SLIPMAP

FAST AND ROBUST MANIFOLD VISUALISATION FOR EXPLAINABLE AI

Anton Björklund, Lauri Seppäläinen, Kai Puolamäki (2024).
Advances in Intelligent Data Analysis XXII, pp. 223–235, LNCS 14642.
DOI: 10.1007/978-3-031-58553-1_18.

https://github.com/edahelsinki/slisemap

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

30th April 2024

These notes contain the spoken parts

SLIPMAP is a supervised dimensionality reduction method, that can be used to explain black box classification or regression models. SLIPMAP is open source and available as a Python library.

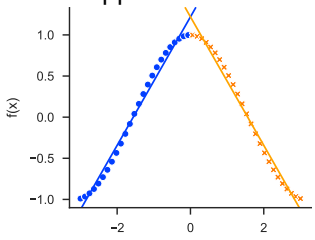
OUTLINE

- **Background**
- SLIPMAP
- Examples
- Conclusions

BACKGROUND

- Local explanations
  - For a selected item
  - Many explanations

Linear approximations as local explanations.

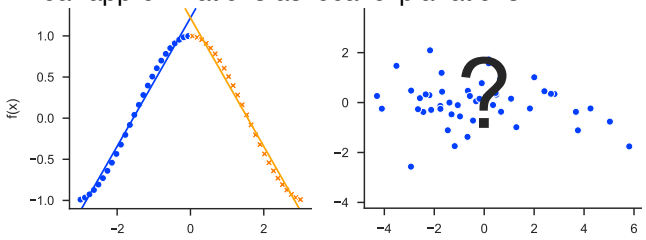


- The goal of explainable artificial intelligence is to extract more information on the predictions of complex black box models than just the output.
- Local explanations give partial views of the black box model, usually centred around a selected data item.
- A common approach is to approximate the complex model with a simple, interpretable model.
- This is demonstrated in the figure where we use linear models to locally approximate a complex function.
- But with every item potentially having a different local explanation, if we want a holistic view we need a way to visualise multiple explanations.

BACKGROUND

- Local explanations
  - For a selected item
  - Many explanations
- Manifold Visualisation
  - 2D embedding
  - Unsupervised

Linear approximations as local explanations.



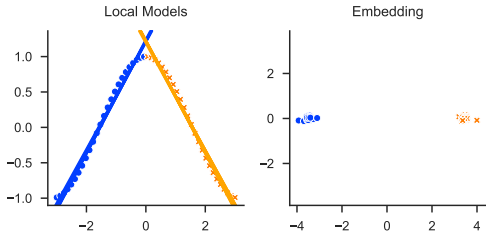
Embedding based on the explanations?

- A common approach for visualising complex datasets is through manifold visualisation methods, such as t-SNE and UMAP.
- They create a low-dimensional embedding to represent high-dimensional datasets.
-
- However, we are not so interested in embeddings based on unsupervised data distances.
- Rather we want to know whether a given local explanation is valid for a given data point.
- But, how would such an embedding look like?

SLISEMAP

“Find an embedding that groups data items with similar local models”.

- Simultaneous optimisation of explanations and embedding.



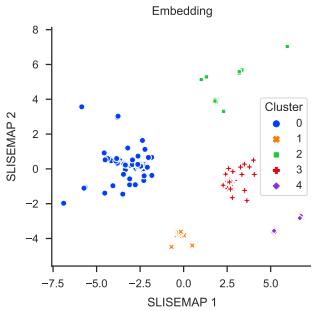
Björklund et al., SLISEMAP: Supervised dimensionality reduction through local explanations, Mach Learn, (2023). DOI: 10.1007/s10994-022-06261-1.

- Our previous method for this idea is called SLISEMAP.
- Where we simultaneously optimise for both the embedding and the local models.
- Such that data items with similar models end up next to each other in the embedding.
-
- And this is the solution that SLISEMAP gives for the previous example.
- There are the two expected groups of local models.
- And the embedding shows the corresponding two groups.
-
- This grouping behaviour also appears in real-world datasets. . .

