Lab Development and Application of Data Mining and Learning Systems: Machine Learning and Data Mining

Master: MA-INF 4306

Preliminary Meeting

Organizers: Florian Seiffarth, Dr. Pascal Welke

Supervisors: Till Schulz, Florian Seiffarth, Dr. Daniel Trabold, Vanessa Toborek
Dorina Weichert, Dr. Pascal Welke

Main Information

- **Information and Contact**

Materials:	ecampus.uni-bonn.de/goto_ecampus_crs_2070077.html
Website:	mlai.cs.uni-bonn.de
Tips and Templates:	mlai.cs.uni-bonn.de/teaching/student-guides-and-templates
Contact:	seiffarth@cs.uni-bonn.de

- **Meeting Link:**
  - https://uni-bonn.zoom.us/j/93761053709?pwd=ZUhBeDZPemVOaEtLWHMwWDd2Zk42UT09

Lab Organization

- Today: Topic Presentation
- Typical task is to answer a scientific question by implementing an algorithm or framework and run experiments
- You can work in groups up to 2 students
- You have to present your progress and final work in four meetings
- You have to submit a written report after the last meeting

Lab Organization

- Each topic has a supervisor
- After the topic assignment and **before** the first meeting contact the supervisor for the next steps
- Arrange **individual appointments** for discussing questions, next steps, presentation slides, and your lab report
- Discuss programming language (and libraries)
- Discuss experimental design

Dates

Meetings	Date
Preliminary Meeting	14.04.2021, 12:00-13:00
First Progress Meeting	04.05.2021, 13:00-15:00
Second Progress Meeting	01.06.2021, 13:00-15:00
Third Progress Meeting	06.07.2021, 13:00-15:00
Final Meeting	07.09.2021, 13:00-15:00

Preliminary Report Deadline	05.09.2021, 23:55
Report Deadline	12.09.2021, 23:55

Next Step

- **Apply for the Lab**
  by sending a mail to **seiffarth@cs.uni-bonn.de** including your Uni-Bonn ID (if you have one) and your topic votes (3 topics) with priorities 1,2,3
  - **Deadline: Thursday, 15.04.2021**
  - We will tell you whether you can participate on Friday, 16.04.2021

Topics

Lab Topics

1) Intermediate Graph Classification (Till Schulz)
2) Generation of "Artificial" Molecule Graphs (Florian Seiffarth)
3) A Detailed Analysis of HoPS (Florian Seiffarth)
4) Analysis of the Loss Function (Vanessa Toborek)
5) Generating IQ-Number Tests for AI (Daniel Trabold)
6) Solving IQ-Number Tests (Daniel Trabold)
7) Numerical Integration of linear constrained Gaussians (Dorina Weichert)
8) Weisfeiler Lehman, Tree Metrics, and the French Railway System (Pascal Welke)

Further Lab Organization

Progress Presentations

- 7 minutes talk + 8 minutes discussion – time constraints are **sharp**
- Discuss the slides with your supervisor in advance
- **Structure of the slides**:
  1) Problem definition
  2) Summary of last presentation (if applicable)
  3) New achievements
  4) Summary of current state,
  5) next steps
- Each slide should contain **title**, **author** and **slide number**
- Demos are welcome (please prepare your demo thoroughly)

Final Presentation

- 15 minutes talk + 5 minutes discussion – time constraints are **sharp**
- Discuss the slides in advance with your supervisor
- **Structure of the slides:**
  1) Problem definition
  2) All the achievements during the entire lab project
  3) Summary of the final state
- Each slide should contain **title**, **author** and **slide number**
- Demos are welcome (please prepare your demo thoroughly)

Lab Report

- **8-10 pages per group** in LaTeX
  - A template can be found here
- Describe
  - the research question of your lab,
  - related work,
  - any preliminaries,
  - what you did to answer your question(s), and
  - analyze your results.
- If working in a group, each student must have a clear sub-focus in the topic and clearly visible individual contribution in the final report (e.g. by authoring separate chapters)
- Check out our homepage for some hints on writing a lab report:
  - mlai.cs.uni-bonn.de/teaching/student-guides-and-templates

Terms for Passing this Module

- Providing a scientific answer to a scientific question
- Providing code that runs in an environment specified by your supervisor(s)
- Successful progress presentations
- Successful final presentation
- Successful written report
- Active participation in all meetings
- Registration in BASIS, Deadline: 30.04.2021
- Email registration for b-it students enrolled in Aachen

Some Hints

Some Hints for Acquiring a Good Grade

- Presentations
  - discuss the outline of your presentations with your supervisor
  - send a preliminary version of your slides to your supervisor **in advance**
- Progress
  - discuss your progress and next steps with your supervisor repeatedly
- Report
  - send preliminary version of the report to your supervisor **in advance** and discuss it with him
  - Cite correctly when using figures and text from the original papers or other sources

Some More Hints

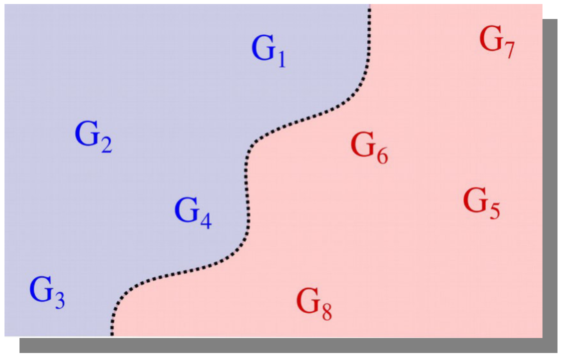
- Slides
    - At most one slide per minute
    - Not too much text or clutter
    - Pictures and animations help a lot
    - Please include slide numbers and your name on each slide
- Presentation
    - your audience consists of people knowledgeable in your field and other people less knowledgeable.
        - your presentation should be understandable, yet interesting for both groups
    - don't just read what is on the slides
    - look at your audience, talk to and with them
    - speak clearly and not too fast

