Machine Learning zur Vorhersage von Kanten in optimalen Lösungen für das Traveling Salesman Problem

FACHBEREICH INFORMATIK, HOCHSCHULE BONN RHEIN SIEG

AHMAD KADER

Inhaltsverzeichnis

Einleitung

- Vorhersage Kanten mithilfe Machine Learning
- Forschungsfragen:
  - ML Lösungen > Lösungen Heuristiken?
  - Wie schneiden Heuristiken untereinander ab?
- Ziele:
  - Implementierung, Training und Testen einer ML-Methode
  - Vergleich der Lösungen der Heuristiken mit den optimalen Lösungen

Stand der Forschung

- Das TSP gilt als einer der meist erforschten kombinatorischen Optimierungsprobleme [1]
- TSP ist ein:
  - Graphenproblem $G = (V, E)$
  - Knotenmenge $V = \{1, \dots, n\}$
  - Kantenmenge $E = \{(u, v) \mid u, v \in V, u \neq v\}$
  - Gewichtungsfunktion $w(e) \rightarrow Z^+$ [1]
- Tour finden, die:
  - Alle Knoten maximal einmal besucht
  - Im Ursprungsknoten endet
  - Die Summe der Gewichte minimiert

Traditionelle Ansätze zur Lösung des Traveling Salesman Problems

- Brute Force
  - Naiv
  - Berechnet optimale Lösung
  - Nicht effizient für große Knotenmengen
- Branch-and-Cut [2]
  - Berechnet Teilgraphen
  - Berechnet eine untere Schranke für die optimale Lösung
- Heuristiken [3]
  - Berechnen nicht die optimale Lösung
  - Berechnen Lösung in effizienter Zeit (die meisten in $O(n^2)$)
  - Relativ einfach zu implementieren
- Dynamische Programmierung [4]
  - TSP wird in kleinere Teilprogramme aufgeteilt
  - Hohe Laufzeit- und Speicheranforderung für größere Problemgrößen

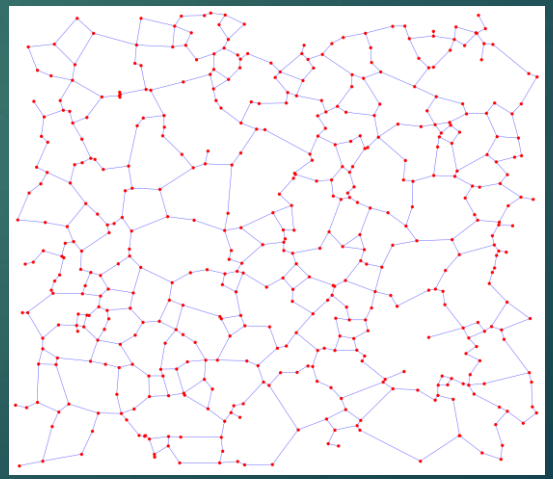
Machine Learning Ansätze zur Lösung des Traveling Salesman Problems

- Reinforcement Learning [5]
  - Training eines Agenten oder Lernalgorithmus
  - Schrittweise Entscheidung zur Auswahl des nächsten Knotens
  - Lernt durch Belohnungen und Bestrafung
- Genetischer Algorithmus [6]
  - Generierung einer zufälligen Population von Touren
  - Die Touren werden durch den Crossover und der Mutation verbessert
  - Der Fitnessscore entscheidet, welche Touren in die nächste Generation weiter genutzt werden
- Neuronale Netze [7]
  - Trainiert eine Funktion, die die optimale Reihenfolge berechnet
  - Als Input dienen die Knoten des TSP
  - Das NN liefert eine Permutation von Knoten

Methodik

- Die verwendeten Daten
  - 1000 TSPs wurden generiert
  - 1000 optimale Lösungen berechnet
  - Je 1000 Lösungen für 8 Heuristiken berechnet
  - Die Größe der TSPs liegen zwischen 300 und 500 Knoten
- Die Analyse der Daten
  - Approximationsgüte $\frac{heu - opt}{opt} * 100$
  - Der Prozentsatz der Kanten, die in der optimalen Lösung liegen $\frac{amount}{dimension} * 100$
  - MSTs als untere Schranke [8]
  - k-RNG
    1. Anzahl optimale Kanten im 1-RNG
    2. Anzahl Kanten des Greedy Algorithmus im 1-RNG
    3. Anzahl optimale Kanten im k-RNG