SLISEMAP ISSUES

- Scales quadratically
- Not a predictive model
- Finds groups in noise

$\Longrightarrow$ New method: SLIPMAP

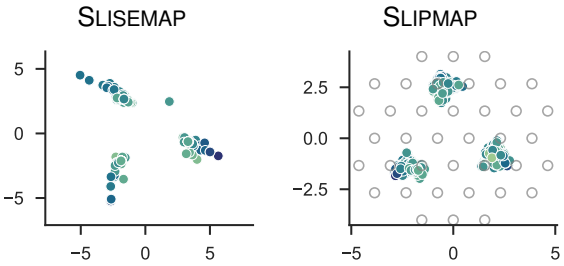


- . . . such as this one about cars.
-
- While SLISEMAP generally works well, there are some issues that could be improved.
- The biggest one being that the time complexity scales quadratically.
- And to add an unseen data point to the embedding requires the target value.
- So we can not use SLISEMAP to predict the target value.
- Finally, we noticed that the SLISEMAP embedding forms groups when given Gaussian noise.
-
- With our new method, called SLIPMAP, we will resolve these issues.

OUTLINE

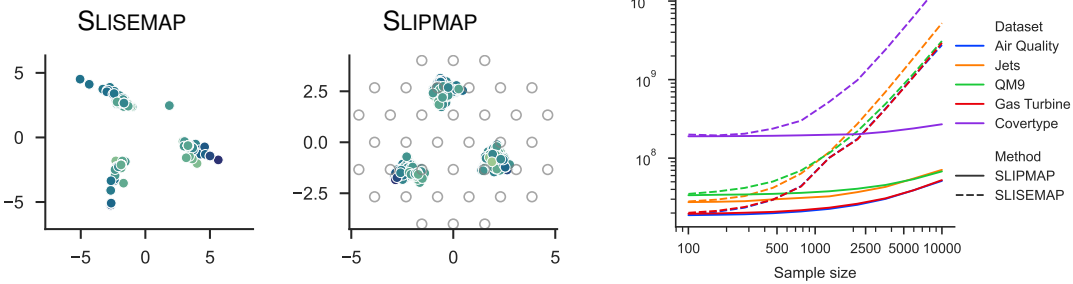
- Background
- **SLIPMAP**
- Examples
- Conclusions

PROBLEM 1: SCALING



- Starting with the scaling.
- In SLISEMAP every data item has its own local model.
- This means we have to compare every data item with every other item.
- With SLIPMAP we don't have individual models, instead we introduce a fixed grid of points, and only the grid points have local models.
- The grid points can be seen in the embedding on the right as the grid of grey circles.

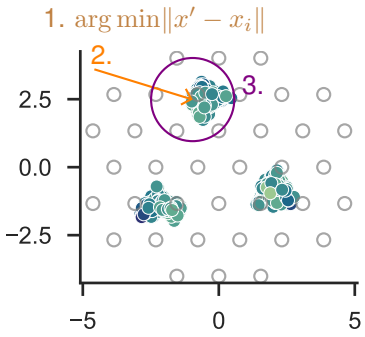
PROBLEM 1: SCALING



- Now we only need to compare the data items to the grid points.
- Which results in a linear scaling, drastically improving the efficiency.

