Machine Learning for Many-Body Physics
Homework 2

Thomas Koutsikos

May 2024

1 Identifying phase transitions using t-SNE

The objective of this problem is to use t-distributed stochastic neighbour embedding (t-SNE) to identify phase transitions based on spin configuration data.

As discussed in Lecture 9, t-SNE is a method that takes a dataset of high-dimensional vectors $\{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_m\}$ (with each $\mathbf{x}_i \in \mathbb{R}^N$ and N large) and creates a lower-dimensional representation of the dataset $\{\mathbf{x}'_1, \mathbf{x}'_2, ..., \mathbf{x}'_m\}$. When $\mathbf{x}'_i \in \mathbb{R}^2$ or $\mathbb{R}^3$ then the representation can be visualized.

Here, you will apply t-SNE to the problem of identifying phases of the two-dimensional classical Ising model and the classical Ising gauge theory. Your datasets will consists of spin configurations from these models at various temperatures. You may start from Laurens van der Maaten's Python implementation of t-SNE for the MNIST dataset of handwritten digits (see Reference [1]). You are also encouraged to look at the "Examples" section of this reference to see the results of applying t-SNE to other datasets.

Submit your code as well as a written summary (including relevant plots) of your answers to the following points.

a) Give a brief explanation of why using dimensional reduction to identify phases of matter from spin configurations is classified as unsupervised learning as opposed to supervised learning. How does the data given to the algorithm differ from the data you used for Problem 2 of Homework 1 (more specifically the Ising model dataset for L = 30)?

Dimensional reduction falls under the category of unsupervised learning because we provide no labels during training. For example, in problem 2 of homework 1 we had to provide labels for each configuration specifying if the configuration is in the feromagnetic or the paramagnetic phase, but this is not necessary in the unsupervised learning case.

b) Look through the code from Reference [1]. Within it, there is a function **tsne** that takes arguments **X**, **no dims**, **initial dims** and **perplexity**. Explain the meaning of each of these arguments.

- **X:** The dataset on which we run t-SNE
- **no dims:** The dataset is reduced to **no dims** dimensions after t-SNE.
- **initial dims:** The dataset is initially **initial dims**-dimensional.

  To summarize the meaning of the previous variables using the notation of the introduction to the homework, $\mathbf{X}$ is the dataset $\{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_m\}$, $\mathbf{x}_i \in \mathbb{R}^N$, N is **initial dims** and if we define $\mathbf{x}' \in \mathbb{R}^M$ $M < N$, M is **no dims**.
- **perplexity:** Perplexity can be thought of as the number of effective nearest neighbors during the dimensionality reduction process. The higher the perplexity the more points will end up being in the same neighborhood and the lower the perplexity the less points will end up being in the same neighborhood during the reduction process.

Hint: For the perplexity, you may find it useful to look at Reference [2] and also at the visualizations on this link. https://distill.pub/2016/misread-tsne/

You will be able to solve the remaining questions after we do tutorial 5 on Friday, April 26th.

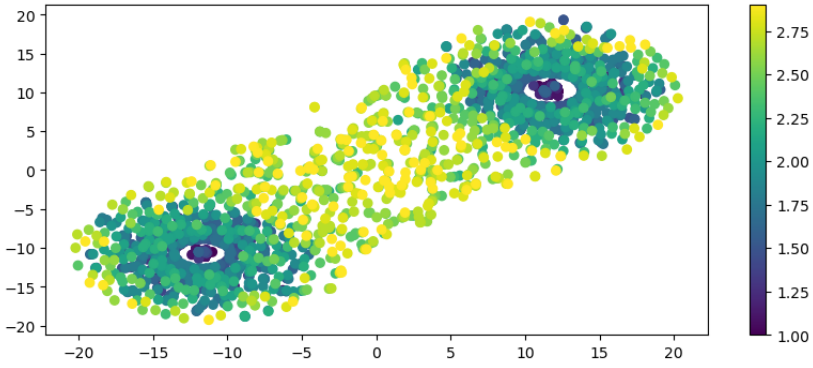
c) Modify the code from Reference [1] such that it applies t-SNE to the datasets from Tutorial 5 for the two-dimensional classical Ising model (from the same GitHub repository). Within your summary, include your results at L = 20, 40 and 80.