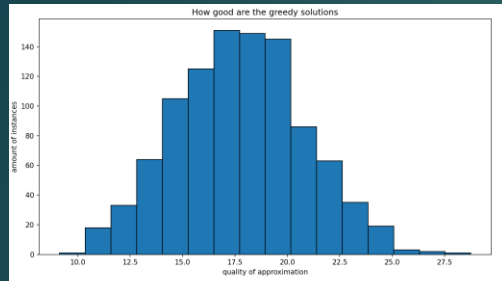
Verfahren	Typ
Greedy Algorithmus	Einfügeheuristik
Nearest Neighbor	Eröffnungsheuristik
Farthest Insertion	Einfügeheuristik
Random Insertion	Einfügeheuristik
Nearest Insertion	Einfügeheuristik
Cheapest Insertion	Einfügeheuristik
MST Heuristik	Eröffnungsheuristik
Christofides Algorithmus	Eröffnungsheuristik

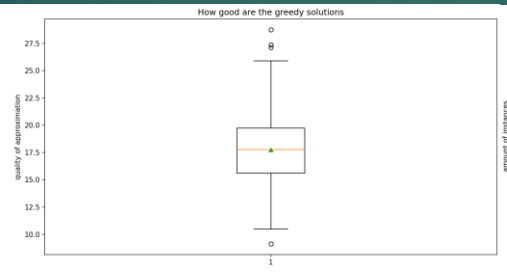


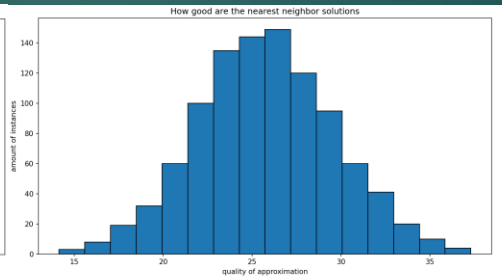
Die ausgewählten Merkmale

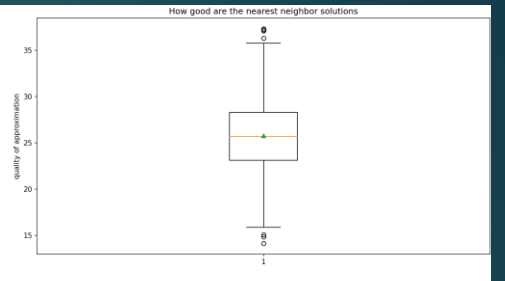
- Merkmale aus den Papers
  - Merkmal aus der linearen Programmierung[1]
  - Merkmal aus MSTs[1]
  - Lokale Merkmale[1]
  - Merkmal aus reduzierten Graphen[1]
  - Merkmale aus Graphen[9]
  - Statistische Merkmale[9]
- Weitere Merkmale
  - Merkmale aus den Heuristiken
  - Merkmale mithilfe von k-RNGs
  - MST zur Wahrscheinlichkeit des Vorkommens einer Kante
- Random Forest Tree
  - 50% Greedy Touren zum Training
  - 50% Greedy Touren zum Testen
  - Optimalen Touren zur Evaluierung und Klassifikation

