# Strengthening structural baselines for graph classification using Local Topological Profile

J. Adamczyk, W. Czech

AGH University of Science and Technology
Faculty of Computer Science, Electronics and Telecommunications
Institute of Computer Science
al. A. Mickiewicza 30, 30-059 Krakow, Poland

# Graph classification

- **task:** assign a label to a whole graph
- **examples:**
  - molecules, e.g. mutagenicity, toxicity
  - proteins, e.g. enzyme or not
  - social networks, e.g. type of community

# Main approaches

**Feature extraction:**

- extract features from the graph into a tabular form
- classify with any classic algorithm
- often called **graph embedding**
- simple, but very fast and scalable

# Main approaches

**Graph kernels:**

- direct all-pairs similarity comparison
- costly and the least scalable

# Main approaches

**Graph Neural Networks (GNNs):**

- learnable, task-specific feature extraction
- the most modern approach
- use: topology, node features, edge features (sometimes)
- suffer from overfitting, oversmoothing, lack of data, training instability etc.
- nevertheless, often seem to give the best results

# Local Degree Profile (LDP)

- Cai, Chen, and Yusu Wang. "A simple yet effective baseline for non-attributed graph classification."
- **structural** (topological) feature extraction method
- designed as a **baseline** method for graph kernels and GNNs, widely used
- **idea:**
  - extract 5 degree features for each node
  - aggregate with histograms (or EDF)
  - classify with kernel SVM

# LDP - details

- **LDP features:**
  - node degree, neighbors degrees statistics: min, max, mean, stddev
- **hyperparameters:**
  - number of bins
  - aggregation: histogram vs EDF
  - degree normalization: per graph vs per dataset
  - use log scale or not

# LDP - problems

- **strictly local:** degree is a strictly local descriptor
- **not scalable:** kernel SVM is not scalable, and requires hyperparameter tuning
- **sequential:** both for feature extraction (NetworkX) and for training (SVM + grid search)
- **many hyperparameters:** too many for a simple baseline
- **worrying methodology:** authors report validation set accuracy

# Solution: LTP

- we analyzed all 5 areas, and **improved all of them**
- our resulting method is **Local Topological Profile (LTP)**
- an iterative improvement, being:
  - 2-3 degrees of magnitude faster
  - better results
  - more scalable

# Local Topological Profile (LTP)

- **more global:** add 3 new, well selected descriptors
- **scalable:** uses Random Forest (RF)
- **parallel:** both for feature extraction (torch-scatter, NetworKit) and for training
- **no hyperparameters:** we show that none are necessary
- **fair methodology:** we use fair comparison procedure for GNNs from:

Errica, Federico, et al. "A fair comparison of graph neural networks for graph classification."

# LTP - additional descriptors

- **edge betweenness centrality** - measures edge importance:

$$EBC(e) = \sum_{s,t;(s,t)\neq(u,v)} \frac{\sigma_{st}(e)}{\sigma_{st}}$$

- **Jaccard index** - measures node clustering structure:

$$JI(u,v) = \frac{|N(u) \cap N(v)|}{|N(u) \cup N(v)|}$$

- **Local Degree Score** - measures "hubness" of nodes:

$$LDS(e) = \max\left(1 - \frac{\ln \operatorname{rank}(v,u)}{\ln \operatorname{degree}(v)}, 1 - \frac{\ln \operatorname{rank}(u,v)}{\ln \operatorname{degree}(u)}\right)$$

# LTP - Random Forest

- Random Forests are fast and **embarassingly parallel**, which makes them much more scalable than SVM
- they are **not sensitive to hyperparameter choice**, so we can omit tuning
- they consider each feature individually - we can **eliminate 2 hyperparameters:** aggregation and normalization

# LTP - hyperparameters

- we experimentally verify that we can set **reasonable defaults** for number of bins, and that we do not need log scale
- combined with RF, this **eliminates all hyperparameters**, with no (or negligible) decrease in accuracy

# LTP - parallelization

- LDP used sequential, Python-based NetworkX for feature extraction
- computing our new features **may be** costly
- we use parallel libraries torch-scatter and NetworKit, written in C++
- this improvement alone **completely offsets**, by a large margin, the added cost

# LTP - methodology

- we use **nested cross-validation** for evaluation:
  - 10-fold CV for testing
  - 5-fold CV for validation
- as described in fair comparison procedure for GNNs we use exactly the same folds as that paper
- we report results both for LDP and for our proposed LTP under this methodology
- benchmark is a standardized collection of 9 reasonably large graph classification datasets

