Proceedings of the 12th Student Computing Research Symposium

Izola, Slovenia
October 8, 2026

Nina Chiarelli
Matjaž Krnc
Domen Šoberl
(Eds.)



https://www.scores.si

Title Proceedings of the 12th Student Computing Research Symposium (SCORES'26)

Izola, Slovenia
October 8, 2026

Editors Nina Chiarelli (UP FAMNIT)
Matjaž Krnc (UP FAMNIT)
Domen Šoberl (UP FAMNIT)

Conference 12th Student Computing Research Symposium (SCORES'26)

Venue University of Primorska
InnoRenew CoE
Livade 6a, 6310 Izola, Slovenia

Published by University of Primorska Press
hippocampus.si
Koper · 2026

Co-published by UM FERI · feri.um.si
UL FRI · fri.uni-lj.si

Program Committee Klemen Berkovič (UM)
Marko Bizjak (UM)
Lucija Brezočnik (UM)
Andrej Brodnik (UL, UP)
Nina Chiarelli (UP)
Uroš Čibej (UL)
Jani Dugonik (UM)
Iztok Fister (UM)
Iztok Fister Jr. (UM)
Luka Fürst (UL)
Branko Kavšek (UP)
Štefan Kohek (UM)
Miklós Krész (InnoRenew CoE)
Matjaž Krnc (UP)
Niko Lukač (UM)
Martin Milanič (UP)
Uroš Mlakar (UM)
Damijan Novak (UM)
Eneko Osaba (TECNALIA)
Jan Popič (UM)
Peter Rogelj (UP)
Domen Šoberl (UL)
Lovro Šubelj (UL)
Grega Vrbančič (UM)

Organizing Committee Nina Chiarelli (UP FAMNIT)
Matjaž Krnc (UP FAMNIT)
Domen Šoberl (UP FAMNIT)

Free Electronic Edition
https://www.hippocampus.si/ISBN/978-961-293-616-7.pdf
https://doi.org/10.26493/scores26

CC BY © Authors

Kataložni zapis o publikaciji (CIP) pripravili
v Narodni in univerzitetni knjižnici v Ljubljani

COBISS.SI-ID 293411587
ISBN 978-961-293-616-7 (PDF)





Proceedings of the 12th Student Computing Research Symposium (SCORES'26)

Izola, Slovenia
October 8, 2026

Nina Chiarelli
Matjaž Krnc
Domen Šoberl
(Eds.)

Editors' Foreword

In the ever-evolving landscape of computer science, where innovation knows no bounds, we welcome you to the 12th Student Computing Research Symposium (SCORES 2026). This annual gathering is a testament to the unwavering commitment of our four initial organizing institutions: the University of Ljubljana, the University of Maribor, the University of Primorska (UP FAMNIT), and ACM Slovenia. This year, we are also thrilled to host the symposium in a new, inspiring setting at the state-of-the-art InnoRenew CoE in Izola.

Recent years have witnessed remarkable strides in computer science, fundamentally accelerated by the artificial intelligence revolution. As AI and machine learning continue to reshape industries, scientific discovery, and daily life, nurturing the next generation of computer scientists becomes more critical than ever. SCORES 2026 is our platform to empower undergraduate and master's students to embrace these technologies and contribute their unique insights to the ever-expanding frontiers of computer science.

Our mission is clear: to provide a stage where these talented students can showcase their research, ideas, and innovations. The symposium is about bridging the gap between academic theory and real-world applications. It is an accessible platform for young minds to experience the scientific peer-review process, present their findings, and build lasting networks with their peers and mentors.

This year's conference program boasts a rich diversity of high-quality submissions that beautifully capture the current zeitgeist of the field. The papers featured here offer fresh perspectives and highly practical solutions: from leveraging deep reinforcement learning for multi-agent traffic control and strategic game theory, to pioneering accessible melanoma screening on mobile devices and deep learning for bridge weigh-in-motion systems. Our student authors have tackled complex theoretical challenges in graph neural networks and algorithms, advanced the processing of endangered languages using large language models, and optimized natural language speech models.

As we embark on this exciting journey—highlighted by keynote discussions exploring the multifaceted impacts of AI—we invite you to join us in celebrating the dedication, creativity, and accomplishments of these emerging computer scientists. Their work represents the bright future of our field, and the Student Computing Research Symposium is the platform where it all comes to life.

SCORES 2026 organizing committee:
Nina Chiarelli, Matjaž Krnc, Domen Šoberl

 DOI: https://doi.org/10.26493/scores26

Conference Program

Plenary Speakers **1**

1 The Evolving Impact of AI on the Practice of Mathematics
Eric Gottlieb

2 Research and Development in the Video Game Industry
Domen Vake

Multi-Agent Systems and Reinforcement Learning **2**

5 Primerjava strategij sprotnega učenja z uglednim mehanizmom v iterativni zapornikovi dilemi in igri piščanca
Nik Bržan, David Bujić and Klemen Berkovič

9 Večagentno prilagodljivo upravljanje semaforjev z globokim okrepitvenim učenjem in medsebojno komunikacijo
Miha Pliberšek, Enej Krabonja, Oskar Forštnarič, Anže Hameršak, Luka Antolinc and Borko Boškovič

13 Hibridni pristop večagentnega iskanja poti s toplotno karto
Nik Damiš, Nik Drozg, Pavel Medved, Igor Polajžer, Blaž Vidovič and Klemen Berkovič

17 Večagentno odločanje v igri Pac-Man z uporabo algoritmov tekmovalnega iskanja
Irina Mićović, Sara Hristova, Borko Bošković and Klemen Berkovič

Natural Language Processing, Speech and Human-Computer Interaction **21**

23 Heuristic Evaluation of Mobile Web E-Commerce Checkout: A Domain-Specific Binary Checklist and Scoring Method
Olha Tushych and Alla Toropenko

27 Sestava specializiranih modelov za diarizacijo in transkripcijo slovenskega govora
Aljoša Golob, Matej Rajter, Jan Vališer, Nejc Fras and Jan Popič

31 Obdelava ogroženih jezikov z velikimi jezikovnimi modeli na osnovi jezikoslovnih virov
Enej Krabonja, Miha Pliberšek, Tjaša Lah, Oskar Forštnarič, Maks Marolt and Borko Boškovič

35 Geometric Characterization of Embedding Spaces Across Chunking Strategies and Embedding Models for Slovenian Text
Anja Jodl and Mitko Nikov

Algorithms, Graphs and Theoretical Computer Science **39**

41 Resilient Incremental LL(1) Parsing Based on Li's Algorithm
Denis Balant

45 Marker Diagrams for String Rewriting Systems
Janez Ignacij Jereb

49 Encoding Graphs without Gradients
Matic Požar

53 Public Transport Anchored Flow Termini Coverage Model for Dock Placement in Bicycle Sharing Systems
Dmytro Tupkalenko, Daniil Baldouski and Miklós Krész

57 Two-Step Relaxation for All-Pairs Shortest Paths in Sparse Graphs
Dani Zugan

Applied Machine Learning, Computer Vision and Signal Processing **61**

63 Deep Learning-based Axle Detection from B-WIM strain signals
David Chonev and Domen Šoberl

67 Toward Accessible Melanoma Screening on Mobile Devices: A Three-Tier Feature-Fusion CAD Prototype
Vuk Ignjatović, Nedim Šišić and Peter Rogelj

71 Cepstral Features for EEG Classification: Comparison with Spectral Features
Luka Matič and Peter Rogelj

75 Attack Detection on Graph Neural Networks
Diana Ioana Catana, Caio Simon, Đorđe Stevanović and Pietro Sestito

Index of Authors **81**

 DOI: https://doi.org/10.26493/scores26

Plenary Speakers

Eric Gottlieb

Associate Professor
Department of Mathematics and Statistics
Rhodes College, Memphis
TN 38112, USA

The Evolving Impact of AI on the Practice of Mathematics

In the span of a few months, the mathematical abilities of AI have evolved from mindless and error-prone parroting to indisputably impressive. The emergence of AI as a powerful mathematical force is already changing the way mathematicians teach and practice their discipline. It is impossible to know with certainty where these changes will lead us. In this talk, I will share some recent mathematical AI accomplishments, some observations about how the use of AI as a tool has affected the teaching and doing of mathematics, some controversies and concerns, and some speculations about what the future might hold for mathematics.



Eric Gottlieb is an Associate Professor in the Department of Mathematics and Statistics at Rhodes College. His training is in algebraic, enumerative, and topological combinatorics, but he has also published in combinatorial game theory, Ramsey theory, and graph theory. A believer in the cultural and educational benefits of international scholarly exchanges, he was a Fulbright Scholar at the University of Talca in Chile from 2004 - 2005 and is currently on Fulbright at the University of Primorska. He is an advocate for the liberal arts educational model.

 DOI: https://doi.org/10.26493/scores26

Domen Vake

PhD student
Department of Information Sciences and Technologies
FAMNIT and InnoRenew CoE
Slovenia

Bridging the Question–Answer Gap in Retrieval-Augmented Generation: Hypothetical Prompt Embeddings

Retrieval-Augmented Generation (RAG) systems synergize retrieval mechanisms with generative language models to enhance the accuracy and relevance of responses. However, bridging the style gap between user queries and relevant information in document text remains a persistent challenge in retrieval-augmented systems, often addressed by runtime solutions (e.g., Hypothetical Document Embeddings (HyDE)) that attempt to improve alignment but introduce extra computational overhead at query time. To address these challenges, we propose Hypothetical Prompt Embeddings (HyPE), a framework that shifts the generation of hypothetical content from query time to the indexing phase. By precomputing multiple hypothetical prompts for each data chunk and embedding the chunk in place of the prompt, HyPE transforms retrieval into a question-question matching task, bypassing the need for runtime synthetic answer generation. This approach does not introduce latency but also strengthens the alignment between queries and relevant context. Our experimental results on six common datasets show that HyPE can improve retrieval context precision by up to 42 percentage points and claim recall by up to 45 percentage points, compared to standard approaches, while remaining compatible with re-ranking, multi-vector retrieval, query decomposition, and other RAG advancements.



Domen Vake received the B.S. and M.S. degrees in computer science from the University of Primorska (UP), Slovenia. He is currently pursuing the Ph.D. degree in computer science with the Faculty of Mathematics, Natural Sciences and Information Technologies (FAMNIT), Department of Information Sciences and Technologies (DIST). Previously, he was a Programmer with InnoRenew CoE. He is an Assistant Researcher with InnoRenew CoE and UP FAMNIT. His research interests include artificial intelligence with a larger focus on large language models and their usage.

Multi-Agent Systems and Reinforcement Learning

Primerjava strategij sprotnega učenja z uglednim mehanizmom v iterativni zapornikovi dilemi in igri piščanca

Nik Bržan, David Bujić and Klemen Berkovič

https://doi.org/10.26493/scores26.02

Večagentno prilagodljivo upravljanje semaforjev z globokim okrepitvenim učenjem in medsebojno komunikacijo

Miha Pliberšek, Enej Krabonja, Oskar Forštnarič, Anže Hameršak, Luka Antolinc and Borko Boškovič

https://doi.org/10.26493/scores26.05

Hibridni pristop večagentnega iskanja poti s toplotno karto

Nik Damiš, Nik Drozg, Pavel Medved, Igor Polajžer, Blaž Vidovič and Klemen Berkovič

https://doi.org/10.26493/scores26.08

Večagentno odločanje v igri Pac-Man z uporabo algoritmov tekmovalnega iskanja

Irina Mićović, Sara Hristova, Borko Bošković and Klemen Berkovič

https://doi.org/10.26493/scores26.15

 DOI: https://doi.org/10.26493/scores26

 DOI: https://doi.org/10.26493/scores26.02

Primerjava strategij sprotnega učenja z uglednim mehanizmom v iterativni zapornikovi dilemi in igri piščanca

Nik Bržan
nik.brzan@student.um.si
Fakulteta za elektrotehniko, računalništvo in informatiko, Univerza v Mariboru
Koroška cesta 46
SI-2000 Maribor, Slovenija

David Bujić
david.bujic@student.um.si
Fakulteta za elektrotehniko, računalništvo in informatiko, Univerza v Mariboru
Koroška cesta 46
SI-2000 Maribor, Slovenija

Klemen Berkovič
klemen.berkovic1@um.si
Fakulteta za elektrotehniko, računalništvo in informatiko, Univerza v Mariboru
Koroška cesta 46
SI-2000 Maribor, Slovenija

POVZETEK

Iterativna zapornikova dilema in igra piščanca sta dve najpogosteje preučevani igri, ki modelirata konflikte med lastnimi interesi in skupno koristjo v večagentnih sistemih. V tem članku razširimo eksperiment, ki implementira Axelrodov turnir iterativne zapornikove dileme, z uvedbo uglednega mehanizma in turnirja igre piščanca. Predlagani mehanizem agentom pred vsakim dvobojem posreduje agregirano informacijo o kooperativni zgodovini nasprotnika v obliki reputacijskega koeficienta. Eksperiment izvedemo v obliki krožnega turnirja z $N = 16000$ krogov na dvoboj za obe igri. Rezultate vrednotimo s stopnjo kooperativnosti. Ugotovitve pokažejo, da ugledni mehanizem pomembno vpliva na kooperativno vedenje agentov v turnirju. V iterativni zapornikovi dilemi uvedba ugleda presenetljivo zmanjša stopnjo kooperativnosti predvsem pri kontekstualnih banditih, saj začetna vrednost sproži preventivno defekcijo in zanko nezaupanja. V iterativni igri piščanca pa je ta negativni učinek še izrazitejši, predvsem pri kontekstualnih banditih.

KLJUČNE BESEDE

Sprotno učenje, iterativna zapornikova dilema, igra piščanca, ugledni mehanizem, večagentni sistem

1 UVOD

Večagentni sistemi (angl. *Multi-Agent Systems*) so danes prisotni na številnih področjih vsakodnevne uporabe. Primeri teh področij so biologija, ekonomija, poleg tega pa problematiko zasledimo tudi v vsakodnevni uporabi, specifično pri prenašanju datotek z uporabo enakovrednega omrežja (angl. *Peer-to-peer*) [3, 9]. Najpomembnejše vprašanje pri načrtovanju takšnih sistemov je, kako se med seboj vedejo agenti, kadar njihovi cilji niso povsem usklajeni. Teorija iger analizira tovrstne interakcije, pri čemer sta iterativna zapornikova dilema (angl. *Iterated Prisoner's Dilemma*, IPD) in igra piščanca (angl. *Game of Chicken*) dve najpogosteje preučevani igri [4, 9]. Definicijo iterativne zapornikove dileme [10] lahko zapišemo v matrični obliki in sicer:

$$G = [N, \{A_i\}_{i \in N}, \{R_i\}_{i \in N}]. \tag{1}$$

V definiciji N predstavlja nabor agentov, medtem ko A_i predstavlja akcije, ki jih lahko i-ti agent izvede. R_i je nagradna funkcija i-tega agenta. Osnoven primer je igra dveh agentov, kjer so možne akcije C (kooperacija) in D (prevara), rezultati pa so T (temptacija), R (nagrada), P (kazen), S (bedak). Za dane akcije in nagrade velja:

$$T > R > P > S \quad \text{in} \quad 2R > T + S. \tag{2}$$

Te so v razmerju, ki je vidno v Tabeli 1.

	C	D
C	R, R	S, T
D	T, S	P, P

Tabela 1: Matrika izplačil za zapornikovo dilemo dveh igralcev

Igra piščanca [4] je podobna zapornikovi dilemi, vendar se razlikuje v tem, da je sodelovanje, kadar nasprotnik uporabi prevaro, boljša izbira, kot vzajemna prevara. Matrika akcij in izidov je identična tabeli 1, vendar je razmerje med vrednostmi izidov drugačno:

$$T > R > S > P. \tag{3}$$

Obe igri modelirata situacije iz resničnega sveta, v katerih se agenti soočijo s konfliktom med lastnimi interesi in skupno koristjo. V enkratni igri je za vsakega agenta optimalno, da izda nasprotnika, v iterativni obliki pa se ustvarijo pogoji, v katerih se kooperativnost lahko razvije in vzdržuje skozi čas.

V zadnjem desetletju so metode sprotnega učenja (angl. *Online Learning*) postale privlačen pristop za modeliranje vedenja agentov v takšnih dilemah. Avtorji članka [8] so z uporabo evolucijskega algoritma in algoritma rojev delcev (angl. *particle swarm*) v okvirju knjižnice Axelrod-Python trenirali strategije na naboru več kot 170 nasprotnikov. Ugotovili so, da naučene strategije v standardnih turnirjih dosledno premagajo vse klasične strategije, vključno s strategijo *Tit-for-Tat*. Ključna skupna lastnost najboljših strategij je bila upoštevanje prvih potez nasprotnika, kjer so strategije z globokim spominom in pogojevanjem na začetno vedenje nasprotnika dosegale bistveno boljše rezultate. Avtorji članka [10] so nato sistematično primerjali celoten spekter algoritmov sprotnega učenja, ki gre od večroknih banditov (angl. *multi-armed bandits*) prek kontekstualnih banditov do Q-učenja. Algoritmi so bili primerjani v IPD turnirju z več agenti. Ugotovili so, da so kontekstualni banditi, ki pri odločanju upoštevajo le trenutno stanje, najslabši pristop v tej vrsti dileme. Večrokni banditi brez konteksta po drugi strani najboljše posnemajo človeško vedenje v IPD, kar nakazuje, da ljudje pri odločanju v tej igri morda ne upoštevajo trenutnega konteksta. Človeška stopnja kooperativnosti v nobenem primeru ni dosežena, kar odpira vprašanje, kateri dodatni mehanizmi bi to omogočili.

Omenjena ugotovitev je motivirala osrednjo idejo tega članka. V delu [1] so avtorji pokazali, da signali o ugledu vodijo do višje stopnje kooperativnosti pri agentih z okrepitvenim učenjem, v delu [12] pa so kombinacijo Q-učenja in uglednega mehanizma uspešno aplicirali v scenariju zapornikove dileme. Avtorji dela [6] so demonstrirali, da globina spomina agenta neposredno vpliva na uspešnost strategije v iterativnih igrah. Na podlagi teh ugotovitev predlagamo razširitev eksperimenta [10] z uvedbo *uglednega mehanizma*, ki agentom pred vsako tekmo posreduje agregirano informacijo o kooperativni zgodovini nasprotnika. Pričakovano je, da bo ugledni mehanizem povzročil povečanje stopnje kooperativnosti v primerjavi z osnovnim eksperimentom.

Preostali del članka je strukturiran kot sledi. V poglavju 2 so predstavljena sorodna dela s področja IPD, igre piščanca in uglednih mehanizmov. Poglavje 3 opisuje ugledni mehanizem. Eksperimentalno okolje in metrike vrednotenja so razloženi v poglavju 4, analiza rezultatov pa v poglavju 5. Poglavje 6 zaključi članek s povzetkom ugotovitev.

2 PREGLED SORODNIH DEL

V članku [12] avtorji z uporabo večagentnega okrepitvenega učenja (angl. *Multi-agent reinforcement learning*) odkrijejo novo strategijo »memory-two bilateral reciprocity« (MTBR), ki se je izkazala boljša kot velik del klasičnih strategij. Pokažejo tudi, da globina spomina neposredno vpliva na uspešnost strategije, pri čemer MTBR izgubi prednost v igrah, kjer velja $2R = T + S$.

Članek [6] modelira zapornikovo dilemo, v kateri ugled agentov določa njihovo migracijsko vedenje. Avtorji so predlagali kvantitativni ugledni mehanizem, kjer se ugled posameznika posodablja glede na izplačila, ki jih po igri prejmejo nasprotniki. Stopnja pozabljanja preteklih interakcij se spreminja glede na faktor razpada α. Rezultati pokažejo, da faktor razpada neposredno vpliva na stopnjo kooperacije in da migracija, ki jo poganja ugledni mehanizem, pospeši razvoj kooperativnega vedenja.

Avtorji članka [1] preučujejo vlogo ugleda pri kooperativnosti agentov. Agenti se s pomočjo Q-učenja naučijo ocenjevati ugled nasprotnikov na podlagi njihovih predhodnih dejanj in to informacijo uporabijo pri odločanju. Rezultati pokažejo, da informacije o ugledu vodijo do višje stopnje sodelovanja v primerjavi z agenti, ki nimajo dostopa do teh informacij.

Članek [7] opisuje iterativno obliko igre jastreb-golob (angl. *Hawk-Dove game*), ki je matematično ekvivalentna igri piščanca. Ugotovijo, da so najuspešnejše strategije tiste, ki vključujejo kratko fazo ocenjevanja, ki ji sledi stabilno razmerje. Trajanje ocenjevanja je odvisno od stroškov boja in razlike v moči nasprotnikov.

V članku [5] avtorji vpeljejo model okrepitvenega učenja v prostorski različici igre kopicarjev (angl. *Snowdrift game*), ki je strukturno identična igri piščanca. Odkrijejo pojav anti-koordiniranih domen, kjer agenti razvijejo vzorce za preklapljanje med sodelovanjem in prevaro, pri čemer visoka stopnja učenja vodi do nestabilnih prehodnih stanj, nizka pa k dolgotrajni stabilizaciji.

3 UGLEDNI MEHANIZEM

V tem poglavju opisujemo metodološki pristop uglednega mehanizma, ki smo ga uporabili pri razširitvi eksperimenta iterativne zapornikove dileme.

Omejitev osnovnega eksperimenta je, da agenti nimajo dostopa do informacij o zgodovinskem vedenju nasprotnika iz prejšnjih dvobojev. Naša razširitev to omejitev odpravlja z uvedbo uglednega mehanizma, ki agentom zagotavlja informacijo o kooperativni zgodovini nasprotnika pred začetkom dvoboja.

Ugled agenta $g_k \in G$ po $n \leq N$ odigranih krogih v celotnem turnirju definiramo kot delež kooperativnih akcij:

$$\rho_k^{(n)} = \frac{1}{n} \sum_{t=1}^{n} \sigma[a_k^t = C], \tag{4}$$

kjer σ označuje indikatorsko funkcijo, ki vrne vrednost 1, če je pogoj izpolnjen, in 0 sicer. Vrednost $\rho_k \in [0, 1]$ predstavlja delež kooperativnih akcij agenta g_k v vseh dosedanjih krogih, pri čemer $\rho_k = 0$ pomeni, da je agent vedno izdal, $\rho_k = 1$ pa, da je vedno sodeloval.

Agenti, ki podpirajo kontekstualno učenje, prejmejo ugled nasprotnika ρ_j kot dodatno vhodno spremenljivko pred vsakim novim dvobojem. Razširjeno stanje agenta je potem definirano na naslednji način:

$$s_t = (a_j^{t-1},\ \rho_j), \tag{5}$$

kjer $a_j^{t-1} \in \{C, D\}$ označuje zadnjo akcijo nasprotnika v tekočem dvoboju. Informacija o ugledu med dvobojem se sproti osvežuje, saj posodobitev ugleda poteka po vsakem zaključenem krogu.

Ugledni mehanizem ni uporabljen v primeru algoritmov, ki ne upoštevajo konteksta, ker mehanizem nima neposrednega učinka nanje in služijo samo kot referenčna skupina za primerjavo. Posodobitev ugleda poteka po vsakem zaključenem krogu dvoboja.

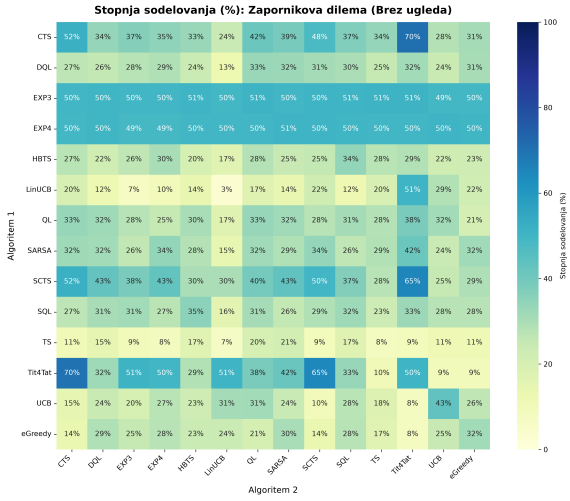
4 EKSPERIMENT

Eksperiment poteka v obliki krožnega turnirja (angl. *round-robin tournament*), v katerem vsak agent $g_k \in \mathcal{G}$ odigra serijo N krogov proti vsakemu drugemu agentu $g_m \in \mathcal{G}$, $m \neq k$. Turnir izvedemo na dva načina: brez uglednega mehanizma, s čimer reproduciramo rezultate članka [10], in z uglednim mehanizmom, ki predstavlja našo nadgradnjo. Enak postopek ponovimo za obe igri.

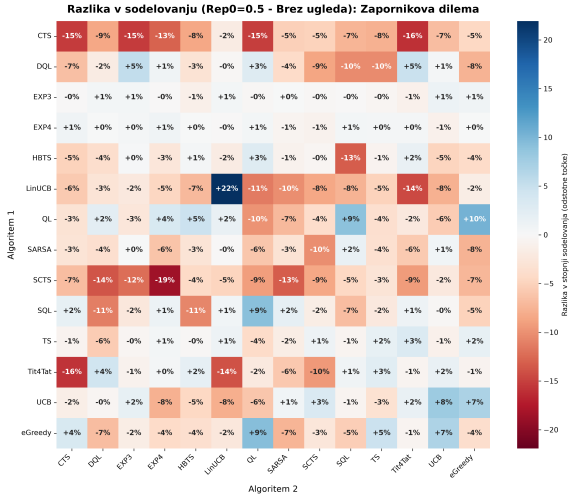
Eksperiment smo implementirali v programskem jeziku Python, pri čemer kot izhodišče uporabimo javno dostopno kodo članka [10], ki je prosto dostopna na repozitoriju GitHub[1]. Ugledni mehanizem smo implementirali kot razširitev obstoječega turnirskega ogrodja. Koda našega eksperimenta je dostopna na repozitoriju GitHub[2].

Standardne vrednosti izplačilne matrike za zapornikovo dilemo so $T = 5$, $R = 3$, $P = 1$, $S = 0$. Za igro piščanca ohranimo enake vrednosti T in R, vrednosti S in P pa zamenjamo tako, da velja pogoj $T > R > S > P$ [4]. Za število krogov na dvoboj smo se odločili na podlagi ugotovitev raziskave [2], ki so pokazali, da se vedenje agentov znotraj velikega razpona bistveno ne spreminja. Ker med 16000 in milijon krogi ni pričakovati opaznih razlik, smo se odločili za $N = 16000$. Izbrana vrednost je dovolj visoka, da agenti dosežejo stabilno kooperativno vedenje, hkrati pa eksperiment ostane računsko obvladljiv in ne potrebuje močne strojne opreme. Parameter m predstavlja globino spomina preteklih potez, ki jo kontekstualni banditi poleg ugleda vključujejo v vektor konteksta. Turnirje izvedemo pri nastavitvah $m = 1$ in $m = 5$.

[1] https://github.com/doerlbh/dilemmaRL
[2] https://github.com/NBrzan/dilemmaRL

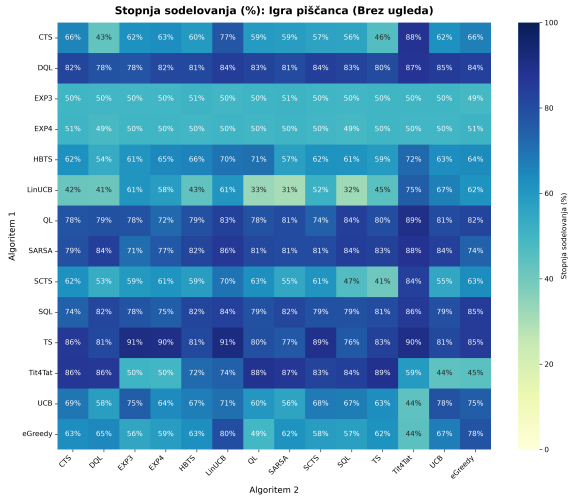


(a)

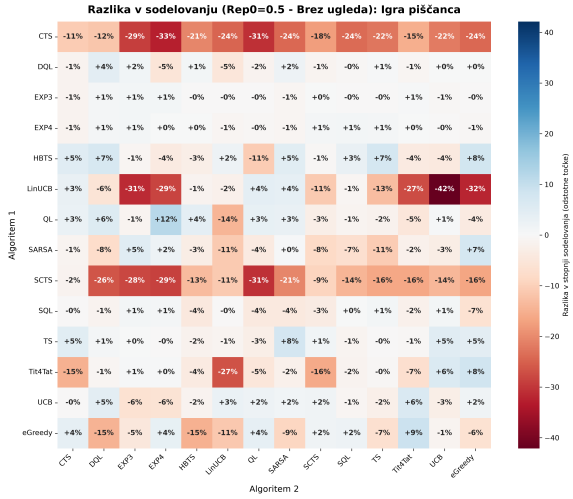


(b)

Slika 1: (a) Prikaz stopnje sodelovanja v iterativni zapornikovi dilemi brez ugleda. (b) Prikaz razlike v stopnji sodelovanja med različicama z ugledom ($\rho_0 = 0.5$) in brez.



(a)



(b)

Slika 2: (a) Prikaz stopnje sodelovanja v igri piščanca brez ugleda. (b) Prikaz razlike v stopnji sodelovanja med različicama z ugledom ($\rho_0 = 0.5$) in brez.

Za primerjavo osnovne in razširjene različice uporabimo naslednje metrike. Stopnja kooperativnosti CR_k meri delež kooperativnih akcij agenta g_k čez vse kroge celotnega turnirja:

$$CR_k = \frac{\sum_{t=1}^{N\cdot(K-1)} \sigma[a_k^t = C]}{N \cdot (K-1)}, \tag{6}$$

pri čemer je K število vseh agentov.

5 ANALIZA REZULTATOV

Rezultati eksperimenta za konfiguracijo $m = 5$ so prikazani na slikah 1 in 2, ki prikazujeta razliko v stopnji kooperativnosti med razširjeno različico z uglednim mehanizmom ($\rho_0 = 0.5$) in osnovno različico brez njega. Negativna vrednost pomeni, da osnovna različica dosega višjo stopnjo kooperativnosti.

