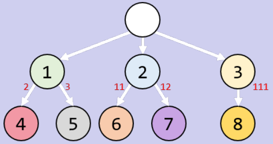


Master Thesis Seminar Talk

Progress Update

Fabrice Beaumont

Department of Information Systems and Artificial Intelligence - **Dr. Pascal Welke**

14. September 2022

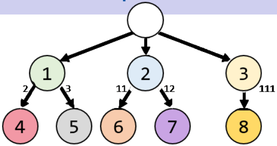
Progress Update

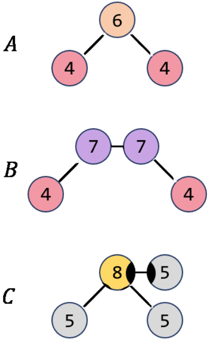
- Implementation of all major code components completed:
  - Reading da TU **Dataset**, cleaning and converting it
  - Constructing a **WLLT** (ability to expand it at will) with edge weights
  - A EdgeWeightLearner-**Interface** and classes to conveniently **evaluate** the quality of the resulting clustering

Current state

- Discarded the idea of **shifting weights**/**keeping** the total weight sum.
- Fine-tuning a **“DefaultLearner”**, inheriting from the interface
Previous goal: Finish one version. Then code other implementations of the interface
Actual situation: Making the Default Learner more and more parameterized

Example of the whole procedure





$^2/_3$		$^1/_3$		
$^1/_2$			$^1/_2$	
	$^3/_4$			$^1/_4$

Tree metric:

$$\begin{array}{c} \begin{array}{ccccc} 4 & 5 & 6 & 7 & 8 \end{array} \\ \begin{pmatrix} \cdot & 2 & 4 & 4 & 4 \\ & \cdot & 4 & 4 & 4 \\ & & \cdot & 2 & 4 \\ & \upuparrows & & \cdot & 4 \\ & & & & \cdot \end{pmatrix} \end{array}$$

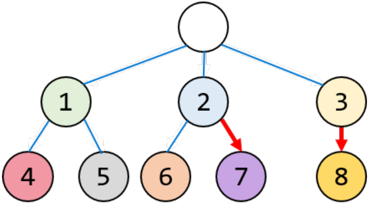
Wasserstein Dist.:

$$\mathcal{W}_t(A, B) = \frac{4}{3}$$
$$\mathcal{W}_t(A, C) = 3$$
$$\mathcal{W}_t(B, C) = 3$$

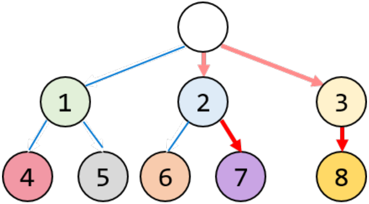
$$d_{\text{WLLT}}(B, C) = 2 * \frac{2}{4} + 4 * \frac{1}{4} + 4 * \frac{1}{4} = \frac{12}{4} = 3$$

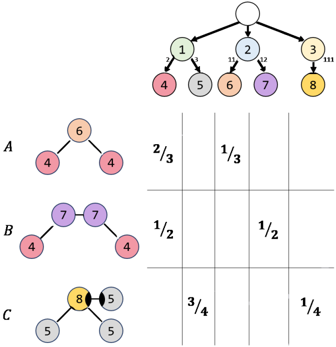
Example of the whole procedure

Local update $P_{7,8}$:



Weighted path update $P_{7,8}$:





Implementation of the Default Learner

- Initialize all edge weights as 1.0
- Compute the *Tree Wasserstein Distance*[1] between two graphs
- Select a **batch**, with **equal distributions** between all classes
- Pic the *n* highest differences in the weighted difference vector [2] (Option: Leaves-only)
- Update rule:

$$w' = w + \lambda \Delta w$$

[1] Normalized weighted distance between their wl-label histograms.

[2] Most expensive earth that had to be moved.