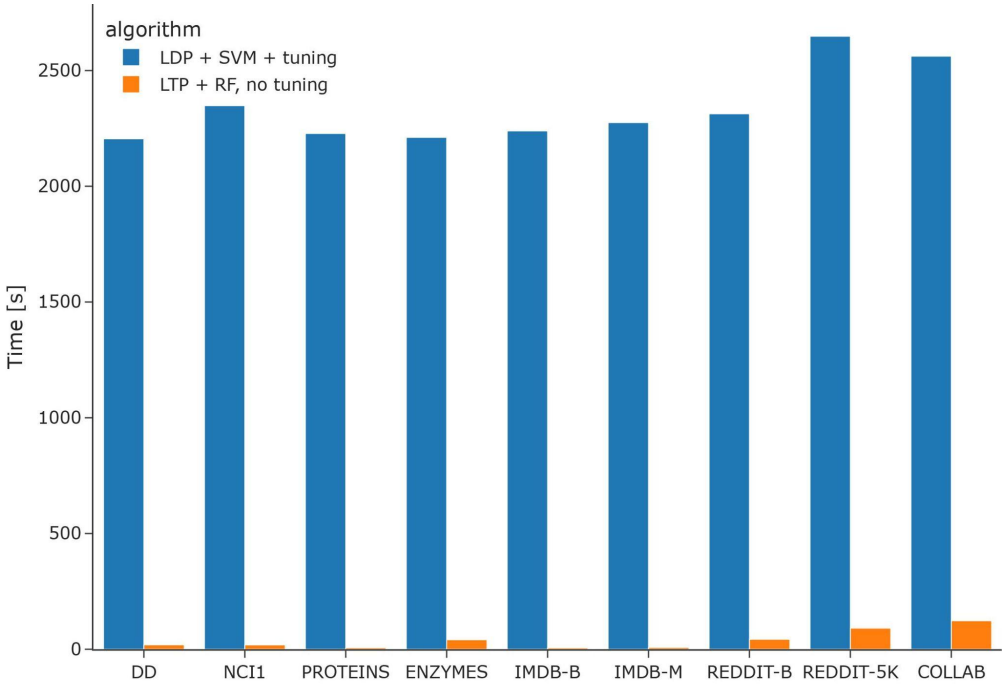
# Datasets

| Dataset | # Graphs | Avg. # Nodes | Avg. # Edges | # Classes |
|---|---|---|---|---|
| DD | 1178 | 284.32 | 715.66 | 2 |
| NCI1 | 4110 | 29.87 | 32.30 | 2 |
| PROTEINS | 1113 | 39.06 | 72.82 | 2 |
| ENZYMES | 600 | 32.63 | 64.14 | 6 |
| IMDB-B | 1000 | 19.77 | 96.53 | 2 |
| IMDB-M | 1500 | 13.00 | 65.94 | 3 |
| REDDIT-B | 2000 | 429.63 | 497.75 | 2 |
| REDDIT-5K | 4999 | 508.82 | 594.87 | 5 |
| COLLAB | 5000 | 74.49 | 2457.78 | 3 |

# Results - scalability

**Note:** we re-implemented LDP with torch-scatter and used parallel grid search.

With original implementation, LDP would be at least 10-100x slower.



# Results - accuracy

| Dataset | Baseline [4] | DGCNN | DiffPool | ECC | GIN | GraphSAGE | LDP | LTP |
|---|---|---|---|---|---|---|---|---|
| DD | **78.4 ±4.5** | 76.6 ±4.3 | 75.0 ±3.5 | 72.6 ±4.1 | 75.3 ±2.9 | 72.9 ±2.0 | 76.0 ±3.0 | 77.1 ±3.7 |
| NCI1 | 69.8 ±2.2 | 76.4 ±1.7 | 76.9 ±1.9 | 76.2 ±1.4 | **80.0 ±1.4** | 76.0 ±1.8 | 77.2 ±1.5 | 77.0 ± 1.9 |
| PROTEINS | **75.8 ±3.7** | 72.9 ±3.5 | 73.7 ±3.5 | 72.3 ±3.4 | 73.3 ±4.0 | 73.0 ±4.5 | 70.6 ±1.7 | 72.7 ±4.2 |
| ENZYMES | **65.2 ±6.4** | 38.9 ±5.7 | 59.5 ±5.6 | 29.5 ±8.2 | 59.6 ±4.5 | 58.2 ±6.0 | 37.4 ±4.0 | 42.5 ±4.1 |
| IMDB-B | 70.8 ±5.0 | 69.2 ±3.0 | 68.4 ±3.3 | 67.7 ±2.8 | 71.2 ±3.9 | 68.8 ±4.5 | 71.3 ±3.3 | **74.5 ±4.3** |
| IMDB-M | 49.1 ±3.5 | 45.6 ±3.4 | 45.6 ±3.4 | 43.5 ±3.1 | 48.5 ±3.3 | 47.6 ±3.5 | 49.0 ±4.4 | **50.0 ±4.6** |
| REDDIT-B | 82.2 ±3.0 | 87.8 ±2.5 | 89.1 ±1.6 | OOR | 89.9 ±1.9 | 84.3 ±1.9 | 89.6 ±1.2 | **91.1 ±1.0** |
| REDDIT-5K | 52.2 ±1.5 | 49.2 ±1.2 | 53.8 ±1.4 | OOR | **56.1 ±1.7** | 50.0 ±1.3 | 51.9 ±1.6 | 53.3 ±1.5 |
| COLLAB | 70.2 ±1.5 | 71.2 ±1.9 | 68.9 ±2.0 | OOR | 75.6 ±2.3 | 73.9 ±1.7 | 75.7 ±2.0 | **79.4 ±2.5** |

# Results - model ranks

| | Baseline [4] | DGCNN | DiffPool | ECC | GIN | GraphSAGE | LDP | LTP |
|---|---|---|---|---|---|---|---|---|
| **Average rank** | 3.8 | 5.2 | 4.6 | 7.6 | 2.7 | 5.4 | 4 | **2.6** |

# Summary

- we analyze existing structural baseline, Local Degree Profile (LDP)
- based on results, we identified 5 areas for improvement
- we propose **Local Topological Profile (LTP)** as a strengthened structural baseline
- LTP has better scalability and accuracy
- it can even outperform modern GNNs

# Questions?