Last but not Least

- Please keep in mind the **registration deadline** in BASIS! (master students of Uni Bonn)
- Please always **cite** other peoples work and reference pictures correctly! (In case of **plagiarism** the final grade will be **5.0**!)
- It is your responsibility to contact your supervisor with your questions and other issues with respect to the lab
- Finally, all programs must run on your supervisor's machine
- Please do not underestimate the required effort and time you need to invest in order to successfully participate in this lab

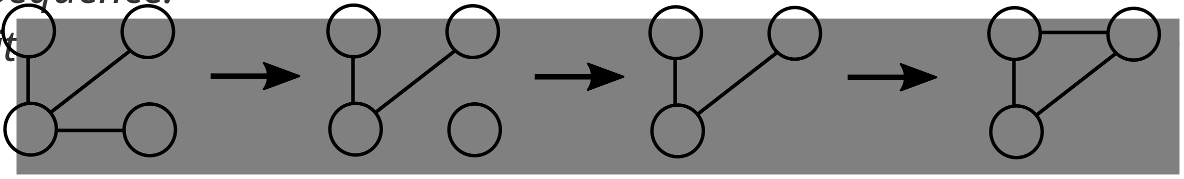
Intermediate Graph Classification

- General (binary) graph classification task:
  "Separate blue from red graphs in some latent space to reason about color of previously unseen graphs"



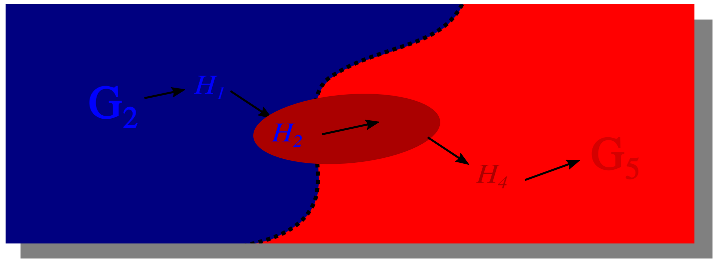
- Most common graph classification approaches:
  - Graph kernels (via Support Vector Machines)
  - Graph Neural Networks

- *Graph Edit Sequence:*
  "A graph edit *into another."*



Intermediate Graph Classification

- Lab topic questions:
  For a pair of red and blue graphs, consider the intermediate graphs defined by an edit sequence.



  - What can be reasoned about the border?
  - How volatile is that border w.r.t. randomness in the learning process?
  - Does the classification confidence of an intermediate graph correlate to the position of that graph in the edit sequence?

- Lab subtasks:
  - Set up a graph classification framework
  - Implement the graph edit sequence generation process
  - Investigate above research questions

Lab Topic: A Detailed Analysis of HoPS

Sommersemester 2021

Introduction

- The base problem is to estimate the count of subtree patterns in big labeled and unlabeled graphs.

Introduction

- The base problem is to estimate the count of subtree patterns in big labeled and unlabeled graphs.
- Exact algorithms are very slow because the problem is NP-hard

Introduction

- The base problem is to estimate the count of subtree patterns in big labeled and unlabeled graphs.
- Exact algorithms are very slow because the problem is NP-hard
- Estimation works (i.e. HOPS estimation algorithm) but there are some parameters which are not studied and probably the algorithm can be optimized.

Introduction

- The base problem is to estimate the count of subtree patterns in big labeled and unlabeled graphs.
- Exact algorithms are very slow because the problem is NP-hard
- Estimation works (i.e. HOPS estimation algorithm) but there are some parameters which are not studied and probably the algorithm can be optimized.
- The lab task is to make detailed experiments evaluating the importance of these parameters.

Applications

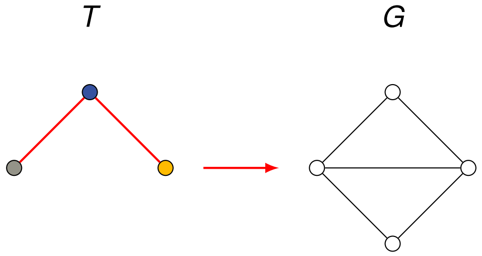
- social network analysis
- analysis of molecules and biological networks

Importance Sampling

- Estimate the number of subtree embeddings using importance sampling for trees, see [WSKW20, FK14, RZRD18]

Importance Sampling

Importance Sampling

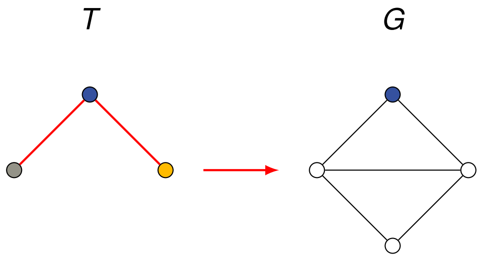


Estimation: 0

- Estimate the number of subtree embeddings using importance sampling for trees, see [WSKW20, FK14, RZRD18]

