Autoencoding Random Forests

Binh Duc Vu[1*]
binh.vu@kcl.ac.uk

Jan Kapar[2,3*]
kapar@leibniz-bips.de

Marvin Wright[2,3]
wright@leibniz-bips.de

David S. Watson[1]
david.watson@kcl.ac.uk

[1]King's College London
[2]Leibniz Institute for Prevention Research and Epidemiology – BIPS
[3]University of Bremen

[*]Equal contribution

NeurIPS 2025
November 20, 2025

Motivation - why autoencoding with random forests?

- Deep learning is state-of-the-art in representation learning, but:
  - Struggles with mixed tabular data
  - Data hungry
  - Tuning-intense
  - Computationally heavy

Motivation - why autoencoding with random forests?

- Deep learning is state-of-the-art in representation learning, but:
  - Struggles with mixed tabular data
  - Data hungry
  - Tuning-intense
  - Computationally heavy
- Random forests:
  - Natural handling of mixed tabular data
  - Good performance on small data
  - Robust off-the-shelf
  - Fast

Motivation - why autoencoding with random forests?

- Deep learning is state-of-the-art in representation learning, but:
  - Struggles with mixed tabular data
  - Data hungry
  - Tuning-intense
  - Computationally heavy
- Random forests:
  - Natural handling of mixed tabular data
  - Good performance on small data
  - Robust off-the-shelf
  - Fast

$\Rightarrow$ Can random forests learn useful representations and reconstruct data?

Primary contributions

1. Introduction and analysis of **(Breiman) random forest kernel**

Primary contributions

1. Introduction and analysis of **(Breiman) random forest kernel**
2. **Encoding**: **Spectral embeddings** via diffusion maps based on RF kernel similarities

Primary contributions

1. Introduction and analysis of **(Breiman) random forest kernel**
2. **Encoding**: **Spectral embeddings** via diffusion maps based on RF kernel similarities
3. **Decoding**: Introduction and theoretical analysis of **exact** and **approximate methods**

Primary contributions

1. Introduction and analysis of **(Breiman) random forest kernel**
2. **Encoding**: **Spectral embeddings** via diffusion maps based on RF kernel similarities
3. **Decoding**: Introduction and theoretical analysis of **exact** and **approximate methods**
4. **Experiments**: **Competitive results** for data **visualization**, **compression**, **clustering** and **denoising**

(Breiman) random forest kernel

- Decision tree kernel for tree b:

$$k^{(b)}(\mathbf{x}, \mathbf{x}') = \mathbb{1}_{\{\text{leaf}_b(\mathbf{x})=\text{leaf}_b(\mathbf{x}')\}}(\mathbf{x}, \mathbf{x}')$$

(Breiman) random forest kernel

- Decision tree kernel for tree b:

$$k^{(b)}(\mathbf{x}, \mathbf{x}') = \mathbb{1}_{\{\text{leaf}_b(\mathbf{x})=\text{leaf}_b(\mathbf{x}')\}}(\mathbf{x}, \mathbf{x}')$$

- **(Breiman) random forest kernel**: Normalized average of tree kernels

$$k_n^{\text{RF}}(\mathbf{x}, \mathbf{x}') = \frac{1}{B} \sum_{b=1}^{B} \frac{k^{(b)}(\mathbf{x}, \mathbf{x}')}{\sum_{i=1}^{n} k^{(b)}(\mathbf{x}, \mathbf{x}_i)}$$

(Breiman) random forest kernel

- Decision tree kernel for tree b:

$$k^{(b)}(\mathbf{x}, \mathbf{x}') = \mathbb{1}_{\{\text{leaf}_b(\mathbf{x})=\text{leaf}_b(\mathbf{x}')\}}(\mathbf{x}, \mathbf{x}')$$

- **(Breiman) random forest kernel**: Normalized average of tree kernels

$$k_n^{\text{RF}}(\mathbf{x}, \mathbf{x}') = \frac{1}{B} \sum_{b=1}^{B} \frac{k^{(b)}(\mathbf{x}, \mathbf{x}')}{\sum_{i=1}^{n} k^{(b)}(\mathbf{x}, \mathbf{x}_i)}$$

(Breiman) random forest kernel

- Decision tree kernel for tree b:

$$k^{(b)}(\mathbf{x}, \mathbf{x}') = \mathbb{1}_{\{\text{leaf}_b(\mathbf{x})=\text{leaf}_b(\mathbf{x}')\}}(\mathbf{x}, \mathbf{x}')$$