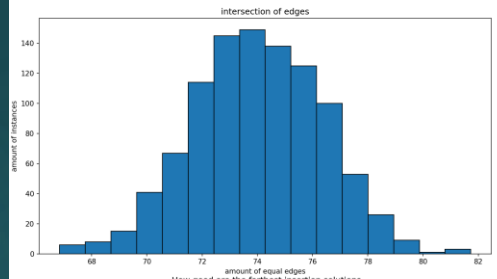
Ergebnisse

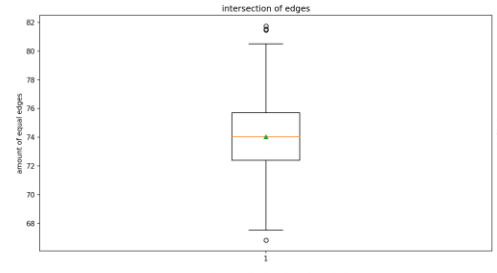

How good are the greedy solutions
amount of instances
quality of approximation

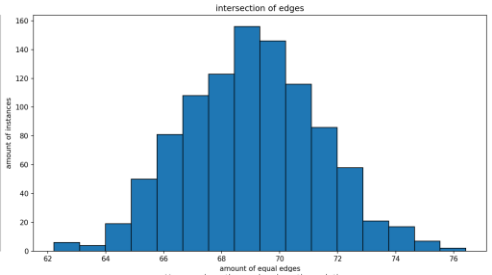

How good are the greedy solutions
quality of approximation

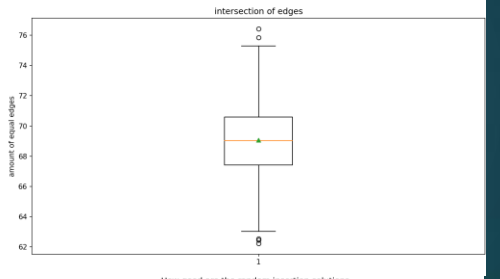

How good are the nearest neighbor solutions
amount of instances
quality of approximation

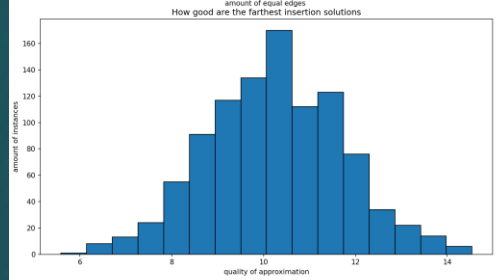

How good are the nearest neighbor solutions
quality of approximation


intersection of edges
amount of instances
amount of equal edges

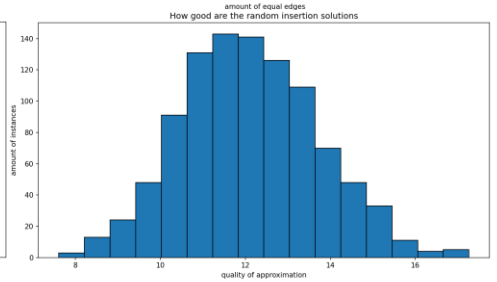

intersection of edges
amount of equal edges


intersection of edges
amount of instances
amount of equal edges

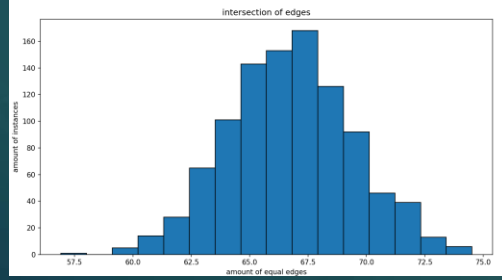

intersection of edges
amount of equal edges

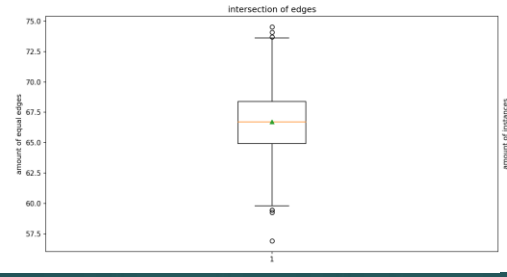

How good are the farthest insertion solutions
amount of instances
quality of approximation

