28462000 - Bourez Nicolas
38502000 - Demaret Dimitri
77102000 - Lhernould Benoit

LINMA2472 - Homework 3 - Part I - Group 9

1 Computing the kernel

1.1 Weisfeiler-Lehman subtree complexity

The asymptotic running time of the Weisfeiler-Lehman algorithm is $\mathcal{O}(hm)$, where h is the number of iterations that the algorithm performs and m is the number of edges in the graph.

It can be proved by showing that, for each iteration, all the steps can be done in $\mathcal{O}(m)$ time.

1. For the first step, we need to assign the multiset of neighboring labels to each node. The number of edges is m, so if we want to find each node, the upper bound is $\mathcal{O}(m)$.
2. For the second step, Sorting all the multisets of the nodes can be perform in $\mathcal{O}(m)$ time using counting sort, which is an instance of bucket sort.

   Bucket sort is a sorting algorithm that works by partitioning an array into a finite number of buckets. Here, we don't have an array, but we have a multiset of labels for each node. We can use this algorithm to sort because the cardinality of the labels is finite and upper bounded by n, the number of nodes. For each multiset, we can assign the elements to their corresponding buckets, which are the labels of the nodes. As there are $\mathcal{O}(m)$ elements in the multisets, we can do this in $\mathcal{O}(m)$ time. If we know from which multiset each node comes, we can just read through all buckets in ascending order and then extract the sorted multiset of labels for each node.
3. For the third step, we can perform the compression step in $\mathcal{O}(m)$ time, because we pass over all strings which are max m.
4. For the last step, we just update the labels of the nodes which can be done in $\mathcal{O}(m)$ time, as well

1.2 Explicit features

Let be $\mathcal{G}$ the space containing all the elements of our dataset $G_i \in \mathcal{G}$ with $i \in \{0, \dots, N\}, N = |\mathcal{G}|$.

To find the explicit embedding space of $k^{(h)}_{WLsubtree}(\cdot,\cdot) : \mathcal{G} \times \mathcal{G} \to \mathbb{R}$, we need to find the mapping function $\phi := \phi^{(h)}_{WLsubtree}(\cdot)$.[1]

Given two graphs G and G' :

- We define $\Sigma_i \subseteq \Sigma$ as the set of letters that occur as node labels at least once in G or G' at the end of the i-iteration of the WL-algorithm
- Σ_0 is the set of the original labels of G and G'
- $\Sigma_i = \{\sigma_{i1}, \dots, \sigma_{i|\Sigma_i|}\}$ are assumed pairwise disjoint and ordered, with σ_{ij} is a letter $\in \Sigma_i$

[1] The explicit kernel is the same as the one in the paper[1]

- We define a map $c_i : \{G, G'\} \times \Sigma_i \to \mathbb{N}$ s.t. $c_i(G, \sigma_{ij})$ is the number of occurrences of the letter σ_{ij} in the Graph G.

Thanks to these, we can explicitly define our kernel :

$$k^{(h)}_{WLsubtree}(G, G') = \langle \phi^{(h)}_{WLsubtree}(G), \phi^{(h)}_{WLsubtree}(G') \rangle$$

with (same for G'):

$$\phi^{(h)}_{WLsubtree}(G) = (c_0(G, \sigma_{01}), \dots, c_0(G, \sigma_{i|\Sigma_0|}), \dots, c_h(G, \sigma_{h1}), \dots, c_i(G, \sigma_{h|\Sigma_h|}))$$

1.3 Explicit embedding versus kernel

- The rank differs from the size of the WL subtree kernel matrix. This means that some feature maps $\phi(G_i)$ are dependent so we can define our implicit embedding space with a lower space.
- The size of one feature map is $h \sum_{i=0}^{h} |\Sigma_i|$
- The lower bound of the implicit embedding space is thus $rank \times h \sum_{i=0}^{h} |\Sigma_i|$, the space in which every feature map is independent.

Here are the different results for the 3 datasets :

dataset	nb_samples	rank
MUTAG	188	175
ENZYMES	600	595
NCI1	4110	4002

2 Visualization

2.1 Kernel centralization

Given the kernel function $k(x, y) = \langle \phi(x), \phi(y) \rangle$ and the kernel matrix K with elements $K_{ij} = k(x_i, x_j)$, we want to prove that the centered kernel matrix $\tilde{K}$ corresponds to the inner products of the centered features. The centered kernel matrix is defined as:

$$\tilde{K} = K - \frac{1}{N} 1_{N \times N} K - \frac{1}{N} K 1_{N \times N} + \frac{1}{N^2} 1_{N \times N} K 1_{N \times N} \tag{1}$$

Where $1_{N \times N}$ is an $N \times N$ matrix of ones. We denote $\bar{\phi} = \frac{1}{N} \sum_{i=1}^{N} \phi(x_i)$ as the mean feature map. We need to prove that:

$$\tilde{K}_{ij} = \langle \phi(x_i) - \bar{\phi}, \phi(x_j) - \bar{\phi} \rangle \tag{2}$$