Hint: When applying the code from Reference [1] to the datasets from Tutorial 5, you may run into numerical issues due to the variable **sumP** within the function **Hbeta** sometimes becoming very small. If you run into these issues, try replacing the line **sumP = sum(P)** with **sumP = max(sum(P),1e-13)**.

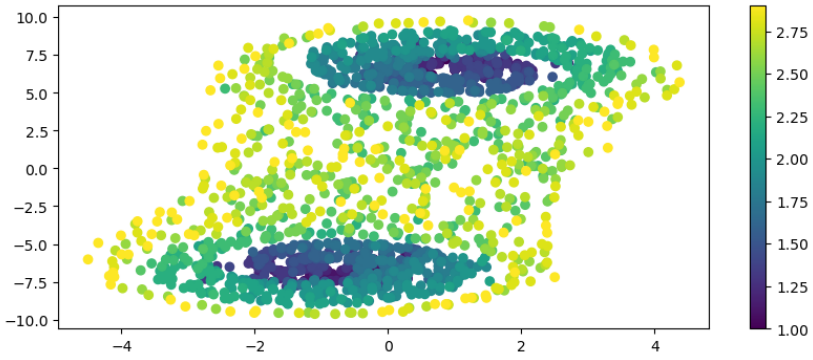
The code was modified and a copy of the notebook is on github. The plots will appear for the subsequent parts of the homework, but something to note was that I also had to regulate the high values of sumP with an additional line after **sumP = max(sum(P),1e-13)**, namely **sumP = min(sumP,1e13)**. This is probably a result of small negative values of D, which lead to positive infinity when exponentiated.

d) Explain your results from part c), including discussion of the following points:

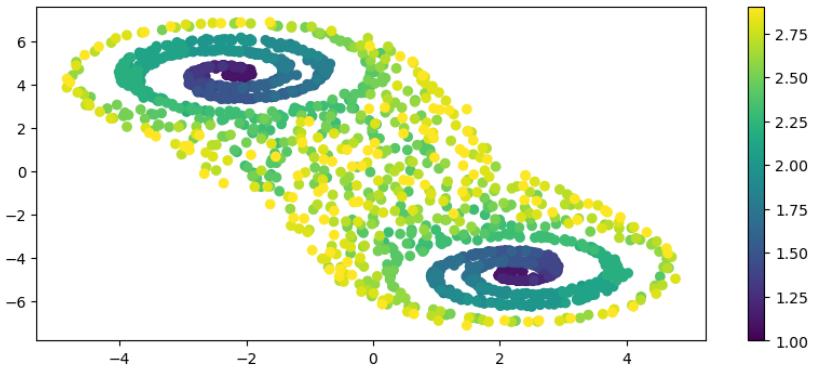
- How do the clustering results from t-SNE change as L increases? Below are the images for the L=20, 40 and 80 classical Ising model.