Importance Sampling

Importance Sampling

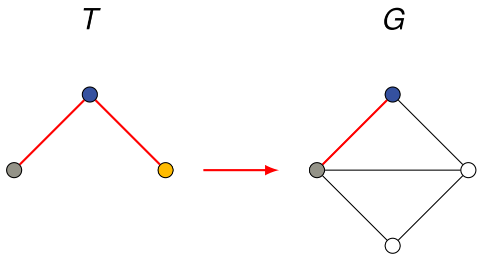


Estimation: 4

- Estimate the number of subtree embeddings using importance sampling for trees, see [WSKW20, FK14, RZRD18]

Importance Sampling

Importance Sampling

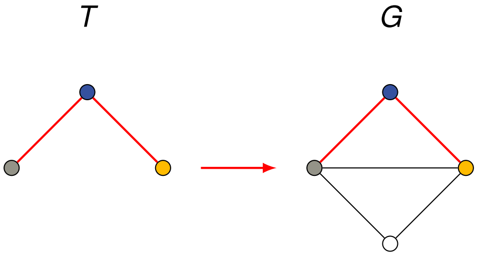


Estimation: $4 \cdot 2$

- Estimate the number of subtree embeddings using importance sampling for trees, see [WSKW20, FK14, RZRD18]

Importance Sampling

Importance Sampling

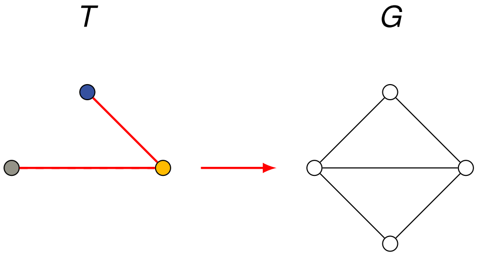


Estimation: $8 = 4 \cdot 2 \cdot 1$

- Estimate the number of subtree embeddings using importance sampling for trees, see [WSKW20, FK14, RZRD18]

Importance Sampling

Importance Sampling

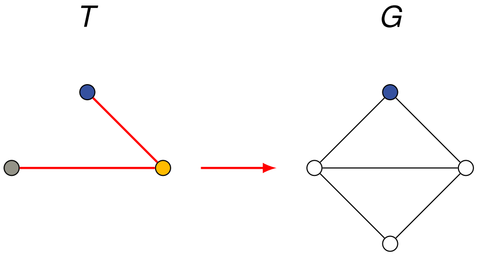


Estimation: 0

- Estimate the number of subtree embeddings using importance sampling for trees, see [WSKW20, FK14, RZRD18]

Importance Sampling

Importance Sampling

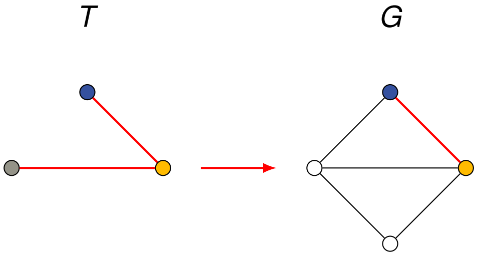


Estimation: 4

- Estimate the number of subtree embeddings using importance sampling for trees, see [WSKW20, FK14, RZRD18]

Importance Sampling

Importance Sampling

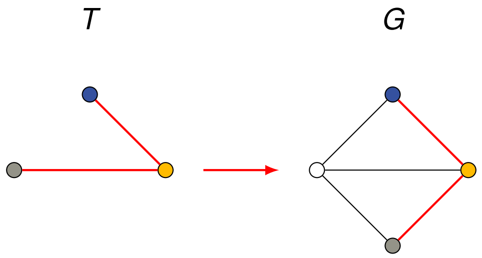


Estimation: $4 \cdot 2$

- Estimate the number of subtree embeddings using importance sampling for trees, see [WSKW20, FK14, RZRD18]

Importance Sampling

Importance Sampling

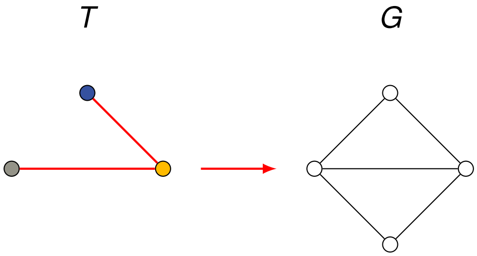


Estimation: $16 = 4 \cdot 2 \cdot 2$

- Estimate the number of subtree embeddings using importance sampling for trees, see [WSKW20, FK14, RZRD18]
- Average over all estimations

Importance Sampling

Importance Sampling



Average Estimation: $12 = \frac{8+16}{2}$

- Estimate the number of subtree embeddings using importance sampling for trees, see [WSKW20, FK14, RZRD18]
- Average over all estimations

Implementation Task I

- Understand Importance Sampling and the HOPS algorithm
- Use the HOPS algorithm and implement variations to look at the following problems:
    - What is the importance of choosing the root node of the pattern tree?
    - What is the importance of choosing the image of the root node in the big graph
- Both questions should be addressed in the unlabeled and labeled case.

Implementation Task II

- Create experiments with different parameters, choosing the pattern root node as the node with:
    - most frequent label frequency based on the big graph
    - less frequent label frequency based on the big graph
    - highest degree
    - lowest degree
    - most central in pattern tree
- Instead of choosing the image of the root node iid considering to distributions where
    - nodes with high degree are more probable images of the root node
    - nodes with low degree are more probable images of the root node

Evaluation Task

- Evaluate the accuracy and number of iterations for the different parameters.
- Is there any correlation between number of iterations and estimation error for constant time but different parameters?
- Are there any significant changes considering only the estimation error and changes in the parameters?