Proof. We expand the right-hand side:

$$\begin{aligned} \langle \phi(x_i) - \bar{\phi}, \phi(x_j) - \bar{\phi} \rangle &= \langle \phi(x_i), \phi(x_j) \rangle - \langle \phi(x_i), \bar{\phi} \rangle - \langle \bar{\phi}, \phi(x_j) \rangle + \langle \bar{\phi}, \bar{\phi} \rangle \\ &= K_{ij} - \frac{1}{N} \sum_{k=1}^{N} K_{ik} - \frac{1}{N} \sum_{k=1}^{N} K_{kj} + \frac{1}{N^2} \sum_{k=1}^{N} \sum_{l=1}^{N} K_{kl} \end{aligned}$$

This aligns with the elements of $\tilde{K}$ as defined in (1). □

2.2 Distance

Proof. Consider the WL subtree kernel between two graphs G_1 and G_2:

$$k_{WLsubtree}(G_1, G_2) = \sum_{h=0}^{H} k_h(G_1^{(h)}, G_2^{(h)})$$

Where $G^{(h)}$ denotes the graph after h iterations of the WL algorithm, and k_h is the corresponding kernel function at iteration h. The pairwise distances between all the graphs in the dataset are computed as:

$$d(G_1, G_2) = \sqrt{k(G_1, G_1) + k(G_2, G_2) - 2k(G_1, G_2)}$$

To prove that this is the classical Euclidean distance in the implicit embedding space of the kernel, we can expand the distance formula in terms of the inner product in the feature space :

$$\begin{aligned}
\|\phi(G_1) - \phi(G_2)\|^2 &= \langle \phi(G_1) - \phi(G_2), \phi(G_1) - \phi(G_2) \rangle \\
&= \langle \phi(G_1), \phi(G_1) \rangle + \langle \phi(G_2), \phi(G_2) \rangle - 2\langle \phi(G_1), \phi(G_2) \rangle \\
&= k(G_1, G_1) + k(G_2, G_2) - 2k(G_1, G_2) \\
&= d(G_1, G_2)^2
\end{aligned}$$

Therefore, the pairwise distance formula corresponds to the Euclidean distance between the feature space embeddings of G_1 and G_2:

$$d(G_1, G_2) = \|\phi(G_1) - \phi(G_2)\|$$

□

2.3 KPCA-visualization and Comparison

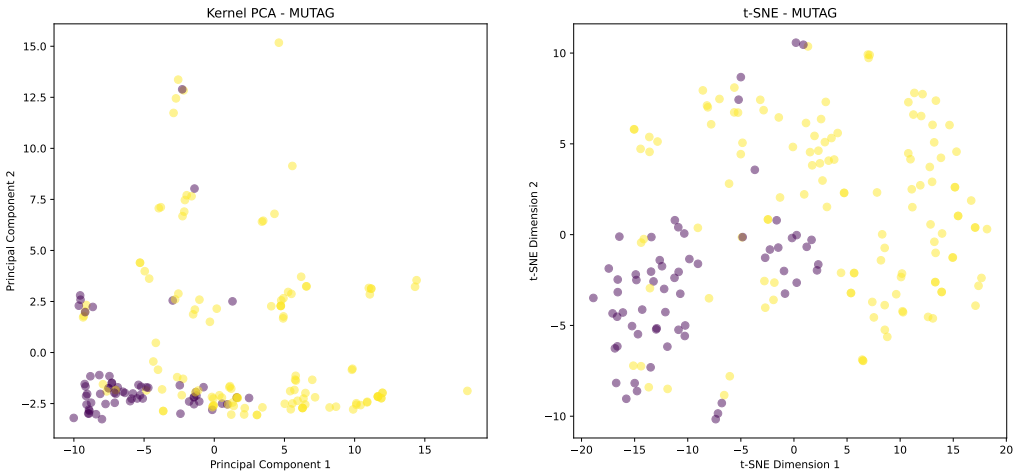

Figure 1: Comparison over the MUTAG dataset

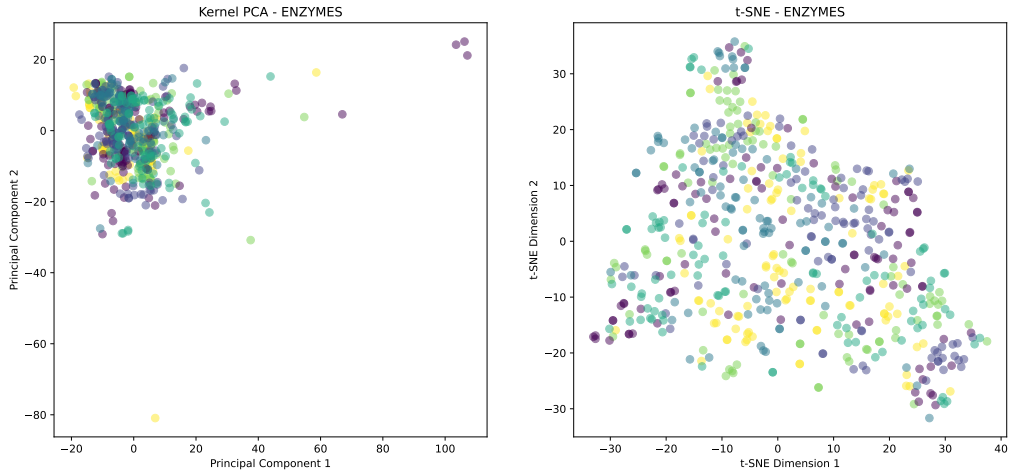

Figure 2: Comparison over the ENZYMES dataset

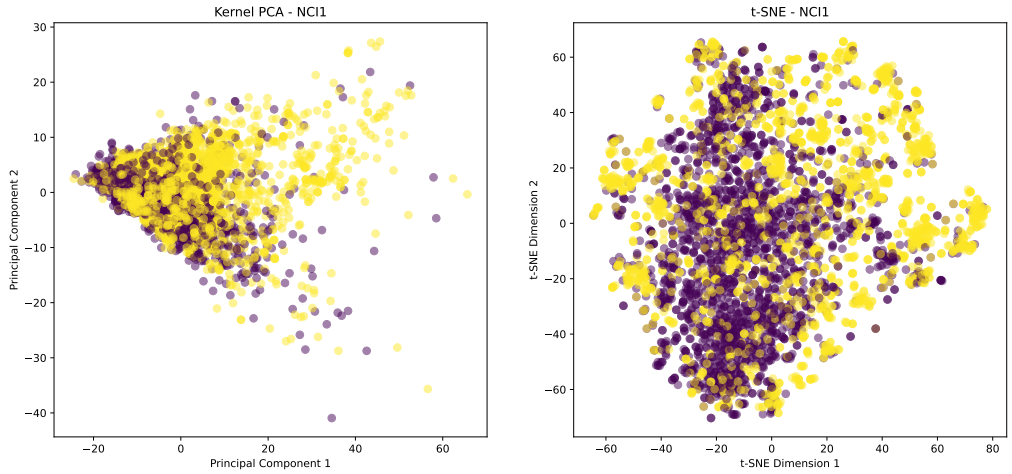

Figure 3: Comparison over the NCI1 dataset

Kernel-PCA tends to preserve the global structure of the dataset, with clearer separation between clusters as seen in the MUTAG dataset. It is effective in illustrating the overall data separability. On the other hand, t-SNE forming clusters are more spare, and less dense than the KPCA, especially noticeable in the ENZYMES and NCI1 datasets. The comparative plots underscore KPCA's ability to highlight distinct groupings, which can be particularly useful for pattern recognition tasks.

3 Classification

In this section, we present the accuracies obtained with our different codes for part 4.3 of part I of the project. This will help us in our observation in the next part of the report.

3.1 A simple baseline

Dataset	Accuracy
MUTAG	0.66
ENZYMES	0.17
NCI1	0.50

Table 1: Accuracy of the best constant model on the three datasets

Those results are the ratio of the more represented labels in the dataset.

3.2 Support Vector Machines (SVM)

Dataset	Accuracy
MUTAG	0.89
ENZYMES	0.53
NCI1	0.84

Accuracy of Optimized SVM Models with WL Subtree Kernel

3.3 Select hyperparameters

Dataset	C	H	Best Accuracy
MUTAG	10000	1	0.92
ENZYMES	10	3	0.54
NCI1	10	8	0.85

Best combination of hyperparameters and their accuracy

4 Observations

Dataset	Best Accuracy
MUTAG	0.92
ENZYMES	0.54
NCI1	0.85

Best Accuracy for MUTAG, ENZYMES, and NCI1 Datasets

We observe that the best accuracy of the SVM occurs with the MUTAG dataset, the NCI1 dataset is close, and over the ENZYMES it has a way smaller accuracy. This can be seen as a relationship with the plot of the kernel-PCA visualization in the fact that the dataset on which we have the best result (MUTAG) is the one on which we can distinguish two groups on the plot (see Figure 1). The second dataset (ENZYMES) has the least significant accuracy, and we can make the parallel with the KPCA plot on which we cannot distinguish the groups that are on top of each other (see Figure 2). The fact that there are more labels also influences the accuracy.

References

[1] Nino Shervashidze et al. "Weisfeiler-Lehman Graph Kernels". In: *Journal of Machine Learning Research* 12.77 (2011), pp. 2539–2561. URL: %5Curl%7Bhttp://jmlr.org/papers/v12/shervashidze11a.html%7D.