V iterativni zapornikovi dilemi ugledni mehanizem v večini primerov negativno vpliva na stopnjo kooperativnosti. Kontekstualni banditi (mCB) so najbolj prizadeti (razlika v povprečju = −0.080) pri konfiguraciji $m = 1$ in (razlika v povprečju = −0.075) pri $m = 5$. Večrokni banditi (mMAB) so, kot je pričakovano, minimalno prizadeti (razlika v povprečju med −0.0032 in 0.0024), algoritmi okrepitvenega učenja so nespremenjeni (mRL $m = 5$, razlika v povprečju = 0.0008), saj pri odločanju ne upoštevajo konteksta in uglednega mehanizma ne morejo neposredno izkoriščati. Izpostavimo lahko algoritem LinUCB, ki v bitki s samim seboj kaže pozitivno reakcijo

na ugledni mehanizem (+22 %). Povprečen upad za vse algoritme je −0.023, najslabše pa ugled vpliva na CTS (povprečno −0.094) in SCTS (povprečno −0.086). Najboljša nastavitev začetne vrednosti je bila ($\rho_0 = 0.5$).

V igri piščanca je negativni učinek uglednega mehanizma izrazitejši kot v iterativni zapornikovi dilemi. Rezultati so pokazali, da je tudi tu začetna vrednost $\rho_0 = 0.5$ najuspešnejša nastavitev. Kontekstualni banditi pri tej nastavitvi beležijo največje upade: mCB $m = 1$ dosega (razlika v povprečju = −0.149), mCB $m = 5$ pa (razlika v povprečju = −0.105), kar je opazna izboljšava iz −0.122 pri $\rho_0 = 0$. Slika 2 (toplotni zemljevid) izpostavi, da algoritem CTS dosega največje negativne razlike v turnirju s povprečnim upadom −0.21 (vrednosti med −11 % in −33 %), sledi algoritem SCTS (povprečni upad −0.163, z razlikami med −2 % in −31 %, medtem ko je pri pesimistični inicializaciji $\rho_0 = 0.0$ beležil upad kar −0.209). Večrokni banditi ostajajo skoraj nespremenjeni (razlika v povprečju med −0.0089 in 0.0003), algoritmi okrepitvenega učenja (mRL $m = 5$) pa kažejo le minimalen upad (razlika v povprečju = −0.0036).

Ugledni mehanizem v obeh igrah ni potrdil izhodiščne hipoteze, da bi dostop do informacije o kooperativni zgodovini nasprotnika izboljšal stopnjo kooperativnosti. Delni razlog je verjetno v tem, da dodatna dimenzija konteksta poveča prostor stanj, kar zahteva daljšo fazo učenja in hkrati povečuje variabilnost odločitev, kar je razvidno iz razlik v povprečju, ki so pri kontekstualnih banditih višje kot pri ostalih razredih. Ta nezmožnost prilagajanja na nedeterministične pogoje se ujema tudi z ugotovitvami iz literature. V članku [11] izvedejo eksperiment, ki dokaže, da veliko strategij potrebuje deterministično okolje, kar pa v realnih problemih ni vedno mogoče zagotoviti. V eksperiment uvedejo negotovost, ki je v Axelrod turnir uvedena preko spremenljive dolžine iger. Podobno kot pri njih, kjer negotovost dolžine iger zmanjša uspešnost strategij, se v našem primeru algoritmi ne znajo spopasti z negotovostjo začetnega stanja. Pomembno vlogo ima tudi začetna vrednost. Rezultati so pokazali, da pesimistična inicializacija ($\rho_0 = 0.0$) kontekstualne agente potisne v zanko nezaupanja, agenti vrednost interpretirajo kot popolno odsotnost kooperativnosti, kar sproži preventivno defekcijo še pred vzpostavitvijo zaupanja. Nevtralna ($\rho_0 = 0.5$) in optimistična ($\rho_0 = 1.0$) inicializacija to zanko uspešno prekineta. V primerjavi s pesimistično inicializacijo se je pri zapornikovi dilemi kooperativnost kontekstualnih banditov pri $\rho_0 = 0.5$ povečala za 21.6 % (z 0.246 na 0.299), LinUCB pa se je podvojil (z 0.059 na 0.118). V igri piščanca se je SCTS pri $\rho_0 = 1.0$ izboljšal z 0.396 na 0.454 (+14.4 %).

6 ZAKLJUČEK

V tem članku smo razširili eksperiment iterativne zapornikove dileme z uvedbo uglednega mehanizma, ki agentom pred vsakim dvobojem posreduje agregirano informacijo o kooperativni zgodovini nasprotnika. Eksperiment smo izvedli v obliki krožnega turnirja za dve igri: iterativno zapornikovo dilemo in igro piščanca, pri čemer smo primerjali tri razrede algoritmov sprotnega učenja: večrokne bandite, kontekstualne bandite in algoritme okrepitvenega učenja.

Rezultati niso potrdili izhodiščne hipoteze. Ugledni mehanizem pri nobeni konfiguraciji ρ_0 ni izboljšal povprečne stopnje kooperativnosti, temveč jo je v večini zmanjšal. Najbolj izrazit negativni učinek smo opazili pri kontekstualnih banditih v obeh igrah, kjer informacija o ugledu poveča kompleksnost prostora stanj. Ključen vpliv ima začetna vrednost ugleda, vrednost $\rho_0 = 0.0$ je pri agentih sprožila preventivno defekcijo in zanko nezaupanja. Kot najbolj uravnotežena in stabilna se je izkazala nevtralna inicializacija $\rho_0 = 0.5$.

Primerjava med igrama je pokazala, da je negativni učinek uglednega mehanizma izrazitejši v igri piščanca kot v zapornikovi dilemi, kar je skladno z ugotovitvami [5], ki opozarjajo na specifične vzorce kooperativnosti v igri piščanca.

Za prihodnje delo predlagamo dve smeri. Prvič, postopno uvajanje informacije o ugledu med turnirjem namesto takojšnje uvedbe bi agentom omogočilo prilagoditev brez nenadne spremembe prostora stanj. Drugič, utežena funkcija ugleda, ki novejšim interakcijam pripisuje večjo težo, bi lahko zmanjšala šum zgodnje faze učenja.

LITERATURA

[1] Nicolas Anastassacos, Julian García, Stephen Hailes, and Mirco Musolesi. 2021. Cooperation and reputation dynamics with reinforcement learning. *arXiv preprint arXiv:2102.07523* (2021).

[2] Lee-Ann Barlow and Jeffrey Tsang. 2015. Play profiles: The effect of infinite-length games on evolution in the iterated Prisoner's Dilemma. In *2015 IEEE Conference on Computational Intelligence and Games (CIG)*. IEEE, 368–375.

[3] Bram Cohen et al. 2003. Incentives build robustness in BitTorrent. In *Workshop on Economics of Peer-to-Peer systems*, Vol. 6. 68–72.

[4] Peter de Heus, Niek Hoogervorst, and Eric van Dijk. 2010. Framing prisoners and chickens: Valence effects in the prisoner's dilemma and the chicken game. *Journal of Experimental Social Psychology* 46, 5 (2010), 736–742. https://doi.org/10.1016/j.jesp.2010.04.013

[5] Zhen-Wei Ding, Ji-Qiang Zhang, Guo-Zhong Zheng, Wei-Ran Cai, Chao-Ran Cai, Li Chen, and Xu-Ming Wang. 2024. Emergence of anti-coordinated patterns in snowdrift game by reinforcement learning. *Chaos, Solitons & Fractals* 184 (2024), 114971.

[6] Zhenghui Fang, Haojie Xu, Chen Xie, Xiuyu Yue, Tina P. Benko, and Changwei Huang. 2025. Evolution of cooperation in multi-agent systems driven by reputation-based migration. *Chaos, Solitons Fractals* 200 (2025), 117115. https://doi.org/10.1016/j.chaos.2025.117115

[7] Cameron L Hall, Mason A Porter, and Marian S Dawkins. 2020. Dominance, sharing, and assessment in an iterated Hawk–Dove game. *Journal of theoretical biology* 493 (2020), 110101.

[8] Marc Harper, Vincent Knight, Martin Jones, Georgios Koutsovoulos, Nikoleta E. Glynatsi, and Owen Campbell. 2017. Reinforcement learning produces dominant strategies for the Iterated Prisoner's Dilemma. *PLOS ONE* 12, 12 (12 2017), 1–33. https://doi.org/10.1371/journal.pone.0188046

[9] Marko Jurišić, Dragutin Kermek, and Mladen Konecki. 2012. A review of iterated prisoner's dilemma strategies. In *2012 Proceedings of the 35th International Convention MIPRO*. 1093–1097.

[10] Baihan Lin, Djallel Bouneffouf, and Guillermo Cecchi. 2022. Online Learning in Iterated Prisoner's Dilemma to Mimic Human Behavior. In *PRICAI 2022: Trends in Artificial Intelligence*, Sankalp Khanna, Jian Cao, Quan Bai, and Guandong Xu (Eds.). Springer Nature Switzerland, Cham, 134–147.

[11] Gabriel POP, Mircea MILENCIANU, and Alexandra POP. 2025. THE SECOND AXELROD TOURNAMENT: A MONTE CARLO EXPLORATION OF UNCERTAINTY ABOUT THE NUMBER OF ROUNDS IN ITERATED PRISONER'S DILEMMA. *Studia Universitatis Babeș-Bolyai Oeconomica* 70, 1 (Mar. 2025), 67–82. https://doi.org/10.2478/subboec-2025-0004

[12] Qi Su, Hongyu Wang, Yu Xia, and Long Wang. 2025. A multi-agent reinforcement learning framework for exploring dominant strategies in iterated and evolutionary games. *Nature Communications* (2025). https://doi.org/10.1038/s41467-025-67178-6

 DOI: https://doi.org/10.26493/scores26.05

Večagentno prilagodljivo upravljanje semaforjev z globokim okrepitvenim učenjem in medsebojno komunikacijo

Miha Pliberšek
miha.plibersek@student.um.si
Fakulteta za elektrotehniko, računalništvo in informatiko
Univerza v Mariboru
Koroška cesta 46
SI-2000 Maribor, Slovenija

Enej Krabonja
enej.krabonja@student.um.si
Fakulteta za elektrotehniko, računalništvo in informatiko
Univerza v Mariboru
Koroška cesta 46
SI-2000 Maribor, Slovenija

Oskar Forštnarič
oskar.forstnaric@student.um.si
Fakulteta za elektrotehniko, računalništvo in informatiko
Univerza v Mariboru
Koroška cesta 46
SI-2000 Maribor, Slovenija

Luka Antolinc
luka.antolinc@student.um.si
Fakulteta za elektrotehniko, računalništvo in informatiko
Univerza v Mariboru
Koroška cesta 46
SI-2000 Maribor, Slovenija

Anže Hameršak
anze.hamersak@student.um.si
Fakulteta za elektrotehniko, računalništvo in informatiko
Univerza v Mariboru
Koroška cesta 46
SI-2000 Maribor, Slovenija

Borko Boškovič
borko.boskovic@um.si
Fakulteta za elektrotehniko, računalništvo in informatiko
Univerza v Mariboru
Koroška cesta 46
SI-2000 Maribor, Slovenija

POVZETEK

V prispevku obravnavamo problem prilagodljivega upravljanja semaforjev z uporabo globokega okrepitvenega učenja v simulacijskem okolju SUMO. Izhajali smo iz obstoječega enoagentnega modela DQN, ki upravlja posamezno semaforizirano križišče, in ga nadgradili z dodatnimi prometnimi in okoljskimi metrikami, večkriterijsko funkcijo nagrade ter komunikacijskim modulom za večagentno koordinacijo med sosednjimi križišči.

Eksperimente smo izvedli na simuliranem petkrižiščnem sistemu z uporabo različnih prometnih scenarijev. Rezultati kažejo, da razširjeni večagentni model dosega stabilnejše in hitrejše učenje, pri vrednotenju na novih scenarijih pa bistveno preseže tako enoagentni model kot klasični nespremenljivi časovnik: povprečna velikost čakalne vrste se je v primerjavi s klasičnim časovnikom zmanjšala za 65 %, emisije CO_2 pa za 28 %.

KLJUČNE BESEDE

večagentni sistemi, globoko okrepitveno učenje, upravljanje semaforjev, komunikacija med agenti, koordinacija križišč

1 UVOD

Prometni zastoji v urbanih cestnih omrežjih povzročajo daljše potovalne čase, večjo porabo goriva in povečane emisije CO_2. Klasični semaforski sistemi z določenimi časovnimi intervali se slabo prilagajajo spreminjajočim se prometnim razmeram, kar je še posebej izrazito v omrežjih z več križišči, kjer odločitev enega semaforja vpliva na pretočnost sosednjih križišč.

Te omejitve naslavljajo pristopi prilagodljivega upravljanja semaforjev, ki temeljijo na globokem okrepitvenem učenju (angl. Deep Reinforcement Learning; DRL). Agent na podlagi opazovanega prometnega stanja (dolžine čakalnih vrst, števila vozil, faze semaforja, povprečne hitrosti) izbira akcije, ki dolgoročno izboljšajo prometni tok.

Naše delo se najbolj naslanja na štiri sorodne raziskave. Izhodiščni članek [4] je obravnaval prilagodljivo upravljanje enega semaforja z modelom DQN in večciljno nagrado, ki je upoštevala tudi okoljski vpliv in varnost, eksperimente pa je izvedel v okolju SUMO. Ta pristop smo privzeli kot izhodišče za naš osnovni model in eksperimentalno okolje. Podoben enoagentni DQN pristop za upravljanje posameznega križišča so predstavili v [5]. V tem delu so nevronsko mrežo uporabili kot aproksimator Q-funkcije in model primerjali s klasičnimi metodami okrepitvenega učenja. Ta pristop smo uporabili kot izhodišče za razvoj našega osnovnega modela in eksperimentalnega okolja. Pri zasnovi večagentne arhitekture smo izhajali iz dela Chu et al. [3], ki so za usklajevanje semaforjev v velikih omrežjih predstavili skalabilen decentraliziran pristop, ki temelji na algoritmu A2C, ter pristop CVDMARL [2], ki so neposredno izmenjavo informacij med agenti združili z dekompozicijo vrednostne funkcije prek omrežja QMIX. Podobno zasnovo komunikacijskega modula, kjer agent zraven lastnega stanja upošteva tudi informacije sosednjih križišč, smo prevzeli tudi v našem pristopu.

Tudi druga sorodna dela so prispevala pomembne vpoglede. V [6] so z Multi-Agent DDQN zmanjšali čakalni čas za do 60 % in povečali povprečno hitrost vozil za do 18 %. V delu [7] so pokazali, da se modeli, naučeni na statičnih scenarijih, slabše prilagodijo spremenljivemu prometu čez dan. Agafonov in Yumaganov [1] pa so z algoritmom Soft Actor-Critic neprekinjeno prilagajali trajanje faz in s tem dosegli manj nenadnih ustavljanj kot DQN.

Kljub uspešnosti omenjenih pristopov ostaja pomembna omejitev pri sistemih, kjer agenti delujejo preveč neodvisno, saj zastoj v enem križišču hitro vpliva na sosednje dele omrežja. Zato smo v tem prispevku obstoječi enoagentni DQN pristop iz dela [4] nadgradili v večagentni model s komunikacijo med sosednjimi križišči, z razširjeno predstavitvijo stanja, večkriterijsko nagrado in algoritmom Double DQN z mehko posodobitvijo ciljne mreže.

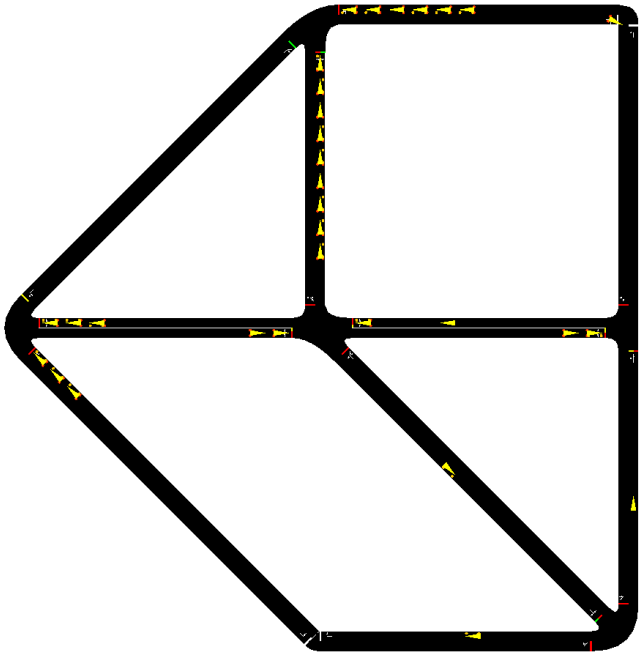
Struktura prispevka je naslednja: poglavje 2 opisuje metodologijo, vključno s formulacijo problema, osnovnim in predlaganim modelom ter eksperimentalno postavitvijo, poglavje 3 predstavi rezultate, poglavje 4 pa povzame ugotovitve in nakaže možnosti nadaljnjega dela.

2 METODOLOGIJA

Sistem deluje v zaprti zanki: agent prek vmesnika TraCI iz simulacijskega okolja SUMO prejme stanje in izračuna nagrado. Pri večagentni različici prejme in pošlje tudi sporočila sosednjih agentov, nato pa v okolju izvede izbrano akcijo nad semaforjem, ki znova vpliva na naslednje stanje. Pri vseh eksperimentih smo uporabili enake prometne scenarije in začetna semena naključnega generatorja, kar omogoča neposredno primerjavo med pristopi. Na ta način razlike v rezultatih izhajajo izključno iz uporabljenega učnega algoritma in komunikacijskega mehanizma.

2.1 Eksperimentalno okolje

Eksperimente smo izvedli v simulacijskem okolju SUMO, povezanim prek vmesnika TraCI, na cestnem omrežju s petimi semaforiziranimi križišči (eno štirikrako, štiri trikraka; dovozne ceste dolžine 500 m, kot prikazuje slika 1). En simulacijski korak ustreza eni sekundi, epizoda obsega $T_{\max} = 6000$ korakov. V vsaki epizodi se v omrežje postopno doda $N_{\text{voz}} = 200$ vozil, s časi prihoda po Weibullovi porazdelitvi ($k = 2$); 75 % vozil vozi naravnost, 25 % zavija, seme generiranja prometa pa je enako zaporedni številki epizode.



Slika 1: Cestno omrežje s petimi križišči.

2.2 Formulacija problema

Problem je formuliran kot Markovski proces odločanja: agent v vsakem koraku opazuje stanje, izbere fazo (ob upoštevanju minimalnega trajanja zelene in rumene semaforske faze) in prejme nagrado. Medtem ko je nagrada v delu [4] temeljila izključno na spremembi skupnega čakalnega časa ($R_t = W_{t-1} - W_t$), smo uvedli večkriterijsko nagrado

$$R_t = -(w_1 D_t + w_2 S_t + w_3 Q_t + w_4 CO_{2t}), \tag{1}$$

kjer D_t, S_t, Q_t in CO_{2t} označujejo skupni prometni zaostanek, število zaustavitev, dolžino vrste in emisije CO_2 (w_1=0,4, w_2=w_3=0,2, w_4=0,0001). Nagrado dodatno normaliziramo glede na trajanje trenutne faze za stabilnejše učenje.

Stanje agenta temelji na izvirnem 80-dimenzionalnem binarnem vektorju zasedenosti cestnih segmentov (8 pasovnih skupin po 10 nelinearno razporejenih odsekov do 500 m od semaforja), ki smo ga razširili z osmimi dodatnimi skalarji (dolžina vrste, število vozil, povprečna hitrost, faza semaforja, skupni čakalni čas, emisije CO_2, relativni čas simulacije, preostala vozila v simulaciji), kar dimenzijo poveča na 88. Vse vhodne vrednosti so normalizirane. Pri večagentni različici je lokalno stanje dodatno razširjeno z vektorjem sporočil sosednjih agentov, ki ga podrobneje opišemo v naslednjem podpoglavju.

2.3 Osnovni in predlagani model

Kot osnovni enoagentni pristop (EA) smo uporabili izhodiščni enoagentni DQN [4], kjer nevronska mreža aproksimira Q-funkcijo, izkušnje pa se vzorčijo iz pomnilnika izkušenj; ciljna mreža se sinhronizira vsakih $\tau = 20$ epizod.

Predlagani večagentni model (VA) razširi osnovni pristop v treh smereh. Vsakemu križišču dodelimo svojega agenta, ki poleg lokalnega stanja pri odločanju upošteva tudi sporočila sosednjih agentov. V eksperimentalnem omrežju je bilo uporabljenih pet agentov, pri čemer je vsak agent upravljal eno semaforizirano križišče. Komunikacijski modul, zasnovan kot vmesni sloj med agenti, vzdržuje slovar sporočil za vsako križišče in pred vsakim izborom akcije sosednjemu agentu posreduje štiri normalizirane vrednosti – dolžino čakalne vrste, število vozil, povprečno hitrost in trenutno fazo semaforja v sosednjem križišču – ki se združijo z lokalnim stanjem agenta v enoten vhodni vektor. Ta vektor ima dimenzijo $88 + 4k$, kjer k označuje število sosednjih križišč posameznega agenta. Ker topologija omrežja ni enaka za vsa križišča, se vhodna dimenzija med agenti lahko razlikuje, zato vsak agent uporablja lastno nevronsko mrežo. S tem agent dobi širši vpogled v prometno stanje sosednjih križišč, ne da bi bilo potrebno popolnoma centralizirano upravljanje celotnega omrežja. Komunikacija med agenti poteka v vsakem odločitvenem koraku simulacije. Ker se izmenjujejo zgolj štiri agregirane prometne značilke, ostaja komunikacijska obremenitev nizka tudi pri večjem številu križišč. Takšna zasnova omogoča razširljivost sistema, saj agentom ni potrebno poznati celotnega globalnega stanja prometnega omrežja.

Namesto standardnega DQN uporabimo Double DQN, kjer glavna mreža izbere najboljšo akcijo, ciljna mreža pa oceni njeno vrednost, kar zmanjša sistematično precenjevanje Q-vrednosti. Izbiro vrednostno usmerjenega algoritma (Double DQN) namesto pristopov, ki temeljijo na politiki, podpira tudi primerjalna študija [8]. Ta je pokazala, da vrednostne DRL metode pri upravljanju semaforjev dosegajo boljše rezultate, pri čemer je za uspešnost posebej pomembna ustrezna zasnova funkcije nagrade, predvsem pri visokih prometnih obremenitvah. Ciljno mrežo posodabljamo z mehko (Polyakovo) posodobitvijo po vsaki epizodi:

$$\theta_{\text{cilj}} \leftarrow \alpha\, \theta_{\text{online}} + (1 - \alpha)\, \theta_{\text{cilj}}, \quad \alpha = 0{,}005, \tag{2}$$

kjer θ_{online} predstavlja parametre glavne nevronske mreže, θ_{cilj} parametre ciljne mreže, α pa koeficient mehke (Polyakove) posodobitve.

Arhitektura nevronske mreže in preostali hiperparametri obeh pristopov so povzeti v tabeli 1.

Tabela 1: Hiperparametri eksperimenta.

Parameter	Vrednost	Opis
$T_{\max}$	6000 s	Dolžina simulacije
Epizode	400	Skupno število epizod
N_{voz}	200	Vozila na epizodo
γ	0,95	Diskontni faktor
ε	EA: $1-e/E$; VA: $0{,}995^e$ (min. 0,05)	ε-pohlepna strategija
Velikost serije	100	Vzorci iz pomnilnika
Pomnilnik izkušenj	50.000	Maks. kapaciteta
τ (EA)/α (VA)	20 epizod/0,005	Sinhr. ciljne mreže (trda/Polyak)
Zelena/rumena faza	10/4 koraki	Trajanje faz
Arhitektura nevronske mreže (D)DQN		
Vhodna plast (EA)	88	Stanje okolja
Vhodna plast (VA)	$88 + 4k$ (k = št. sosedov)	Stanje okolja
Skriti plasti	2×400 nevronov	Linearna / ReLU
Izhodna plast	4	Linearna aktivacija
Optimizator	Adam	Privzeta stopnja učenja
Izguba	Huber	/

2.4 Učenje agentov

Učenje je potekalo skozi 400 epizod. V vsaki epizodi so agenti postopoma gradili pomnilnik izkušenj, iz katerega so se naključno vzorčili mini paketi za posodobitev nevronske mreže. Akcije so bile med učenjem izbrane po ε-pohlepni strategiji, pri kateri agent večino časa izbere trenutno najboljšo akcijo, z določeno verjetnostjo pa izvede tudi naključno raziskovanje.

Pri večagentnem pristopu je vsak agent sprejemal odločitve neodvisno, vendar na podlagi lokalnega stanja in informacij, prejetih od sosednjih križišč. S tem je bilo omogočeno implicitno usklajevanje semaforjev brez potrebe po centralnem nadzornem agentu. Po vsaki epizodi so bile posodobljene ciljne mreže, pri čemer je VA model uporabljal mehko Polyakovo posodobitev, ki zagotavlja stabilnejše učenje in manjša nihanja ocenjenih Q-vrednosti.

2.5 Primerjalne metode in vrednotenje

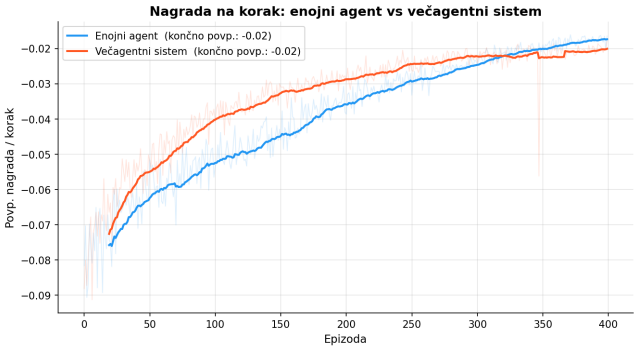
Primerjali smo tri pristope, testirane na identičnih prometnih scenarijih z enakimi semeni: klasični fiksni časovnik, ki zaporedno in brez prilagajanja kroži skozi štiri zelene faze z desetimi koraki, DQN brez komunikacije (EA), kjer agent opazuje izključno lokalno stanje svojega križišča, in predlagani VA model s komunikacijo, kjer agent pri odločanju upošteva tudi stanja sosednjih agentov.

Uspešnost smo merili s povprečno nagrado na korak, dolžino čakalne vrste, zamudo vozil, številom zaustavitev, emisijami CO_2 na korak in pretočnostjo (deležem vozil, ki uspešno prečka omrežje), prek vmesnika TraCI pa smo spremljali tudi varnostna kazalnika nenadnih zaviranj in trkov.

3 REZULTATI

3.1 Rezultati učenja

Slika 2 prikazuje povprečno nagrado na korak skozi 400 epizod učenja za oba pristopa. Obe krivulji kažeta jasno konvergenco: nagrada narašča od začetnih vrednosti okoli −0,075 proti vrednostim blizu −0,02 po 400 epizodah. Večagentni sistem (VA) izkazuje hitrejšo konvergenco skozi celotno fazo učenja – pri epizodi 100 doseže nagrado ≈ −0,034, medtem ko enoagentni pristop (EA) v isti točki doseže le ≈ −0,050. Podoben vzorec hitrejše zgodnje konvergence se kaže tudi pri čakalni vrsti in emisijah CO_2 (tabela 2).



Slika 2: Povprečna nagrada na korak skozi 400 epizod učenja.

Pri zamudi vozil je razmerje med pristopoma manj enoznačno. VA na začetku učenja beleži nekoliko večjo zamudo (≈ −0,120 s) kot EA (≈ −0,125 s), kar odraža večji vhodni prostor stanja in dodatne komunikacijske stroške v zgodnjih fazah učenja VA agenta. V srednji fazi učenja (epizode 100–250) ima EA nižjo zamudo, VA pa ga dohiti šele v poznejši fazi, ko oba pristopa konvergirata na enako končno vrednost −0,02 s. Pretočnost (delež od $N = 200$ generiranih vozil, ki uspešno prečka omrežje) se v zgodnjih epizodah giblje med 192 in 194 vozili, proti koncu učenja pa se pri obeh pristopih ustali pri ≈ 198–199 vozil, pri VA pristopu pa v posameznih epizodah doseže tudi 100 %.

Tabela 2: Povzetek ključnih metrik: učenje (povprečje zadnjih 50 epizod) in vrednotenje na novih scenarijih (ε=0), normalizirano na korak.

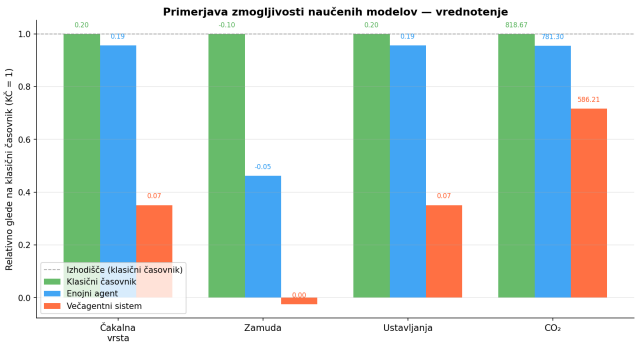
	Učenje		Vrednotenje		
Metrika	EA	VA	Klas.	EA	VA
Nagrada	−0,02	−0,02	/	/	/
Zamuda [s]	−0,02	−0,02	−0,10	−0,05	≈ 0,00
Čak. vrsta	0,06	0,08	0,20	0,19	0,07
Zaustavitve	0,06	0,08	0,20	0,19	0,07
CO_2 [mg]	561,18	589,56	818,67	781,30	586,21

Rezultati kažejo, da komunikacija med agenti ne vpliva zgolj na končno zmogljivost sistema, temveč tudi na stabilnost učenja.

Variabilnost metrik med posameznimi epizodami je bila pri VA pristopu manjša kot pri EA pristopu, kar nakazuje, da imajo agenti zaradi dodatnih informacij boljši vpogled v globalno prometno stanje. To je posebej pomembno v primerih, ko zastoji nastajajo zaradi vpliva sosednjih križišč in jih ni mogoče zaznati zgolj z lokalnim opazovanjem. Posebej izrazita je razlika med učenjem in vrednotenjem. Medtem ko so metrike pri učenju pri EA in VA razmeroma podobne, se prednost komunikacije pokaže predvsem pri posploševanju na nove prometne scenarije. To nakazuje, da dodatne informacije sosednjih križišč izboljšujejo robustnost naučene politike in zmanjšujejo odvisnost od specifičnih prometnih vzorcev, prisotnih med učenjem.