Martin Fürer and Shiva Prasad Kasiviswanathan.
Approximately counting embeddings into random graphs.
Comb. Probab. Comput., 23(6):1028–1056, 2014.

Irma Ravkic, Martin Znidarsic, Jan Ramon, and Jesse Davis.
Graph sampling with applications to estimating the number of pattern embeddings and the parameters of a statistical relational model.
Data Min. Knowl. Discov., 32(4):913–948, 2018.

Pascal Welke, Florian Seiffarth, Michael Kamp, and Stefan Wrobel.
HOPS: probabilistic subtree mining for small and large graphs.
In Rajesh Gupta, Yan Liu, Jiliang Tang, and B. Aditya Prakash, editors, *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, pages 1275–1284. ACM, 2020.

Lab Topic: Generation of “Artificial” Molecule Graphs

Sommersemester 2021

UNIVERSITÄT BONN

Idea

1. Use different algorithms (GNN, GANs, VAE) on molecule graph data to generate artificial molecule graphs, e.g. ([LABG18], [SK18], [LVD$^+$18], [KPH17], [RNS20])

Idea

1. Use different algorithms (GNN, GANs, VAE) on molecule graph data to generate artificial molecule graphs, e.g. ([LABG18], [SK18], [LVD$^+$18], [KPH17], [RNS20])
2. Look for algorithms which can validate if the generated graph is a valid molecule.

Idea

1. Use different algorithms (GNN, GANs, VAE) on molecule graph data to generate artificial molecule graphs, e.g. ([LABG18], [SK18], [LVD$^+$18], [KPH17], [RNS20])
2. Look for algorithms which can validate if the generated graph is a valid molecule.
3. Evaluate different algorithms on different measures (variety of the generated graphs, number of valid molecules generated etc.)

Idea

1. Use different algorithms (GNN, GANs, VAE) on molecule graph data to generate artificial molecule graphs, e.g. ([LABG18], [SK18], [LVD^{+}18], [KPH17], [RNS20])
2. Look for algorithms which can validate if the generated graph is a valid molecule.
3. Evaluate different algorithms on different measures (variety of the generated graphs, number of valid molecules generated etc.)
4. Discuss the results, advantages and disadvantages of the different generation methods

Tasks

Implementation:

- Implement a pipeline which in the first step reads molecule graphs from graph datasets, then applies different graph generation algorithms on the data and at the end provide some checking/evaluation if the output graphs are valid molecules.

Testing/Evaluation:

- Run pipeline on different datasets, and compare the output of the algorithms
- Use and develop different evaluation measures for the output molecules

Matt J. Kusner, Brooks Paige, and José Miguel Hernández-Lobato.
Grammar variational autoencoder.
In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1945–1954. PMLR, 2017.

Qi Liu, Miltiadis Allamanis, Marc Brockschmidt, and Alexander L. Gaunt.
Constrained graph variational autoencoders for molecule design.
In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018,*

December 3-8, 2018, Montréal, Canada, pages 7806–7815, 2018.

Yujia Li, Oriol Vinyals, Chris Dyer, Razvan Pascanu, and Peter W. Battaglia.
Learning deep generative models of graphs.
CoRR, abs/1803.03324, 2018.

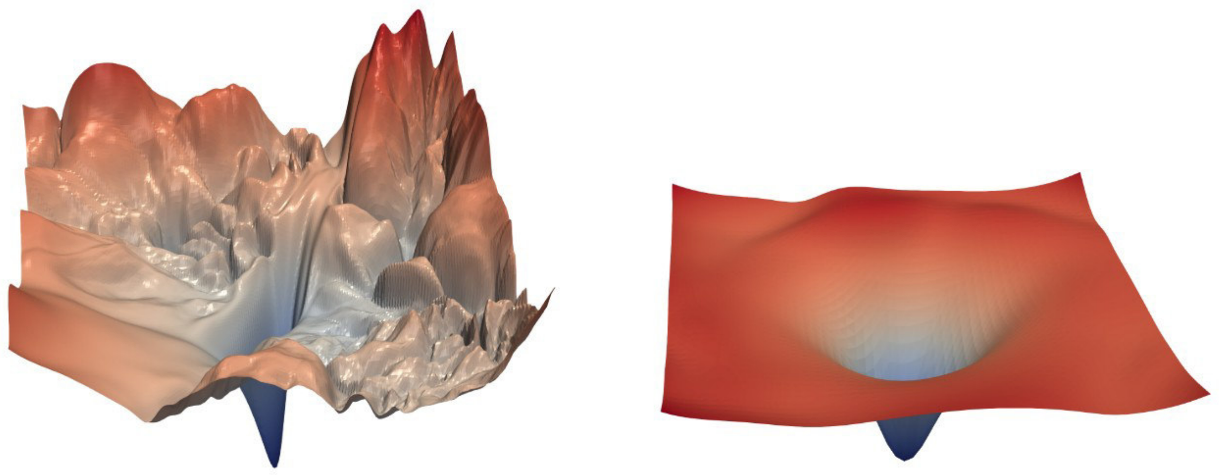
Davide Rigoni, Nicolò Navarin, and Alessandro Sperduti.
Conditional constrained graph variational autoencoders for molecule design, 2020.