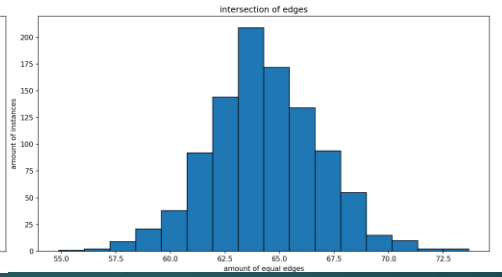

How good are the farthest insertion solutions
quality of approximation

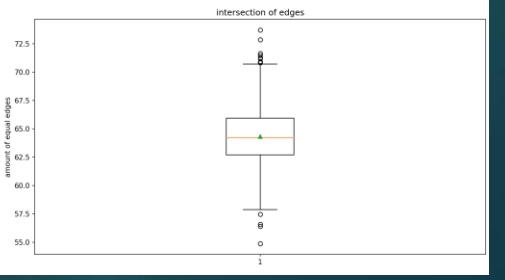

How good are the random insertion solutions
amount of instances
quality of approximation


How good are the random insertion solutions
quality of approximation

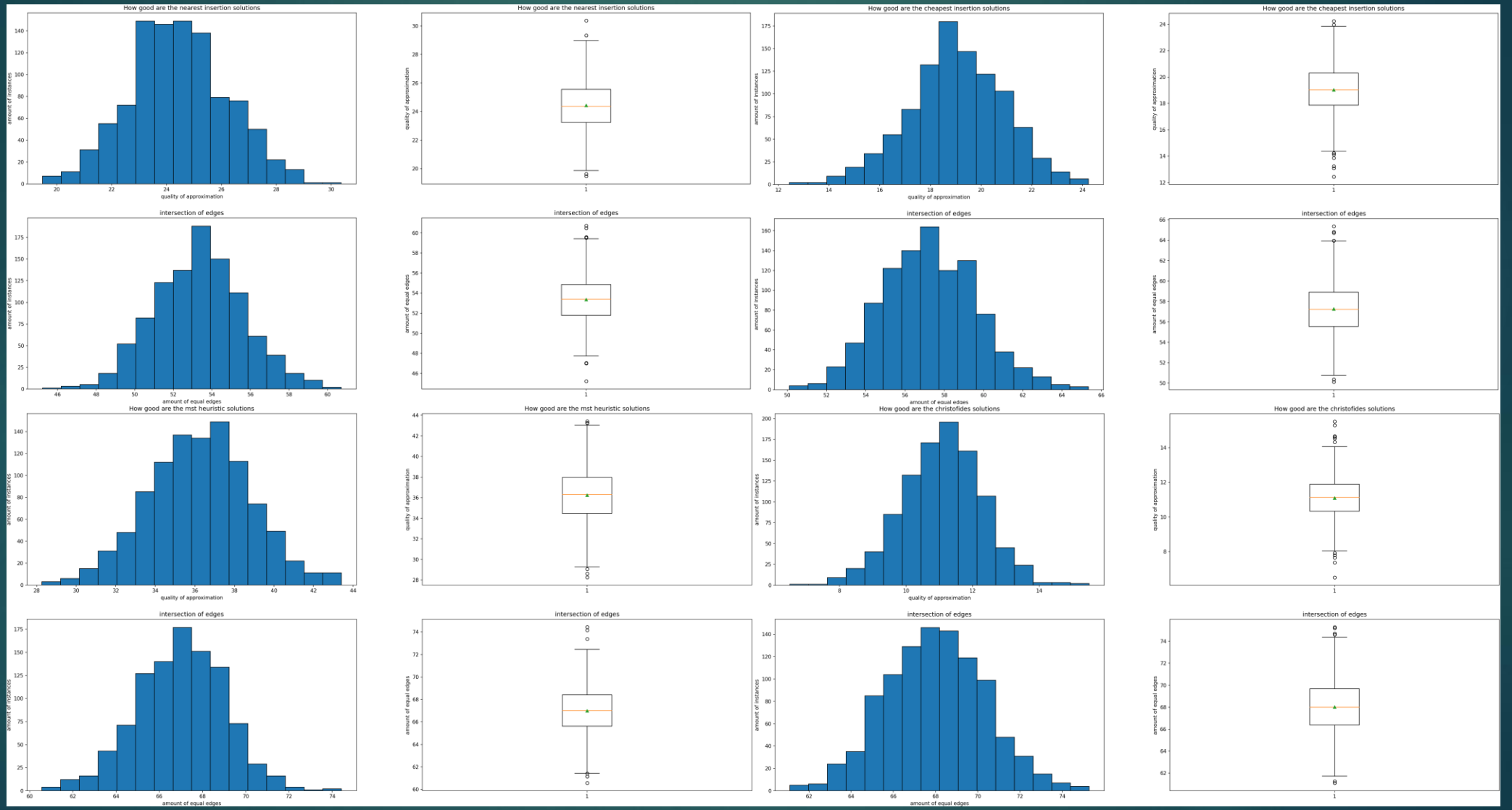

intersection of edges
amount of instances
amount of equal edges


intersection of edges
amount of equal edges


intersection of edges
amount of instances
amount of equal edges

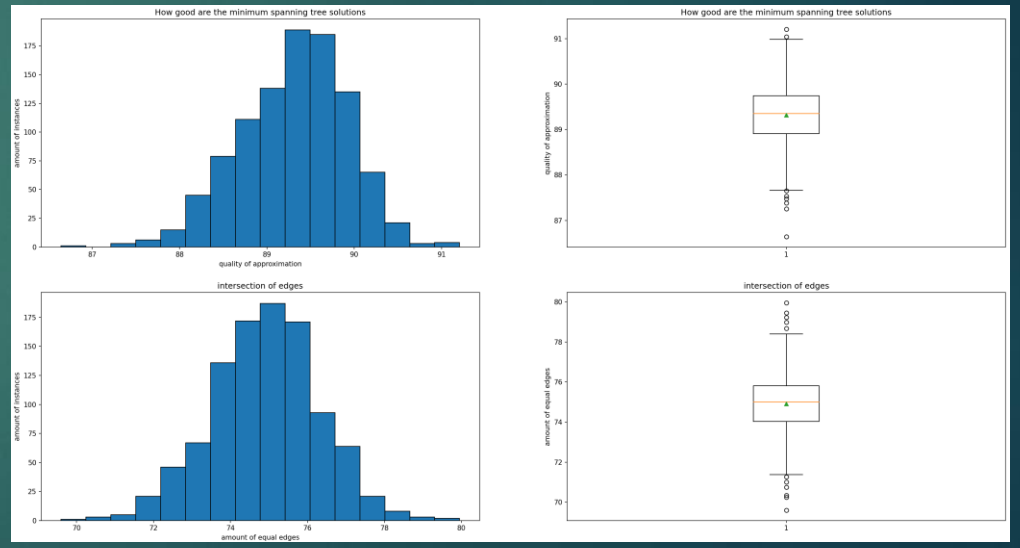

intersection of edges
amount of equal edges

Ergebnisse

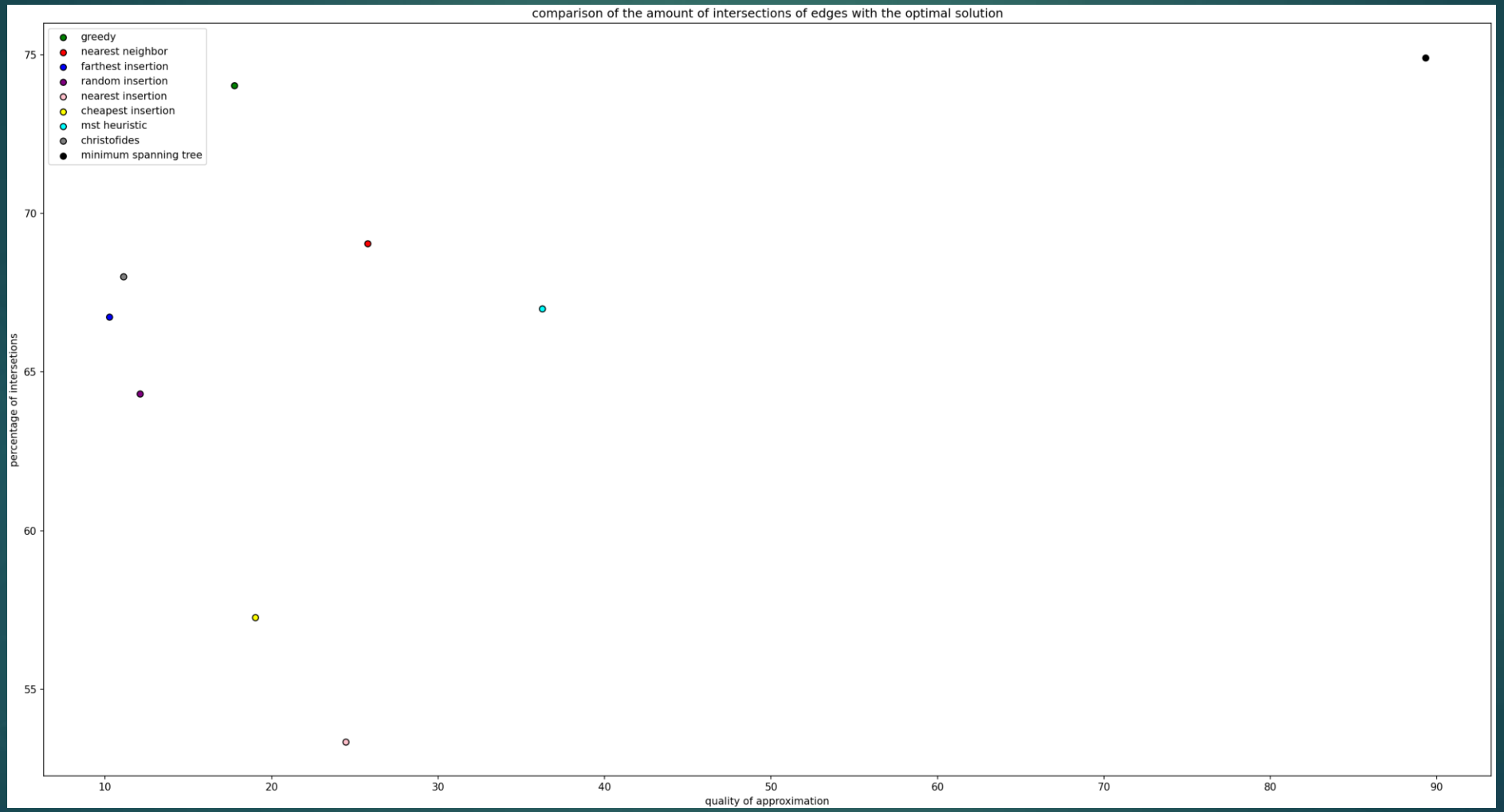

How good are the nearest insertion solutions
amount of instances