3.2 Vrednotenje na novih scenarijih

Pri vrednotenju na novih, med učenjem nevidenih prometnih scenarijih, z izklopljeno eksploracijo (ε=0), oba pristopa bistveno presežeta klasični časovnik (slika 3). Večagentni sistem zmanjša čakalno vrsto in število zaustavitev za 65 %, emisije CO_2 za 28 %, zamuda vozil pa pade na zanemarljivo vrednost. Enoagentni pristop brez komunikacije v enakih pogojih doseže le skromno, ≈ 5-odstotno izboljšanje, kar potrjuje, da je medagentna komunikacija ključni dejavnik izboljšave.



Slika 3: Primerjava zmogljivosti naučenih modelov pri vrednotenju, normalizirano glede na klasični časovnik (KČ = 1).

Razkorak med na videz primerljivimi vrednostmi učenja in veliko razliko pri vrednotenju pojasnjuje različna strategija razpadanja ε: EA uporablja linearni razpad, ki minimum (0,05) doseže že pri epizodi ≈ 380, medtem ko VA uporablja eksponentni razpad ($0{,}995^e$), ki pri epizodi 400 še vedno znaša ≈ 13,5 %. Učne krivulje VA pristopa zato delno odražajo naključno raziskovanje in ne le naučene politike, ki šele pri vrednotenju brez eksploracijskega šuma razkrije svojo dejansko, koordinirano kakovost. Varnostna kazalnika (nenadna zaviranja, trki) sta pri obeh pristopih skozi celotno učenje ostala zanemarljiva oziroma ničelna. Ugotovitev kaže, da numerične vrednosti same po sebi med učenjem ne odražajo nujno dejanske kakovosti naučene politike, kar velja smiselno upoštevati pri vrednotenju podobnih večagentnih sistemov.

Dobljeni rezultati potrjujejo ugotovitve sorodnih raziskav, da koordinacija med križišči postaja pomembnejša z večanjem kompleksnosti prometnega omrežja. Medtem ko lahko enoagentni pristopi uspešno optimizirajo posamezno križišče, pogosto ne upoštevajo posledic svojih odločitev na sosednje dele omrežja. Predlagani pristop ta problem delno odpravi z neposredno izmenjavo ključnih prometnih informacij med agenti, kar omogoča bolj usklajeno upravljanje prometnih tokov.

4 ZAKLJUČEK

V tem delu smo predstavili večagentni sistem za prilagodljivo upravljanje semaforjev, ki temelji na algoritmu Double DQN in komunikaciji med sosednjimi križišči ter pri tem razširili predstavitev stanja in funkcijo nagrade z dodatnimi prometnimi in okoljskimi metrikami.

Predlagani pristop smo ovrednotili v simulacijskem okolju SUMO na omrežju s petimi semaforiziranimi križišči, kjer smo primerjali klasični fiksni časovnik, enoagentni DQN (EA) brez komunikacije in predlagani večagentni VA model s komunikacijo. Rezultati so pokazali, da večagentni sistem med učenjem konvergira hitreje kot enoagentni, na novih prometnih scenarijih pri vrednotenju pa bistveno preseže klasični časovnik: čakalna vrsta in število zaustavitev sta se zmanjšala za 65 %, emisije CO_2 za 28 %, zamuda vozil pa je padla na zanemarljivo vrednost. Enoagentni pristop brez komunikacije je v enakih pogojih dosegel le ≈ 5-odstotno izboljšanje, kar potrjuje, da je medagentna komunikacija ključni dejavnik izboljšave. Varnostna kazalnika nenadnih zaviranj in trkov sta pri obeh pristopih skozi celotno učenje ostala zanemarljiva.

Za nadaljnje delo predlagamo izenačitev strategije razpadanja ε med EA in VA pristopoma. Smiselna razširitev bi bila tudi vpeljava nagrade, ki bi poleg lokalne komponente upoštevala tudi uteženo vsoto nagrad sosednjih agentov, s čimer bi bil vsak agent posredno spodbujen k odločitvam, ki koristijo celotnemu omrežju in ne le lastnemu križišču. Zanimiva bi bila tudi razširitev na gostejša mestna omrežja z večjim številom križišč, kjer bi prišla komunikacija med agenti še bolj do izraza, kar bi predlagani pristop približalo realni uporabi pri zmanjševanju prometnih zastojev in emisij v mestih.

Literatura

[1] Alexander Agafonov, Anton in Yumaganov. 2025. Adaptive traffic signal control with Soft Actor-Critic: A phase duration optimization approach. *Optical Memory and Neural Networks* 34, 4 (2025), 553–565. doi:10.3103/S1060992X25600648

[2] Yuting in Wang Chunguang in Bie Yiming Chang, Ande in Ji. 2024. CVD-MARL: A communication-enhanced value decomposition multi-agent reinforcement learning traffic signal control method. *Sustainability* 16, 5 (2024), 2160. doi:10.3390/su16052160

[3] Jie in Codecà Lara in Li Zhaojian Chu, Tianshu in Wang. 2019. Multi-agent deep reinforcement learning for large-scale traffic signal control. *IEEE Transactions on Intelligent Transportation Systems* 21, 3 (2019), 1086–1095. doi:10.1109/TITS.2019.2901791

[4] Sherif M. in Shwaly Elsayed Abd-Elazem in Shahdah Usama Elrawy Elharoun, Marwa in El-Badawy. 2025. Adaptive traffic signal control using deep reinforcement learning: A multi-objective approach for single and multi-intersection scenarios. *IATSS Research* 49, 4 (2025), 481–492. doi:10.1016/j.iatssr.2025.10.004

[5] Pin in Chan Ching-Yao in Askary-Sid Guo, Mengyu in Wang. 2019. A reinforcement learning approach for intelligent traffic signal control at urban intersections. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, 4242–4247. doi:10.1109/ITSC.2019.8917268

[6] Zhuo-Yu Hu, Ta-Yin in Li. 2024. A multi-agent deep reinforcement learning approach for traffic signal coordination. *IET Intelligent Transport Systems* 18, 8 (2024), 1428–1444. doi:10.1049/itr2.12521

[7] Qian in Guo Weiwei in Xie Na in Liu Fuyu in Wei Jing in Zhang Xinwei Tan, Jiyuan in Yuan. 2022. Deep reinforcement learning for traffic signal control model and adaptation study. *Sensors* 22, 22 (2022), 8732. doi:10.3390/s22228732

[8] Inhi in Ma Zhenliang Wu, Chunliang in Kim. 2023. Deep reinforcement learning based traffic signal control: A comparative analysis. *Procedia Computer Science* 220 (2023), 275–282. doi:10.1016/j.procs.2023.03.036

 DOI: https://doi.org/10.26493/scores26.08

Hibridni pristop večagentnega iskanja poti s toplotno karto

Nik Drozg
nik.drozg@student.um.si
Fakulteta za elektrotehniko, računalništvo in informatiko, Univerza v Mariboru
Koroška cesta 46
SI-2000 Maribor, Slovenia

Pavel Medved
pavel.medved@student.um.si
Fakulteta za elektrotehniko, računalništvo in informatiko, Univerza v Mariboru
Koroška cesta 46
SI-2000 Maribor, Slovenia

Nik Damiš
nik.damis@student.um.si
Fakulteta za elektrotehniko, računalništvo in informatiko, Univerza v Mariboru
Koroška cesta 46
SI-2000 Maribor, Slovenia

Igor Polajžer
igor.polajzer@student.um.si
Fakulteta za elektrotehniko, računalništvo in informatiko, Univerza v Mariboru
Koroška cesta 46
SI-2000 Maribor, Slovenia

Blaž Vidovič
blaz.vidovic6@student.um.si
Fakulteta za elektrotehniko, računalništvo in informatiko, Univerza v Mariboru
Koroška cesta 46
SI-2000 Maribor, Slovenia

Klemen Berkovič
klemen.berkovic1@um.si
Fakulteta za elektrotehniko, računalništvo in informatiko, Univerza v Mariboru
Koroška cesta 46
SI-2000 Maribor, Slovenia

Povzetek

Koordinacija med večjim številom agentov v dinamičnih okoljih predstavlja enega ključnih izzivov v vseživljenjskem večagentnem iskanju poti, kjer agenti neprekinjeno prejemajo nove naloge. Obstoječe decentralizirane metode, ki temeljijo na lokalnih opazovanjih, pri večjem številu agentov pogosto vodijo do slabše koordinacije in povečanega števila konfliktov. V tem delu predlagamo hibridni pristop, ki razširi ogrodje PRIMAL2 z globalnim signalom v obliki skupne toplotne karte zasedenosti. Ta agentom ob ohranjanju decentraliziranega odločanja posreduje informacije o gostoti prometa na mapi. Pristop primerjamo z izbranimi metodami iz literature: Conformant-CBS, Guided-PIBT, Learn to Follow, MAPF-GPT in izvirnim PRIMAL2. Vse primerjave so izvedene v okolju, ki meri 160×160 celic z gostoto ovir 33 %. V vsakem scenariju nastopa do 128 agentov, pri čemer merimo prepustnost, število trkov in čas izvajanja. Rezultati kažejo, da dodatne globalne informacije v večjih okoljih zmanjšajo število trkov in izboljšajo koordinacijo, medtem ko prepustnost ostaja primerljiva z izvirnim PRIMAL2. Ugotovitve potrjujejo, da je predlagani pristop, ki združuje lokalno odločanje z omejenimi globalnimi informacijami, smiseln za skalabilno reševanje problemov vseživljenjskega večagentnega iskanja poti.

Ključne besede

vseživljenjsko večagentno iskanje poti, večagentni sistemi, načrtovanje poti, vzpodbujevalno učenje, decentralizirano odločanje

1 Uvod

Večagentno iskanje poti (angl. *Multi-Agent Path Finding* – MAPF) obravnava problem načrtovanja poti za več agentov, ki morajo doseči svoje ciljne lokacije brez trkov z drugimi agenti ali ovirami v okolju [9]. Problem se pojavlja v okoljih, kjer več agentov hkrati deluje na omejenem območju, na primer v skladiščnih sistemih, kjer neusklajeno gibanje povzroča zastoje in zmanjšuje učinkovitost sistema [5].

V praksi se MAPF pogosto razširi v vseživljenjsko večagentno iskanje poti (angl. *Lifelong MAPF – LMAPF*), kjer agenti po dosegu cilja prejmejo nove naloge in nadaljujejo z delovanjem [2, 5]. V takšnem okolju optimizacijski cilj ni več zgolj iskanje posamezne optimalne poti, temveč maksimizacija prepustnosti (angl. *throughput*), ki predstavlja povprečno število opravljenih nalog v časovni enoti [2].

Sodobni pristopi za reševanje LMAPF pogosto temeljijo na metodah globokega vzpodbujevalnega učenja ali imitacijskega učenja, ki omogočajo decentralizirano odločanje agentov na podlagi lokalnih opazovanj [1, 8]. Kljub temu se z naraščanjem števila agentov njihova zmogljivost pogosto poslabša, kar se kaže v povečanju števila konfliktov in slabši koordinaciji v gostih okoljih [2, 4].

To omejitev ponazarja ogrodje PRIMAL2 [3], kjer agenti delujejo v delno opazljivem okolju in imajo dostop le do omejenega lokalnega vidnega polja. Zaradi pomanjkanja informacij o širšem stanju okolja lahko pride do neučinkovite koordinacije, zlasti v ozkih prehodih, kjer agenti nimajo vpogleda v stanje izven svojega zaznavanja [3].

V tem članku predlagamo razširitev pristopa PRIMAL2 z uvedbo dodatnega globalnega signala v obliki skupne toplotne karte (angl. *global heatmap*), ki agentom omogoča boljši vpogled v stanje okolja ob ohranjanju decentraliziranega odločanja. Cilj predlagane metode je izboljšati pretok sistema z zmanjšanjem zastojev in konfliktov med agenti.

V poglavju Sorodna dela predstavimo obstoječe pristope za reševanje problemov MAPF in LMAPF. V poglavju Metodologija opišemo osnovni pristop PRIMAL2 ter predlagano razširitev z uporabo skupne toplotne karte. V poglavju Eksperimenti predstavimo eksperimentalno okolje, uporabljene metrike in rezultate primerjave z obstoječimi pristopi. Na koncu podamo analizo rezultatov ter glavne ugotovitve raziskave.

2 Sorodna dela

Raziskave na področju večagentnega iskanja poti so se skozi leta razvile iz reševanja statičnih v reševanje dinamičnih problemov v sistemih, ki podpirajo neprekinjeno delovanje tisočih agentov.

Obstoječe pristope v literaturi lahko v osnovi razdelimo na algoritmične metode iskanja, ki temeljijo na hevristikah, ter na sodobne metode strojnega učenja, ki poudarjajo decentralizacijo [9].

2.1 Različni pristopi k reševanju MAPF

Raziskovalci za reševanje MAPF in LMAPF izrabljajo različne tehnike, pri čemer vsaka prinaša specifične prednosti in omejitve, kot je povzeto v Tabeli 1.

Tradicionalne metode, kot je iskanje na podlagi sporov (angl. *Conflict-Based Search* – CBS), so se izkazale za izjemno učinkovite pri manjšem številu agentov, saj zagotavljajo optimalne rešitve. Vendar se te metode soočajo s kritičnimi težavami pri skaliranju na večje sisteme [7]. Za naslavljanje potreb po vseživljenjskem načrtovanju v obsežnih okoljih so se uveljavile metode z drsečim horizontom (angl. *Rolling-Horizon Collision Resolution* – RHCR). Problem LMAPF razbijejo na manjše, časovno omejene enote, kar poveča odzivnost in skalabilnost sistema, a prepustnost ni globalno optimizirana, saj kolizije zunaj trenutnega časovnega okna ostanejo nerešene [5].

V zadnjem obdobju narašča zanimanje za pristope, temelječe na globokem učenju, ki poudarjajo decentralizirano odločanje. Ogrodje PRIMAL$_2$ [3] uvaja decentralizirano vzpodbujevalno učenje (angl. *Reinforcement Learning* – RL), ki agentom omogoča sprotno načrtovanje poti v delno opaznih okoljih. Novejši pristopi, kot je MAPF-GPT [1], izkoriščajo arhitekture transformerjev za imitacijsko učenje (angl. *Imitation Learning* – IL). Ti modeli se učijo iz obsežnih baz strokovnih rešitev, kar jim omogoča posplošitev brez predhodnega prilagajanja (angl. *zero-shot generalization*) na nove tipe okolij brez potrebe po dodatni komunikaciji med agenti.

Kljub napredku se metode, ki uporabljajo zgolj globoko učenje, pogosto soočajo s težavami pri dolgoročni koordinaciji v zelo gostih okoljih. Zaradi teh inherentnih omejitev posameznih tehnik so se raziskovalci začeli usmerjati v hibridne pristope, ki poskušajo združiti zanesljivost algoritmičnega načrtovanja s prožnostjo nevronskih mrež.

Tabela 1: Primerjava ključnih pristopov k reševanju MAPF.

Metoda	Prednosti	Slabosti
CBS	Optimalnost, popolnost	Slabo skaliranje
RHCR	Učinkovitost v LMAPF	Lokalna optimalnost
RL	Skalabilnost, odzivnost	Težavna koordinacija
IL	Generalizacija (zero-shot)	Odvisnost od podatkov

2.2 Hibridni pristopi

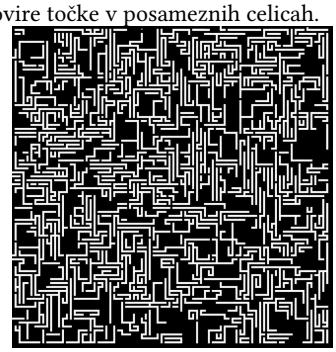
Hibridni pristopi pri problemu večagentnega iskanja poti združujejo klasično planiranje in metode strojnega učenja z namenom izkoriščanja prednosti obeh pristopov. Planiranje omogoča učinkovitejšo obravnavo globalne strukture problema, medtem ko učni pristopi prispevajo večjo prilagodljivost v dinamičnih in delno opazovanih okoljih. Učinkovitost hibridnih pristopov temelji na tesni povezavi algoritmičnega načrtovanja in metod strojnega učenja, saj le njuna medsebojna dopolnitev omogoča izkoriščanje prednosti obeh strategij. To pokaže tudi pristop Learn to Follow [8], kjer avtorji ugotavljajo, da je natančna povezava med komponento planiranja in komponento strojnega učenja ključna, saj algoritem brez katerekoli od njiju deluje bistveno slabše.

Podoben pristop uporablja hibridno ogrodje CHS [6], ki združuje globalni načrtovalnik D* Lite z večagentnim vzpodbujevalnim učenjem (angl. *Multi-Agent Reinforcement Learning* – MARL). Sistem dinamično preklaplja med globalnim in lokalnim planiranjem glede na gostoto agentov v okolici, pri čemer deli le inkrementalne spremembe okolja namesto celotnega globalnega stanja. Na ta način ohranja tako globalno optimalnost poti kot lokalno prilagodljivost, kar je še posebej učinkovito v gostih in dinamičnih okoljih. Navdihnjeni s tem pristopom, v tem delu testiramo, ali lahko podobno globalno usmeritev zagotovimo algoritmu PRIMAL2 [3] z uvedbo skupne toplotne karte, ki agentom posreduje informacije o pogosto zasedenih območjih, ne da bi pri tem posegali v decentralizirano naravo algoritma.

3 Hibridni pristop PRIMAL2 s toplotno karto

V tem delu predstavimo predlagan hibridni pristop za večagentno iskanje poti, ki temelji na nadgradnji ogrodja PRIMAL2 z dodatnim globalnim signalom v obliki toplotne karte zasedenosti. S tem smo poskušali izboljšati odločanje agentov ter povečati njihovo učinkovitost delovanja v okolju.

Okolje algoritma PRIMAL2 ter naše izboljšane verzije je predstavljeno kot diskretna 2D mreža, predstavljena na Sliki 1, kjer so agenti, cilji in ovire točke v posameznih celicah.



Slika 1: Primer testnega okolja velikosti 160 × 160.

V algoritmu PRIMAL2 vsak agent pošilja 10 kanalov, ki predstavljajo okolje v nevronsko mrežo. Kanali so sledeči:

- Položaj agenta,
- Lastni cilj,
- Cilji ostalih agentov,
- Ovire,
- Mapa razdalj do cilja,
- Blokiranje hodnikov,
- Orientacija hodnika X,
- Orientacija hodnika Y,
- Napovedane prihodnje pozicije agentov (trije časovni koraki).

Agenti sprejemajo odločitve vzporedno, pri čemer v vsakem časovnem koraku izberejo eno izmed možnih akcij: premik ali čakanje. Te odločitve generira kopija nevronske mreže, ki deluje kot njihov pravilnik gibanja (π).

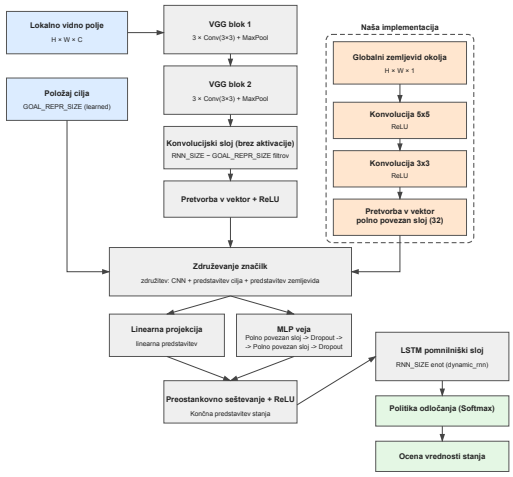
Sama nevronska mreža vsebuje razne konvolucijske nivoje in bolj pomembno, dolgoročni kratkoročni spomin (angl. *Long Short-Term Memory* – LSTM) agentu omogoča pomnjenje preteklih stanj, kar izboljša odločanje v delno opazljivem okolju.

3.1 Toplotna karta

Nadgradnja našega pristopa temelji na uvedbi globalnega znanja o stanju mape, ki ga posameznim agentom posreduje **oddajni agent**. Že sam algoritem PRIMAL2 uporablja enega izmed aktivnih agentov v okolju, ki koordinira korake in posreduje informacije iz okolja in nagrade v skupni pomnilnik.

Predlagana izboljšava razširi koncept učenja sledenja [8] tako, da oddajnemu agentu dodelimo še eno nalogo. Agent dodatno pridobi podatke o gostoti prometa na mapi in jih posreduje v skupni pomnilnik z ostalimi informacijami, da lahko do teh podatkov dostopajo vsi ostali agenti.

Takšen pristop agentom omogoča učinkovitejše načrtovanje poti, saj lokalne odločitve sprejemajo na podlagi globalnega vpogleda v stanje sistema. S tem se izboljša usklajenost gibanja agentov in zmanjša možnost, da bi se večje število agentov hkrati usmerilo v ozke predele na mapi.



Slika 2: Vizualizacija nevronske mreže nad lastno implementacijo.

V naši implementaciji smo uporabili globalno toplotno karto velikosti 40 × 40 namesto popolne ločljivosti okolja, ki je v našem primeru 160 × 160. Odločitev temelji na dejstvu, da ni potrebe po popolni ločljivosti toplotne karte, saj je cilj pridobiti informacije o okvirni gostoti okolja, ne pa kje se nahajajo vsi agenti, saj je to že centraliziran pristop k reševanju problema. Hkrati pa tudi prihranimo na računski zahtevnosti algoritma in lažja je tudi implementacija vhoda v nevronsko mrežo. Karta vsebuje položaje agentov v določenem trenutku, pri čemer so njihove pozicije predstavljene z enicami. V vsaki iteraciji se karta prek enega agenta posreduje v skupni prostor, do katerega imajo dostop vsi agenti.

Nevronski mreži smo dodali dodaten vhodni tok, ki sprejme podatke toplotne karte in jih obdela s konvolucijskim slojem. Izhod tega sloja se nato splošči in združi z ostalimi vhodnimi podatki, kot je prikazano na Sliki 2.

Samo nevronsko mrežo smo zaradi sprememb ponovno trenirali. Uporabili smo enake parametre kot avtorji pristopa PRIMAL2 [3]. Dolžina epizod je znašala 256 korakov, dolžina epizod pri učenju s posnemanjem pa 64 korakov. Uporabljen je bil optimizator NAdam s hitrostjo učenja $2 \cdot 10^{-5}$, diskontni faktor γ pa je bil nastavljen na 0,95. Treniranje je potekalo na lokalnem sistemu s procesorjem Ryzen 7 7800X3D, 8 jedri in 32 GB pomnilnika ter je trajalo 32 tisoč epizod.

4 Eksperiment in analiza rezultatov

Eksperimentalno okolje je zasnovano po vzoru ogrodja PRIMAL2, kjer je prostor predstavljen kot diskretna mreža, v kateri agenti, cilji in ovire zasedajo posamezne celice.

V okviru eksperimentalnega dela smo izvedli primerjalno analizo naslednjih pristopov:

- Conformant-CBS [7],
- Guided-PIBT [9],
- Learn to Follow (Follower in FollowerLite) [8],
- MAPF-GPT [1],
- PRIMAL2 [3],
- PRIMAL2 z razširjenim modelom, ki vključuje globalno karto zasedenosti.

Za zagotavljanje poštene primerjave so bili vsi algoritmi zagnani pod enakimi pogoji, kar vključuje uporabo istega nabora zemljevidov, enakega števila agentov ter enako število časovnih korakov. Takšen pristop omogoča neposredno primerjavo rezultatov, saj smo za vrednosti parametrov algoritmov uporabili enake vrednosti kot v osnovnih člankih.

Eksperimente smo za vsak pristop izvedli 25-krat, pri čemer smo analizirali povprečne vrednosti rezultatov. S tem smo zmanjšali vpliv naključnosti in povečali zanesljivost ugotovitev. Največje število časovnih korakov simulacije je bilo omejeno na 256 korakov, kar je omogočilo enotno primerjavo vseh pristopov.

Za vrednotenje smo uporabili naslednje metrike:

- prepustnost, ki predstavlja povprečno število uspešno zaključenih nalog na časovni korak simulacije:

$$prepustnost = \frac{N_{tasks}}{T}$$

  kjer N_{tasks} predstavlja število zaključenih nalog, T pa skupni čas izvajanja,
- število trkov, ki meri konfliktne situacije med agenti,
- celotni čas izvajanja simulacije.

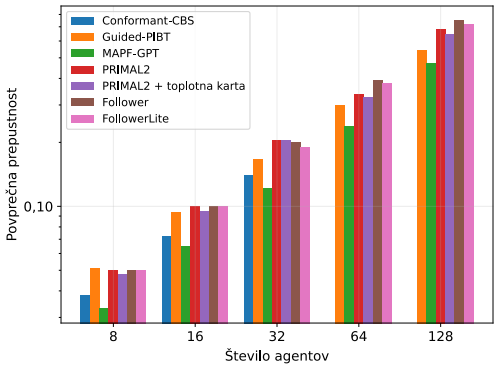
Eksperiment smo izvedli v naslednji konfiguraciji:

- velikost okolja: 160 × 160,
- število agentov: 8, 16, 32, 64, 128,
- gostota ovir: 33 %,
- struktura okolja: mreža z naključno generiranimi ovirami in krajšimi zidovi.

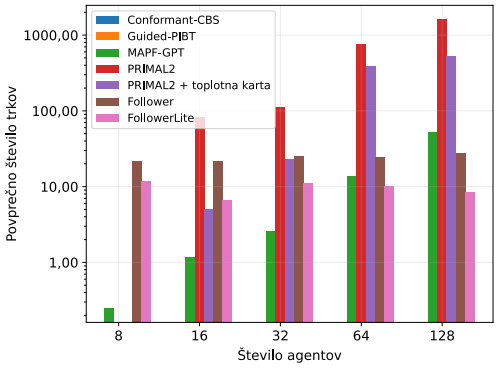
Vsi algoritmi so bili testirani pod enakimi pogoji, kar omogoča neposredno primerjavo njihove učinkovitosti.

4.1 Analiza rezultatov

Iz Slike 3 je razvidno, da prepustnost pri vseh algoritmih narašča z večanjem števila agentov. Najboljšo prepustnost pri 128 agentih doseže Follower, zelo blizu mu sledi FollowerLite, potem PRIMAL2 in naša izboljšana različica s toplotno karto. Guided-PIBT in MAPF-GPT pri največjem številu agentov zaostajata za PRIMAL2. Conformant-CBS dosega nižjo prepustnost in pri 64 agentih odpove, zato ni primeren za večje število agentov.

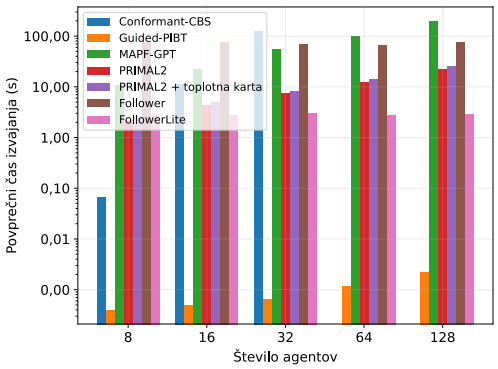


Slika 3: Povprečna prepustnost (logaritemska skala).



Slika 4: Povprečno število trkov (logaritemska skala).

Slika 4 prikazuje, da imajo Conformant-CBS, Guided-PIBT, MAPF-GPT in Follower skozi nabor števila agentov relativno nizko število trkov. Najbolj odstopa PRIMAL2 (0, 83, 113, 762 in 1594), pri katerem se število trkov pri 64 in predvsem pri 128 agentih izrazito poveča. Manjše odstopanje je opazno tudi pri naši izboljšani različici PRIMAL2 (0, 5, 23, 388 in 521).



Slika 5: Povprečni čas trajanja do zaključka vseh nalog (logaritemska skala).

Na Sliki 5 vidimo, da sta Guided-PIBT in Follower najhitrejša, saj čas izvajanja ostaja zelo nizek skozi celoten razpon agentov. PRIMAL2 in naša izboljšana različica PRIMAL2 s toplotno karto z večanjem števila agentov naraščata postopno, vendar ostajata bistveno hitrejša od MAPF-GPT. Največje povečanje časa izvajanja opazimo pri MAPF-GPT, ki pri 128 agentih doseže skoraj 200 sekund. Conformant-CBS pri 32 agentih že preseže 125 sekund, pri 64 agentih pa odpove, zato se pri večjem številu agentov ne izkaže kot primeren.

5 Zaključek

V članku smo obravnavali problem vseživljenjskega večagentnega iskanja poti v dinamičnih okoljih. Analizirali smo obstoječe pristope za reševanje MAPF in LMAPF ter njihove omejitve in lastnosti reševanja pri večjem številu agentov.

Na podlagi analize sorodnih del smo nadgradili algoritem PRIMAL2 z uporabo skupne toplotne karte zasedenosti. Cilj izboljšave je bil zmanjšati število konfliktov in izboljšati koordinacijo med agenti.

Rezultati so pokazali, da lahko dodatne globalne informacije v določenih primerih izboljšajo delovanje sistema. V večjih okoljih smo opazili manj trkov in boljšo koordinacijo agentov. Pri manjšem številu agentov naša implementacija ni prinesla izboljšav, saj se zaradi premajhnega števila agentov niso oblikovala območja z visoko gostoto agentov. Z večanjem njihovega števila pa je toplotna karta postala učinkovita.

Ugotovili smo, da kombinacija lokalnega odločanja in omejenih globalnih informacij predstavlja smiseln pristop za reševanje problema LMAPF. V prihodnje bi bilo smiselno izboljšati način posredovanja globalnih informacij ter pristop preizkusiti na večjih in bolj kompleksnih okoljih.