Martin Simonovsky and Nikos Komodakis.
Graphvae: Towards generation of small graphs using variational autoencoders.
In Vera Kurková, Yannis Manolopoulos, Barbara Hammer, Lazaros S. Iliadis, and Ilias Maglogiannis, editors, *Artificial Neural Networks and Machine Learning - ICANN 2018 - 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part I,*

volume 11139 of *Lecture Notes in Computer Science*, pages 412–422. Springer, 2018.

Topic ##: Analysis of the Loss Function

- Topic: Analyze and compare neural networks by inspecting the topology of their loss functions
- For the problem of image classification: how does the topolgy of the loss function differ between a network that has learned to correctly classify the image and one that has not?



(a) without skip connections (b) with skip connections

Zhang, Q., Cao, R., Zhang, S., Redmonds, M., Wu, Y. N., & Zhu, S.-C. (2017). Interactively Transferring CNN Patterns for Part Localization. http://arxiv.org/abs/1708.01783

Thema ##: Analysis of the Loss Function

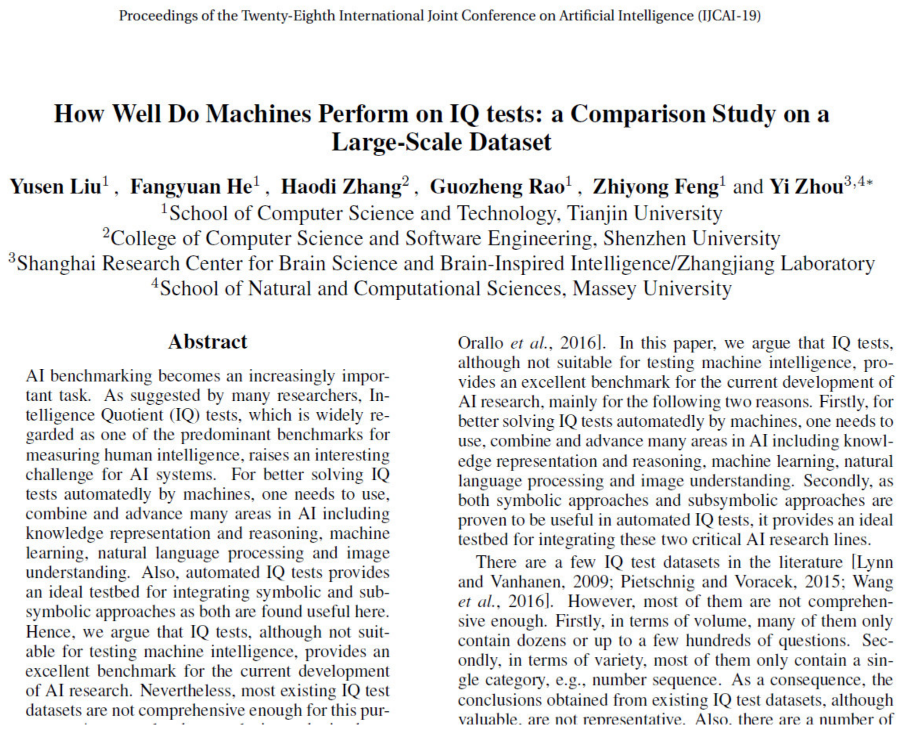
- Algorithms:
  - Visualization
    - Zhang, Q., Cao, R., Zhang, S., Redmonds, M., Wu, Y. N., & Zhu, S.-C. (2017). **Interactively Transferring CNN Patterns for Part Localization**. http://arxiv.org/abs/1708.01783
  - Flatness Measure
    - Petzka, H., Adilova, L., Kamp, M., & Sminchisescu, C. (2019). **A Reparameterization-Invariant Flatness Measure for Deep Neural Networks**. NeurIPS, 1–14. http://arxiv.org/abs/1912.00058
    - Rangamani, A., Nguyen, N. H., Kumar, A., Phan, D., Chin, S., & Tran, T. D. (2019). **A scale invariant flatness measure for deep network minima**. ArXiv, 1–17.
- Task:
  - Implementation of the visualization methods and flatness measures comparing two differently trained CNNs.

Topic X: Generating IQ-Number Tests for AI

Supervisor: Dr. Daniel Trabold

- The setting**:**
  - Human IQ is measured by tests
  - Number sequences are one format used in IQ tests

- The goal:
  - Develop a program that can generate number sequences for IQ tests
  - Compare AI approaches based on the generated number sequences



Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI-19)

How Well Do Machines Perform on IQ tests: a Comparison Study on a Large-Scale Dataset

Yusen Liu[1], **Fangyuan He**[1], **Haodi Zhang**[2], **Guozheng Rao**[1], **Zhiyong Feng**[1] and **Yi Zhou**[3,4*]
[1]School of Computer Science and Technology, Tianjin University
[2]College of Computer Science and Software Engineering, Shenzhen University
[3]Shanghai Research Center for Brain Science and Brain-Inspired Intelligence/Zhangjiang Laboratory
[4]School of Natural and Computational Sciences, Massey University

Abstract

AI benchmarking becomes an increasingly important task. As suggested by many researchers, Intelligence Quotient (IQ) tests, which is widely regarded as one of the predominant benchmarks for measuring human intelligence, raises an interesting challenge for AI systems. For better solving IQ tests automatedly by machines, one needs to use, combine and advance many areas in AI including knowledge representation and reasoning, machine learning, natural language processing and image understanding. Also, automated IQ tests provides an ideal testbed for integrating symbolic and subsymbolic approaches as both are found useful here. Hence, we argue that IQ tests, although not suitable for testing machine intelligence, provides an excellent benchmark for the current development of AI research. Nevertheless, most existing IQ test datasets are not comprehensive enough for this pur-