quality of approximation
How good are the nearest insertion solutions
quality of approximation
How good are the cheapest insertion solutions
amount of instances
quality of approximation
How good are the cheapest insertion solutions
quality of approximation
intersection of edges
amount of instances
amount of equal edges
intersection of edges
amount of equal edges
intersection of edges
amount of instances
amount of equal edges
intersection of edges
amount of equal edges
How good are the mst heuristic solutions
amount of instances
quality of approximation
How good are the mst heuristic solutions
quality of approximation
How good are the christofides solutions
amount of instances
quality of approximation
How good are the christofides solutions
quality of approximation
intersection of edges
amount of instances
amount of equal edges
intersection of edges
amount of equal edges
intersection of edges
amount of instances
amount of equal edges
intersection of edges
amount of equal edges

Ergebnisse

- Die Analyse der Heuristiken hat gezeigt, dass:
  - Farthest Insertion hat die beste Approximationsgüte mit 10,1%
  - Die meisten optimalen Kanten hat der Greedy Algorithmus mit 74%
  - Nearest Insertion hat die wenigsten optimalen Kanten mit 53,8%
  - MST Heuristik hat die schlechteste Approximationgüte mit 36,5%
- Analyse der MSTs
  - Die Approximationsgüte gilt als untere Schranke der TSPs
  - Approximationsgüte liegt bei 89,5%
  - Der Prozentsatz der optimalen Kanten liegt bei 75,5%

