; E_z \|z\|_0$$

$$+ \; \sum_{d=1}^{D} \left(\frac{d}{0.5} \right)$$

Gaussian STochastic Gates (STG)

$\boxed{\text{The idea: use a truncated Gaussian to relax the Bernoulli distribution}}$

Each gate is controlled by a trainable parameter

Learn model and gate parameters by minimizing

$$b = \arg\min_{2\,R^D} \ \mathrm{Tr}\ X^T P_x X \ + \ \mathbb{E}_z\|z\|_0$$

$$+ \sum_{d=1}^{D} \left(\frac{d}{0.5} \right)$$

Gaussian STochastic Gates (STG)

$\boxed{\text{The idea: use a truncated Gaussian to relax the Bernoulli distribution}}$

Each gate is controlled by a trainable parameter

Learn model and gate parameters by minimizing

$$b = \arg\min_{\in R^D} \mathrm{Tr}\, X^T P_x X + E_z\|z\|_0 + \sum_{d=1}^{D} \left(\frac{d}{0.5}\right)$$

Gaussian STochastic Gates (STG)

The idea: use a truncated Gaussian to relax the Bernoulli distribution

Each gate is controlled by a trainable parameter

Learn model and gate parameters by minimizing

$$b = \arg\min_{2 R^D} \mathrm{Tr}\; X^T P_x X + \; E_z \|z\|_0 \;+\; \sum_{d=1}^{D} \left(\frac{d}{0.5} \right)$$

Results: Noisy Two-moons

Consider an informative data $\tilde{X}$ concatenated with nuisance variables $X = [\tilde{X} | \epsilon_1 | \ldots | \epsilon_{D_n}] \in \mathbb{R}^{N \times (D)}$, where $D = D_c + D_n$, and $\epsilon_{i,j} \sim N(0, 1)$

Results: Noisy Two-moons

Consider an informative data $\tilde{X}$ concatenated with nuisance variables $X = \tilde{X} | {}_1 | \dots | {}_{D_n} \quad N \times (D)$, where $D = D_c + D_n$, and ${}_{i;j} \quad N(0; 1)$

DUFS performance for different number of nuisance variables (d)

Results: MNIST

Use digits of 3s and 8s from noisy MNIST

Results: MNIST

Use digits of 3s and 8s from noisy MNIST

Results: MNIST

Selected features and clustering accuaracy

Results: MNIST

Selected features and clustering accuaracy

Results: MNIST

Selected features and clustering accuaracy

Results: MNIST

Selected features and clustering accuaracy

Results: Real Data

K-means accuracy on benchmark datasets[3] (# of selected features)

Datasets	LS	MCFS	NDFS	LLCFS	SRCFS	CAE	**DUFS**	All	Dim/Samples/Classes
GISETTE	75.8 (50)	56.5 (50)	69.3 (250)	72.5 (50)	68.5 (50)	77.3 (250)	**99.5 (50)**	74.4	4955 / 6000 / 2
PIX10	76.6 (150)	75.9 (200)	76.7 (200)	69.1 (300)	76.0 (300)	**94.1 (250)**	88.4 (50)	74.3	10000 / 100 / 10
COIL20	55.2 (250)	59.7 (250)	60.1 (300)	48.1 (300)	59.9 (300)	65.6 (200)	**65.8 (250)**	53.6	1024 / 1444 / 20
Yale	42.7 (300)	41.7 (300)	42.5 (300)	42.6 (300)	46.3 (250)	45.4 (250)	**47.9 (200)**	38.3	1024 / 165 / 15
TOX-171	47.5 (200)	42.5 (100)	46.1 (100)	46.7 (250)	45.8 (150)	44.4 (150)	**49.1 (50)**	41.5	2000 / 62 / 4
ALLAML	73.2 (150)	68.4 (100)	69.4 (100)	**77.8 (50)**	67.7 (250)	72.2 (200)	74.5 (100)	67.3	7192 / 72 / 2
PROSTATE	57.5 (300)	57.3 (300)	58.3 (100)	57.8 (50)	60.6 (50)	56.9 (250)	**64.7 (150)**	58.1	5966 / 102 / 2
RCV1	54.9 (300)	50.1 (150)	55.1 (150)	55.0 (300)	53.7 (300)	54.9 (300)	**60.2 (300)**	50.0	47236 / 21232 / 2
SRBCT	41.1(300)	43.7(250)	41.0(50)	34.6(150)	33.49(50)	**62.6 (200)**	51.7 (50)	39.6	2308 / 83 / 4
BIASE	83.8 (200)	95.5 (300)	100 (100)	52.2 (300)	50.8 (50)	85.1 (250)	**100 (50)**	41.8	25683 / 56 / 4
INTESTINE	43.2 (300)	48.2 (300)	42.3 (100)	63.3 (200)	58.1 (300)	51.9 (50)	**71.9 (250)**	54.8	3775 / 238 / 13
FAN	42.9 (150)	45.5 (150)	48.8 (100)	29.0 (50)	29.0 (100)	35.2 (300)	**49.0 (50)**	37.5	25683 / 56 / 8
POLLEN	46.9 (150)	**66.5 (300)**	48.9 (50)	35.0 (100)	34.9 (300)	58.0 (250)	60.2 (50)	54.9	21810 / 301 / 4
Median rank	4	6	4	4	5	3	**1**		
Mean rank	4.1	6	3.9	4.6	4.7	3.4	**1.3**		

[3] https://jundongl.github.io/scikit-feature/

Conclusion and Future Work

- "Cleaning" the Laplacian prior to calculation of the LS is cruical
- DUFS is also applicable for Manifold Learning
- Potential applications in computational biology, medicine
- Extending the method to handle correlated features
- Code available at https://github.com/Ofirlin/DUFS