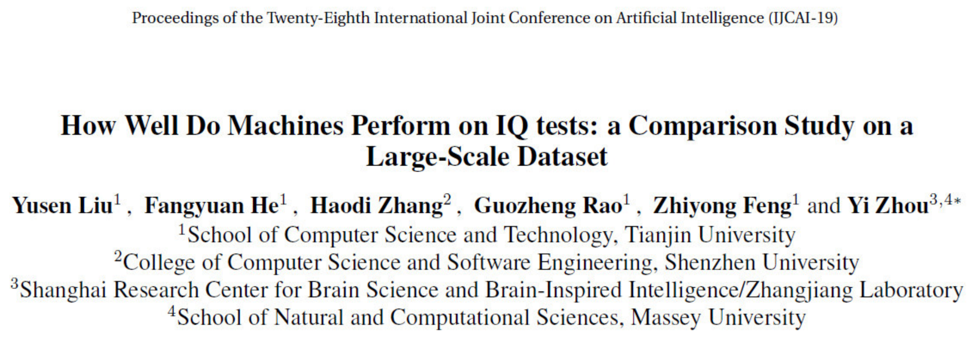
Orallo *et al.*, 2016]. In this paper, we argue that IQ tests, although not suitable for testing machine intelligence, provides an excellent benchmark for the current development of AI research, mainly for the following two reasons. Firstly, for better solving IQ tests automatedly by machines, one needs to use, combine and advance many areas in AI including knowledge representation and reasoning, machine learning, natural language processing and image understanding. Secondly, as both symbolic approaches and subsymbolic approaches are proven to be useful in automated IQ tests, it provides an ideal testbed for integrating these two critical AI research lines.

There are a few IQ test datasets in the literature [Lynn and Vanhanen, 2009; Pietschnig and Voracek, 2015; Wang *et al.*, 2016]. However, most of them are not comprehensive enough. Firstly, in terms of volume, many of them only contain dozens or up to a few hundreds of questions. Secondly, in terms of variety, most of them only contain a single category, e.g., number sequence. As a consequence, the conclusions obtained from existing IQ test datasets, although valuable, are not representative. Also, there are a number of

https://www.ijcai.org/Proceedings/2019/846

Topic X: Generating IQ-Number Tests for AI

Supervisor: Dr. Daniel Trabold



Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI-19)

How Well Do Machines Perform on IQ tests: a Comparison Study on a Large-Scale Dataset

Yusen Liu[1], **Fangyuan He**[1], **Haodi Zhang**[2], **Guozheng Rao**[1], **Zhiyong Feng**[1] and **Yi Zhou**[3,4*]
[1]School of Computer Science and Technology, Tianjin University
[2]College of Computer Science and Software Engineering, Shenzhen University
[3]Shanghai Research Center for Brain Science and Brain-Inspired Intelligence/Zhangjiang Laboratory
[4]School of Natural and Computational Sciences, Massey University

https://www.ijcai.org/Proceedings/2019/846

- Examples:
  - 1 2 3 4 5 _
  - 2 8 4 16 8 _
  - 10 15 10 22 10 _
  - 5 25 7 49 31 _

- Your task:
  - Program a python library to generate number sequences
  - Design and implement a mechanism to control the complexity
  - Repeat the experiments from Liu et al.

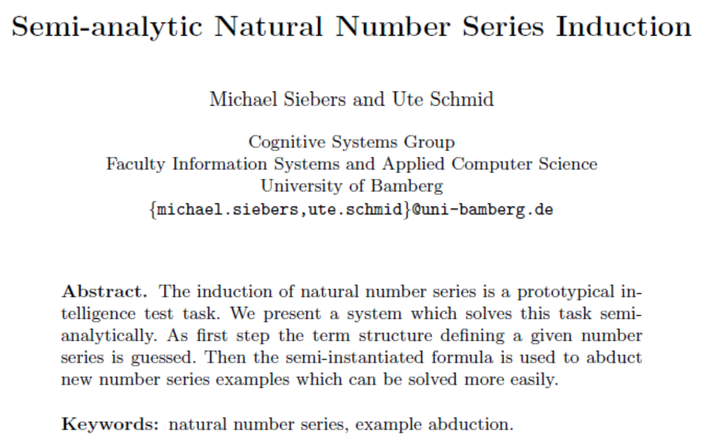
https://www.uni-bamberg.de/en/cogsys/research/projects/numberseries/

Topic Y: Solving IQ-Number Tests

Supervisor: Dr. Daniel Trabold

- The setting:
  - Siebers and Schmid published a program to solve IQ sequence tests

- The goal:
  - Improve the program by Siebers and Schmid



Semi-analytic Natural Number Series Induction

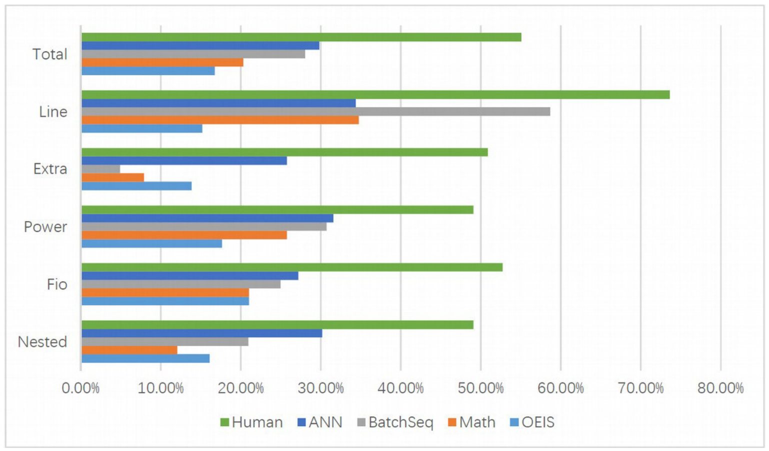
Michael Siebers and Ute Schmid

Cognitive Systems Group
Faculty Information Systems and Applied Computer Science
University of Bamberg
{michael.siebers,ute.schmid}@uni-bamberg.de