PROBLEM 2: PREDICTIONS

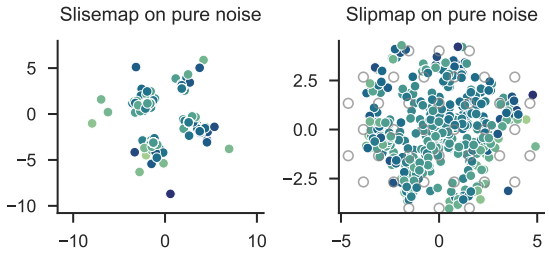
- Procedure:
  1. Find the most similar training data point
  2. Copy the embedding
  3. Find the nearby grid points
  4. Use the local models for prediction
- Could replace the black box model



- For producing predictions we propose the following procedure:
- First we find the nearest neighbour in the training data and copy the embedding.
- Then we collect the nearby grid points.
- Finally we take a weighted average of the predictions from use the local models.
-
- This gives better results than just a nearest neighbour model.
- And the procedure is interpretable, since we “select a local model that works well for similar data”.
- In several cases the predictions are good enough that we could consider replacing the black box model.

PROBLEM 3: NOISE

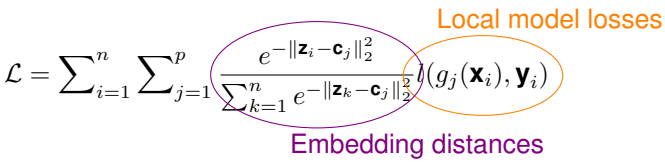
- SLISEMAP: overfits on noise
  - Finds groups in pure noise
- SLIPMAP: noise looks like noise
  - More general local models
  - Squared kernels
    $e^{-\|\cdots\|_2} \to e^{-\|\cdots\|_2^2}$



- As mentioned before, the embedding of SLISEMAP tends to overfit on noise.
- With SLIPMAP noise looks more like noise.
- This is due to the grid points having slightly more general local models.
- And a small tweak to the loss function in the form of squared distances in the kernel.

LOSS FUNCTION

"Multiply embedding weights by local model losses".

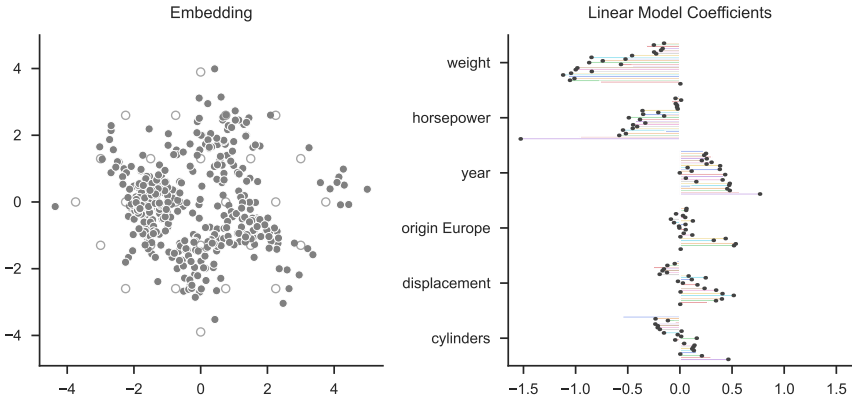


- Speaking of the loss function, this is the core of it.
- We turn distances in the embedding into weights using a softmax kernel and multiply by the losses from the local models.
- This creates a strong connection between the embedding and the explanations.
-
- As default local models we use linear or logistic regression.
- And to optimise the loss we use the LBFGS optimiser combined with a greedy heuristic to escape local optima.

OUTLINE

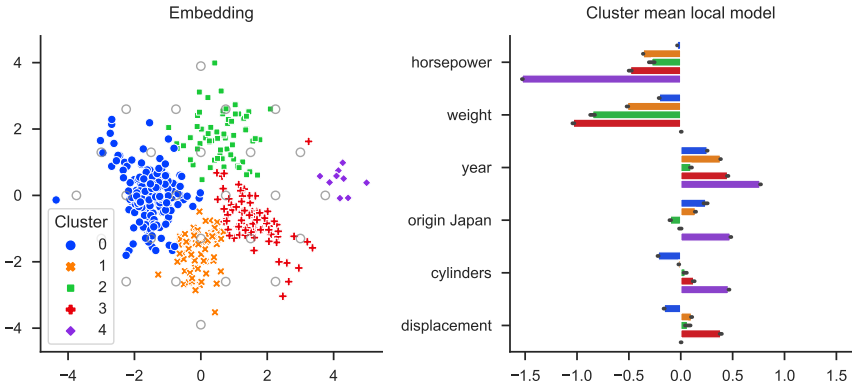
- Background
- SLIPMAP
- **Examples**
- Conclusions