- **(Breiman) random forest kernel**: Normalized average of tree kernels

$$k_n^{\text{RF}}(\mathbf{x}, \mathbf{x}') = \frac{1}{B}\sum_{b=1}^{B} \frac{k^{(b)}(\mathbf{x}, \mathbf{x}')}{\sum_{i=1}^{n} k^{(b)}(\mathbf{x}, \mathbf{x}_i)}$$

- Properties:
  - **Positive semi-definite**
  - Kernel matrix $\mathbf{K} \coloneqq \left(k_n^{\text{RF}}(\mathbf{x}_i, \mathbf{x}_j)\right)_{i,j=1,\dots,n} \in [0,1]^{n\times n}$ doubly stochastic
  - Asymptotically **universal**
  - Asymptotically **characteristic**

Random forest autoencoder (RFAE)

- **Encoding:**

Random forest autoencoder (RFAE)

- **Encoding:**
  1. Train **random forest** (supervised/unsupervised)

Random forest autoencoder (RFAE)

- **Encoding:**
  1. Train **random forest** (supervised/unsupervised)
  2. Calculate **kernel matrix** $\mathbf{K}$ for training data

Random forest autoencoder (RFAE)

- **Encoding:**
  1. Train **random forest** (supervised/unsupervised)
  2. Calculate **kernel matrix** $\mathbf{K}$ for training data
  3. Calculate **eigen-decomposition** $\mathbf{K} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T$

Random forest autoencoder (RFAE)

- **Encoding:**
  1. Train **random forest** (supervised/unsupervised)
  2. Calculate **kernel matrix** $\mathbf{K}$ for training data
  3. Calculate **eigen-decomposition** $\mathbf{K} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T$
  4. **Training data embeddings:** Apply **diffusion map** at desired timestep t and latent dimension $d_{\mathbf{Z}}$ to calculate spectral embedding $\mathbf{Z} = \sqrt{n}\mathbf{V}_{[d_Z]}\mathbf{\Lambda}^t_{[d_Z]}$

Random forest autoencoder (RFAE)

- **Encoding:**
  1. Train **random forest** (supervised/unsupervised)
  2. Calculate **kernel matrix** $\mathbf{K}$ for training data
  3. Calculate **eigen-decomposition** $\mathbf{K} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T$
  4. **Training data embeddings:** Apply **diffusion map** at desired timestep t and latent dimension $d_{\mathbf{Z}}$ to calculate spectral embedding $\mathbf{Z} = \sqrt{n}\mathbf{V}_{[d_z]}\mathbf{\Lambda}^t_{[d_z]}$
  5. **Test data embeddings**: Calculate test-train cross-kernel matrix $\mathbf{K}_0$ and use **Nyström formula** $\mathbf{Z}_0 = \mathbf{K}_0\mathbf{Z}\mathbf{\Lambda}^{-1}_{[d_z]}$

Random forest autoencoder (RFAE)

- **Encoding:**
  1. Train **random forest** (supervised/unsupervised)
  2. Calculate **kernel matrix** $\mathbf{K}$ for training data
  3. Calculate **eigen-decomposition** $\mathbf{K} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T$
  4. **Training data embeddings:** Apply **diffusion map** at desired timestep t and latent dimension $d_{\mathbf{Z}}$ to calculate spectral embedding $\mathbf{Z} = \sqrt{n}\mathbf{V}_{[d_z]}\mathbf{\Lambda}^t_{[d_z]}$
  5. **Test data embeddings**: Calculate test-train cross-kernel matrix $\mathbf{K}_0$ and use **Nyström formula** $\mathbf{Z}_0 = \mathbf{K}_0\mathbf{Z}\mathbf{\Lambda}^{-1}_{[d_z]}$
- **Decoding:**
  - Best-performing in practice: **k-Nearest Neighbors** in latent space
  - Universally consistent, fast, robust

Results

Results

- **Reconstruction:** competitive on MNIST and tabular data.