Literatura

[1] Anton Andreychuk, Konstantin Yakovlev, Aleksandr Panov, and Alexey Skrynnik. 2025. MAPF-GPT: Imitation Learning for Multi-Agent Pathfinding at Scale. *Proceedings of the AAAI Conference on Artificial Intelligence* 39, 22 (Apr. 2025), 23126–23134. https://doi.org/10.1609/aaai.v39i22.34477

[2] Zhe Chen, Daniel Harabor, Jiaoyang Li, and Peter J Stuckey. 2024. Traffic flow optimisation for lifelong multi-agent path finding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38. 20674–20682. https://doi.org/10.1609/aaai.v38i18.30054

[3] Mehul Damani, Zhiyao Luo, Emerson Wenzel, and Guillaume Sartoretti. 2021. PRIMAL _2: Pathfinding via reinforcement and imitation multi-agent learning-lifelong. *IEEE Robotics and Automation Letters* 6, 2 (2021), 2666–2673. https://doi.org/10.1109/LRA.2019.2903261

[4] Andrea Giuffrida, Nicola Basilico, and Francesco Amigoni. 2025. An empirical evaluation of learning-based multi-agent path finding algorithms in warehouse environments. *Robotics and Autonomous Systems* 194 (2025), 105149. https://doi.org/10.1016/j.robot.2025.105149

[5] Jiaoyang Li, Andrew Tinka, Scott Kiesel, Joseph W Durham, TK Satish Kumar, and Sven Koenig. 2021. Lifelong multi-agent path finding in large-scale warehouses. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 11272–11281. https://doi.org/10.1609/aaai.v35i13.17344

[6] Ning Liu, Sen Shen, Xiangrui Kong, Hongtao Zhang, and Thomas Bräunl. 2025. Cooperative Hybrid Multi-Agent Pathfinding Based on Shared Exploration Maps. *Journal of Intelligent & Robotic Systems* 111 (2025), 97. https://doi.org/10.1007/s10846-025-02304-6

[7] Tomer Shahar, Shashank Shekhar, Dor Atzmon, Abdallah Saffidine, Brendan Juba, and Roni Stern. 2021. Safe Multi-Agent Pathfinding with Time Uncertainty. *J. Artif. Int. Res.* 70 (May 2021), 923–954. https://doi.org/10.1613/jair.1.12397

[8] Alexey Skrynnik, Anton Andreychuk, Maria Nesterova, Konstantin Yakovlev, and Aleksandr Panov. 2024. Learn to follow: Decentralized lifelong multi-agent pathfinding via planning and learning. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 38. 17541–17549. https://doi.org/10.1609/aaai.v38i16.29704

[9] Roni Stern. 2019. *Multi-Agent Path Finding – An Overview.* Springer International Publishing, Cham, 96–115. https://doi.org/10.1007/978-3-030-33274-7_6

 DOI: https://doi.org/10.26493/scores26.15

Večagentno odločanje v igri Pac-Man z uporabo algoritmov tekmovalnega iskanja

Sara Hristova
sara.hristova@student.um.si
Fakulteta za elektrotehniko, računalništvo in informatiko,
Univerza v Mariboru
Koroška cesta 46
SI-2000 Maribor, Slovenija

Irina Mićović
irina.micovic@student.um.si
Fakulteta za elektrotehniko, računalništvo in informatiko,
Univerza v Mariboru
Koroška cesta 46
SI-2000 Maribor, Slovenija

Borko Bošković
borko.boskovic@um.si
Fakulteta za elektrotehniko, računalništvo in informatiko,
Univerza v Mariboru
Koroška cesta 46
SI-2000 Maribor, Slovenija

Klemen Berkovič
klemen.berkovic1@um.si
Fakulteta za elektrotehniko, računalništvo in informatiko,
Univerza v Mariboru
Koroška cesta 46
SI-2000 Maribor, Slovenija

POVZETEK

Prispevek obravnava večagentno odločanje v igri Pac-Man. Razvili smo algoritem HybridMCTSAgent (HMA), ki temelji na Monte Carlo simulacijah s posodobljeno hevristično evalvacijsko funkcijo, ter ga primerjali z algoritmi Minimax, Alpha-Beta Pruning, Expectimax in Monte Carlo Tree Search (MCTS). Eksperimente smo izvedli na štirih zemljevidih. Pri klasičnih algoritmih smo preverili globine iskanja 1, 2 in 3, pri MCTS in HMA pa 10, 20 in 40 simulacij v kombinaciji z globinami simulacije 4, 6 in 8. Skupaj smo ovrednotili 108 konfiguracij. Za vsako konfiguracijo je bilo načrtovanih 50 iger, skupaj 5400. Posamezna igra je bila omejena na 25 s, zato je bilo uspešno zaključenih 5110 iger, 290 pa jih je bilo prekinjenih zaradi časovne omejitve igre. Pri analizi smo za vsak algoritem na posameznem zemljevidu izbrali najboljšo konfiguracijo ter jih primerjali glede na povprečno oceno igre in delež zmag. Na zemljevidu `openClassic` je HMA dosegel 72 % zmag, MCTS pa 54 %. Na `trickyClassic` je HMA dosegel najvišjo povprečno oceno 1032,77, na `minimaxClassic` pa je bil najuspešnejši MCTS z 82 % zmag in povprečno oceno 333,78. Rezultati kažejo, da se uspešnost algoritmov in konfiguracij razlikuje glede na zemljevid in parametre.

KLJUČNE BESEDE

večagentni sistemi, Pac-Man, Monte Carlo Tree Search, HybridMCTSAgent, tekmovalno iskanje

1 UVOD

Večagentni sistemi obravnavajo okolja, kjer agenti sprejemajo odločitve glede na lastne cilje in vedenje drugih agentov. Pac-Man je primerno okolje, saj vključuje agente z različnimi vlogami in cilji. Pac-Man se po labirintu premika v dovoljenih smereh ter zbira hrano in kapsule. Drugi agenti so duhovi, ki lahko ob stiku s Pac-Manom povzročijo njegov poraz.

Ko Pac-Man zaužije kapsulo, duhovi začasno preidejo v prestrašeno stanje, v katerem jih lahko zaužije in pridobi dodatne točke. Igra se konča z zmago, ko Pac-Man zaužije vso hrano, izgubi pa, če ga ujame aktiven oziroma neprestrašen duh. Pri izbiri akcij mora zato upoštevati zbiranje hrane, uporabo kapsul, položaj duhov in nevarnost, ki jo predstavljajo. Zaradi velikega prostora stanj, negotovosti prihodnjih potez in nasprotujočih si ciljev je takšno okolje primerno za primerjavo klasičnega tekmovalnega iskanja in simulacijskih metod.

Obravnavani problem lahko formalno opišemo z množico stanj S in množico dovoljenih akcij $A(s)$ za posamezno stanje $s \in S$. Stanje igre vsebuje informacije o položaju Pac-Mana in duhov, preostali hrani in kapsulah ter trenutnem rezultatu igre. Izvedba akcije $a \in A(s)$ privede v naslednje stanje s', zato lahko potek igre predstavimo kot zaporedje stanj

$$s_0 \rightarrow s_1 \rightarrow \cdots \rightarrow s_t.$$

Cilj Pac-Mana je izbirati akcije, ki vodijo do ugodnejših stanj ob upoštevanju vedenja drugih agentov.

V delu primerjamo Minimax, Alpha-Beta Pruning, Expectimax in MCTS z lastnim algoritmom HMA. Raziskovalno vprašanje je, ali lahko MCTS z domensko specifičnim hevrističnim vrednotenjem doseže boljše rezultate od osnovnega MCTS in klasičnih algoritmov. HMA zato ne spreminja ideje Monte Carlo iskanja, temveč način vrednotenja simuliranih stanj dopolni z informacijami o hrani, kapsulah in oddaljenosti od duhov.

V poglavju 2 predstavimo sorodna dela, pomembna za zasnovo HMA. V poglavju 3 opišemo osnovni MCTS in našo hevristično razširitev. V poglavju 4 opišemo eksperimentalno okolje in analiziramo rezultate in v poglavju 5 povzamemo ključne ugotovitve.

2 SORODNA DELA

Pregled MCTS in njegovih razširitev v [6] je pomemben, saj obravnava načine prilagajanja posameznih faz MCTS, med drugim strategije izbire vozlišč in vrednotenja simuliranih stanj. MCTS ocenjuje akcije z večkratnim vzorčenjem možnih prihodnjih potekov. Takšen pristop je smiseln v prostorih z velikim številom stanj, kjer je popolno preiskovanje neizvedljivo. To je neposredno izhodišče za HMA, kjer smo oceno simulacije dopolnili z domensko specifično hevristiko.

V študiji primerjave igralnih agentov [5] je bil Minimax z Alpha-Beta obrezovanjem ovrednoten pri različnih globinah iskalnega drevesa. Minimax predpostavlja nasprotno vedenje drugega igralca, Alpha-Beta obrezovanje pa ohrani enako odločitveno pravilo, vendar zmanjša število pregledanih vozlišč z odstranjevanjem vej, ki ne morejo vplivati na končno odločitev. Avtorja sta eksperimentalno preverila globine iskanja od 1 do 4 in pri izbiri nastavitev upoštevala uspešnost agenta in čas, potreben za izvedbo poteze. Zaradi računske zahtevnosti sta pri globini 4 izvedla manj ponovitev kot pri globinah 1, 2 in 3. Rezultati kažejo kompromis med globino iskanja, uspešnostjo agenta in računskim stroškom. Zato tudi v našem eksperimentu globino iskanja obravnavamo kot eksperimentalni parameter ter za Minimax, Alpha-Beta Pruning in Expectimax primerjamo globine 1, 2 in 3.

Delo [4] obravnava učinkovitost odločanja v okolju Pac-Man ter vrednotenje uspešnosti in časa izvajanja, kar smo upoštevali pri izboru eksperimentalnih metrik.

V delu [3] so izpostavljeni problemi nestacionarnosti, koordinacije in skalabilnosti pri večagentnem ojačitvenem učenju. V [2] je okolje Pac-Man predstavljeno kot primerno za večagentno iskanje ter za obravnavo deterministočnih, stohastičnih in tekmovalnih problemov. V našem delu je to pomembno pri primerjavi algoritmov Minimax in Expectimax: Minimax predpostavlja, da nasprotni agent izbere akcijo, ki je za Pac-Mana najmanj ugodna, medtem ko Expectimax možne akcije drugih agentov obravnava glede na njihove verjetnosti in izbiro akcije temelji na pričakovani vrednosti.

Pregled MCTS v [1] pokaže prednosti vzorčenja v prostorih z velikim številom stanj in predstavlja teoretično osnovo za naš simulacijski pristop. Skupaj ta dela utemeljujejo primerjavo treh različnih načinov odločanja: Minimax in Alpha-Beta Pruning predstavljata deterministično tekmovalno iskanje, Expectimax uporablja pričakovano vrednotenje možnih potez drugih agentov, MCTS in HMA pa temeljita na Monte Carlo simulacijah.

3 ALGORITEM HYBRIDMCTSAGENT

Razvili smo algoritem HybridMCTSAgent (HMA), ki simulacijski pristop Monte Carlo Tree Search (MCTS) razširja z domensko specifičnim hevrističnim vrednotenjem stanj igre Pac-Man. Pri ocenjevanju možnih akcij poleg rezultatov simulacij upoštevamo tudi informacije o hrani, kapsulah ter položaju in stanju duhov.

Klasični MCTS temelji na štirih fazah: izbiri, razširitvi, simulaciji in povratnem posodabljanju. V fazi izbire izbere perspektivno vozlišče, pri razširitvi doda novo stanje, nato simulira nadaljnji potek igre do določene globine, rezultat pa povratno prenese skozi obiskana vozlišča in posodobi njihove ocene. Simulacijski del uporabljamo za vrednotenje dovoljenih akcij.

HMA združuje Monte Carlo simulacije z domensko specifično hevristično funkcijo $h(s)$. Hevristična funkcija upošteva trenutni rezultat igre, oddaljenost do najbližje hrane, število preostalih kosov hrane in kapsul ter oddaljenost in stanje duhov. Za neterminalno stanje jo definiramo kot

$$h(s) = S(s) + \frac{20}{d_f(s)+1} - 4N_f(s) - 15N_c(s) + \sum_{g\in G}\phi_g(s), \quad (1)$$

kjer je $S(s)$ trenutni rezultat igre, $d_f(s)$ Manhattanova razdalja med Pac-Manom in najbližjim preostalim kosom hrane, $N_f(s)$ število preostalih kosov hrane, $N_c(s)$ pa število preostalih kapsul. Množica G vsebuje vse duhove v trenutnem stanju igre. Če na zemljevidu ni več hrane, se člen z $d_f(s)$ ne uporabi.

Za posameznega duha $g \in G$ označimo z $d_g(s)$ Manhattanovo razdaljo med trenutnim položajem Pac-Mana in položajem tega duha. Prispevek duha modeliramo kot:

$$\phi_g(s) = \begin{cases} \dfrac{30}{d_g(s)+1}, & \text{če je duh prestrašen,} \\ -1000, & \text{če duh ni prestrašen in } d_g(s) \le 1, \\ -\dfrac{100}{d_g(s)}, & \text{če duh ni prestrašen in } 1 < d_g(s) \le 3, \\ 2d_g(s), & \text{če duh ni prestrašen in } d_g(s) > 3. \end{cases} \quad (2)$$

Zmagovalno stanje nastopi, ko Pac-Man zaužije vso preostalo hrano, poraženo pa ob stiku z aktivnim, neprestrašenim duhom. Pri zmagovalnem stanju hevristična funkcija vrne $+\infty$, pri poraženem stanju pa $-\infty$. Ti vrednosti ne predstavljata dejanskega rezultata igre, temveč posebni vrednosti evalvacijske funkcije, s katerima ima zmaga pri primerjavi stanj največjo, poraz pa najmanjšo prioriteto. Takšno hevristično vrednotenje spodbuja Pac-Mana k približevanju hrani in prestrašenim duhovom, zmanjševanju preostale hrane in kapsul ter izogibanju aktivnim duhovom.

Za vsako dovoljeno akcijo a HMA najprej ustvari naslednje stanje s_a. Iz tega stanja izvede N naključnih simulacij do največje globine d_r oziroma do terminalnega stanja, če je to doseženo prej. Končno doseženo stanje posamezne simulacije ovrednotimo z že definirano hevristično funkcijo $h(s)$. Povprečno simulacijsko oceno akcije zapišemo kot

$$\overline{V}_{\text{sim}}(a) = \frac{1}{N}\sum_{i=1}^{N} h\left(s_i^{(a)}\right), \quad (3)$$

kjer $s_i^{(a)}$ predstavlja stanje, doseženo na koncu i-te simulacije po izbiri akcije a, bodisi zaradi dosežene omejitve globine bodisi zaradi terminalnega stanja.

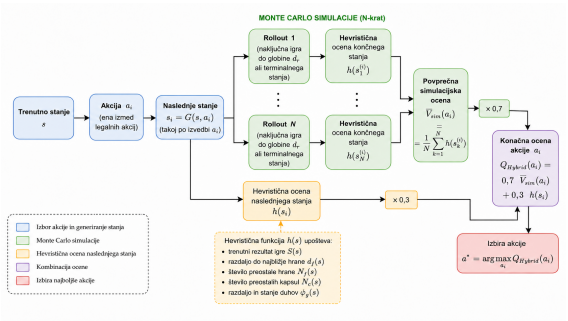
HMA poleg simulacij neposredno ovrednoti tudi stanje s_a, ki nastane takoj po izvedbi obravnavane akcije. Končna ocena akcije je določena kot

$$Q_{\text{Hybrid}}(a) = 0.7\,\overline{V}_{\text{sim}}(a) + 0.3\,h(s_a). \quad (4)$$

Agent nato izbere akcijo z največjo vrednostjo $Q_{\text{Hybrid}}(a)$.Povprečna Monte Carlo ocena tako predstavlja 70 % končne ocene, neposredna hevristična ocena naslednjega stanja pa preostalih 30 %. Uteži posameznih členov hevristične funkcije ter razmerje 0,7/0,3 smo določili hevristično, brez ločenega postopka optimizacije, pri čemer smo večji vpliv namenili rezultatom Monte Carlo simulacij. S tem simulacije predstavljajo glavni del odločitve, hevristično vrednotenje pa usmerja izbiro proti stanjem, ki so glede na trenutno razporeditev hrane, kapsul in duhov ugodnejša za Pac-Mana.

Slika 1 prikazuje postopek vrednotenja posamezne dovoljene akcije v algoritmu HMA. Iz naslednjega stanja po izbrani akciji izvedemo N simulacij, njihova končna stanja pa ovrednotimo s hevristično funkcijo iz enačbe (1). Povprečna ocena teh simulacij predstavlja 70 % končne ocene akcije, neposredna hevristična ocena naslednjega stanja pa preostalih 30 %. Agent nato izbere akcijo z največjo vrednostjo Q_{Hybrid}.

 DOI: https://doi.org/10.26493/scores26.15



Slika 1: Postopek vrednotenja akcij in komponente hevristične funkcije v algoritmu HybridMCTSAgent (HMA).

Implementacija temelji na odprtokodnem okolju Pac-Man[1], ki ponuja vmesnik za različne odločitvene agente ter izvajanje eksperimentov na različnih zemljevidih. Pac-Man predstavlja agenta, ki se obnaša glede na uporabljeni algoritem, okolje pa na podlagi izbrane akcije izvede njegov premik. Duhovi se v uporabljenem okolju obnašajo skladno s privzetim algoritmom RandomGhost, ki dovoljene akcije izbira naključno. HMA je implementiran v programskem jeziku Python kot dodatni odločitveni agent in uporablja obstoječe mehanizme okolja za pridobivanje dovoljenih akcij ter generiranje naslednjih stanj igre.

4 EKSPERIMENT IN REZULTATI

Eksperimente smo izvedli v programskem jeziku Python v odprtokodnem okolju Pac-Man. Eksperimentalne skripte avtomatizirajo izbiro agenta, zemljevida in vrednosti parametrov ter rezultate shranijo v datoteke CSV. Primerjali smo pet algoritmov: Minimax, Alpha-Beta Pruning, Expectimax, MCTS in HMA.

Uspešnost algoritmov smo primerjali glede na povprečno oceno in delež zmag. Poročamo tudi standardni odklon ocen iger in povprečni čas izvajanja posamezne konfiguracije algoritma. Čas posamezne igre smo omejili na 25 s. Daljše igre so bile prekinjene. Na štirih obravnavanih zemljevidih je bilo načrtovanih 5400 iger, od katerih je bilo uspešno zaključenih 5110, 290 pa jih je bilo prekinjenih zaradi časovne omejitve igre.

Ocena igre je vsota nagrad in kazni, ki jih Pac-Man prejme med igro. V uporabljeni implementaciji se ocena poveča za 10 točk ob zaužitju hrane in za 200 točk ob zaužitju prestrašenega duha, zmaga doda 500 točk, poraz pa 500 točk odšteje. Čas igranja zmanjša vrednost ocene igre, pri čemer vsaka sekunda igranja zmanjša oceno za 1. Rezultat igre zapišemo kot

$$S = 10N_{\text{food}}^{\text{eat}} + 200N_{\text{ghost}}^{\text{eat}} + 500I_{\text{win}} - 500I_{\text{lose}} - N_t, \qquad (5)$$

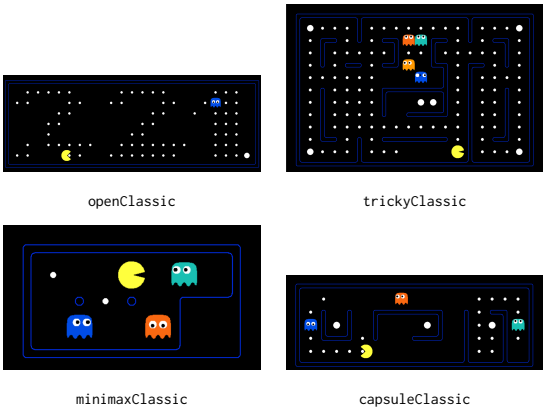
kjer $N_{\text{food}}^{\text{eat}}$ pomeni število zaužitih kosov hrane, $N_{\text{ghost}}^{\text{eat}}$ število zaužitih prestrašenih duhov, N_t čas igranja agenta, I_{win} je indikator zmage in I_{lose} indikator poraza. Ocena igre je lahko tudi negativna, ker negativna vrednost poraza in časa igranja preseže pozitivno vrednost hrane in zaužitih duhov.

Konfiguracija pomeni kombinacijo algoritma, njegovih parametrov in zemljevida. Pregled uporabljenih parametrov in obsega eksperimenta je prikazan v tabeli 1. Pri klasičnih algoritmih smo uporabili

[1]https://github.com/Abdelaal495/Multi-Agent-Pac-Man

Tabela 1: Eksperimentalni prostor parametrov.

Parameter	Vrednosti
Zemljevidi	openClassic, trickyClassic, minimaxClassic, capsuleClassic
Klasični algoritmi	Minimax, Alpha-Beta, Expectimax
Globina klasičnega iskanja	1, 2, 3
MCTS / HybridMCTS simulacije	10, 20, 40
Globina simulacije	4, 6, 8
Ponovitve na konfiguracijo	50
Konfiguracije	108
Načrtovane / zaključene igre	5400 / 5110



Slika 2: Uporabljeni zemljevidi v eksperimentu.

globine iskanja 1, 2 in 3. Pri MCTS in HMA smo uporabili tri vrednosti števila Monte Carlo simulacij (10, 20 in 40) ter tri globine simulacije (4, 6 in 8). Na posameznem zemljevidu smo preverili 27 konfiguracij: po tri za vsak klasični algoritem ter po devet za MCTS in HMA. Na štirih zemljevidih je bilo skupaj preverjenih 108 konfiguracij. Za vsako konfiguracijo je bilo načrtovanih 50 iger, skupaj 5400 iger.

V oznakah konfiguracij `d` predstavlja globino iskanja, `s` število Monte Carlo simulacij, `r` pa globino simulacije. Tako `Minimax_d3` pomeni Minimax z globino 3, `MCTS_s40_r8` pa MCTS s 40 simulacijami in globino 8. Delež zmag smo izračunali kot razmerje med številom zmag in vsemi 50 načrtovanimi igrami posamezne konfiguracije. Pri izračunu deleža zmag smo časovno prekinjene igre obravnavali kot poraze. Povprečna ocena je aritmetična sredina rezultatov izvedenih iger posamezne konfiguracije, standardni odklon pa opisuje razpršenost posameznih rezultatov okoli povprečne vrednosti. Povprečni čas izvajanja je povprečni čas zagona algoritma izbrane konfiguracije.

Slika 2 prikazuje štiri uporabljene zemljevide, ki se razlikujejo po velikosti, geometriji hodnikov, razporeditvi hrane in kapsul ter položajih Pac-Mana in duhov. `openClassic` je bolj odprt, `trickyClassic` vsebuje zahtevnejše prehode, `capsuleClassic` poudarja vlogo kapsul, `minimaxClassic` pa predstavlja kompaktnejšo postavitev.

4.1 Analiza rezultatov

Ker smo za posamezne algoritme uporabili več različnih nastavitev parametrov, vseh 108 konfiguracij v končni primerjavi ne prikazujemo. Za izbrane konfiguracije poročamo povprečno oceno igre, standardni odklon ocene, število zmag, število zaključenih iger od 50 načrtovanih, delež zmag, povprečni čas izvajanja in status.

Tabela 2: Primerjava najboljših konfiguracij algoritmov na posameznih zemljevidih.

Zemljevid	Konfiguracija	Povp. ocena	Std. odklon	Zmage	Zaključene igre	Delež zmag	Povp. čas [s]	Status
openClassic	Minimax_d3	-1279,78	751,57	0	36/50	0,00%	**15,92**	✖
openClassic	AlphaBeta_d3	-873,70	652,85	1	27/50	2,00%	17,77	✖
openClassic	Expectimax_d3	-940,60	631,82	0	10/50	0,00%	22,27	✖
openClassic	MCTS_s40_r8	**875,89**	325,14	27	28/50	54,00%	20,28	✖
openClassic	Hybrid_s40_r8	868,72	**223,13**	**36**	36/50	**72,00%**	**15,92**	✖
trickyClassic	Minimax_d3	548,44	**310,10**	0	18/50	0,00%	22,17	✖
trickyClassic	AlphaBeta_d3	416,25	574,18	0	20/50	0,00%	20,36	✖
trickyClassic	Expectimax_d3	816,90	336,74	0	10/50	0,00%	23,59	✖
trickyClassic	MCTS_s40_r6	937,60	654,74	**10**	47/50	**20,00%**	12,73	✖
trickyClassic	Hybrid_s40_r6	**1032,77**	822,01	8	48/50	16,00%	**11,79**	✖
minimaxClassic	Minimax_d1	290,96	422,59	39	50/50	78,00%	0,10	✔
minimaxClassic	AlphaBeta_d1	88,54	503,75	29	50/50	58,00%	**0,08**	✔
minimaxClassic	Expectimax_d3	148,08	487,93	32	50/50	64,00%	0,09	✔
minimaxClassic	MCTS_s40_r8	**333,78**	**391,56**	**41**	50/50	**82,00%**	0,11	✔
minimaxClassic	Hybrid_s40_r4	127,88	496,84	31	50/50	62,00%	0,11	✔
capsuleClassic	Minimax_d2	-270,76	**220,14**	0	50/50	0,00%	**0,18**	✔
capsuleClassic	AlphaBeta_d3	-109,64	380,92	2	50/50	4,00%	0,33	✔
capsuleClassic	Expectimax_d3	-125,10	314,65	1	50/50	2,00%	0,51	✔
capsuleClassic	MCTS_s20_r4	70,16	540,73	**10**	50/50	**20,00%**	0,76	✔
capsuleClassic	Hybrid_s10_r6	**115,06**	607,79	8	50/50	16,00%	0,62	✔

Tabela 2 prikazuje izbrane konfiguracije vseh petih algoritmov na štirih zemljevidih, pri čemer smo za vsak algoritem na posameznem zemljevidu izbrali konfiguracijo z najvišjo povprečno oceno igre. V stolpcu status znak ✖ pomeni, da zaradi časovne omejitve 25 s niso bile zaključene vse načrtovane igre posamezne konfiguracije, znak ✔ pa pomeni, da je bilo odigranih vseh 50 načrtovanih iger. Krepke vrednosti označujejo najboljše rezultate v posameznih stolpcih za vsak zemljevid.

Na zemljevidu `openClassic` imata algoritma MCTS in HMA najvišji povprečni oceni. MCTS s 40 simulacijami in globino simulacije 8 je dosegel povprečno oceno 875,89 ter 54 % zmag. HMA z enakima kontrolnima parametroma pa 868,72 in 72 % zmag. Klasični algoritmi so dosegli negativne povprečne ocene in zelo nizke deleže zmag.

Na zemljevidu `trickyClassic` je najvišjo povprečno oceno dosegel HMA s 40 simulacijami in globino 6, in sicer 1032,77, MCTS pa 937,60. MCTS je imel višji delež zmag, 20,00 % v primerjavi s 16,00 % pri HMA. Med klasičnimi algoritmi je bil najboljši Expectimax z globino 3 in povprečno oceno 816,90.

Na zemljevidu `minimaxClassic` je bil glede na povprečno oceno igre in delež zmag najuspešnejši MCTS s 40 simulacijami in globino 8, s povprečno oceno 333,78 in 82 % zmag. Najboljši klasični algoritem je bil Minimax z globino 1 (290,96 in 78 % zmag). HMA s 40 simulacijami in globino simulacije 4 je dosegel povprečno oceno 127,88 ter 62 % zmag.

Na zemljevidu `capsuleClassic` je najvišjo povprečno oceno dosegel HMA z 10 simulacijami in globino 6, in sicer 115,06. MCTS z 20 simulacijami in globino 4 je dosegel nekoliko nižjo povprečno oceno 70,16, vendar višji delež zmag, 20 % v primerjavi s 16 % pri HMA. Klasični algoritmi so imeli negativne povprečne ocene in med 0 % in 4 % zmag.

Najuspešnejši algoritem ni enak na vseh zemljevidih, višja povprečna ocena pa ne pomeni nujno višjega deleža zmag, kar je posebej vidno na zemljevidih `trickyClassic` in `capsuleClassic`. Rezultati zato potrjujejo smiselnost hkratnega upoštevanja povprečne ocene igre in deleža zmag pri primerjavi izbranih konfiguracij.

5 ZAKLJUČEK

V delu smo primerjali štiri uveljavljene pristope z algoritmom HMA, ki MCTS dopolni z domensko specifičnim hevrističnim vrednotenjem. Eksperimente smo izvedli na štirih zemljevidih in preverili različne vrednosti parametrov posameznih algoritmov. Skupaj je bilo načrtovanih 5400 iger, od katerih je bilo zaradi časovne omejitve 25 s uspešno zaključenih 5110.

Analiza najboljših konfiguracij je pokazala, da se uspešnost algoritmov in izbira parametrov razlikujeta glede na zemljevid. MCTS in HMA sta na več zemljevidih dosegla višje povprečne ocene in deleže zmag od klasičnih algoritmov. Na `openClassic` je HMA dosegel 72 % zmag, MCTS pa nekoliko višjo povprečno oceno. Na `trickyClassic` je najvišjo povprečno oceno dosegel HMA, MCTS pa višji delež zmag. Na `minimaxClassic` je bil glede na povprečno oceno in delež zmag najuspešnejši MCTS, na `capsuleClassic` pa je HMA dosegel višjo povprečno oceno, MCTS pa višji delež zmag.

Rezultati kažejo, da pri vrednotenju algoritmov ni smiselno upoštevati le ene metrike, saj višja povprečna ocena ne pomeni vedno višjega deleža zmag. Prav tako ista konfiguracija ni bila najuspešnejša na vseh zemljevidih, zato struktura zemljevida in izbrani parametri vplivajo na uspešnost algoritma.

Nadaljnje delo bi lahko vključevalo več zemljevidov, večje število ponovitev ter nadaljnjo prilagoditev simulacijskih parametrov in hevristične funkcije glede na strukturo zemljevida. Smiselno bi bilo raziskati tudi uporabo nevronskih mrež in evolucijskih agentov ter njihovo uspešnost primerjati s HMA v enakih eksperimentalnih pogojih.

VIRI

[1] Cameron B. Browne et al. 2020. Monte Carlo Tree Search: A Review of Recent Modifications and Applications. *arXiv preprint arXiv:2008.01508* (2020).

[2] John DeNero and Dan Klein. 2010. Teaching Introductory Artificial Intelligence with Pac-Man. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 24. 1885–1889. https://doi.org/10.1609/aaai.v24i3.18829

[3] Sven Gronauer and Klaus Diepold. 2021. Multi-agent deep reinforcement learning: a survey. *Artificial Intelligence Review* 55 (2021), 895–943. https://doi.org/10.1007/s10462-021-09996-w

[4] Artem Novikov and Volodymyr Yanovskyi. 2024. Analysis of the decision-making algorithm efficiency in complex game environments on the example of Pac-Man. *Information Technologies and Computer Engineering* 21, 3 (2024), 108–118. https://doi.org/10.63341/vitce/3.2024.108

[5] Santiago Videgaín and Pablo García Sánchez. 2021. Performance Study of Minimax and Reinforcement Learning Agents Playing the Turn-based Game Iwoki. *Applied Artificial Intelligence* 35, 10 (2021), 717–744. https://doi.org/10.1080/08839514.2021.1934265

[6] Maciej Świechowski et al. 2021. Monte Carlo Tree Search: A Review of Recent Modifications and Applications. *Artificial Intelligence Review* 54, 7 (2021), 5497–5542.