Abstract. The induction of natural number series is a prototypical intelligence test task. We present a system which solves this task semi-analytically. As first step the term structure defining a given number series is guessed. Then the semi-instantiated formula is used to abduct new number series examples which can be solved more easily.

Keywords: natural number series, example abduction.

https://www.uni-bamberg.de/en/cogsys/research/projects/numberseries/



https://www.ijcai.org/Proceedings/2019/846

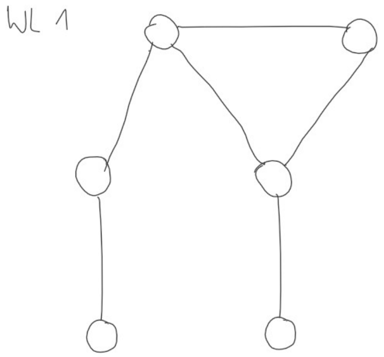
Topic Y: Solving IQ-Number Tests

Supervisor: Dr. Daniel Trabold

- Examples**:**
  - 9 16 25 36 49 _ → x(0) = 9; x(n) = (x(n-1)+5)+(n*2)
  - -9 -16 -25 -36 -49 _ → x(0) = -9 ; x(n) = 1+[x(0) = -17 , x(n) = (5^n)+(n*-31)];
  - Expected: x(n) = (n+3)^2 or -(n+3)^2
- Your task:
  - Reimplement the program by Siebers and Schmid
  - Improve the induction for the above example and others (e.g., alternating, short series)
  - Add an option to control induced positions per series

https://www.uni-bamberg.de/en/cogsys/research/projects/numberseries/

Weisfeiler Lehman, Tree Metrics, and the French Railway System

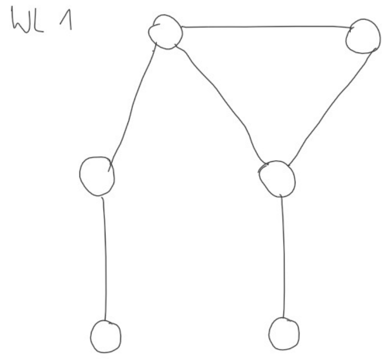


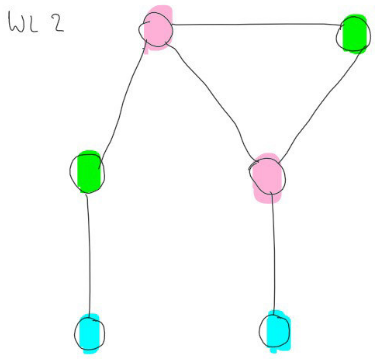
Weisfeiler Lehman (WL) Relabeling

for each vertex *v*:

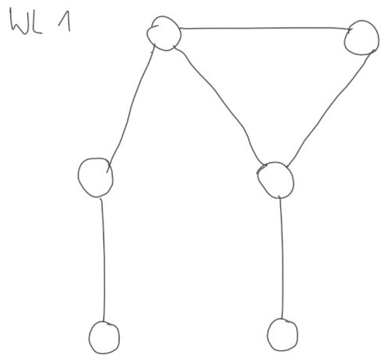
$$wl_{i+1}(v) = wl_i(v)\ \mathrm{sorted}(wl_i(w) \text{ for } w \text{ in } \mathrm{neighbors}(v))$$

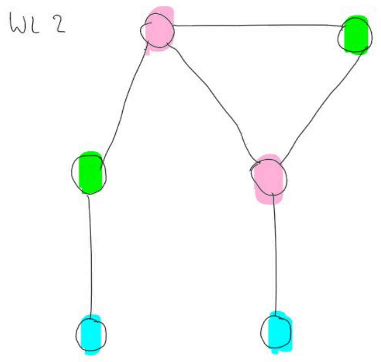
Supervisor: Pascal Welke

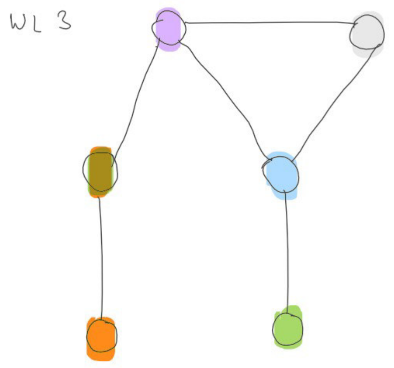




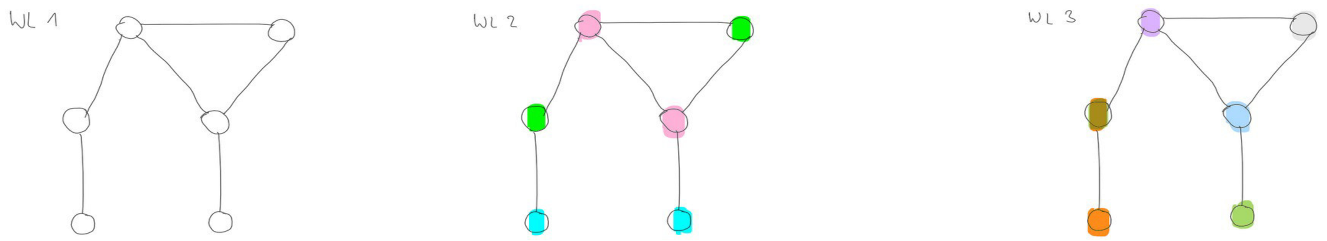
Applying WL Relabeling to a graph results in new labels