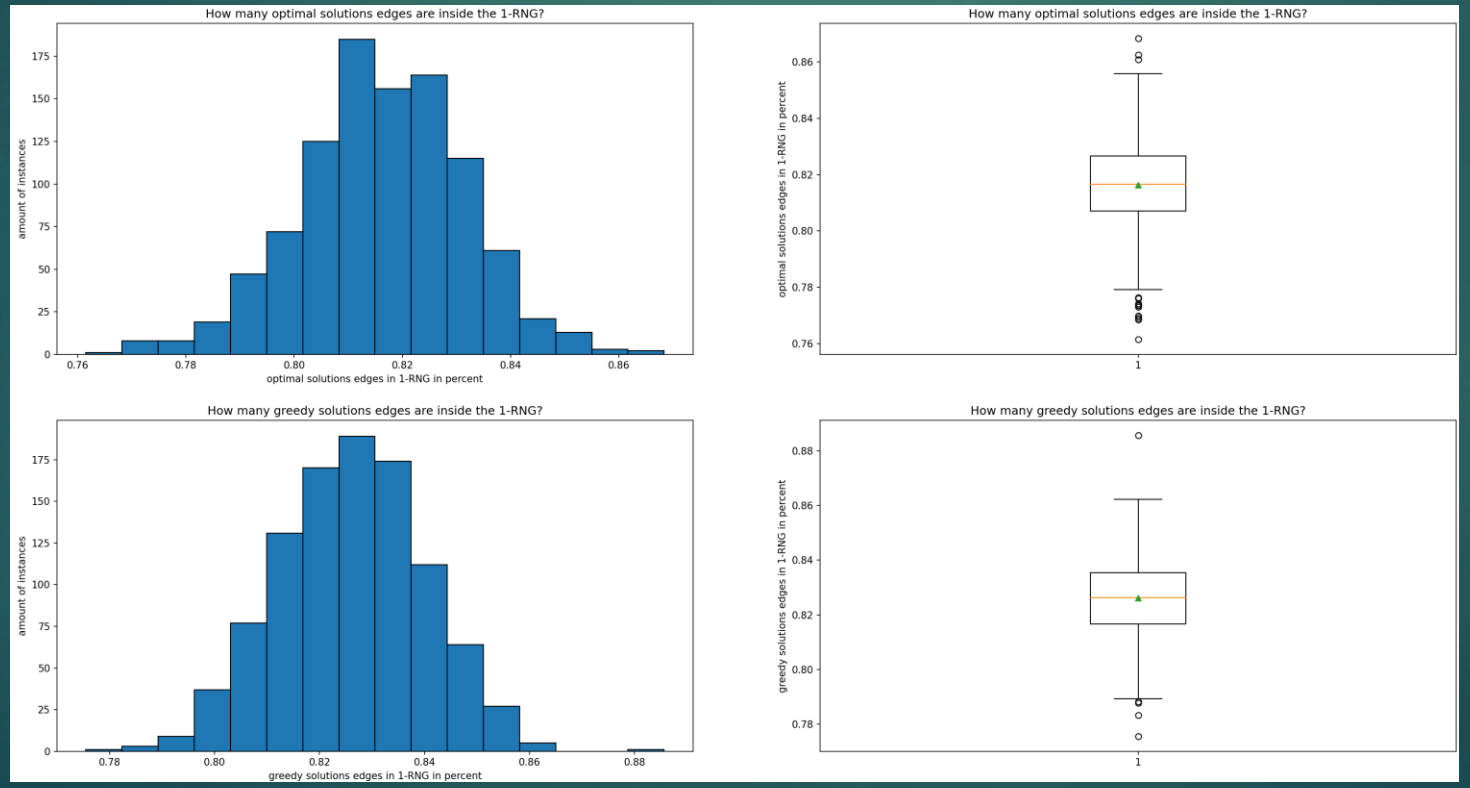


Ergebnisse


comparison of the amount of intersections of edges with the optimal solution
greedy
nearest neighbor
farthest insertion
random insertion
nearest insertion
cheapest insertion
mst heuristic
christofides
minimum spanning tree
percentage of intersetions
quality of approximation
75
70
65
60
55
10
20
30
40
50
60
70
80
90