Natural Language Processing, Speech and Human-Computer Interaction

Heuristic Evaluation of Mobile Web E-Commerce Checkout: A Domain-Specific Binary Checklist and Scoring Method

Olha Tushych and Alla Toropenko

https://doi.org/10.26493/scores26.01

Sestava specializiranih modelov za diarizacijo in transkripcijo slovenskega govora

Aljoša Golob, Matej Rajter, Jan Vališer, Nejc Fras and Jan Popič

https://doi.org/10.26493/scores26.03

Obdelava ogroženih jezikov z velikimi jezikovnimi modeli na osnovi jezikoslovnih virov

Enej Krabonja, Miha Pliberšek, Tjaša Lah, Oskar Forštnarič, Maks Marolt and Borko Boškovič

https://doi.org/10.26493/scores26.04

Geometric Characterization of Embedding Spaces Across Chunking Strategies and Embedding Models for Slovenian Text

Anja Jodl and Mitko Nikov

https://doi.org/10.26493/scores26.18

 DOI: https://doi.org/10.26493/scores26

 DOI: https://doi.org/10.26493/scores26.01

Heuristic Evaluation of Mobile Web E-Commerce Checkout: A Domain-Specific Binary Checklist and Scoring Method

Olha Tushych
tshch92@gmail.com
Odesa Polytechnic National University
Odesa, Ukraine

Alla Toropenko
toropenko.a.v@op.edu.ua
Odesa Polytechnic National University
Odesa, Ukraine

ABSTRACT

We present a heuristic checklist for the checkout of online stores on the mobile web: 58 binary criteria in seven categories, from the cart up to, but not including, the payment screen. Every criterion is based on HCI guidance, web-form research, WCAG 2.2, or research on dark patterns. General heuristics, such as Nielsen's, say nothing about typical mobile web checkout problems, such as small tap targets, input formats, or consent. We built the checklist from an expert audit of 100 Ukrainian online stores and applied it to the same stores. It gives a transparent overall score and seven category sub-scores that show clear differences between checkouts, before any usability test is run.

KEYWORDS

heuristic evaluation, mobile web checkout, e-commerce, web forms

1 INTRODUCTION

For an online store, the checkout is where intent becomes revenue, and where it is most easily lost. Known reasons for abandonment include total cost not shown upfront, required registration, and long or confusing forms [1, 2]. Usability testing finds such problems well, but it is resource-intensive. Heuristic evaluation is a faster and low-cost alternative, which only requires researchers to check an interface against a list of usability principles [3, 4].

However, Nielsen's classic heuristics are very general. They say nothing about the typical problems of mobile web checkout: small tap targets, phone fields without an input mask, validation that works only after the form is sent, or marketing consent that is checked by default. So practitioners have to use a variety of separate sources, such as industry guidelines, accessibility standards, and data-protection law. This makes consistent, repeatable evaluation difficult.

In this paper we offer an instrument for this domain. Our contributions are: (1) a checklist of 58 binary criteria in seven categories that covers usability, accessibility, and consent in mobile web checkout, where most criteria check only one condition and each criterion is based on a reliable source; (2) a transparent scoring procedure that anyone can apply and that turns the checklist into an overall percentage and a profile of seven categories; and (3) a first benchmark of Ukrainian online stores. We did not invent heuristic evaluation. We built our instrument with the first stages of an existing methodology (Section 3) and wrote each criterion as a yes/no check, so that the evaluator needs less interpretation.

2 RELATED WORK

Heuristic evaluation was formalized by Nielsen and Molich [3]. Nielsen's ten classic heuristics, still used today, were published in 1994 [5]. The method has known weaknesses: different evaluators find different problems; this is called the evaluator effect [6]. Also, inspection and usability testing find different kinds of problems [7].

General heuristics miss many problems of a specific domain. For this reason, researchers often build domain-specific heuristic sets [8], and Quiñones et al. [9] described a step-by-step methodology for this.

Several instruments target online stores. Bonastre and Granollers [10] give 64 e-commerce heuristics with a scoring scheme, but their set covers e-commerce websites as a whole: checkout is only one of six groups of heuristics, and there is nothing about accessibility. Swaid and Suid [11] propose heuristics for m-commerce, but for native apps, not the mobile web. The checklist by Hashim et al. [12] for mobile shopping apps is not checkout-specific either: it checks only one quality attribute, operability, with 14 questions, and gives no overall score. The work closest to our combination of usability and accessibility is by Pascual-Almenara and Granollers-Saltiveri [13]: thirty-two evaluators scored eight e-commerce sites with a general heuristic instrument of 60 questions and, separately, against WCAG 2.1, and the two results were averaged into one index. The authors used general heuristics on purpose, not e-commerce ones, checked accessibility only on the homepage, and note themselves that averaging two separate instruments counts some problems twice. Finally, the Baymard Institute has the most developed checkout instrument (110+ guidelines, a 7-point weighted benchmark) [2], but its full guidelines are available only through a paid subscription, and it uses weighted ratings, not simple binary criteria.

Three further lines of work inform our criteria. Checkout is largely a web form: Bargas-Avila et al. [14] derived 20 guidelines for web forms, and Seckler et al. [15] showed in an experiment that forms improved with them are completed faster and with fewer submission attempts. For consent, Gray et al. [16] give a taxonomy of dark patterns for UX practitioners, and Mathur et al. [17] crawled 11,000 shopping sites and found dark patterns on 1,254 of them, including pre-selected options and hidden additions (sneaking); our user-control category covers these consent problems. For operationalisation, tools such as the consistency checker SHERLOCK [18] and Baymard's AI-based audit UX-Ray [19] already turn heuristics into conditions that can be checked; our binary criteria follow this idea.

As far as we know, no existing instrument has all of the following: yes/no criteria focused on checkout; a transparent scoring procedure that anyone can apply; benchmarking across stores; and a clear focus on mobile web, accessibility, and consent together.

3 METHOD: BUILDING THE INSTRUMENT

We built the instrument from an expert audit of 100 Ukrainian online stores in December 2025. We found the stores with Google searches for self-chosen product categories, so this is a convenience sample, not a random one. For each store, the first author went from the cart through checkout on the mobile website as an ordinary user in Google Chrome on a physical iPhone 15 Pro Max, checked the pages in Google Chrome DevTools with iPhone 14 Pro Max emulation, and compared the screens with established guidance [2, 20, 21]. All stores were scored by one evaluator, the first author, a UX designer with ten years of experience; a paid research assistant collected screenshots but did not score any criteria.

Following Quiñones et al. [9], we turned two kinds of audit findings into criteria: recurring problems, and solutions seen in some stores that established guidance, such as Baymard [2], recommends as best practice. Each criterion is written in a positive way (met = good) and is based on at least one reliable source. Most criteria check one observable condition; a few combine closely related conditions (see Section 6). We split broad topics. For example, "accessibility" became separate criteria for contrast and tap-target size (WCAG SC 1.4.3/1.4.11, SC 2.5.8) and for text size (common type-scale conventions, because WCAG sets no minimum). The result is 58 criteria in seven categories. This follows the methodology's advice to keep the number of heuristics small and to add a checklist when more detail is needed [9]: the seven categories work as heuristics, and the 58 criteria as their checklist.

The derivation was both inductive and deductive: the audit showed which failures and best practices really appear in the stores, and we kept a candidate only if a reliable source supported it. We completed only the first stages of the methodology: exploring the domain and the literature, putting together the criteria, and describing each one. The validation stages (an independent expert review, controlled experiments against another instrument, and user testing) would still be needed. So the instrument is fully specified, but not yet formally validated.

The scope has three limits: only the mobile web checkout, not native apps; only accessibility criteria that can be checked in the rendered interface, with contrast measured by WCAG contrast-checking tools (e.g., Stark, Chrome DevTools), not a full WCAG audit or checks with assistive technology (screen readers, ARIA); and only the flow from the cart through checkout, without the payment screen.

Binary criteria leave less room for interpretation. But with one evaluator we could not measure agreement between evaluators [6].

4 THE INSTRUMENT

The seven categories group the criteria by principle, not by their place on the screen: (C1) user control and autonomy, (C2) transparency and predictability, (C3) logical structure and cognitive load, (C4) error prevention and feedback, (C5) accessibility (WCAG), (C6) consistency, conventions and affordances, and (C7) input efficiency and effort economy.

Table 1 lists all 58 criteria with a short source code for each. The criteria have their own numbers, separate from the categories (e.g., #31, not "C31", so they are not confused with C1–C7). The legend below the table links each source code to the references.

Table 1: Mobile web e-commerce checkout heuristic checklist (58 binary criteria). *Src* gives grounding source(s); [c.] = conditional (excluded from the denominator when N/A).

C1 — User control & autonomy	
1 Guest checkout available	B,N
2 Guest checkout ≥ as prominent as sign-in	B
3 Cart quantity editable	N
4 Quantity via +/− steppers	B
5 Marketing consent opt-in (unchecked)	GDPR, G18, M19
6 Terms/privacy consent opt-in	GDPR,M19
7 "No confirmation call" not pre-selected	G18,M19
8 Order does not force a manager call	G18
C2 — Transparency & predictability	
9 Item prices & payable total clear	N94
10 Currency labelled once, unambiguously	N94
11 Purpose of phone/e-mail explained	B
12 Unavailable pay/delivery hidden or explained	N94
13 Delivery cost shown	B
14 Delivery time shown	B
15 Free-delivery threshold shown [c.]	B
16 Next-step button labels state next step	Sm
C3 — Logical structure & cognitive load	
17 Natural name-field order	B
18 Logical field sequence	N,BA11
19 Forms >4 fields split into blocks	B
20 Law of proximity respected	N,Sm
21 Order: personal → delivery → payment	Sm
22 Carrier ordering matches input method [c.]	B,W3Ci18n
23 Address ordering matches method & country [c.]	W3Ci18n, GLAI, UPU, B
24 Single-column fields	Bmc,Sm
25 No distracting elements in checkout	Goog
26 Collapsed accordion reveals key content [c.]	B
C4 — Error prevention & feedback	
27 Required fields clearly marked	B,BA11
28 Optional fields clearly marked [c.]	B,Sil,BA11
29 Every field keeps a persistent label	Sil,N,Sm
30 Placeholder shows format, not duplicate label	N,Sil
31 Phone field with input mask [c.]	B
32 On-blur inline validation present	N,BA11,Sk14
33 Inline validation correct (no false errors)	N,B
34 Error fields visually distinct	N
35 Filled vs empty fields distinct	N94
36 No disabled/inactive buttons	Sil
37 Password reveal toggle [c.]	N
38 No forced password re-entry [c.]	N
C5 — Accessibility (WCAG; scoped)	
39 Contrast meets WCAG: text & non-text UI	WCAG,Sm,B
40 Adequate text size	Ap,Sm
41 Minimum tap-target size (SC 2.5.8)	WCAG,B
42 Key content not hidden by scroll	N94,WCAG
43 Content reflows to one column	WCAG
C6 — Consistency, conventions & affordances	
44 Consistent interface language	N94
45 System messages in interface language	N94
46 Visual consistency of fields/buttons	N94
47 Tidy alignment of fields and labels	B,N
48 Standard radio vs checkbox conventions	N
49 Dropdowns show an affordance indicator	N
50 Buttons/links distinct from plain text	N
51 Conventional social-login buttons [c.]	N
52 Primary controls use conventional terms	N94
C7 — Input efficiency & effort economy	
53 Browser autocomplete works, not misleading	Sm,N
54 City search surfaces regional centres [c.]	B
55 Branch searchable by its number [c.]	B
56 "Other recipient" option offered [c.]	B
57 Social login / SSO offered as an option [c.]	N,Sun13
58 Few or no optional fields	Sil,B,BA11

Src codes: B Baymard [2] · Bmc Baymard (multicolumn) [22] · N NN/g [23–28] · N94 Nielsen [4, 5] · WCAG W3C WCAG 2.2 [21] · W3Ci18n W3C i18n address formats [29] · GLAI Google libaddressinput [30] · UPU UPU S42 [31] · Ap Apple HIG [32] · Sm Holst, Smashing Mag. [33] · Sil A. Silver [34] · Goog Google Retail Playbook [20] · G18 Gray et al. 2018 [16] · M19 Mathur et al. 2019 [17] · BA11 Bargas-Avila et al. 2010 [14] · Sk14 Seckler et al. 2014 [15] · Sun13 Sun et al. 2013 [35] · GDPR GDPR Recital 32 [36].

Table 2: Category sub-scores and overall score (%) for an anonymised sample of six stores (two top, two median, two bottom) and the corpus median (n = 100). C1–C7 are the seven category sub-scores.

Store	C1	C2	C3	C4	C5	C6	C7	Overall
A	100	88	80	92	60	100	67	**86**
B	88	75	80	92	80	100	67	**84**
C	100	50	70	92	60	78	50	**74**
D	100	62	60	83	40	89	67	**74**
E	100	38	70	50	40	89	50	**64**
F	75	38	70	75	20	22	50	**53**
Median	88	50	74	83	40	89	50	**74**

5 SCORING AND BENCHMARKING

The checklist gives a numerical score of conformance with the heuristics. It is not a validated measure of usability (see Section 6). Each criterion gets 1 (met) or 0 (not met). If a criterion does not apply to a store (e.g., there is no courier delivery or no password field), it is not counted in that store's denominator. The overall score of a store is the number of met criteria divided by the number of applicable criteria, in percent. We also calculate seven category sub-scores in the same way. Together they form a diagnostic profile that shows where a checkout is strong or weak, not just one number.

We chose equal weights and a binary scale on purpose, instead of severity weights. We do not claim that all criteria have the same effect on users or on business. Severity ratings are subjective and differ between evaluators. If the main score depended on them, it would be harder to reproduce, and reproducibility is the main reason to use a checklist. With equal weights and yes/no answers, the score stays transparent and anyone can recompute it.

To show how the instrument works, we applied it to all 100 audited stores and calculated the overall and category scores for each store. Table 2 shows an anonymised sample (Store A–F) from the top, middle, and bottom of the distribution. Figure 1 shows the category profile of the corpus median next to the best and the worst store in the sample.

All 58 criteria were scored for all 100 stores, using the same December 2025 audit records. Conditional criteria (e.g., the order of carrier branches or of address fields) are marked as not applicable and are not counted when a store does not have that feature.

Across all stores (n = 100), overall scores go from 53% to 86% (median 74%, mean 74%, SD 5.5). So even the best stores in our sample fail about one criterion in seven. The weakest category is accessibility (C5, median 40%), followed by transparency and input efficiency (C2 and C7, both median 50%). The strongest are user control (C1) and consistency (C6), with medians of 88% and 89%.

6 DISCUSSION AND LIMITATIONS

The benchmark shows that the instrument reveals clear differences: scores differ by up to 33 percentage points, and stores with the same overall score can have very different profiles. Stores C and D in Table 2 both have 74% but differ by 20 points in accessibility and 17 in input efficiency. One total number would hide this difference. Accessibility is weak almost everywhere: 91 of 100 stores meet

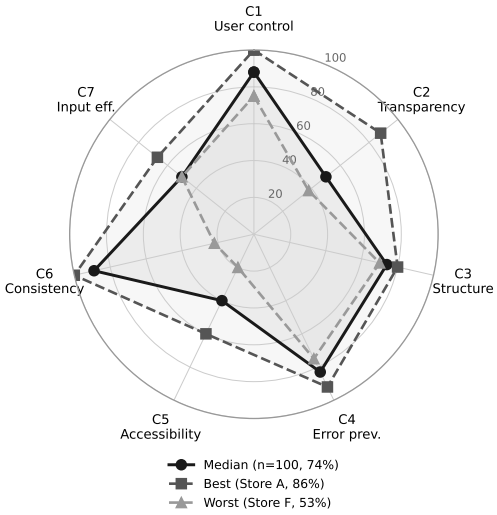


Figure 1: Category sub-score profile (C1–C7, %) for the corpus median (n = 100, 74% overall), the best-scoring store (Store A, 86% overall), and the worst-scoring store (Store F, 53% overall).

at most 60% of the accessibility criteria (C5), and none meets all of them. The accessibility median of 40% covers only the part of WCAG that can be checked in the rendered interface, so it is an optimistic upper limit.

The instrument is not yet validated with usability testing or conversion data. Because inspection and testing find different problems [7], it should support testing, not replace it. Equal weights are a simplification, not an empirical result; weights measured in usability tests, rather than expert severity ratings, could refine the score without losing reproducibility. A few criteria are based on best practice that depends on context (e.g., offering SSO). Some criteria (e.g., #9 price/total clarity, #39 WCAG contrast) combine related conditions that a future version should split.

A more serious limitation is that we built the criteria from the same 100-store audit that we then scored. So the benchmark describes the stores that shaped the instrument; it is not an independent measurement. Failures that were common in these stores were more likely to become criteria, so the checklist looks more complete on this sample than it may be, and failures absent from Ukrainian checkouts are probably missing. So the benchmark shows applicability, not external validity. A natural validation step would be to apply it to an independent sample, ideally outside Ukraine, scored by evaluators who did not write the criteria.

Criteria #22–#23 and #54–#56 follow Ukrainian delivery services, and the phone mask (#31) applies only to a verified local format; other markets are untested.

Accessibility is limited to criteria that can be checked in the rendered interface (Section 3). Checks that need assistive technology or code inspection (e.g., SC 2.1.1, 2.4.7, 4.1.2, 1.1.1) were not audited, so we cannot say whether the sites meet them. Finally, one evaluator, who also wrote the criteria, scored all stores. So the evaluator effect [6] and confirmation bias are not controlled. An external UX expert was consulted during criterion development,

which only partly reduces this risk. Measuring agreement between evaluators and an independent expert review of the criteria would address this.

7 CONCLUSIONS

We presented a heuristic checklist for the mobile web e-commerce checkout: 58 binary criteria in seven categories, most of which check one condition. The criteria are based on HCI, web-form, accessibility, and dark-pattern sources. We also presented a transparent scoring procedure that anyone can apply, and a first benchmark. Because the criteria are designed to limit interpretation and the score is transparent, teams can measure heuristic conformance before any usability test. The category profile shows which parts of the checkout need work first. Two questions remain open: whether different evaluators reach the same scores with the checklist, and whether the score relates to conversion.

ACKNOWLEDGMENTS

The authors thank Anna Krupina for paid research assistance with data collection for the audit, and Dmytro Yatsenko for expert consultations on checkout best practices.

REFERENCES

[1] Baymard Institute. 2026. Unexpected Costs Are the Leading Cause of Checkout Abandonment. https://baymard.com/premium/quant-insights/GC050. Baymard Premium. Accessed September 2026.

[2] Baymard Institute. 2026. Cart & Checkout Usability Research. https://baymard.com/research/checkout-usability. Cart abandonment statistics: https://baymard.com/lists/cart-abandonment-rate. Accessed September 2026.

[3] Jakob Nielsen and Rolf Molich. 1990. Heuristic Evaluation of User Interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '90)*. ACM, New York, NY, USA, 249–256. https://doi.org/10.1145/97243.97281

[4] Jakob Nielsen. 1994. Enhancing the Explanatory Power of Usability Heuristics. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '94)*. ACM, New York, NY, USA, 152–158. https://doi.org/10.1145/191666.191729

[5] Jakob Nielsen. 1994. Heuristic Evaluation. In *Usability Inspection Methods*, Jakob Nielsen and Robert L. Mack (Eds.). John Wiley & Sons, New York, NY, 25–62.

[6] Morten Hertzum and Niels Ebbe Jacobsen. 2003. The Evaluator Effect: A Chilling Fact About Usability Evaluation Methods. *International Journal of Human-Computer Interaction* 15, 1 (2003), 183–204. https://doi.org/10.1207/S15327590IJHC1501_14

[7] Robin Jeffries, James R. Miller, Cathleen Wharton, and Kathy M. Uyeda. 1991. User Interface Evaluation in the Real World: A Comparison of Four Techniques. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '91)*. ACM, New York, NY, USA, 119–124. https://doi.org/10.1145/108844.108862

[8] Daniela Quiñones and Cristian Rusu. 2017. How to Develop Usability Heuristics: A Systematic Literature Review. *Computer Standards & Interfaces* 53 (2017), 89–122. https://doi.org/10.1016/j.csi.2017.03.009

[9] Daniela Quiñones, Cristian Rusu, and Virginica Rusu. 2018. A Methodology to Develop Usability/User Experience Heuristics. *Computer Standards & Interfaces* 59 (2018), 109–129. https://doi.org/10.1016/j.csi.2018.03.002

[10] Laia Bonastre and Toni Granollers. 2014. A Set of Heuristics for User Experience Evaluation in E-commerce Websites. In *Proceedings of the 7th International Conference on Advances in Computer-Human Interactions (ACHI 2014)*. IARIA, 27–34. http://www.thinkmind.org/download.php?articleid=achi_2014_2_10_20126

[11] Samar I. Swaid and Taima Z. Suid. 2019. Usability Heuristics for M-Commerce Apps. In *Advances in Usability, User Experience and Assistive Technology*. Advances in Intelligent Systems and Computing, Vol. 794. Springer, Cham, 79–88. https://doi.org/10.1007/978-3-319-94947-5_8

[12] Nor Laily Hashim, Ihediohamma Raphael Uzoma, and Norhanisha Yusof. 2025. Mobile Shopping Apps: Operability Measures in Checklists of Heuristic Evaluation. In *Digital Innovation in Knowledge Management*. Information Systems Engineering and Management, Vol. 49. Springer Nature Switzerland, Cham, 218–228. https://doi.org/10.1007/978-3-031-91485-0_17

[13] Afra Pascual-Almenara and Toni Granollers-Saltiveri. 2021. Combining Two Inspection Methods: Usability Heuristic Evaluation and WCAG Guidelines to Assess e-Commerce Websites. In *Human-Computer Interaction (HCI-COLLAB 2021) (Communications in Computer and Information Science, Vol. 1478)*. Springer International Publishing, Cham, 1–16. https://doi.org/10.1007/978-3-030-92325-9_1

[14] J. A. Bargas-Avila, O. Brenzikofer, S. P. Roth, A. N. Tuch, S. Orsini, and K. Opwis. 2010. Simple but Crucial User Interfaces in the World Wide Web: Introducing 20 Guidelines for Usable Web Form Design. In *User Interfaces*, Rita Mátrai (Ed.). InTech. https://doi.org/10.5772/9500

[15] Mirjam Seckler, Silvia Heinz, Javier A. Bargas-Avila, Klaus Opwis, and Alexandre N. Tuch. 2014. Designing Usable Web Forms: Empirical Evaluation of Web Form Improvement Guidelines. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 1275–1284. https://doi.org/10.1145/2556288.2557265

[16] Colin M. Gray, Yubo Kou, Bryan Battles, Joseph Hoggatt, and Austin L. Toombs. 2018. The Dark (Patterns) Side of UX Design. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, Article 534, 14 pages. https://doi.org/10.1145/3173574.3174108

[17] Arunesh Mathur, Gunes Acar, Michael J. Friedman, Eli Lucherini, Jonathan Mayer, Marshini Chetty, and Arvind Narayanan. 2019. Dark Patterns at Scale: Findings from a Crawl of 11K Shopping Websites. *Proceedings of the ACM on Human-Computer Interaction* 3, CSCW, Article 81 (2019), 32 pages. https://doi.org/10.1145/3359183

[18] Rohit Mahajan and Ben Shneiderman. 1997. Visual and Textual Consistency Checking Tools for Graphical User Interfaces. *IEEE Transactions on Software Engineering* 23, 11 (1997), 722–735. https://doi.org/10.1109/32.637386

[19] Baymard Institute. 2025. UX-Ray. https://baymard.com/product/ux-ray. Accessed December 2025.

[20] Google. 2018. UX Playbook for Retail: Collection of Best Practices to Delight Your Users. https://services.google.com/fh/files/events/pdf_retail_ux_playbook.pdf. Accessed December 2025.

[21] W3C. 2024. Web Content Accessibility Guidelines (WCAG) 2.2. W3C Recommendation, https://www.w3.org/TR/WCAG22/. Accessed September 2026.

[22] Edward Scott. 2023. Form Field Usability: Avoid Extensive Multicolumn Layouts (16% Make This Form Usability Mistake). Baymard Institute, https://baymard.com/premium/research-articles/avoid-multi-column-forms. Accessed December 2025.

[23] Kathryn Whitenton. 2016. Website Forms Usability: Top 10 Recommendations. Nielsen Norman Group, https://www.nngroup.com/articles/web-form-design/. Accessed September 2026.

[24] Rachel Krause. 2019. 10 Design Guidelines for Reporting Errors in Forms. Nielsen Norman Group, https://www.nngroup.com/articles/errors-forms-design-guidelines/. Last reviewed 12 December 2024. Accessed September 2026.

[25] Anna Kaley. 2018. The Mobile Checkout Experience. Nielsen Norman Group, https://www.nngroup.com/articles/mobile-checkout-ux/. Accessed September 2026.

[26] Raluca Budiu. 2017. A Checklist for Registration and Login Forms on Mobile. Nielsen Norman Group, https://www.nngroup.com/articles/checklist-registration-login/. Accessed September 2026.

[27] Hoa Loranger. 2015. Beyond Blue Links: Making Clickable Elements Recognizable. Nielsen Norman Group, https://www.nngroup.com/articles/clickable-elements/. Accessed September 2026.

[28] Huei-Hsin Wang. 2026. Does Your Form Really Need a Dropdown List? Nielsen Norman Group, https://www.nngroup.com/articles/dropdown-list/. Accessed September 2026.

[29] W3C. 2026. Address Formats Around the World. https://www.w3.org/International/questions/qa-address-formats. Accessed September 2026.

[30] Google. 2026. libaddressinput — Per-Country Address Field Metadata. https://github.com/google/libaddressinput. Accessed September 2026.

[31] Universal Postal Union. 2026. S42: International Postal Address Components and Templates. https://www.upu.int/en/Postal-Solutions/Programmes-Services/Addressing-Solutions. Accessed September 2026.

[32] Apple. 2026. Human Interface Guidelines: Typography. https://developer.apple.com/design/human-interface-guidelines/typography. Accessed September 2026.

[33] Christian Holst. 2011. Fundamental Guidelines of E-Commerce Checkout Design. Smashing Magazine, https://www.smashingmagazine.com/2011/04/fundamental-guidelines-of-e-commerce-checkout-design/. Accessed September 2026.

[34] Adam Silver. 2018. *Form Design Patterns*. Smashing Media AG, Freiburg, Germany.

[35] San-Tsai Sun, Eric Pospisil, Ildar Muslukhov, Nuray Dindar, Kirstie Hawkey, and Konstantin Beznosov. 2013. Investigating Users' Perspectives of Web Single Sign-On: Conceptual Gaps and Acceptance Model. *ACM Transactions on Internet Technology* 13, 1, Article 2 (2013), 35 pages. https://doi.org/10.1145/2532639

[36] European Union. 2016. Regulation (EU) 2016/679 (General Data Protection Regulation), Recital 32. https://eur-lex.europa.eu/eli/reg/2016/679/oj. Cf. Law of Ukraine "On Personal Data Protection" No. 2297-VI (2010).

 DOI: https://doi.org/10.26493/scores26.03

Sestava specializiranih modelov za diarizacijo in transkripcijo slovenskega govora

Aljoša Golob
aljosa.golob1@student.um.si
Fakulteta za elektrotehniko računalništvo in informatiko,
Univerza v Mariboru
Koroška cesta 46
SI-2000 Maribor, Slovenija

Nejc Fras
nejc.fras@student.um.si
Fakulteta za elektrotehniko računalništvo in informatiko,
Univerza v Mariboru
Koroška cesta 46
SI-2000 Maribor, Slovenija

Matej Rajter
matej.rajter@student.um.si
Fakulteta za elektrotehniko računalništvo in informatiko,
Univerza v Mariboru
Koroška cesta 46
SI-2000 Maribor, Slovenija

Jan Vališer
jan.valiser@student.um.si
Fakulteta za elektrotehniko računalništvo in informatiko,
Univerza v Mariboru
Koroška cesta 46
SI-2000 Maribor, Slovenija

Jan Popič
jan.popic1@um.si
Fakulteta za elektrotehniko računalništvo in informatiko,
Univerza v Mariboru
Koroška cesta 46
SI-2000 Maribor, Slovenija

POVZETEK

V članku obravnavamo samodejno pretvorbo govorjenega posnetka v strukturiran besedilni zapis, v katerem so posamezni govorci jasno označeni. Predstavimo modularni cevovod, ki združuje razločevanje govorcev z modelom pyannote.audio in razpoznavanje govora z za slovenščino specializiranim modelom Whisper. Naš glavni doprinos je zasnova cevovoda, ki zaporedne segmente istega govorca najprej združi, nato pa vsak segment prepiše ločeno, namesto da bi celoten posnetek prepisal naenkrat in ga naknadno poravnal na diarizacijske segmente. Cevovod na slovenski govorni zbirki ARTUR-J neposredno primerjamo s sistemom WhisperX. Pri tem dosežemo nižjo stopnjo napake razločevanja govorcev in občutno krajši čas obdelave, in to ob uporabi bistveno manjšega modela za razpoznavanje govora, kar predlagani pristop uvršča med računsko manj zahtevne rešitve za slovenski govor.

KLJUČNE BESEDE

razločevanje govorcev, diarizacija, razpoznavanje govora

1 UVOD

Razločevanje govorcev je proces določanja, kdo govori in kdaj v zvočnem posnetku, ki pogovor razdeli na segmente in vsakemu dodeli oznako govorca [8, 11]. Uporaben je pri sestankih, intervjujih, medijskih vsebinah in podobnih aplikacijah.

Sodobni pristopi temeljijo na globokem učenju. Med vodilnimi odprtokodnimi orodji na tem področju je pyannote.audio [3, 6], ki v modularen cevovod združuje segmentacijo z nevronskimi mrežami ter vektorske predstavitve govorcev.