Implementation of the Learner - Update Rule

- Update rule:

$$\begin{aligned} w' &= w + \lambda \Delta w \\ &= w + \lambda \big(p_{\text{pull/push}} \; c_{\text{imba}} \; w \big) \end{aligned}$$

Where:

- Learning rate λ
- Push-Pull-Factor:
  $$p_{\text{pull/push}} = \begin{cases} +p_{\text{push}} & \text{different class} \\ -p_{\text{pull}} & \text{same class} \end{cases}$$
- Class sample imbalance factor:
  c_{imba} account for the fact, that when selecting two samples the probability to have two samples from the same class (pulling) is lower than having two samples from different classes (pushing).

Current todos / Outlook

? Why are the plots for MUTAG class -1 often zero? Is there an indexing error?
(E.g. Max Intra Cl Dist C-1)

? Why are the mean weights of mid-layers changing equally? Plotting error, update error or graph structure?

? Why are weights in other layers changing when setting to “only-leaves”

? Why is so few weight added? Is the balance factor correct?

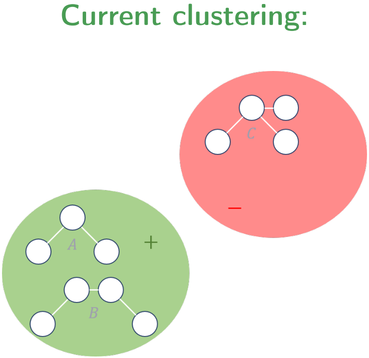
! Switch to go from relative to absolute push/pull factors

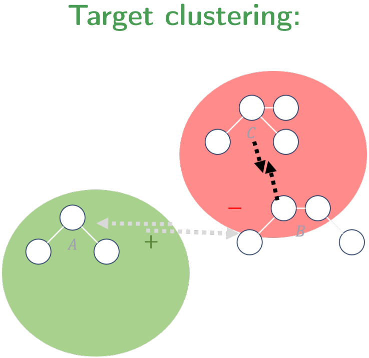
! How to deal with different batch sizes

Thank you all for listening.

I will be happy to answer any **questions** and hear your **comments**.

Example of the whole procedure





Idea: Reduce distance between B and C, by updating the edge weights.

Preparation of the performance comparison

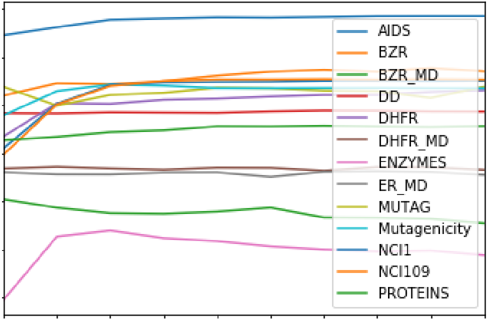


Figure: Classification accuracies on databases using Weisfeiler-Lehman.

```
grakel.kernels.WeisfeilerLehman(n_iter=[1-10], base=grakel.kernels.VertexHistogram, normalize=True)
grakel.utils.cross_validate_Kfold_SVM(K, y, n_iter=10)
```

Implementation road-map 1/2

- **WLLT Construction**:
  - Write to file and read from file. Construct WL-iteration based.
  - All weights *equal*.
  - (*Random* initial weights.)
  - (Use *a priori* knowledge.)
- **Wasserstein-Distance feedback**:
  - “Biggest pile of dirt”. (“Smallest”, to increase the distance.)
  - Distribution proportional to the pile size.
  - Distribution proportional to the cost of moving the pile size.

Implementation road-map 2/2

- **Update rule**:
  - Value:
    - Constant λ.
    - *Gradient descent*.
  - Location:
    - *Local*: Only update the first and last edge weights of the connecting path.
    - *Weighted path*: Update all edge weights on the path, with less magnitude for edges closer to the root.
    - *Path*: Update all edges on the path.
    - *Global*: Update all edges, related to all occurring labels.