(a) $L = 20$



(b) $L = 40$



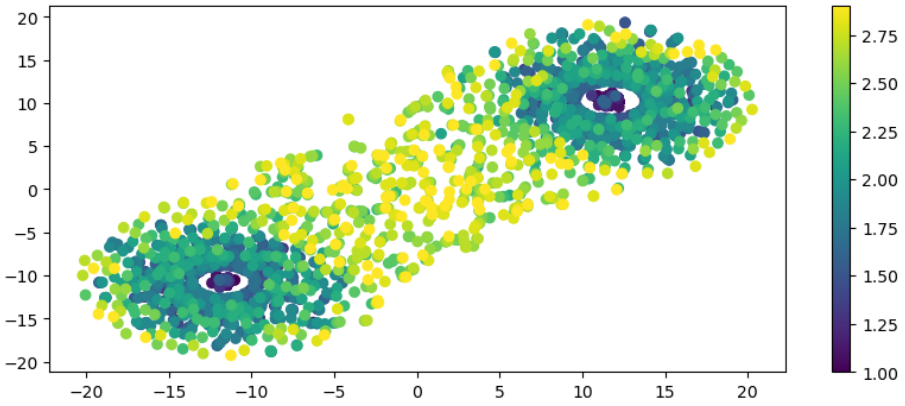
(c) $L = 80$

Figure 1: Classical Ising t-SNE

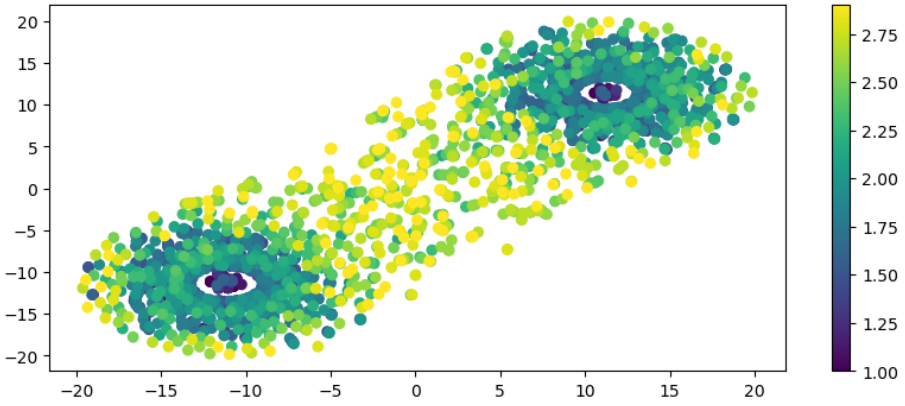
We notice from the axes that as L increases the clusters get closer to each other. However, the clusters are distinct.

- Show that if you apply the t-SNE code twice to the same dataset, you get different results. Why do such differences occur?

  Below are two plots that resulted from running t-SNE on the same L20 lattice twice.



(a) Run 1



(b) Run 2

Figure 2: Comparison of different t-SNE runs on $L = 20$ classical Ising

During test running the code we verified that the results can be quite different and not as similar as in this specific run of the code (the results are still different if one pays close attention). The general features of the plot however remained the same in all runs, i.e. the existence of clusters. The reason why we expect these differences is the stochasticity of t-SNE.

- Compare your results for the Ising model with Figure 6 of Reference [3]. In this figure, t-SNE was applied to the two-dimensional Ising model on a lattice with L = 30. Do you observe similar behaviour?

  Below is figure 6 of reference 3

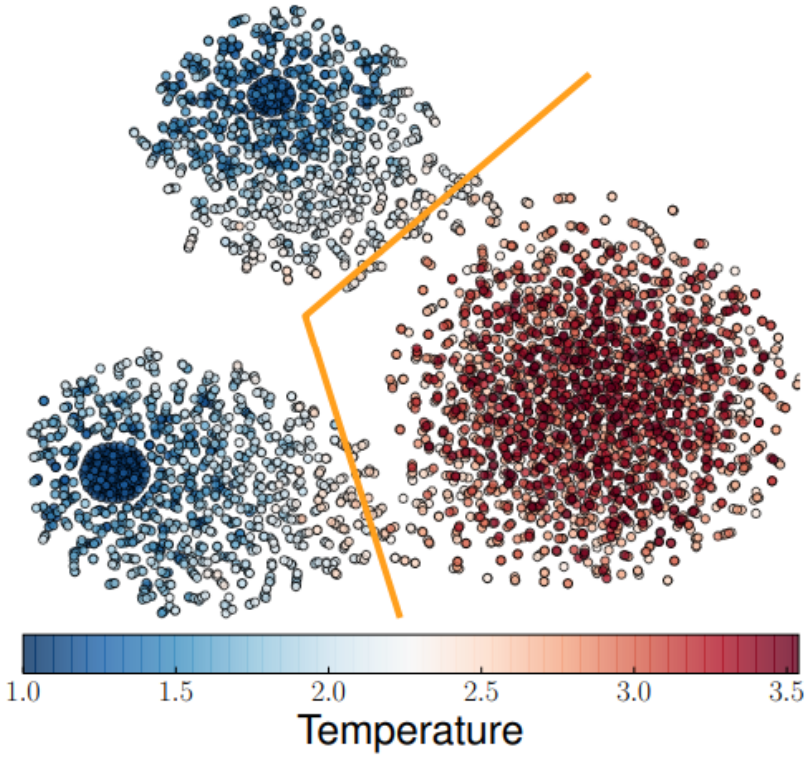


Figure 3: Figure 6 of reference 3

As we can see, the results are qualitatively similar. We observe distinct clustering between the two phases. In our case we did not include many temperatures above the critical point, so what would be our red cluster is more underdeveloped. In our diagrams it is the yellow-ish region connecting the two blue clusters.