Celostni sistemi, kot je WhisperX [1], sicer združujejo razpoznavanje govora, časovno poravnavo in razločevanje govorcev, vendar se opirajo na velike večjezične modele in niso posebej prilagojeni slovenščini. Celovita rešitev, ki bi za slovenski govor hkrati razločila govorce in jih natančno prepisala z za slovenščino specializiranim modelom, zato predstavlja zanimiv raziskovalni izziv.

V tem delu predstavljamo modularni cevovod za samodejno generiranje strukturiranih prepisov slovenskega govora, ki združuje razločevanje govorcev z modelom pyannote.audio [4] in razpoznavanje govora s specializiranim slovenskim modelom Whisper [9, 10].

Naš prispevek obsega zasnovo cevovoda za združevanje diarizacijskih segmentov istega govorca in njihovo razpoznavanje po posameznih odsekih, neposredno primerjavo tako zasnovanega cevovoda s sistemom WhisperX na slovenski zbirki ARTUR-J ter razčlenitev napak in porabe virov, s katero pokažemo, iz česa izhajata prednost pri diarizaciji in zaostanek pri besedni napaki.

2 OPIS CEVOVODA

Slika 1 prikazuje celoten potek obdelave enega posnetka: od razločevanja govorcev in združevanja zaporednih segmentov istega govorca do izreza odsekov in njihove transkripcije.

2.1 Razločevanje govorcev

Prva faza cevovoda iz zvočnega posnetka pridobi časovno organizirane segmente govora in vsakemu pripiše oznako govorca. Pred obdelavo večkanalne posnetke pretvorimo v mono, tako da izračunamo povprečje vseh kanalov. Mono signal posredujemo modelu pyannote/speaker-diarization-3.1 [3, 5], ki segmentacijo izvede na kratkem drsečem oknu in za vsak segment izračuna vektorsko vložitev govorca; segmente nato glede na podobnost vložitev razvrsti v gruče in vsaki gruči pripiše oznako govorca.

Tako dobljene segmente dodatno obdelamo: zaporedne segmente istega govorca združimo, kadar je časovni razmik med njima manjši od 0,5 s, s čimer zmanjšamo razdrobljenost izpisa brez prekomerne izgube ločljivosti med govorci. Izhod te faze je seznam segmentov, pri katerih so podani začetni in končni čas ter oznaka govorca.

2.2 Razpoznavanje govora

Druga faza vsakemu segmentu iz prve faze pripiše ustrezno besedilo. Za vsak segment iz mono signala izrežemo pripadajoči zvočni odsek glede na njegov začetni in končni čas. Segmente, krajše od 0,5 s,

izpustimo iz razpoznavanja in jim pripišemo prazno besedilo, saj model na tako kratkih odsekih pogosto proizvede napačen izpis. Gre za preprosto hevristiko; podobne težave z zelo kratkimi odseki ima tudi WhisperX, natančnejša obravnava pa ostaja odprta za nadaljnje delo.

Preostale odseke posredujemo specializiranemu slovenskemu modelu Whisper [10], pri čemer vsak odsek razpoznamo neodvisno od drugih. Modelu eksplicitno določimo ciljni jezik in nalogo prepisa, kar prepreči samodejno zaznavanje jezika in s tem zmanjša tveganje za napačno določen jezik na kratkih ali akustično manj jasnih odsekih. Daljše odseke model med dekodiranjem razdeli na prekrivajoča se okna dolžine 20 s s prekrivanjem 2 s. Ker je samodejno razpoznavanje govora na šumnih ali nejasnih odsekih nagnjeno k ponavljanju besed in halucinacijam, dekodiranje dodatno omejimo: uporabimo kazen za ponavljanje (angl. *repetition penalty*) z vrednostjo 1,1, ki zniža verjetnost že izpisanih besed, ter prepoved ponavljanja enakih zaporedij (angl. *no-repeat n-gram*) dolžine treh besed.

Razpoznano besedilo pripnemo ustreznemu segmentu iz prve faze, s čimer dobimo končni strukturiran prepis. Čas, porabljen za razločevanje govorcev, in čas, porabljen za razpoznavanje besedila, merimo ločeno za vsak posnetek; njuna vsota predstavlja skupni čas obdelave posnetka, uporabljen pri izračunu faktorja realnega časa (RTF), opisanega v razdelku 3.2. Izhod cevovoda je strukturiran prepis v obliki:

```
SPEAKER_00 [0.50s-4.23s]: Besedilo prvega govorca.
SPEAKER_01 [4.80s-9.11s]: Besedilo drugega govorca.
```

3 EVALVACIJA

Uspešnost cevovoda ovrednotimo ločeno za razločevanje govorcev in za razpoznavanje besedila, rezultate pa neposredno primerjamo s sistemom WhisperX.

3.1 Podatkovni nabor

Cevovod smo ovrednotili na delu slovenske govorne zbirke ARTUR [12], ki je nastala v okviru projekta Razvoj slovenščine v digitalnem okolju (RSDO) in je dostopna v okviru repozitorija CLARIN.SI.

Za evalvacijo smo uporabili sklop javnega govora ARTUR-J, ki vsebuje avtentične posnetke javnih govornih dogodkov, kot so novinarske konference, okrogle mize, intervjuji, predavanja in spletni dogodki. Sklop obsega 208 ur posnetkov z 811 govorci, od tega je ročno prepisanih 62 ur s 100 posnetki. Ročne transkripcije so podane v pogovornem in standardiziranem zapisu, shranjene v formatu TRS [2]. Vsi posnetki so v formatu WAV, vzorčeni s 44,1 kHz, 16-bitni, mono.

Od 100 ročno prepisanih posnetkov smo zaradi omejenih računskih virov ovrednotili 33 posnetkov, saj je obdelava obeh sistemov terjala približno devet ur računanja na grafični kartici. Da se izognemo pristranski izbiri akustično lažjih posnetkov, smo jih uredili po njihovih identifikatorjih in zaporedoma obdelali prvih 33; tako izbrani podnabor obsega vseh pet vrst dogodkov, ki jih zbirka označuje z oznako teme, in posnetke z od 1 do 11 govorci. Skupaj zajema 18,2 ure zvoka, od tega 17,8 ure referenčnega govora in 147.064 referenčnih besed; dolžine posameznih posnetkov se gibljejo od 7 minut do 1 ure in 6 minut, povprečna dolžina pa znaša 33 minut. Za ocenjevanje razločevanja govorcev smo anotacije TRS pretvorili v format RTTM [7], standardizirane prepise pa uporabili kot referenco za izračun besedne stopnje napake.

3.2 Evalvacijske metrike

Za vrednotenje uspešnosti diarizacije smo uporabili standardne metrike, ki primerjajo referenčno anotacijo in napoved modela. Osrednja metrika je **stopnja napake diarizacije** (**DER**, angl. *Diarization Error Rate*), ki podaja delež napačno diariziranega govora glede na skupno trajanje referenčnega govora. Definirana je kot:

$$DER = \frac{T_{miss} + T_{fa} + T_{conf}}{T_{ref}} \tag{1}$$

kjer T_{miss} predstavlja trajanje zgrešenega govora, T_{fa} trajanje lažno zaznanega govora, T_{conf} trajanje govora, pripisanega napačnemu govorcu, in T_{ref} skupno trajanje referenčnega govora.

Komponenta **zgrešenega govora** (angl. *Miss*) predstavlja delež govornih segmentov, ki jih model ni zaznal, in je definirana kot:

$$Miss = \frac{T_{miss}}{T_{ref}} \tag{2}$$

Komponenta **lažnega alarma** (angl. *False Alarm*) predstavlja delež segmentov, ki jih je model napačno označil kot govor, čeprav v referenčni anotaciji niso govor, in je definirana kot:

$$FA = \frac{T_{fa}}{T_{ref}} \tag{3}$$

Komponenta **zamenjave govorca** (angl. *Confusion*) predstavlja delež govora, ki je bil pripisan napačnemu govorcu, in je definirana kot:

$$Conf = \frac{T_{conf}}{T_{ref}} \tag{4}$$

Kakovost razpoznavanja besedila merimo z **besedno stopnjo napake** (**WER**, angl. *Word Error Rate*), izračunano med prepisom cevovoda in referenčnim prepisom zbirke ARTUR-J. Definirana je kot:

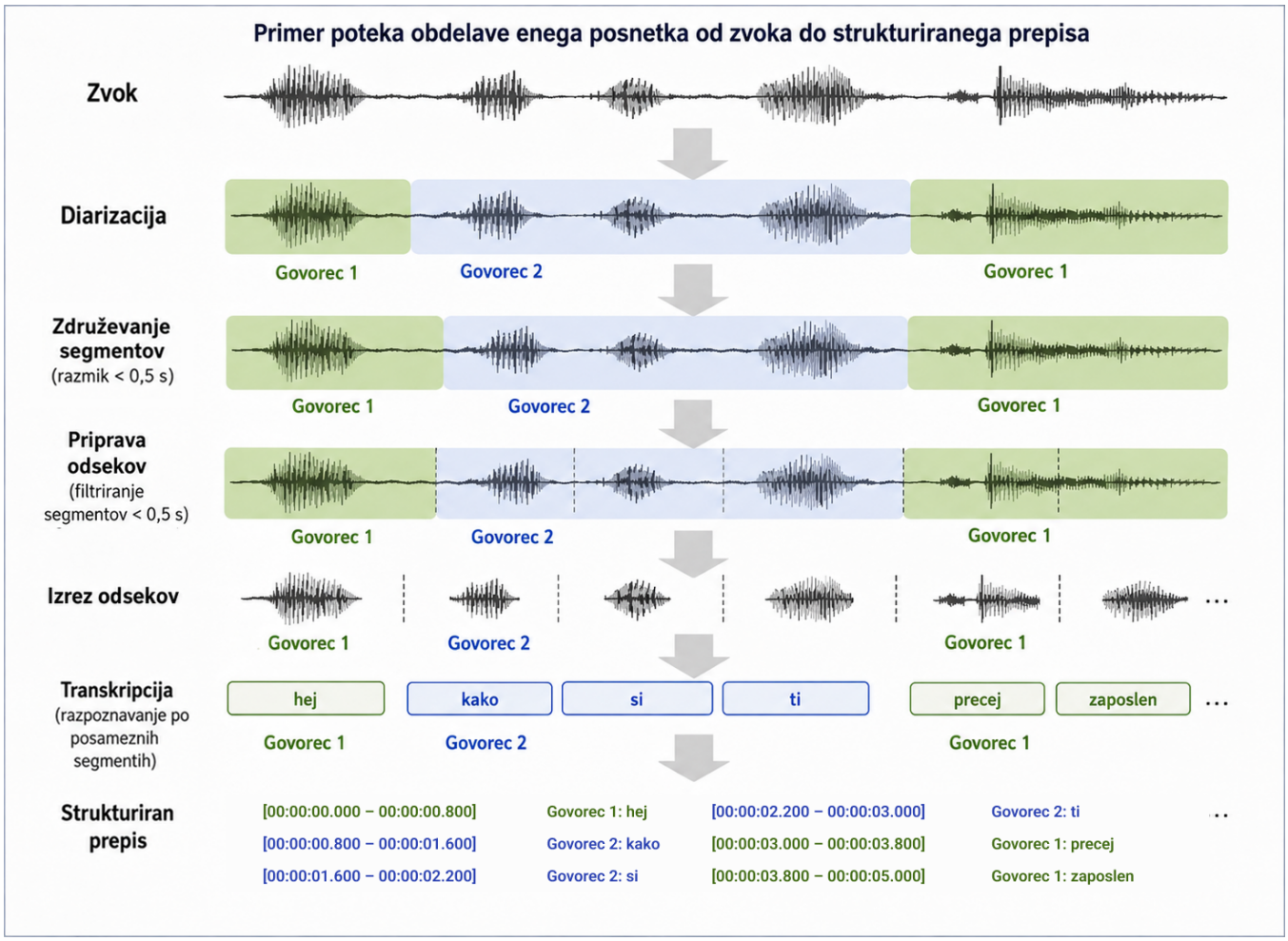
$$WER = \frac{S + D + I}{N} \tag{5}$$

kjer S predstavlja število zamenjanih besed, D število izpuščenih besed, I število vrinjenih besed in N skupno število besed v referenčnem prepisu. Pred primerjavo oba prepisa pretvorimo v male črke in odstranimo ločila. Poleg skupne vrednosti poročamo tudi deleže S, D in I glede na N, saj razmerje med njimi pokaže, ali napake izhajajo iz napačno razpoznanih ali iz izpuščenih besed.

Za primerjavo računske zahtevnosti dodatno poročamo **faktor realnega časa** (**RTF**, angl. *Real-Time Factor*), definiran kot razmerje med časom obdelave posnetka in dolžino posnetka. Vrednost RTF manjša od 1 pomeni, da sistem posnetek obdela hitreje, kot traja njegovo predvajanje. Vse metrike poročamo mikropovprečene, torej utežene s trajanjem referenčnega govora oziroma s številom referenčnih besed posameznega posnetka.

3.3 Okolje

Eksperiment smo izvedli v programskem jeziku Python 3.12.7. Kot strojno opremo smo uporabili grafično kartico NVIDIA GeForce RTX 5070 (12 GB VRAM) s tehnologijo CUDA (angl. Compute Unified Device Architecture) 12.8. Za razločevanje govorcev smo uporabili model pyannote/speaker-diarization-3.1 [5], kot izhodišče za primerjavo pa sistem WhisperX [1] z modelom Whisper large-v3.



Slika 1: Cevovod predlaganega pristopa

Oba sistema smo izvajali na isti strojni opremi in na istih posnetkih, zato so izmerjeni časi neposredno primerljivi.

4 REZULTATI

Glavni rezultat je primerjava našega cevovoda s sistemom WhisperX na 33 posnetkih slovenske zbirke ARTUR-J (tabela 1). Naš cevovod doseže opazno nižji DER, in sicer 6,14 % proti 8,87 % pri ničelni toleranci (angl. *collar*) oziroma 5,37 % proti 7,61 % pri toleranci 0,25 s, kar pomeni natančnejše razločevanje govorcev. WhisperX doseže nižji WER (20,03 % proti 30,22 %), kar je pričakovano, saj uporablja bistveno večji model large-v3. Naš cevovod hkrati posnetke obdela občutno hitreje: čas obdelave enega, v povprečju 1989 s (33 minut) dolgega posnetka znaša 283 s v primerjavi s 700 s pri sistemu WhisperX, kar ustreza približno 2,5-kratni pohitritvi (RTF 0,14 proti 0,35) in potrjuje manjšo računsko zahtevnost predlaganega cevovoda, vendar ob višji besedni napaki.

Razčlenitev DER pokaže, da pri obeh sistemih prevladuje zgrešeni govor: pri sistemu WhisperX pojasni 7,18 od 8,87 odstotne točke, pri našem cevovodu pa 4,42 od 6,14, medtem ko sta lažni alarm (0,84 %

Tabela 1: Primerjava cevovoda s sistemom WhisperX na slovenski zbirki ARTUR-J (33 posnetkov)

Sistem	DER	$DER_{0,25}$	WER	RTF
WhisperX (large-v3)	8,87	7,61	**20,03**	0,35
Naš cevovod (whisper-small-sl)	**6,14**	**5,37**	30,22	**0,14**

pri obeh) in zamenjava govorca (0,85 % oziroma 0,88 %) majhna in med sistemoma praktično enaka. Prednost našega cevovoda torej izhaja iz manj zgrešenega govora in ne iz zanesljivejšega razvrščanja govorcev v gruče.

Podobno razčlenimo besedno napako: zamenjave besed so pri našem cevovodu skoraj dvakrat pogostejše kot pri sistemu WhisperX (17,07 % proti 8,93 %), deleža izpuščenih (11,17 % proti 9,58 %) in vrinjenih besed (1,98 % proti 1,52 %) pa sta precej bližje, zato zamenjave pojasnijo približno štiri petine razlike v WER. Ker bi razpoznavanje po segmentih in izpuščanje najkrajših odsekov povečala predvsem izpuste, to kaže, da razliko v veliki meri pojasni

manjša zmogljivost razpoznavalnega modela. Med najpogostejšimi zamenjavami prevladujejo razlike med pogovorno in standardizirano obliko (*veliko* → *velik*, *imamo* → *mamo*) ter kratke slovnične besede (*s* → *z*, *kar* → *ker*), zato izmerjeni WER pri obeh sistemih precenjuje napake, ki bi motile bralca prepisa.

Čas obdelave je pri obeh sistemih linearno odvisen od dolžine posnetka (R^2 = 0,93 pri našem cevovodu in R^2 = 0,83 pri sistemu WhisperX), vrednosti RTF našega cevovoda pa so bistveno manj razpršene (od 0,11 do 0,23 s standardnim odklonom 0,025) kot pri sistemu WhisperX (od 0,25 do 0,54 z odklonom 0,084); naš cevovod je hitrejši na vseh 33 posnetkih. Po vrstah dogodkov doseže nižji DER pri vseh petih vrstah, pri čemer so najzahtevnejši intervjuji (9,91 % proti 13,26 %) s pogostim menjavanjem govorcev, najlažje pa novinarske konference z dolgimi monološkimi nastopi (4,55 % proti 7,43 %).

Pri razpoznavanju besedila je standardni model Whisper na zbirki ARTUR-J proizvajal slabše prepise, medtem ko je specializirani model whisper-small-slovenian dal bistveno boljše rezultate.

5 DISKUSIJA

Predlagani cevovod za razpoznavanje govora uporablja bistveno manjši model kot WhisperX, kar se odraža v višji besedni stopnji napake; to predstavlja kompromis med kakovostjo prepisa na eni strani ter hitrostjo in računsko nezahtevnostjo na drugi. Ker približno štiri petine razlike v WER izhajajo iz zamenjav besed in ne iz izpuščenega govora, pričakujemo, da bi večji ali dodatno prilagojen slovenski model to razliko zmanjšal ob ohranjeni hitrostni prednosti cevovoda. Koliko k zaostanku vseeno prispeva razpoznavanje po segmentih, bi dokončno pokazala ablacija z nespremenjeno diarizacijo; zanimiva bi bila tudi kombinacija diarizacije z modelom pyannote in razpoznavanja z modelom large-v3, ki bi predvidoma dosegla nizka DER in WER, vendar pri računski zahtevnosti, primerljivi s sistemom WhisperX.

Razčlenitev napake diarizacije kaže, da je prostor za izboljšave predvsem pri zaznavanju govora, saj zgrešeni govor pojasni skoraj tri četrtine DER. Nizka vrednost zamenjave govorca hkrati kaže, da je pripisovanje govorcev znotraj posnetka razmeroma robustno tudi ob spremembah glasu, na primer zaradi utrujenosti ali oddaljenosti od mikrofona; ker pa oznake govorcev niso vezane na identiteto, s to metriko preverjamo le doslednost znotraj enega posnetka in ne prepoznavanja istega govorca med posnetki.

Ker sta razločevanje govorcev in razpoznavanje govora ločeni in zamenljivi fazi, lahko posamezni model nadgradimo neodvisno od preostalega cevovoda. Vezava razpoznavanja neposredno na diarizacijske segmente hkrati odpravi potrebo po naknadni časovni poravnavi in s tem povezane napake, kar je skupaj z manjšo računsko zahtevnostjo pomembno zlasti za jezike z manj viri.

Glavni omejitvi raziskave sta ovrednotenje na eni sami slovenski zbirki ter omejeni računski viri, ki niso dopuščali obsežnejše optimizacije in primerjave z drugimi modeli; ker rezultati temeljijo na tretjini ročno prepisanega dela sklopa ARTUR-J, bi ovrednotenje na celotnem sklopu dalo zanesljivejše ocene. Za prihodnje delo predlagamo razširitev na večje in raznolike zbirke, podporo prekrivajočemu govoru, sprotno (angl. *online*) razločevanje govorcev, natančnejšo obravnavo zelo kratkih segmentov ter zgoraj opisane ablacije.

6 ZAKLJUČEK

Predstavili smo modularni cevovod za razločevanje govorcev in razpoznavanje slovenskega govora, ki iz zvočnega posnetka ustvari strukturiran prepis z jasno označenimi govorci. Na slovenski zbirki ARTUR-J cevovod doseže nižji DER kot sistem WhisperX ob približno 2,5-krat krajšem času obdelave, pri čemer za razpoznavanje uporablja bistveno manjši, za slovenščino specializiran model. Prednost pri diarizaciji je dosledna pri vseh vrstah posnetkov in izhaja iz manj zgrešenega govora, večino zaostanka pri besedni napaki pa pojasnijo zamenjave besed in ne razpoznavanje po posameznih segmentih.

Rezultati kažejo, da je tudi z računsko nezahtevnim, jezikovno specializiranim modelom mogoče doseči natančno razločevanje govorcev in uporabne prepise slovenskega govora, kar je obetavno zlasti za okolja z omejenimi računskimi viri. Modularna zasnova omogoča neodvisno nadgradnjo komponent, zato pričakujemo, da bo kakovost prepisov z napredkom modelov za slovenščino še naraščala.

LITERATURA

[1] Max Bain, Jaesung Huh, Tengda Han, in Andrew Zisserman. 2023. WhisperX: Time-Accurate Speech Transcription of Long-Form Audio. V *INTERSPEECH 2023*. 10.21437/Interspeech.2023-78

[2] Claude Barras, Edouard Geoffrois, Zhibiao Wu, in Mark Liberman. 2001. Transcriber: Development and use of a tool for assisting speech corpora production. *Speech Communication* 33, 1–2 (2001), 5–22. 10.1016/S0167-6393(00)00067-4

[3] Hervé Bredin. 2023. pyannote. audio 2.1 speaker diarization pipeline: principle, benchmark, and recipe. V *24th Interspeech Conference (INTERSPEECH 2023)*. ISCA, 1983–1987.

[4] Hervé Bredin. 2026. pyannote/speaker-diarization-3.1. https://github.com/pyannote/pyannote-audio. Dostopano: 2026.

[5] Hervé Bredin in Alexis Plaquet. 2023. pyannote/speaker-diarization-3.1. https://huggingface.co/pyannote/speaker-diarization-3.1. Hugging Face model repository.

[6] Herve Bredin, Ruiqing Yin, Juan Coria, Gregory Gelly, Pavel Korshunov, Diego Fustes, Hadrien Titeux, Wassim Bouaziz, in Marie-Philippe Gill. 2020. Pyannote.Audio: Neural Building Blocks for Speaker Diarization. V *Pyannote.Audio: Neural Building Blocks for Speaker Diarization*. 7124–7128. 10.1109/ICASSP40776.2020.9052974

[7] National Institute of Standards and Technology. 2009. *The 2009 (RT-09) Rich Transcription Meeting Recognition Evaluation Plan*. Technical Report. NIST. https://web.archive.org/web/20100606041157/http://www.itl.nist.gov/iad/mig/tests/rt/2009/docs/rt09-meeting-eval-plan-v2.pdf

[8] Douglas O'Shaughnessy. 2025. Speaker Diarization: A Review of Objectives and Methods. *Applied Sciences* 15, 4 (2025). 10.3390/app15042002

[9] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, in Ilya Sutskever. 2022. Robust Speech Recognition via Large-Scale Weak Supervision. arXiv:2212.04356 [eess.AS] https://arxiv.org/abs/2212.04356

[10] samolego. 2024. whisper-small-slovenian: Fine-tuned Whisper Small for Slovenian. https://huggingface.co/samolego/whisper-small-slovenian. Fine-tuned on the ARTUR 1.0 dataset. Hugging Face model repository.

[11] S.E. Tranter in D.A. Reynolds. 2006. An overview of automatic speaker diarization systems. *IEEE Transactions on Audio, Speech, and Language Processing* 14, 5 (2006), 1557–1565. 10.1109/TASL.2006.878256

[12] Darinka Verdonik, Andreja Bizjak, Andrej Žgank, Mitja Bernjak, Špela Antloga, Simona Majhenič, Peter Čakš, Matevž Pucer, Mitja Cvetko, Marijana Zelenik, Jani Pavlič, Simon Dobrišek, Janez Križaj, Gregor Strle, Marija Ivanovska, Klemen Grm, Marko Bajec, Iztok Lebar Bajec, Tjaša Jelovšek, Jure Lokovšek, Jure Longyka, Mitja Trojar, Jerneja Žganec Gros, Aleš Mihelič, Boštjan Vesnicer, Naum Dretnik, in David Bordon. 2023. ASR database ARTUR 1.0 (audio). http://hdl.handle.net/11356/1776 Slovenian language resource repository CLARIN.SI.

 DOI: https://doi.org/10.26493/scores26.04

Obdelava ogroženih jezikov z velikimi jezikovnimi modeli na osnovi jezikoslovnih virov

Enej Krabonja
enej.krabonja@student.um.si
Fakulteta za elektrotehniko, računalništvo in informatiko
Univerza v Mariboru
Koroška cesta 46
SI-2000 Maribor, Slovenija

Miha Pliberšek
miha.plibersek@student.um.si
Fakulteta za elektrotehniko, računalništvo in informatiko
Univerza v Mariboru
Koroška cesta 46
SI-2000 Maribor, Slovenija

Oskar Forštnarič
oskar.forstnaric@student.um.si
Fakulteta za elektrotehniko, računalništvo in informatiko
Univerza v Mariboru
Koroška cesta 46
SI-2000 Maribor, Slovenija

Tjaša Lah
tjasa.lah1@student.um.si
Fakulteta za elektrotehniko, računalništvo in informatiko
Univerza v Mariboru
Koroška cesta 46
SI-2000 Maribor, Slovenija

Maks Marolt
maks.marolt@student.um.si
Fakulteta za elektrotehniko, računalništvo in informatiko
Univerza v Mariboru
Koroška cesta 46
SI-2000 Maribor, Slovenija

Borko Boškovič
borko.boskovic@um.si
Fakulteta za elektrotehniko, računalništvo in informatiko
Univerza v Mariboru
Koroška cesta 46
SI-2000 Maribor, Slovenija

POVZETEK

Prispevek obravnava obdelavo ogroženih jezikov z velikimi jezikovnimi modeli na osnovi jezikoslovnih virov. Izhodišče je sistem LingoLLM, ki z vključitvijo morfološke analize, slovarja in slovnice omogoča prevajanje jezikov brez vzporednih učnih podatkov. Predlagamo tri optimizacije: uporabo lokalnega modela, dinamično pridobivanje slovničnih odlomkov in stiskanje pozivov z LLMLingua. Eksperiment na jeziku manchu in slovenščini pokaže, da dinamično pridobivanje slovničnih pravil bistveno zmanjša število vhodnih žetonov, hkrati pa ohrani ali izboljša kakovost prevodov.

KLJUČNE BESEDE

ogroženi jeziki, veliki jezikovni modeli, strojno prevajanje, dinamično pridobivanje z jezikovnimi viri, stiskanje pozivov

1 UVOD

V zadnjih letih so veliki jezikovni modeli (angl. *large language models*; LLM) bistveno izboljšali kakovost obdelave naravnega jezika, med drugim tudi na področju strojnega prevajanja. Modeli, kot je ChatGPT, pri prevajanju visoko zastopanih jezikov pogosto dosegajo primerljive ali celo boljše rezultate od specializiranih prevajalskih sistemov. Vendar pa ta prednost ni univerzalna. Avtorji v delu [5] ugotavljajo, da zmogljivost modela ChatGPT izrazito upade pri jezikih z omejenimi viri, saj so ti jeziki v učnih podatkih praviloma slabo zastopani ali povsem odsotni. To predstavlja pomembno omejitev pri uporabi sodobnih jezikovnih modelov za globalno večjezično podporo. Problem je posebej izrazit pri ogroženih jezikih, kjer pogosto ne obstajajo vzporedni korpusi, obsežni digitalni slovarji ali zadostne količine besedil za dodatno učenje modelov.

Po ocenah je danes ogroženih več kot 40 % svetovnih jezikov, pri čemer številni raziskovalci napovedujejo nadaljnje pospešeno izumiranje jezikovne raznolikosti. Zaradi tega postaja razvoj metod za digitalno podporo ogroženim jezikom pomemben tako z vidika računalništva kot jezikovne revitalizacije. Pomemben prispevek na tem področju predstavlja raziskava [12], v kateri avtorji analizirajo možnosti uporabe metod obdelave naravnega jezika pri revitalizaciji čerokejščine. Predlagajo več ključnih nalog, med njimi gradnjo govornih korpusov, razvoj morfoloških analizatorjev ter strojno prevajanje. Posebej poudarijo, da tehnološke rešitve same po sebi niso zadostne, temveč morajo biti tesno povezane z obstoječimi jezikovnimi viri in sodelovanjem z jezikovno skupnostjo.

Na tej osnovi delo [13] predstavi sistem LingoLLM, ki velikim jezikovnim modelom omogoča obdelavo ogroženih jezikov z uporabo eksplicitnih jezikoslovnih virov. Namesto dodatnega učenja model uporablja slovnice, slovarje in rezultate morfološke analize neposredno v pozivu. Poziv je celotno besedilo, ki ga sistem posreduje jezikovnemu modelu kot vhod. Tak pristop omogoča prevajanje tudi za jezike brez vzporednih učnih podatkov. Avtorji pokažejo, da lahko s tem pristopom bistveno izboljšajo kakovost prevajanja v primerjavi z neposredno uporabo velikega jezikovnega modela brez dodatnega konteksta.

Sorodna dela dodatno potrjujejo potencial uporabe strukturiranega jezikovnega znanja pri prevajanju jezikov z malo viri. Dela [1, 3] pokažejo, da je mogoče prevajanje izboljšati že z uporabo slovnic in slovarjev brez klasičnih paralelnih korpusov. Raziskave na področju večjezičnih LLM [4, 7, 8] medtem predlagajo dodatno učenje, medjezikovno sklepanje in optimizacijo vhodnih podatkov za izboljšanje večjezičnih sposobnosti modelov.

Kljub obetavnim rezultatom ima LingoLLM pomembno omejitev: celotna slovnica se pri vsakem klicu ponovno vključuje v poziv modela. To povečuje število vhodnih žetonov, podaljšuje čas obdelave in povečuje računske stroške, zlasti pri večjih slovničnih opisih. Ta omejitev je posebej problematična pri praktični uporabi lokalno gostovanih modelov, kjer so pomnilniške in računske omejitve izrazitejše. V tem prispevku predstavimo optimizacijo sistema LingoLLM s tremi izboljšavami: uporabo lokalnih jezikovnih modelov, dinamičnim pridobivanjem relevantnih slovničnih odlomkov z metodo RAG (angl. *Retrieval Augmented Generation*) in stiskanjem pozivov z orodjem LLMLingua. Cilj predlaganega pristopa je

zmanjšati računsko zahtevnost ob ohranitvi ali izboljšanju kakovosti prevajanja pri jezikih z omejenimi viri.