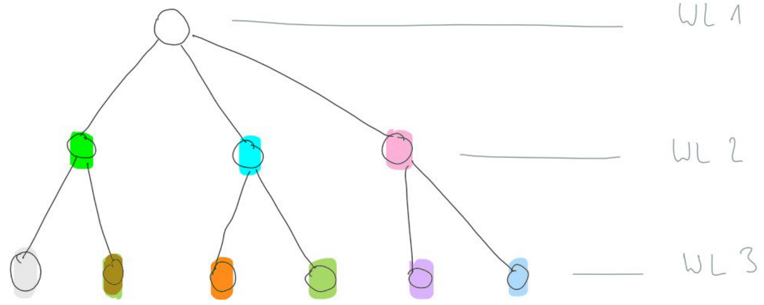


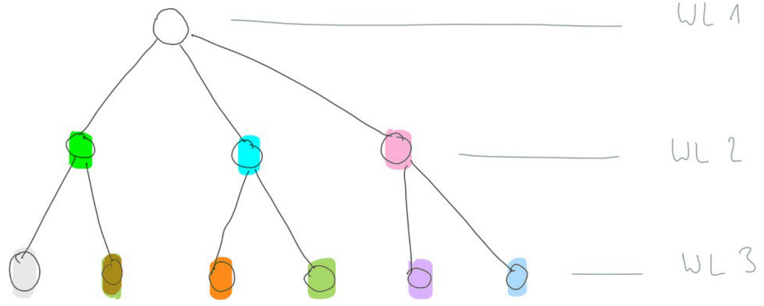


We say that vertices with identical labels are equivalent

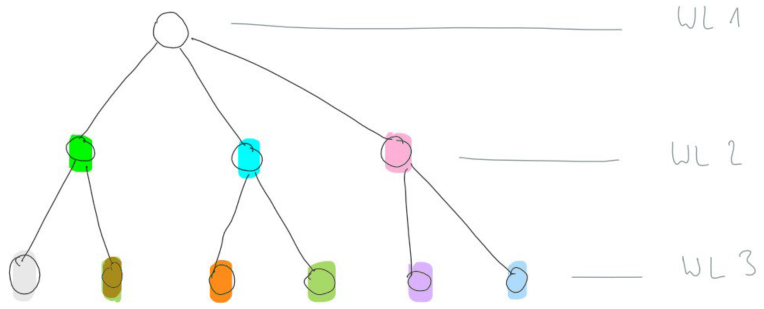


- The equivalence classes form a tree
- Each equivalence class has a parent (from previous iteration) and at least one child in the next iteration





- Each graph can be mapped to such a tree
- We want to use these trees to compute similarities among graphs
- **To this end, you should compute useful weights for the edges of these trees.**

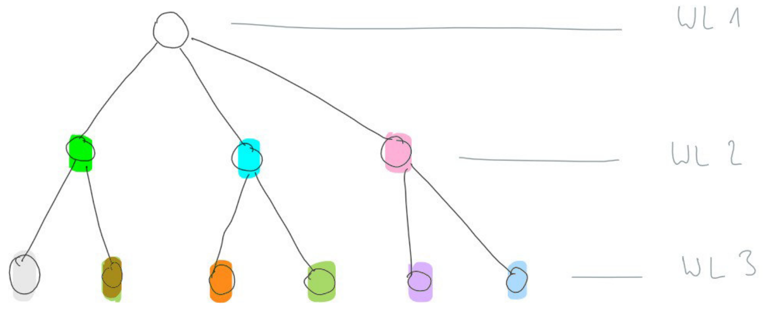


- Consider the tree *T* of *all WL labels* in your graph dataset
- The pairwise distance between siblings *v,w* can be computed using L1 / Hamming distance
- To obtain edge weights for the edges, we can use the French Railway Metric

$$d(v,w) = d_1(v,p) + d_1(p,w)$$

  where *p* is the barycenter of all siblings

- The edge weight between a parent and its child *v* is then set to $d_1(v,p)$



- Distances between the trees of two graphs can now be computed by the Wasserstein distance over the tree metric defined by *T*
- We can define a *graph kernel* using this distance function and use *kernel methods* (e.g. SVM) to learn over graph databases

Your Task

- Write code that:
  - computes the label tree T for a given graph database *D*
  - Computes edge weights for T
  - Computes a distance/kernel matrix for the graphs in *D*
- Evaluate the kernel by comparing its predictive performance to traditional Weisfeiler Lehman kernels and Wasserstein Weisfeiler Lehman kernels

References

- Le et al (2019): Tree-Sliced Variants of Wasserstein Distances. NeurIPS 2019: 12283-12294
- Schulz et al (2021): A Generalized Weisfeiler-Lehman Graph Kernel. ArXiv/2101.08104
- Togninalli et al (2019): Wasserstein Weisfeiler-Lehman Graph Kernels. NeurIPS 2019: 6436-6446