EXAMPLE



- Here we apply SLIPMAP on the same dataset of fuel efficiency in cars.
- On the left you can see the embedding with 30 grid points.
- On the right are the coefficients of the corresponding local models.
- However, one bar per grid point and variable is quite a lot, so to make the visualisation easier to read we, post-hoc, cluster the local models by their coefficients.

EXAMPLE



- Here we see thatthe error bars on the coefficients in the bar plot are small.
- And that the model clusters correspond to contiguous areas in the embedding.

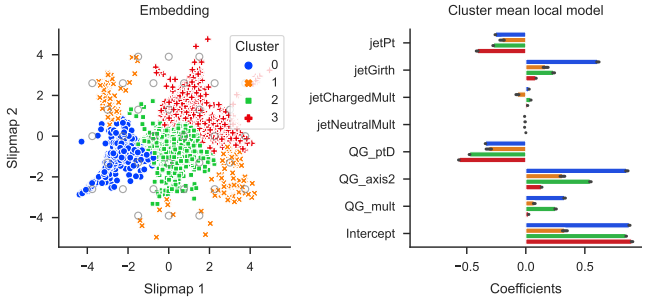
IMPLEMENTATION

Installation

```
pip install slisemap
```

Code

```
from slisemap import Slipmap
# Use Lasso regularisation
sm = Slipmap(X, y, lasso=0.01)
sm.optimise()
sm.plot(clusters=5, bars=6)
```



- SLIPMAP is implemented in Python, using Pytorch for the optimisation.
- If you want to try it, the open source library can be installed via pip.
- And, using SLIPMAP requires only a couple of lines of code.
- Here is another example plot with data from high energy physics.

SUMMARY

SLIPMAP is a supervised manifold visualisation method that embeds data items into a lower-dimensional space such that nearby data items share the same white box model.

A. Björklund, L. Seppäläinen, K. Puolamäki (2024).
SLIPMAP: Fast and Robust Manifold Visualisation for Explainable AI.
Advances in Intelligent Data Analysis XXII, pp. 223–235, LNCS 14642.
DOI: 10.1007/978-3-031-58553-1_18.

https://github.com/edahelsinki/slisemap

- To summarise, SLIPMAP is a supervised manifold visualisation method that simultaneously finds local models and constructs an embedding based on the local models.
- Please read our paper or check out our GitHub repo for further documentation.
- Thank you for your attention.

BONUS: PROBLEM DEFINITION

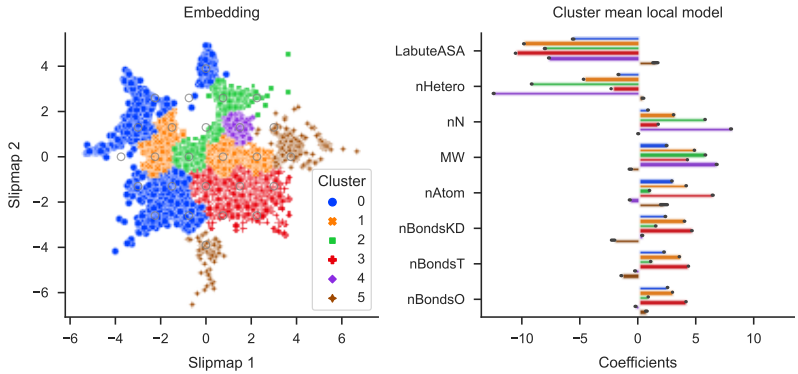
- Given a dataset of n items: $(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n)$.
- And a grid of p prototypes in the embedding: $\mathbf{c}_1, \ldots, \mathbf{c}_p$
- Find the embedding coordinates $\mathbf{z}_1, \ldots, \mathbf{z}_n$ and the local models $g_1, \ldots, g_p$ that minimise:

$$\mathcal{L} = \sum_{i=1}^{n} \sum_{j=1}^{p} \frac{e^{-\|\mathbf{z}_i - \mathbf{c}_j\|_2^2}}{\sum_{k=1}^{n} e^{-\|\mathbf{z}_k - \mathbf{c}_j\|_2^2}} l(g_j(\mathbf{x}_i), \mathbf{y}_i)$$

- Where l is a loss function for the local models.
- Under the constraint $\frac{1}{n} \sum_{i=1}^{n} \sum_{k=1}^{d} \mathbf{z}_{ik}^2 = r^2$.

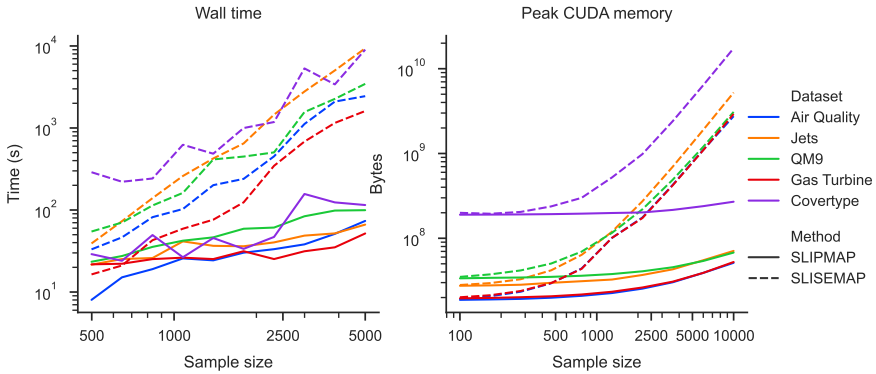
- We express the objective of SLIPMAP as a loss function.
- Where the important parts are:
- The losses from the local models.
- And the distances in the embedding, which we turn into weights using softmax.
- Combined they create a strong connection between the embedding and the explanations.
-
- As default local models we use linear or logistic regression.
- In both cases we also add lasso and ridge regularisation to avoid overfitting.
- Lasso regularisation can cause sparse models, which is important for interpretability.
-
- To optimise the loss we use the LBFGS optimiser combined with a greedy heuristic to escape local optima.

BONUS: ORGANIC MOLECULES



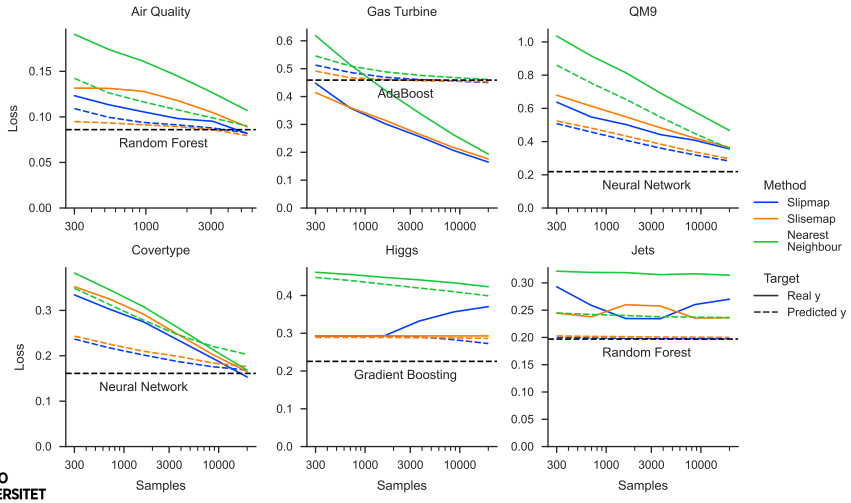
- Here is an example from organic chemistry.