Prispevek je organiziran tako, da poglavje 2 predstavi predlagane optimizacije sistema LingoLLM, poglavje 3 eksperimentalno metodologijo in rezultate vrednotenja, poglavje 4 pa povzame glavne ugotovitve in možnosti nadaljnjega dela.

2 OPTIMIZACIJA SISTEMA LINGOLLM

Čeprav pristop v delu [13] dosega odlične rezultate pri prevajanju jezikov brez učnih podatkov, je izjemno potraten z vidika računalniških virov. Celotna slovnica se vsakič znova pošlje v LLM, kar povečuje zakasnitev, porabo pomnilnika in pri uporabi zunanjih storitev tudi stroške. Naša strategija za optimizacijo prvotnega pristopa naslavlja omenjene pomanjkljivosti s tremi glavnimi gradniki: lokalni LLM (Ollama) - nadomešča komercialne programske aplikacije za vmesnike (angl. *Application Programming Interface*; API) [9], RAG - dinamično pridobivanje le tistih delov slovnice, ki so relevantni za trenutni stavek [2] in kompresija pozivov (LLMLingua) - dodatno zmanjšanje števila žetonov z ohranjanjem ključnih informacij [6].

2.1 Lokalni jezikovni model

Namesto klica GPT-4 smo uporabili lokalno gostovan model z orodjem Ollama [9]. S tem smo odpravili stroške klica do storitve in zagotovili zasebnost podatkov. Ollama omogoča preprost zagon kvantiziranih modelov, kot so Mistral 7B, Llama 3 ali Mixtral, s prilagodljivim nadzorom porabe pomnilnika.

2.2 RAG - dinamično pridobivanje slovničnih pravil

Namesto celotne slovnice v poziv vključimo le tiste odlomke, ki so pomembni za trenutni vhodni stavek [2]. Postopek poteka v štirih korakih:

- priprava podatkov: slovnico razdelimo na smiselne dele (poglavja, odstavke),
- vložitve (angl. *embeddings*): za vsak del izračunamo vektorsko predstavitev in ga vstavimo v vektorsko bazo,
- pridobivanje: za vsak vhodni stavek izračunamo njegovo vektorsko predstavitev in v bazi poiščemo semantično najbližje odlomke; privzeto vrnemo tri najustreznejše,
- združevanje: pridobljene odlomke vstavimo v poziv pred stavkom in navodilom.

2.3 Kompresija pozivov

Poziv vključuje navodila za prevajanje, slovnične odlomke, pridobljene z metodo RAG, in vhodni stavek. Kljub metodi RAG lahko poziv še vedno vsebuje odvečne besede, primere ali ponavljanja. Zato število žetonov dodatno zmanjšamo z odstranitvijo manj pomembnih delov poziva, pri čemer ohranimo ključne informacije. Uporabimo knjižnico LLMLingua [6], ki z manjšim pomožnim modelom oceni pomembnost vsakega žetona in odstrani tiste z najnižjo oceno. Kompresijo izvedemo po sestavljanju končnega poziva, tako da pridobljeni odlomki in vhodni stavek skupaj tvorijo izhodišče za stiskanje. Rezultat je kompakten poziv, ki ga posredujemo jezikovnemu modelu.

3 EKSPERIMENT

3.1 Pridobivanje podatkov

Za primerljivost s sistemom iz [13] smo uporabili dva nabora podatkov: korpus jezika manchu in lasten korpus za slovenščino.

Za manchu smo prevzeli celoten nabor podatkov iz [13], ki vsebuje 101 par stavkov za prevajanje med manchu in angleščino, 15 stavkov za matematično sklepanje, demonstracijske primere za učenje iz nekaj primerov (*few-shot*), morfološki slovar in označeno slovnico. Gre za isti nabor, na katerem je izhodiščni sistem dosegel vrednost 0,5 metrike BLEU (angl. Bilingual Evaluation Understudy) pri neposrednem GPT-4 in BLEU 10 pri LingoLLM, zato omogoča neposredno primerjavo.

Slovenščino smo vključili kot srednje zastopan jezik s kompleksno morfologijo [11]. Uporabili smo referenčni nabor FLORES-200 [10], demonstracijske primere pa izbrali tako, da pokrivajo sklanjanje samostalnikov, spreganje glagolov, dvojino in naslonko *se*.

3.2 Priprava okolja

Izhodiščni sistem LingoLLM je temeljil na klicih vmesnika OpenAI in statičnem nalaganju celotnih slovničnih opisov v poziv modela. V predstavljeni raziskavi smo sistem razširili s štirimi komponentami, od katerih vsaka zahteva posebno nastavitev: z lokalnim jezikovnim modelom, modulom za dinamično pridobivanje odlomkov, modulom za kompresijo poziva in s podporo za slovenščino kot dodaten testni jezik.

3.3 Postopek

Eksperiment smo izvedli z lokalnimi modeli, s čimer smo odpravili odvisnost od komercialnih storitev. Preizkusili smo lokalno nameščene kvantizirane različice modelov Qwen2.5 in Qwen3.6. Uporabili smo model Qwen2.5 32B s 4-bitno in 5-bitno kvantizacijo ter model Qwen3.6 35B s 5-bitno kvantizacijo. Kvantizacija zmanjša porabo pomnilnika ob minimalnem vplivu na kakovost rezultatov. RAG je bil vključen v vse različice sistema. Namesto celotne slovnice sistem za vsak poziv poišče semantično najbližje odlomke z vektorskim iskanjem in jih vstavi v poziv modela. Na ta način se zmanjša skupna dolžina vhodnega besedila in izboljša ustreznost posredovanih informacij.
Stiskanje poziva je bilo vključeno v različice, ki vsebujejo daljše slovnične opise, saj pri kratkih pozivih stiskanje ni smiselno, ker bi lahko povzročilo izgubo informacij. Kompresija ohranja tiste dele vhodnega besedila, ki so semantično najbližje vhodnemu stavku.

Po zaključenem eksperimentu smo prevode ovrednotili z izračunom metrik BLEU in chrF za vsak stavek posebej ter s povprečjem po celotnem testnem nizu. Metrika chrF je bila glavni kriterij primerjave, ker je za morfološko bogate in jezike z omejenimi viri robustnejša od metrike BLEU.

3.4 Rezultati

Tabeli 1 in 2 prikazujeta primerjavo različnih konfiguracij. Pri manchu slovarski pristop dosledno preseže neposredno prevajanje: najboljša konfiguracija (Gemini z RAG) doseže vrednost BLEU 21,13 in chrF 51,57, najboljši neposredni sistem pa BLEU 12,20 in chrF 37,57.

Tabela 1: Pregled metrik – manchu.

Slovar	Konfiguracija	BLEU	chrF
Da	Gemini (slovar)	18,87	46,81
Da	Gemini + RAG (slovar + slovnica)	21,13	51,57
Da	Gemini + RAG + kompresija	16,82	44,31
Da	Qwen3.6 35B Q5_K_M + RAG + kompresija	15,16	39,63
Da	gpt-4-1106-preview [13]	11,10	38,89
Da	Qwen2.5 32B Q5_K_M	9,79	36,32
Da	Qwen2.5 32B Q4_K_M	7,59	34,88
Ne	Gemini	11,30	37,63
Ne	Gemini + kompresija	11,25	37,18
Ne	Gemini + RAG + kompresija	12,20	37,57
Ne	gpt-4-1106-preview [13]	3,57	23,03
Ne	Qwen2.5 32B Q5_K_M	0,17	14,97
Ne	Qwen2.5 32B Q4_K_M	0,16	14,63

Tabela 2: Pregled metrik – slovenščina.

Slovar	Konfiguracija	BLEU	chrF
Da	Gemini (slovar)	34,03	64,96
Da	Gemini + RAG (slovar + slovnica)	33,83	65,55
Da	Gemini + RAG + kompresija	34,94	64,26
Da	Qwen3.6 35B Q5_K_M + RAG + kompresija	33,80	63,27
Da	Qwen3.6 35B Q5_K_M	0,44	11,61
Ne	Gemini	37,51	66,25
Ne	Gemini + kompresija	37,34	65,65
Ne	Gemini + RAG + kompresija	37,09	65,50
Ne	Qwen3.6 35B Q5_K_M	31,68	63,58

RAG se pri modelu Gemini izkaže kot ključna optimizacija. V primerjavi z izhodiščnim GPT-4 [13] izboljša rezultat za BLEU 10,03 in chrF 12,68. Dodana kompresija nekoliko zmanjša kakovost prevoda, vendar zmanjša število vhodnih žetonov in računske stroške. Pri neposrednem prevajanju brez slovarja vpliv RAG ostane majhen.

Lokalni model Qwen2.5 32B brez slovarskega konteksta pri manchu praktično odpove (vrednost BLEU 0,16–0,17), s slovarjem pa doseže BLEU 7,59–9,79. Prehod s kvantizacije Q4 na Q5 prinese zmerno izboljšanje, večji napredek pa povečanje modela na Qwen3.6 35B, ki doseže BLEU 15,16 in chrF 39,63. Kljub temu ne doseže ravni Gemini z RAG, kar kaže na razliko med lokalnimi in komercialnimi modeli pri ogroženih jezikih.

Pri slovenščini so rezultati višji, saj je jezik dobro zastopan v učnih podatkih modelov. Neposredno prevajanje z modelom Gemini doseže vrednost BLEU 37,51 in preseže vse slovarske konfiguracije, kar kaže, da dodatni jezikovni viri pri poznanem jeziku ne prinesejo enake koristi kot pri manchu. Nasprotno lokalni model Qwen3.6 35B brez slovarskega konteksta doseže le BLEU 0,44, z RAG in stiskanjem pa BLEU 33,80, kar potrjuje pomen jezikoslovnega konteksta tudi pri jeziku s srednje veliko količino virov.

3.4.1 Učinkovitost obdelave. Število vhodnih in izhodnih žetonov je neposredna mera računske zahtevnosti posamezne konfiguracije. Jezikovni modeli porabijo pomnilnik in čas sorazmerno s številom žetonov v vhodu, število izhodnih žetonov pa odraža obsežnost generiranega odgovora. Tabela 3 prikazuje primerjavo štirih konfiguracij za slovenščino. Največji učinek ima RAG, ki zmanjša število vhodnih žetonov s 4536 na 487 (9,3-krat), hkrati pa izboljša kakovost prevoda na BLEU 27,96 in chrF 59,02. To kaže, da vključevanje celotne slovnice modelu ne koristi, saj predolg poziv poveča količino nerelevantnega konteksta..

Tabela 3: Primerjava učinkovitosti konfiguracij – slovenščina (gemini-3.1-flash-lite, slovarski pristop).

Konfiguracija	Povp. čas (s)	V. žetoni	I. žetoni	BLEU/chrF
Osnovna	96,78	4536	678	25,04/56,29
RAG	91,57	487	647	27,96/59,02
Stiskanje	150,91	1012	984	1,53/26,06
RAG + stiskanje	91,88	444	604	27,14/56,94

Samo stiskanje brez RAG se izkaže za neučinkovito: čas obdelave naraste za 56 %, kakovost pa pade na BLEU 1,53 in chrF 26,06, ker se pomembni deli slovnice odstranijo že pred izbiro relevantnega konteksta.

Kombinacija RAG in stiskanja doseže najmanj vhodnih (444) in izhodnih (604) žetonov ob kakovosti, primerljivi z uporabo samega RAG, zato predstavlja računsko najučinkovitejšo konfiguracijo.

Tabela 4: Primerjava učinkovitosti konfiguracij – manchu (gemini-3.1-flash-lite, slovarski pristop).

Konfiguracija	Povp. čas (s)	V. žetoni	I. žetoni	BLEU/chrF
Osnovna	38,73	1527	444	25,91/60,05
RAG	39,53	1491	436	21,50/53,22
Stiskanje	51,84	1186	475	21,28/49,58
RAG + stiskanje	51,87	1199	474	28,40/63,09

Rezultati za manchu (tabela 4) se razlikujejo zaradi bistveno krajše slovnice. RAG zmanjša število vhodnih žetonov le za 36, zato ne izboljša niti časa obdelave niti kakovosti prevoda. Samo stiskanje ponovno poslabša rezultate in podaljša čas izvajanja.

Najboljše rezultate pri manchu doseže šele kombinacija RAG in stiskanja (vrednost BLEU 28,40, chrF 63,09). Čeprav je nekoliko počasnejša, doseže najvišjo kakovost prevoda, saj učinkovito izbere in zgosti le relevantne dele slovnice. Številčne rezultate dopolnjuje kvalitativna primerjava prevodov.

3.4.2 Kvalitativna primerjava prevodov. Tabela 5 prikazuje primer prevoda iz jezika manchu v angleščino. Najboljši rezultat doseže Gemini z RAG in kompresijo ob uporabi slovarja, saj pravilno ohrani vzročno razmerje in vlogo osebka. Izhodiščni GPT-4 napačno interpretira subjekt in časovni okvir, lokalni model Qwen2.5 pa brez ustreznega slovarskega konteksta od referenčnega prevoda opazno odstopa.

4 ZAKLJUČEK

Cilj prispevka je bil optimizirati sistem LingoLLM za prevajanje ogroženih jezikov z zmanjšanjem računske zahtevnosti ob ohranitvi oz. izboljšanju kakovosti prevodov. V ta namen smo sistem nadgradili z lokalnim jezikovnim modelom, dinamičnim pridobivanjem slovničnih odlomkov z metodo RAG ter kompresijo pozivov.

Tabela 5: Primerjava prevodov izbranega stavka iz jezika manchu v angleščino.

Konfiguracija	Prevod
Vhod (manchu)	*emu guqu tutafi jimeofi bi elhexeme aliyakiyame yabure jakade tuttu jime goidaha*
Referenca	As a friend of mine was delayed and coming late, I was travelling slowly to wait for him, and because of this my coming has taken such a long time.
GPT-4 + slovar (izhodišče)	One companion, having remained behind, came to the place, leisurely lingering, going to the vicinity like that, came a long time ago.
Qwen2.5 32B Q5 + slovar	Long ago, my friend remained behind, and I waited leisurely for him to come, thus he came late.
Gemini + RAG + komp. (brez slovarja)	A friend having stopped and come, because I was idling and waiting, it has been a long time since they came like that.
Gemini + RAG + komp. (slovar)	A friend stayed behind and came to join me, but because I was walking slowly and lingering, it took a long time to arrive.

Eksperimente smo izvedli na jeziku manchu in slovenščini ter primerjali različne konfiguracije sistema. Rezultati kažejo, da RAG bistveno zmanjša število vhodnih žetonov (v izbranih konfiguracijah), pri slovenščini z 4536 na 487, hkrati pa izboljša kakovost prevoda (z vrednosti BLEU 25,04 na 27,96 in chrF 56,29 na 59,02). Pri manchu najboljše rezultate doseže kombinacija RAG in kompresije (BLEU 28,40, chrF 63,09). Rezultati kažejo tudi, da učinkovitost metode RAG ni odvisna zgolj od same uporabe pridobivanja konteksta, temveč predvsem od obsega in kompleksnosti razpoložljivih jezikoslovnih virov. Pri daljših slovničnih opisih RAG bistveno zmanjša število vhodnih žetonov in izboljša kakovost prevoda, medtem ko je pri krajših slovnicah njegov prispevek manj izrazit.

V nadaljnjem delu bi bilo smiselno pristop ovrednotiti še na večjem številu ogroženih jezikov ter raziskati naprednejše strategije pridobivanja in izbire jezikoslovnih virov. S tem bi sistem dodatno približali praktični uporabi pri digitalni podpori ogroženim jezikom.

Literatura

[1] Antonios Hus, Jonathan in Anastasopoulos. 2024. Back to School: Translation Using Grammar Books. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, Mohit in Chen Yun-Nung Al-Onaizan, Yaser in Bansal (Ed.). Association for Computational Linguistics, Miami, Florida, ZDA, 20207–20219. doi:10.18653/v1/2024.emnlp-main.1127

[2] Patrick Lewis in Ethan Perez in Aleksandra Piktus in Fabio Petroni in Vladimir Karpukhin in Naman Goyal in Heinrich Küttler in Mike Lewis in Wen-tau Yih in Tim Rocktäschel in Sebastian Riedel in Douwe Kiela. 2021. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. doi:10.48550/arXiv.2005.11401

[3] Garrett Tanzer in Mirac Suzgun in Eline Visser in Dan Jurafsky in Luke Melas-Kyriazi. 2024. A Benchmark for Learning to Translate a New Language from One Grammar Book. doi:10.48550/arXiv.2309.16575

[4] Wen Lai in Mohsen Mesgar in Alexander Fraser. 2024. LLMs Beyond English: Scaling the Multilingual Capability of LLMs with Cross-Lingual Feedback. doi:10.18653/v1/2024.findings-acl.488

[5] Nathaniel R. Robinson in Perez Ogayo in David R. Mortensen in Graham Neubig. 2023. ChatGPT MT: Competitive for High- (but not Low-) Resource Languages. doi:10.18653/v1/2023.wmt-1.40

[6] Zhuoshi Pan in Qianhui Wu in Huiqiang Jiang in Menglin Xia in Xufang Luo in Jue Zhang in Qingwei Lin in Victor Rühle in Yuqing Yang in Chin-Yew Lin in H. Vicky Zhao in Lili Qiu in Dongmei Zhang. 2024. LLMLingua-2: Data Distillation for Efficient and Faithful Task-Agnostic Prompt Compression. doi:10.18653/v1/2024.findings-acl.57

[7] Haoyang Huang in Tianyi Tang in Dongdong Zhang in Wayne Xin Zhao in Ting Song in Yan Xia in Furu Wei. 2023. Not All Languages Are Created Equal in LLMs: Improving Multilingual Capability by Cross-Lingual-Thought Prompting. doi:10.48550/arXiv.2305.07004

[8] Xiaoyu Chen Jiaxin Guo in Hao Yang in Zongyao Li Daimeng Wei in Hengchao Shang. 2024. A Novel Paradigm Boosting Translation Capabilities of Large Language Models. doi:10.48550/arXiv.2403.11430

[9] Ollama. n.d.. Ollama Documentation. https://docs.ollama.com/. Dostopano: 22. 3. 2026.

[10] Open Language Data. n.d.. Datasets at Hugging Face. https://huggingface.co/datasets/openlanguagedata/flores_plus Dostopano: 3. 5. 2026.

[11] Marko Ulčar, Matej in Robnik-Šikonja. 2023. Sequence-to-sequence pretraining for a less-resourced Slovenian language. *Frontiers in Artificial Intelligence* Izdaja 6 - 2023 (2023). doi:10.3389/frai.2023.932519

[12] Ben in Bansal Mohit Zhang, Shiyue in Frey. 2022. How can NLP Help Revitalize Endangered Languages? A Case Study and Roadmap for the Cherokee Language. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Preslav in Villavicencio Aline Muresan, Smaranda in Nakov (Ed.). Association for Computational Linguistics, Dublin, Irska, 1529–1541. doi:10.18653/v1/2022.acl-long.108

[13] Yee in Song Zhenqiao in He Taiqi in Wang William Yang in Li Lei Zhang, Kexun in Choi. 2024. Hire a Linguist!: Learning Endangered Languages in LLMs with In-Context Linguistic Descriptions. In *Findings of the Association for Computational Linguistics: ACL 2024*, Andre in Srikumar Vivek Ku, Lun-Wei in Martins (Ed.). Association for Computational Linguistics, Bangkok, Tajska, 15654–15669. doi:10.18653/v1/2024.findings-acl.925

 DOI: https://doi.org/10.26493/scores26.18

Geometric Characterization of Embedding Spaces Across Chunking Strategies and Embedding Models for Slovenian Text

Anja Jodl
anja.jodl@student.um.si
Faculty of Electrical Engineering and Computer Science,
University of Maribor
SI-2000 Maribor, Slovenia

Mitko Nikov
mitko.nikov@um.si
Faculty of Electrical Engineering and Computer Science,
University of Maribor
SI-2000 Maribor, Slovenia

ABSTRACT

This paper investigates whether geometric properties of embedding spaces can provide useful intrinsic indicators for evaluating text chunking strategies and embedding models without relying on downstream tasks. We evaluate several text chunking strategies with different overlap configurations using five embedding models on a large collection of historical Slovenian newspapers. Our results show that embedding-space geometry is strongly model-dependent, while chunking strategies substantially affect local continuity, cluster separation, and how well cluster centroids represent the original article. Token-based chunking with approximately 256 tokens and 25% overlap provided the most consistent balance between the evaluated geometric properties and computational requirements. The results also show that no individual geometric metric defines embedding-space quality and that complementary properties should therefore be interpreted jointly.

KEYWORDS

text embeddings, document chunking, embedding space geometry, semantic similarity, low-resource languages, retrieval, historical documents

1 INTRODUCTION

Dense retrieval systems convert sections of text, so-called chunks, into vectors using embedding models, and retrieve content whose vector representations are similar to the representation of the user's query. In retrieval-augmented generation (RAG), these retrieved text sections provide context for generating accurate answers. The choice of where to split the text, also known as the chunking strategy, determines which text is represented together, how much content is repeated across chunks, and how many representations must be stored and processed. Since the resulting chunks are encoded via an embedding model, chunking strategies and embedding models are interdependent and their selection should be considered together when designing a retrieval pipeline [11].

Chunking strategies can be evaluated through downstream retrieval or task performance. Although such evaluations are essential for establishing effectiveness in a particular application, their results reflect the combined effects of multiple components, including chunking, embedding models, retrieval algorithms, and reranking methods [14, 19]. Final performance scores alone therefore provide limited insight into how chunking and embedding choices affect the underlying vector representations.

When task-specific queries and relevance judgements are unavailable, an additional challenge is how to characterize these effects without first constructing a downstream evaluation dataset. Therefore, previous work has used geometric properties to analyze embedding quality [4, 17], while others applied a cluster-validity measure over sentence embeddings for reference-free text segmentation evaluation [8].

This research investigates how chunking strategies and embedding models affect local continuity between consecutive chunks, article-level separation, the global distribution of embedding variance, and agreement between chunk-based and unchunked article representations. The objective is to characterize these geometric properties jointly and examine where they provide consistent or conflicting indications about representation behavior. They are treated as intrinsic diagnostics rather than as validated predictors of downstream retrieval performance.

Hence, the contributions of this work are:

- A geometric evaluation protocol combining complementary measures of continuity, clustering, variance, and representation preservation.
- Operational definitions of semantic drop rate (SDR) and mean representation degradation (MRD).
- An empirical comparison of 17 chunking configurations and five embedding models, accompanied by publicly available code and experimental results [5].

2 METHODOLOGY

To evaluate how chunking strategies and embedding models affect the resulting embedding spaces, a multi-stage experimental pipeline was constructed. The pipeline consists of OCR digitization of archived newspapers; statistical analysis of the extracted articles; filtering and cleaning the articles based on their text quality and statistical outliers; chunking them using various strategies; generating embeddings with different models; and computing geometric metrics to evaluate the resulting embedding spaces.

2.1 Dataset

To compile the machine-readable dataset, 3430 scanned Slovenian-language newspapers published between 1900 and 2015 and provided by the Digital Library of Slovenia [10] were processed using MinerU [15] and the PaddleOCR engine [2]. The extracted content was stored in Markdown format.

Individual articles were isolated by splitting newspaper files according to Markdown header structures. In total, 167 977 articles were extracted, of which 90 307 were retained after cleaning and filtering.

The cleaning procedure removed non-text elements introduced during extraction, duplicate articles, and articles failing text-quality requirements based on length, letter and vowel ratios, and average

 DOI: https://doi.org/10.26493/scores26.18

word length. Additional filtering removed articles with excessive numeric or special characters and statistical outliers in text-structure measures using median absolute deviation-based bounds. The complete filtering configuration and implementation are publicly available in the accompanying repository [5].

Most articles are written in historical standard Slovenian, which differs from contemporary Slovenian in vocabulary, grammar, and orthography. This variation presents an additional challenge for embedding models typically trained on modern language corpora.

2.2 Chunking Strategies

Following extraction and cleaning, the articles were processed using the chunking strategies shown in Table 1. These included no-operation (no-op) chunking, in which the complete article was retained as a single text segment; paragraph chunking without overlap; sentence-based chunking using groups of N sentences with an overlap of M sentences (SN/M); token-based chunking using chunks of N tokens with an overlap of M tokens (TN/M); and recursive chunking using the LangChain recursive splitter [6], with chunks of N characters and an overlap of M characters (RN/M).

Table 1: Chunking strategy statistics.

Chunking	Chunks	C/A	Ch/C	Memory	Overlap
No-op	90,307	1.00	2471.59	213 MB	–
P	750,276	8.31	295.69	212 MB	0.0%
S1/0	2,641,788	29.25	83.21	210 MB	0.2%
S4/0	694,511	7.69	319.33	212 MB	0.0%
S4/1	880,608	9.75	326.66	274 MB	26.2%
T32/0	2,346,500	25.98	95.52	214 MB	0.0%
T128/0	619,968	6.87	360.65	213 MB	0.0%
T128/16	689,322	7.63	366.55	241 MB	14.9%
T256/0	330,820	3.66	675.61	213 MB	0.0%
T256/32	359,251	3.98	694.67	238 MB	17.8%
T256/64	397,002	4.40	712.65	270 MB	30.6%
T256/128	529,784	5.87	743.24	376 MB	53.4%
T256/192	930,188	10.30	764.82	678 MB	76.0%
R400/0	778,317	8.62	284.99	212 MB	0.0%
R400/64	804,092	8.90	298.73	229 MB	12.7%
R1000/0	317,105	3.51	700.96	212 MB	0.0%
R1000/100	319,127	3.53	710.34	216 MB	5.4%

C/A = mean chunks per article, Ch/C = mean characters per chunk, Overlap = mean overlap between consecutive chunks

Table 1 reports the number of generated chunks, mean chunks per article (C/A), mean characters per chunk (Ch/C), and the observed mean overlap between consecutive chunks. The latter may differ from the configured overlap due to article boundaries.

Figure 1 shows the resulting chunk-length distributions and indicates that paragraph- and sentence-based strategies produce more variable chunk sizes, whereas token-based strategies produce lengths more concentrated around their configured sizes.

T32/0 was included as an extreme small-chunk configuration to assess whether the evaluated geometric properties distinguish it from configurations using longer contexts. No-op chunking provides an unchunked reference representation of each article and is used as a baseline for metrics comparing chunk-based and complete-article representations.

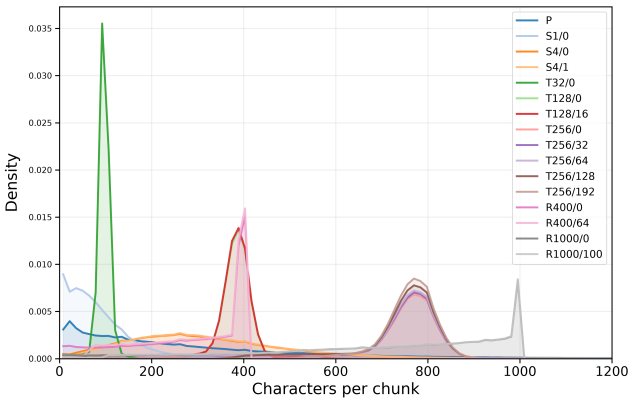


Figure 1: Probability densities of chunk character counts for different chunking strategies. The area under each distribution integrates to 1.

2.3 Embedding Models

Five open-source embedding models were selected to evaluate whether the effects of chunking remain consistent across different embedding spaces, as shown in Table 2: Microsoft's Multilingual E5 Large (E5) [16], Google's 300-million-parameter EmbeddingGemma [13], Alibaba Cloud's 600-million-parameter Qwen3 Embedding model [18], BAAI's BGE-M3 [1], and Novosel's SloVlo V1 [9], which is based on Microsoft's Multilingual E5 Base model. No instruction prefixes were used for any of the embedding models.

Table 2: Architectures of the evaluated embedding models.

Model	Released	Params.	Layers	Dim.
Multilingual E5 Large	Jun. 2023	600M	24	1024
SloVlo V1	Jun. 2024	300M	12	768
BGE-M3	Jan. 2024	560M	24	1024
Qwen3 Embedding 0.6B	Jun. 2025	600M	28	1024
EmbeddingGemma	Sep. 2025	300M	24	768

Params = parameters, Dim = embedding dimensions

2.4 Geometric Evaluation

After generating embeddings for each combination of chunking strategy and embedding model, the resulting embedding spaces were evaluated through complementary geometric properties.

A meaningful embedding space should preserve semantic relationships by placing related content close together while maintaining separation between distinct content. For chunked articles, this includes preserving continuity between consecutive chunks, separating content from different articles, avoiding excessive concentration along particular directions of the embedding space, and preserving article-level information when an article is represented through its chunks [4, 7, 12].

The evaluation therefore examined local semantic continuity, separation between articles, the global distribution of embeddings, and the preservation of article-level representations. All embeddings were normalized to unit length before evaluation to ensure consistent cosine similarity comparisons.

3 RESULTS

The complete pipeline was executed on a workstation equipped with an Intel i9-13900K CPU, an NVIDIA RTX 4090 with 24 GB of VRAM, and 64 GB of RAM. Article extraction and cleaning required approximately 30 minutes, application of all chunking strategies close to 4.5 hours, and embedding generation approximately 124 hours. Embedding generation therefore accounted for the majority of the computational cost. Results and compute metadata were stored in a SQLite database of approximately 320 GB. All code and complete results are available in the accompanying repository [5].

As an initial comparison, pairwise cosine similarity distributions were examined for random chunk pairs from different articles (Negative), chunk pairs from the same article (Positive), and consecutive chunks from the same article (Transition). Mean negative cosine similarity (MNCS), mean positive cosine similarity (MPCS), and mean transition cosine similarity (MTCS) are reported for selected configurations in Table 3. MNCS and MPCS describe article-level separability, while MTCS reflects local continuity.

Figure 2 illustrates the distributions for Gemma T256/64 and E5 T256/64. The distributions differ substantially between models, demonstrating that cosine similarity values should be interpreted relative to the geometry of each embedding space rather than compared directly across models.