Ergebnisse

- Analyse der 1-RNGs
  - 81,8% der optimalen Kanten liegen im 1-RNG
  - 82.7% der Kanten des Greedy Algorithmus liegen im 1-RNG



Ergebnisse

k	absoluten Zahlen	Prozentanteile
0	328339	81.87
1	54893	13.69
2	13051	3.25
3	3407	0.85
4	974	0.24
5	288	0.07
6	80	0.02
7	29	0.01
8	8	0
9	1	0
10	0	0
11	1	0

Schluss

- Merkmale und RFT nicht impementiert
- Forschungsfragen:
  - ML Lösungen > Lösungen Heuristiken? ✗
  - Wie schneiden Heuristiken untereinander ab? ✓
- Ziele:
  - Implementierung, Training und Testen einer ML-Methode ✗
  - Vergleich der Lösungen der Heuristiken mit den optimalen Lösungen ✓

Zusammenfassung

- Kurze Erklärung zum Thema
- Überblick über die Ziele und Forschungsfragen der Arbeit
- TSP kurz erklärt
- Traditionelle- und Machine Learning Ansätze zur Lösung vorgestellt
- Methodik der Analysen vorgestellt
- Merkmale für den Machine Learning Ansatz vorgestellt
- Ergebnisse der Analysen vorgestellt
- Aufgezeigt welche Ziele und Forschungsfragen erreicht bzw. beantwortet wurden

Danke für ihre Aufmerksamkeit!

Literaturverzeichnis

[1] J. Fitzpatrick, D. Ajwani, and P. Carroll, “Learning to sparsify travelling salesman problem instances,” 2021.

[2] S. Ait Bouziaren and B. Aghezzaf, “An improved augmented ε -constraint and branch-andcut method to solve the tsp with profits,” IEEE Transactions on Intelligent Transportation Systems, vol. 20, pp. 195–204, Jan 2019.

[3] D. Nuraiman, F. Ilahi, Y. Dewi, and E. A. Z. Hamidi, “A new hybrid method based on nearest neighbor algorithm and 2-opt algorithm for traveling salesman problem,” in 2018 4th International Conference on Wireless and Telematics (ICWT), pp. 1–4, 2018.

[4] V. B. Lobo, B. B. Alengadan, S. Siddiqui, A. Minu, and N. Ansari, “Traveling salesman problem for a bidirectional graph using dynamic programming,” in 2016 International Conference on Micro-Electronics and Telecommunication Engineering (ICMETE), pp. 127–132, 2016.

Literaturverzeichnis

[5] H. Yang and M. Gu, “A new baseline of policy gradient for traveling salesman problem,” in 2022 IEEE 9th International Conference on Data Science and Advanced Analytics (DSAA), pp. 1–7, 2022.

[6] C. Ding, Y. Cheng, and M. He, “Two-level genetic algorithm for clustered traveling salesmanproblem with application in large-scale tsps,” Tsinghua Science and Technology, vol. 12, no. 4, pp. 459–465, 2007.

[7] Z. Xing and S. Tu, “A graph neural network assisted monte carlo tree search approach to traveling salesman problem,” IEEE Access, vol. 8, pp. 108418–108428, 2020.

[8] G. Reinelt, “TSPLIB—A Traveling Salesman Problem Library,” INFORMS Journal on Computing, vol. 3, pp. 376–384, November 1991.

[9] Y. Sun, A. Ernst, X. Li, and J. Weiner, “Generalization of machine learning for problem reduction: a case study on travelling salesman problems,” OR Spectrum, vol. 43, pp. 607–633, sep 2020.

Literaturverzeichnis

[10] A. Hagberg, P. Swart, and D. S Chult, “Exploring network structure, dynamics, and function using networkx,” Technical report, 2008.