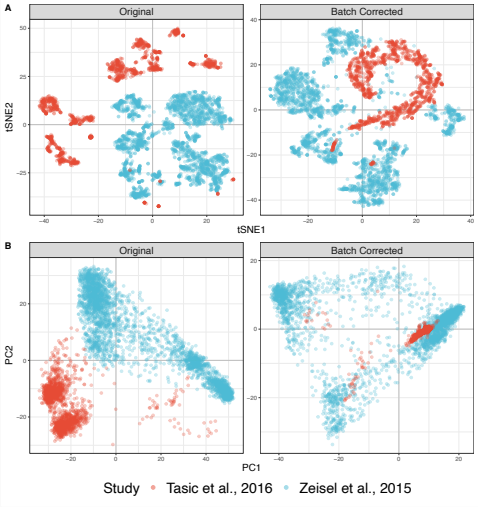
Results

- **Reconstruction:** competitive on MNIST and tabular data.
- **Benchmark:** best performance in 12/20 datasets.

Dataset	RFAE	TVAE	TTVAE	AE	VAE
abalone	**0.167 (0.002)**	0.309 (0.005)	0.260 (0.003)	0.230 (0.025)	0.211 (0.006)
adult	0.326 (0.007)	**0.158 (0.005)**	0.195 (0.007)	0.401 (0.003)	0.391 (0.004)
banknote	**0.100 (0.012)**	0.312 (0.013)	0.276 (0.023)	0.724 (0.023)	0.771 (0.013)
bc	0.333 (0.003)	0.564 (0.003)	0.359 (0.005)	**0.287 (0.008)**	0.578 (0.003)
car	0.320 (0.011)	0.195 (0.014)	**0.107 (0.015)**	0.349 (0.012)	0.313 (0.011)
churn	**0.352 (0.012)**	0.603 (0.011)	0.422 (0.014)	0.861 (0.005)	0.731 (0.006)
credit	**0.315 (0.004)**	0.450 (0.005)	0.375 (0.011)	0.450 (0.005)	0.456 (0.004)
diabetes	**0.479 (0.016)**	0.726 (0.007)	0.643 (0.014)	0.799 (0.011)	0.895 (0.004)
dry_bean	0.137 (0.002)	0.273 (0.002)	0.303 (0.008)	**0.083 (0.014)**	0.206 (0.001)
forestfires	**0.575 (0.008)**	0.804 (0.003)	0.705 (0.008)	0.782 (0.007)	0.790 (0.003)
hd	**0.432 (0.008)**	0.582 (0.003)	0.605 (0.006)	0.892 (0.003)	0.916 (0.002)
king	**0.308 (0.008)**	0.352 (0.006)	0.348 (0.008)	0.377 (0.011)	0.518 (0.004)
marketing	0.292 (0.009)	0.304 (0.005)	**0.259 (0.011)**	0.357 (0.007)	0.372 (0.004)
mushroom	0.083 (0.001)	0.093 (0.003)	**0.011 (0.003)**	0.055 (0.004)	0.035 (0.004)
obesity	**0.227 (0.008)**	0.354 (0.004)	0.299 (0.008)	0.306 (0.009)	0.358 (0.003)
plpn	**0.176 (0.006)**	0.282 (0.006)	0.224 (0.011)	0.384 (0.013)	0.410 (0.009)
spambase	0.558 (0.005)	0.825 (0.002)	0.807 (0.003)	**0.446 (0.010)**	0.784 (0.001)
student	**0.371 (0.002)**	0.424 (0.001)	0.426 (0.004)	0.536 (0.003)	0.551 (0.002)
telco	0.177 (0.003)	0.155 (0.003)	**0.091 (0.007)**	0.128 (0.005)	0.130 (0.005)
wq	**0.240 (0.005)**	0.691 (0.008)	0.759 (0.006)	0.467 (0.019)	0.708 (0.004)
Average Rank	**1.80**	3.38	2.45	3.27	4.10

Results

- **Reconstruction:** competitive on MNIST and tabular data.
- **Benchmark:** best performance in 12/20 datasets.
- **Applications:** denoising scRNA-seq, latent clustering, compression.



Discussion & Outlook

- **Contributions:** Theoretical kernel foundation, practical encoding/decoding, strong empirical results.
- **Advantages:**
  - Works with any RF variant (RF, URF, ARF)
  - No end-to-end training required
- **Limitations:** Computational cost, sensitive to hyperparameters.
- **Next:**
  - Distilled Random Forests
  - Adaptive RF kernels
  - Tree-based generative models (XGBoost, GBM)