BONUS: SCALING



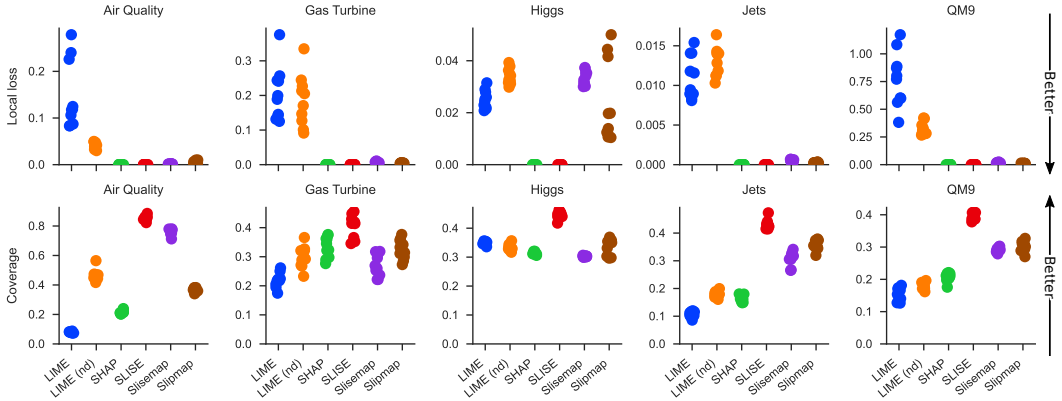
- Here we compare SLIPMAP to SLISEMAP as the size of the data increases.
- Note the logarithmic scales.
- SLISEMAP scales quadratically in both time and memory, while SLIPMAP scales linearly.
- This makes SLIPMAP much faster.
- On a GPU both methods are much faster, but memory becomes the new bottleneck for SLISEMAP.

BONUS: PREDICTIONS



- With SLIPMAP we perform predictions for unseen data by finding and copying the embedding of the nearest training data and taking a weighted average of the nearby local models.
- As a comparison we perform the same procedure for SLISEMAP and just use the nearest neighbour without the local models.
- SLIPMAP generally performs the best.
-
- Learning to approximate the predictions of a complex model often gives better results than training on "raw data".
- And even if the goal was not to compete with the black box models, in some cases we could replace the black box model with a more interpretable procedure without meaningfully affecting the loss.

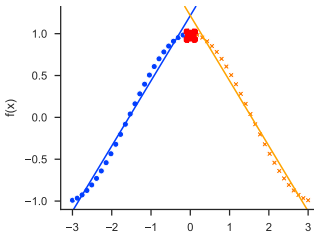
BONUS: XAI COMPARISON



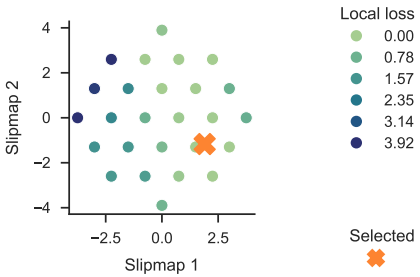
Inline with some other local explanation methods.

- We compare SLIPMAP against some other model-agnostic local explanation methods.
- Using a couple metrics.
- And we find that the local explanations are inline with the other methods.

BONUS: ALTERNATIVE EXPLANATIONS



The choice of local model can be arbitrary.



A data item with multiple viable explanations.

- Finally, most local explanations are not unique (including those of SLIPMAP).
- For example, in the left figure, both linear models suit the highlighted point equally well.
- With SLIPMAP we can find alternative explanations by calculating losses.
- On the right we select one data item and colour all other points based on how well that local model works for the selected item.
- There are natural reasons for this behaviour:
  - Overlapping local models, such as the left example
  - Correlated variables
  - Explanations with different levels of locality
  - etc..
- Further investigation of this phenomenon a direction for future work.