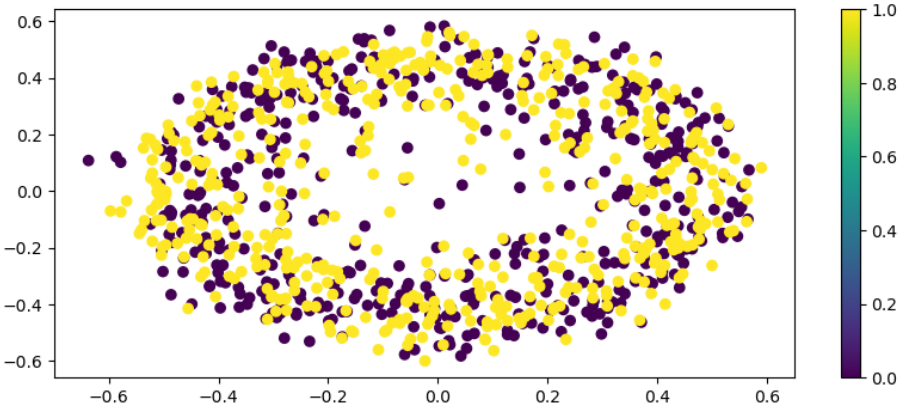
- Compare your results with the results of Tutorial 5, which studied the same problem of phase identification for Ising spin configurations, but used PCA rather than t-SNE. Discuss any differences that you observe.

  The results with t-SNE are similar to those with PCA with some inter-

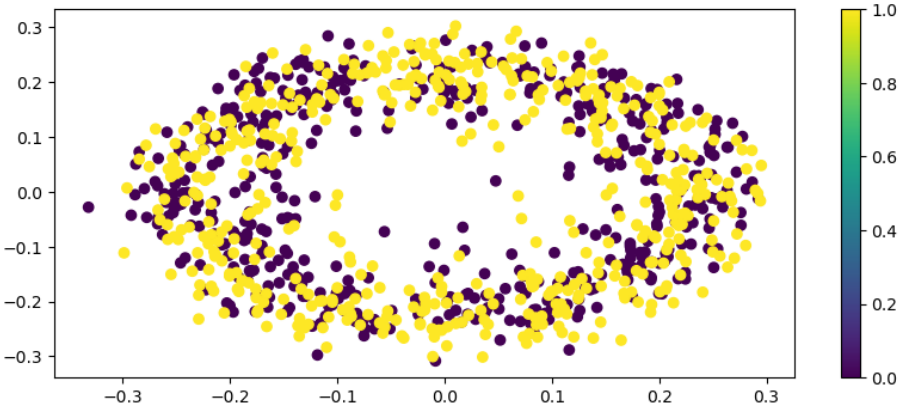
esting differences. For example, in PCA the clusters would become more distinct as L increased, while the opposite is true in t-SNE. We still observe similar clustering patterns corresponding to the feromagnetic and the paramagnetic phase for the classical ising theory, but similarlly to t-SNE, PCA fails for the gauge theory. The greatest qualitative difference between the two is that t-SNE is stocahstic, whereas PCA is not.

e) Repeat part c) for the given classical Ising gauge theory datasets with L = 20 and 40. You should find that, like PCA, t-SNE fails and is unable to produce distinct clusters when applied to this data.

Below we present the results for the classical Ising gauge theory



(a) $L = 20$



(b) $L = 40$

Figure 4: Classical Ising gauge t-SNE

Indeed t-SNE fails to produce distinct clusters for the classical Ising gauge

theory. f) Explain why t-SNE fails to produce distinct clusters when applied to classical Ising gauge theory data.

Hint: Recall that the t-SNE method uses a probability distribution defined in terms of the Euclidean distance between datapoints, and then prioritizes preserving the short-distance information within the lower-dimensional space. Can you find two different topologically-ordered configurations that are not "close" in terms of their Euclidean distance?

t-SNE fails to produce clusters because there exist configurations of the same phase that are not close in terms of Eucledian distance (the metric that t-SNE uses as an input for the probability distribution used to distinguish neighborhoods). The easiest such example would be any topological phase and the "inverse" configuration, as in the configuration where spins up are exchanged for spin downs and vice versa.

In terms of Eucledian distance the two configurations are as far as they can get (antidiametric), however both of them will be topologically ordered (in fact whenever the size of an axis is even, the Wilson loop's value does not change, so they can even be in the same class in the topologically ordered phase). The two configurations should be in the same cluster, but they are not close in Euclidean distance. This prevents clustering.