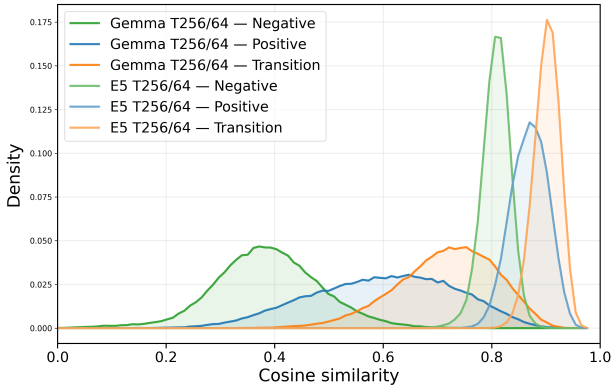


Figure 2: Pairwise cosine similarity distributions for Gemma T256/64 and E5 T256/64 embeddings, shown for negative, positive, and transition chunk pairs.

High MTCS alone does not necessarily indicate better representations, as local continuity should be interpreted relative to the similarity of unrelated content. The results show substantial differences between embedding models as the older E5 and SloVlo models exhibit higher cosine similarity values across all three metrics than BGE, Qwen3, and Gemma, indicating more compressed embedding spaces with less separation between articles.

To further evaluate the distinction between different types of chunk pairs, a Gaussian Naive Bayes (GNB) classifier was trained for each combination of embedding model and chunking strategy using 100 000 randomly sampled pairs per class. Cosine similarity was used as the only feature, with 80% of the pairs used for training and 20% for testing. Classification was performed between positive and negative pairs (N/P) and between negative and transition pairs (N/T).

Classification accuracy depends strongly on both the embedding model and chunking strategy. Accuracy was consistently higher for distinguishing negative from transition pairs, which is expected because consecutive chunks naturally share semantic and linguistic context. Although E5 and SloVlo exhibit higher overall cosine similarities, they achieved higher overall classification accuracy than BGE, Qwen3, and Gemma on this dataset.

To quantify local continuity, semantic drop rate (SDR) was defined as the proportion of consecutive chunk pairs whose cosine similarity falls below the mean negative cosine similarity (MNCS) of the corresponding model and chunking strategy. Lower SDR indicates better preservation of continuity.

Overlapping strategies substantially reduce SDR compared with non-overlapping strategies, as shared content increases the similarity between consecutive chunks. Even relatively small overlaps can produce considerable improvements, as observed for R1000/100.

The Davies-Bouldin index (DBI) [3] was computed to evaluate article-level cluster separation, with lower values indicating better separation. Token-based strategies, particularly T256 with overlap, consistently achieve lower DBI values than the other strategies, indicating more compact and better-separated article clusters.

Mean representation degradation (MRD) was used to measure how closely the centroid of an article's chunk embeddings corresponds to the embedding of the original article. MRD is defined as the mean cosine distance between the original article embedding and the corresponding chunk centroid. Because cosine similarity distributions differ between models, MRD is used only for comparing chunking strategies within the same model. In most cases, token-based strategies, particularly T256 with overlap, achieve lower MRD values than recursive chunking.

Finally, principal component analysis (PCA) was used to examine the global distribution of variance in the embedding spaces. The cumulative variance ratios of the first 1, 5, and 10 principal components are reported as PC1, PC5, and PC10 in Table 3. No clear patterns were observed among chunking strategies or consistent differences between models.

4 CONCLUSION

This research demonstrates that geometric properties of embedding spaces can provide diagnostic intrinsic indicators for evaluating chunking strategies and embedding models when task-specific queries and relevance judgements are unavailable.

The results show that embedding-space geometry is strongly model-dependent, while chunking strategies substantially affect local semantic continuity, article separation, and the preservation of article-level representations. No individual metric defines embedding-space quality, therefore the evaluated properties should be interpreted jointly.

Token-based chunking with moderate overlap provided the most consistent balance across the evaluated metrics. Even small overlaps produced substantial improvements, while overlap above 25% generally resulted in diminishing benefits relative to the increased number of chunks and computational requirements. Among the evaluated configurations, Qwen3 T256/64 and Gemma T256/64 provided a strong overall balance, with Gemma achieving comparable results using approximately half as many parameters.

Table 3: Selected set of the evaluation results combined by chunking strategy and embedding model.

Model	Chunking	MNCS ↓	MPCS↑	MTCS ↑	SDR ↓	DBI ↓	MRD ↓	PC1 ↓	PC5 ↓	PC10 ↓	GNB N/P ↑	GNB N/T ↑
E5	P	0.7770	0.8154	0.8286	10.44%	4.5847	0.0529	0.0450	0.1156	0.1668	72.60%	78.10%
	S1/0	0.7808	0.8002	0.8171	15.49%	8.1291	0.0745	0.0585	0.1262	0.1803	62.44%	72.16%
	S4/1	0.7854	0.8307	0.8811	0.62%	4.2962	0.0509	0.0299	0.1025	0.1548	76.57%	94.55%
	T32/0	0.8091	0.8286	0.8445	7.54%	7.7732	0.0796	0.0294	0.0993	0.1511	65.59%	77.09%
	T256/0	0.8041	0.8605	0.8729	3.83%	3.2015	0.0471	0.0330	0.1172	0.1751	82.87%	88.27%
	T256/64	0.8078	0.8692	0.9010	0.04%	3.1413	0.0455	0.0348	0.1146	0.1735	84.42%	97.01%
	R1000/100	0.7933	0.8461	0.8625	3.56%	3.2629	0.0485	0.0315	0.1093	0.1656	79.45%	86.95%
SloVlo	P	0.6668	0.7383	0.7553	9.70%	4.5019	0.0895	0.0501	0.1518	0.2203	74.18%	78.17%
	S1/0	0.7052	0.7381	0.7617	18.28%	7.9990	0.1315	0.0800	0.1849	0.2541	61.58%	69.62%
	S4/1	0.6617	0.7396	0.8218	0.52%	4.2035	0.0857	0.0349	0.1314	0.2024	75.64%	93.16%
	T32/0	0.7254	0.7576	0.7820	9.93%	7.7314	0.1525	0.0410	0.1395	0.2114	64.53%	74.10%
	T256/0	0.6718	0.7723	0.7970	2.68%	3.1663	0.0845	0.0424	0.1393	0.2113	82.04%	88.52%
	T256/64	0.6740	0.7851	0.8431	0.03%	3.0946	0.0826	0.0444	0.1386	0.2109	83.22%	96.19%
	R1000/100	0.6610	0.7593	0.7901	3.41%	3.1964	0.0823	0.0423	0.1362	0.2075	80.20%	87.53%
BGE	P	0.3461	0.4442	0.4916	11.16%	4.6069	0.1597	0.0289	0.1074	0.1655	72.46%	79.08%
	S1/0	0.3624	0.4168	0.4701	13.68%	8.0167	0.2127	0.0402	0.1191	0.1805	63.07%	74.33%
	S4/1	0.3667	0.4900	0.6548	0.52%	4.2885	0.1601	0.0306	0.1082	0.1675	75.77%	95.05%
	T32/0	0.3723	0.4342	0.4974	6.19%	7.5657	0.2277	0.0246	0.0989	0.1620	66.53%	79.94%
	T256/0	0.3854	0.5443	0.5950	2.35%	3.2251	0.1721	0.0315	0.1140	0.1738	82.80%	90.24%
	T256/64	0.3907	0.5698	0.6949	0.02%	3.1278	0.1684	0.0328	0.1169	0.1768	83.87%	98.00%
	R1000/100	0.3741	0.5255	0.5824	3.64%	3.2920	0.1700	0.0313	0.1142	0.1736	80.17%	88.37%
Qwen3	P	0.3179	0.4060	0.4572	15.19%	4.8259	0.1274	0.0398	0.1321	0.1973	66.83%	74.08%
	S1/0	0.2941	0.3409	0.3835	22.61%	8.9303	0.2005	0.0555	0.1361	0.1954	59.60%	67.30%
	S4/1	0.3514	0.4658	0.6171	2.28%	4.5609	0.1318	0.0367	0.1213	0.1863	71.58%	90.42%
	T32/0	0.3348	0.3864	0.4275	14.59%	8.5227	0.2099	0.0291	0.0986	0.1556	61.93%	70.36%
	T256/0	0.3718	0.5303	0.5805	4.50%	3.3189	0.1110	0.0382	0.1280	0.1951	78.92%	86.47%
	T256/64	0.3784	0.5559	0.6771	0.22%	3.2401	0.1135	0.0396	0.1310	0.1982	80.43%	95.33%
	R1000/100	0.3648	0.5161	0.5740	4.84%	3.3719	0.1067	0.0386	0.1305	0.1983	76.02%	85.10%
Gemma	P	0.3445	0.4783	0.5034	13.33%	4.7226	0.1206	0.0644	0.1675	0.2313	71.29%	74.41%
	S1/0	0.4068	0.4670	0.4936	23.88%	8.6883	0.2461	0.0716	0.1657	0.2277	60.00%	64.49%
	S4/1	0.3556	0.4970	0.6494	1.49%	4.3987	0.1179	0.0395	0.1331	0.2004	72.76%	90.69%
	T32/0	0.4473	0.5087	0.5410	18.07%	8.2395	0.2732	0.0359	0.1183	0.1820	61.36%	67.19%
	T256/0	0.3773	0.5691	0.6127	4.51%	3.2290	0.0768	0.0451	0.1495	0.2202	80.38%	86.30%
	T256/64	0.3862	0.5989	0.7155	0.09%	3.1156	0.0796	0.0477	0.1514	0.2204	81.53%	95.24%
	R1000/100	0.3794	0.5505	0.6033	5.41%	3.3065	0.0752	0.0429	0.1427	0.2096	76.20%	83.80%

5 FUTURE WORK

Future work should validate the relationship between the proposed geometric metrics and downstream retrieval performance on datasets containing queries and relevance judgements. Additionally, the geometric metrics developed in this research could potentially be used not only for evaluation but also to guide the chunking process itself. That includes investigating adaptive chunking algorithms that determine chunk boundaries by optimizing intrinsic geometric properties such as intra-chunk cohesion, anisotropy, or neighborhood preservation.

6 ACKNOWLEDGMENTS

The research paper was elaborated within the research project no. J7-60128: AID HCH - Breakthrough in humanities and cultural heritage with artificial intelligence, funded by the Slovenian Research and Innovation Agency (ARIS).

REFERENCES

[1] Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. M3-Embedding: Multi-Lingual, Multi-Functionality, Multi-Granularity Text Embeddings Through Self-Knowledge Distillation. arXiv:2402.03216 [cs.CL]

[2] Cheng Cui et al. 2025. PaddleOCR 3.0 Technical Report. arXiv:2507.05595 [cs.CV] https://arxiv.org/abs/2507.05595

[3] David L. Davies and Donald W. Bouldin. 1979. A Cluster Separation Measure. PAMI-1, 2 (1979), 224–227. https://doi.org/10.1109/TPAMI.1979.4766909

[4] Kawin Ethayarajh. 2019. *How Contextual Are Contextualized Word Representations? Comparing the Geometry of BERT, ELMo, and GPT-2 Embeddings.* https://doi.org/10.48550/arXiv.1909.00512 arXiv:1909.00512 [cs.CL]

[5] Anja Jodl and Mitko Nikov. 2026. Git Repository for Geometric Characterization of Embedding Spaces. https://github.com/GeMMA-FERI/embedding-space-characterization

[6] LangChain. 2026. Documentation for RecursiveCharacterTextSplitter. https://reference.langchain.com/. Accessed 26 August 2026.

[7] Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. 2020. *On the Sentence Embeddings from Pre-trained Language Models.* https://doi.org/10.48550/arXiv.2011.05864 arXiv:2011.05864 [cs.CL]

[8] Evan Lucas, Dylan Kangas, and Timothy Havens. 2023. A Reference-Free Segmentation Quality Index (SegReFree). In *Findings of the Association for Computational Linguistics: EMNLP 2023.* Association for Computational Linguistics, Singapore, 2957–2968. https://doi.org/10.18653/v1/2023.findings-emnlp.195

[9] Rok Novosel. 2024. SloVlo V1: Embeddings for the Slovenian Language. https://rokn.io/posts/slovlo-v1-embeddings-for-the-slovenian-language

[10] NUK. 2026. Digital Library of Slovenia. https://www.dlib.si/

[11] Renyi Qu, Ruixuan Tu, and Forrest Bao. 2025. Is Semantic Chunking Worth the Computational Cost?. In *Findings of the Association for Computational Linguistics: NAACL 2025.* Association for Computational Linguistics, 2155–2177. https://doi.org/10.18653/v1/2025.findings-naacl.114

[12] Sara Rajaee and Mohammad Taher Pilehvar. 2022. An Isotropy Analysis in the Multilingual BERT Embedding Space. In *Findings of the Association for Computational Linguistics: ACL 2022* (Dublin, Ireland, 2022-05). Association for Computational Linguistics, 1309–1316. https://doi.org/10.18653/v1/2022.findings-acl.103

[13] Henrique Schechter Vera, Sahil Dua, et al. 2025. EmbeddingGemma: Powerful and Lightweight Text Representations. (2025). https://arxiv.org/abs/2509.20354

[14] Muhammad Arslan Shaukat, Muntasir Adnan, and Carlos C. N. Kuhn. 2026. *A Systematic Investigation of Document Chunking Strategies and Embedding Sensitivity.* https://doi.org/10.48550/arXiv.2603.06976 arXiv:2603.06976 [cs.CL]

[15] Bin Wang et al. 2026. MinerU2.5-Pro: Pushing the Limits of Data-Centric Document Parsing at Scale. *arXiv preprint arXiv:2604.04771* (2026).

[16] Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024. Multilingual E5 Text Embeddings: A Technical Report. https://doi.org/10.48550/arXiv.2402.05672 arXiv:2402.05672 [cs.CL]

[17] Brendan Whitaker, Denis Newman-Griffis, Aparajita Haldar, Hakan Ferhatosmanoglu, and Eric Fosler-Lussier. 2019. Characterizing the Impact of Geometric Properties of Word Embeddings on Task Performance. In *Proceedings of the 3rd Workshop on Evaluating Vector Space Representations for NLP.* Association for Computational Linguistics, Minneapolis, USA, 8–17. https://doi.org/10.18653/v1/W19-2002

[18] Yanzhao Zhang et al. 2025. Qwen3 Embedding: Advancing Text Embedding and Reranking Through Foundation Models. *arXiv preprint arXiv:2506.05176* (2025).

[19] Yongjie Zhou, Shuai Wang, Bevan Koopman, and Guido Zuccon. 2026. *Beyond Chunk-Then-Embed: A Comprehensive Taxonomy and Evaluation of Document Chunking Strategies for Information Retrieval.* https://doi.org/10.48550/arXiv.2602.16974 arXiv:2602.16974 [cs.IR]

Algorithms, Graphs and Theoretical Computer Science

Resilient Incremental LL(1) Parsing Based on Li's Algorithm

Denis Balant

https://doi.org/10.26493/scores26.06

Marker Diagrams for String Rewriting Systems

Janez Ignacij Jereb

https://doi.org/10.26493/scores26.09

Encoding Graphs without Gradients

Matic Požar

https://doi.org/10.26493/scores26.11

Public Transport Anchored Flow Termini Coverage Model for Dock Placement in Bicycle Sharing Systems

Dmytro Tupkalenko, Daniil Baldouski and Miklós Krész

https://doi.org/10.26493/scores26.14

Two-Step Relaxation for All-Pairs Shortest Paths in Sparse Graphs

Dani Zugan

https://doi.org/10.26493/scores26.17

 DOI: https://doi.org/10.26493/scores26

 DOI: https://doi.org/10.26493/scores26.06

Resilient Incremental LL(1) Parsing Based on Li's Algorithm

Denis Balant
db59676@student.uni-lj.si
University of Ljubljana
Faculty of Computer and Information Science
SI-1000 Ljubljana, Slovenia

ABSTRACT

A language server must update its syntax tree after every edit, even when the document is incomplete. Li's incremental LL(1) algorithm reuses subtrees from the previous parse but assumes valid input and stops at the first syntax error. We correct inconsistencies in the original presentation of the algorithm and state the conditions under which a subtree may be reused. We then add simple table-driven error recovery, extending subtree reuse to invalid input. On synthetic Prev programs, editing a single term takes 29 µs on average at 10 000 source lines, compared to 4 400 µs for a full reparse.

KEYWORDS

incremental parsing, LL(1) parsing, error recovery, resilient parsing, language server

1 INTRODUCTION

A language server implementing the Language Server Protocol (LSP) [10] must keep its syntax tree up to date as the document changes. A straightforward approach is to relex and reparse the whole document after every edit, but its cost grows with document size even when the edit changes only a few characters.

An *incremental parser* instead avoids reanalyzing unchanged parts of the program by reusing the results of the previous analysis [11], so the cost of processing an edit depends primarily on the size and syntactic impact of the change.

We extend Li's incremental LL(1) algorithm [9] with table-driven error recovery and apply it to Prev, a small statically typed imperative language used in a compilers course at the University of Ljubljana, Faculty of Computer and Information Science.

2 INCREMENTAL LEXING AND PARSING

2.1 Document edit model

Let T be the set of lexical symbols, and let $xyz \in T^*$ be the original token string. We model a document edit as replacing the substring y with $y' \in T^*$, i.e.,

$$xyz\$ \rightsquigarrow xy'z\$,$$

where \$ marks the end of input. An insertion is the special case $y = \varepsilon$, and a deletion $y' = \varepsilon$, where ε represents an empty string.

The goal of incremental parsing is to produce the same result as parsing the full string $xy'z\$$, while reusing as much structure belonging to the unchanged substrings x and z as possible.

2.2 Extended LL parse table

Let $G = \langle N, T, P, S\rangle$ be a reduced \$-augmented LL(1) grammar, where N is the set of nonterminals, P is the set of productions, and S is the start symbol whose sole production has the form $S \to \omega\$$ for some $\omega \in (N \cup T)^*$.

Li's algorithm [9] extends table-driven LL(1) parsing by allowing the parser input to contain both lexer tokens and previously constructed subtrees. The nonterminal at the root of such a subtree serves as its lookahead symbol. The parse table therefore requires one additional column for each nonterminal of the grammar.

To construct additional entries, Li generalizes the conventional FIRST and FOLLOW sets [1] to include nonterminal symbols. For $A \in N$ and $\alpha \in (N \cup T)^*$, the extended sets are defined as

$$\text{FIRST'}(\alpha) = \{X \in (N \cup T) \mid \alpha \Rightarrow^* X\beta\} \cup \{\varepsilon \mid \alpha \Rightarrow^* \varepsilon\}, \quad (1)$$

$$\text{FOLLOW'}(A) = \{X \in (N \cup T) \mid S \Rightarrow^* \gamma A X \beta\}, \quad (2)$$

where $\beta, \gamma \in (N \cup T)^*$ and $\Rightarrow^*$ denotes a derivation in zero or more steps.

Note that if we remove their nonterminal elements, they coincide with the usual FIRST and FOLLOW sets.

Remark. *The original presentation of these definitions contains several inconsistencies, which we correct here:*

1. FOLLOW'(A) *is defined to contain* ε *and not* \$ *even though* \$ *is used in the paper example. This special case is avoided with the* \$*-augmented grammar.*
2. *Since* $\Rightarrow^*$ *is reflexive,* (1) *implies that* $X \in$ FIRST'(X) *for every* $X \in (N \cup T)$ *while the sets computed in Li's example omit this.*
3. FOLLOW'(E) = FOLLOW'(E') = {), \$}, *not* {(, id} *as listed.*

The extended sets can be computed using the usual fixed-point procedure [3, Algorithm 3.13], with FIRST'(X) initialized to $\{X\}$ for every symbol $X \in (N \cup T)$. The extended parse table M' is then constructed using the standard algorithm [1, Algorithm 4.31], with FIRST' and FOLLOW' in place of FIRST and FOLLOW.

We next show that any conflicts in the extended table occur only in entries that the parser never consults.

Lemma 2.1. *Let* $A, X \in N$, *and suppose* $X \Rightarrow^* a\delta$ *for some* $a \in T$. *Every production entered in* $M'[A, X]$ *is also entered in* $M'[A, a]$.

Proof. By construction of the table, a production $A \to \alpha$ is entered in $M'[A, X]$ only in one of the following two cases:

1. If $X \in$ FIRST'(α), then
$$\alpha \Rightarrow^* X\beta \Rightarrow^* a\delta\beta$$
for some β, hence $a \in$ FIRST(α).
2. If $X \in$ FOLLOW'(A) and $\alpha \Rightarrow^* \varepsilon$, then
$$S \Rightarrow^* \gamma A X \beta \Rightarrow^* \gamma A a \delta \beta$$
for some γ, β, hence $a \in$ FOLLOW(A).

In either case, $A \to \alpha$ is also entered in $M'[A, a]$. □

Lemma 2.2. *Let G be an LL(1) grammar. Its extended table contains no conflicts in any terminal column or nonterminal column $X \in N$ for which $X \Rightarrow^* a\delta$ for some $a \in T$ and $\delta \in (N \cup T)^*$.*

Proof. Since the extended sets match the conventional ones on terminals, the terminal columns are identical to those in the ordinary LL(1) table and are therefore without conflicts.

For any $A \in N$ and any $X \in N$ with $X \Rightarrow^* a\delta$, two distinct productions in $M'[A, X]$ would also, by Lemma 2.1, both occupy $M'[A, a]$, contradicting the assumption that G is LL(1). □

Example 2.3. The condition $X \Rightarrow^* a\delta$ in Lemma 2.2 is necessary. Consider the grammar with productions

$$A \to Xa \mid Xb \text{ and } X \to \varepsilon.$$

The grammar is LL(1), as $\text{FIRST}(Xa)$ and $\text{FIRST}(Xb)$ are disjoint. But $X \in \big(\text{FIRST'}(Xa) \cap \text{FIRST'}(Xb)\big)$, which means that the extended entry $M'[A, X]$ contains a conflict.

This conflict is harmless since subtrees with empty yield are discarded before they can serve as lookahead (Example 2.6).

2.3 Subtree reuse

The parser keeps two stacks: a *parse stack* of predicted symbols (Li's *P-stack*) and an *input stack* holding the remaining input – new tokens and previously built subtrees (Li's *I-stack*). The I-stack is needed to support the undo operation described below. Its top element serves as the lookahead symbol.

We now present the main subtree reuse mechanism. It is based on preserving the *yield* of a subtree, i.e. the string of its leaves (tokens), read from left to right.

Lemma 2.4. *Let W be a previously constructed subtree with nonempty yield u. If u and the token following it remain unchanged, reusing W results in the same parser configuration and parse tree structure as parsing u.*

Proof. We prove this by induction on the height of W. Let X be the root symbol of W.

If $X \in T$, W is a leaf and u is a single token, so the claim holds.

Otherwise $X \in N$. Let a be the first token of u. Since u is the yield of W, $X \Rightarrow^* a\delta$ for some δ.

(1) If X is at the top of the parse stack, the incremental run *reattaches* W (the root symbols match). A batch parse reconstructs the same subtree from u since each step of an LL(1) parser depends only on symbols pushed after expanding X and on unchanged lookahead (tokens of u and the first token following them).
(2) If some other $A \in N$ is at the top of the parse stack and $M'[A, X]$ contains a production p, then p is unique by Lemma 2.2 and also belongs to $M'[A, a]$ by Lemma 2.1 (since $X \Rightarrow^* a\delta$). Thus, both runs predict the same production p.
(3) In all remaining cases (terminal on top of the stack or an empty entry), the incremental parser *undoes* W, replacing it by its children with nonempty yield. Their yields still concatenate to u and each is followed by the same token as before, so by the induction hypothesis (children have lower height), reading them has the same effect as reading u.

Thus, in all cases, after reusing W, the parser reaches the same configuration as it does after reading its yield u. □

2.4 Divide mechanism

Since all parsing decisions in an LL(1) parser up to matching the last terminal of x depend only on unchanged lookahead tokens, the old and new runs share the same leftmost derivation

$$S \Rightarrow^*_{\text{lm}} x\alpha,$$

where α is the parse stack content after x has been consumed. Reparsing can therefore resume with α on the stack and $y'z\$$ on the input.

This configuration can be reconstructed from the old tree. Let the *frontier* of a partial parse tree be the sequence of its (nonterminal or terminal) leaves, read from left to right. At each step, it represents the current sentential form: the matched terminal prefix followed by the parse stack content [1]. Hence, immediately after x has been matched, the frontier is $x\alpha$.

Let the *spine* of a leaf t be the path from the root to t. Since predictive parsing always expands the leftmost unexpanded leaf, the roots of the right siblings of nodes on the spine of the last terminal of x form the unmatched part of the frontier and thus the parse stack content α.

Cutting the tree along this spine therefore restores the partial tree for $S \Rightarrow^*_{\text{lm}} x\alpha$ and detaches the subtrees previously used in the derivation $\alpha \Rightarrow^*_{\text{lm}} yz\$$.

Following Li [9], we refer to this splitting procedure as the *divide operation*. As shown in Figure 1, it is applied twice:

(1) along the spine of the last terminal of x: the pruned tree is kept as the starting point (Li's *X-partial-tree*) and pending symbols α become the initial parse stack (Li's *X-sequence*),
(2) at the last terminal of xy, collecting the detached subtrees with nonempty yield (Example 2.6) whose concatenated yield is z (Li's *Z-subtrees*).

If $x = \varepsilon$, there is no spine to cut, so divide restores the initial parser configuration – an empty root with the start symbol on the stack. If $z = \varepsilon$, there are no Z-subtrees, so the new input is y' alone. With both empty, the incremental parse reduces to a batch parse of $y'\$$.

Subtrees lying between the two cuts derive y and are discarded. The new input consists of y' followed by the Z-subtrees, and therefore has yield $y'z\$$. Parsing then resumes from this recovered configuration: y' is parsed as normal, while Z-subtrees are either reattached or undone as described in Lemma 2.4.

Theorem 2.5. *For every syntactically valid input $xy'z\$$, the incremental parser produces the same parse tree as a batch parse of $xy'z\$$.*

Proof. The first divide restores the configuration $x\alpha$, matching a batch parse of $xy'z\$$ after consuming x. The reconstructed input has yield $y'z\$$, and each Z-subtree satisfies the conditions of Lemma 2.4: it was built by the parser, has a nonempty yield, and is followed within z by unchanged tokens, the last by \$.

Processing each Z-subtree thus has the same effect as processing its yield, so both parses produce the same parse tree structure. □

 DOI: https://doi.org/10.26493/scores26.06

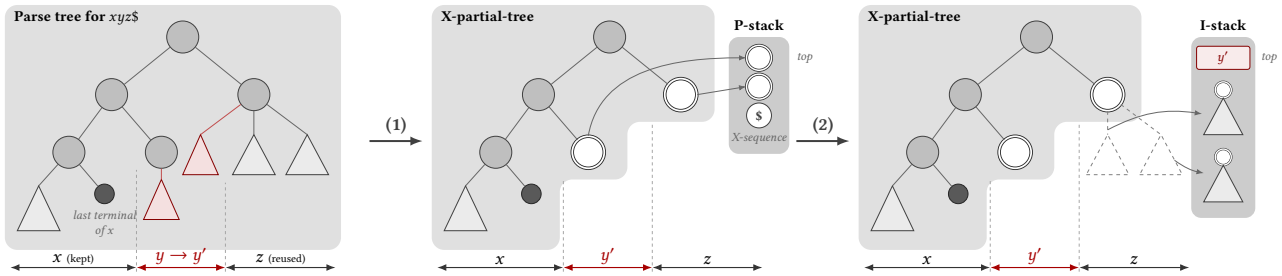


Figure 1: The two divide operations. Cut (1) at the last terminal of x yields the X-partial-tree and the X-sequence; cut (2) at the last terminal of xy yields the Z-subtrees.

Example 2.6. Discarding subtrees with an empty yield is necessary for correctness. Consider the LL(1) grammar

$$S \to aAB \mid bBA, \qquad A \to a \mid \varepsilon, \qquad B \to b \mid \varepsilon$$

and the edit $ab \to bb$, meaning $z = b$. Parsing ab used $S \to aAB$, so divide detaches the subtrees $A(\varepsilon)$ and $B(b)$, where $N(\ldots)$ denotes a subtree with root N.

With $A(\varepsilon)$ discarded, the parser predicts $S \to bBA$, matches b, reattaches $B(b)$, and accepts. However, with $A(\varepsilon)$ retained, the empty subtree $A(\varepsilon)$ is used as lookahead, and the reparse fails on a valid input:

P-stack	I-stack	action
$S\ \$$	$b\ A(\varepsilon)\ B(b)\ \$$	predict $S \to bBA$
$b\ B\ A\ \$$	$b\ A(\varepsilon)\ B(b)\ \$$	match b
$B\ A\ \$$	$A(\varepsilon)\ B(b)\ \$$	predict $B \to \varepsilon$
$A\ \$$	$A(\varepsilon)\ B(b)\ \$$	reattach $A(\varepsilon)$
$\$$	$B(b)\ \$$	error

2.5 Incremental lexing

An incremental lexer relexes text around user edits and integrates the resulting tokens into the current token stream, allowing the parser to identify changed regions [5].

General incremental lexers track each token's lookahead dependency, relex tokens whose dependencies overlap the edit, and continue until an unchanged token with the same kind, content, and relative offset is reached [5].

We use a simpler approach based on two properties of the Prev lexer: it has no explicit states and, because there are no multiline comments, tokens do not span line breaks. An edit can therefore only affect tokens on the lines it touches.

Since edits can change surrounding token boundaries, we relex from the start of the line containing the start of the edit to the first token after the end of the line containing the end of the edit, so that whitespace is attached correctly.

The relexed tokens are used as y', while x and z are recovered from the old tree, so the previous token stream does not need to be stored separately. Because we relex whole lines, some unchanged tokens are reparsed rather than reused. We accept this for simplicity, assuming lines are relatively short.

3 RESILIENT PARSING

Building a parser suitable for interactive use imposes an additional requirement: *error recovery*. Documents are often incomplete while being edited, so the parser must produce a usable tree even for syntactically invalid input, and reuse it for subsequent updates. Such a tree can contain recovery artifacts and may no longer be a parse tree of the grammar, so we call it a *syntax tree*.

Li's algorithm assumes valid input and stops at the first syntax error. We extend its table with recovery actions while preserving the key invariant that every parser step is determined solely by the symbol on top of the stack and the lookahead.

Unlike classical LL repair, these recovery decisions do not inspect the broader parsing context, such as the entire stack or the surrounding input [4, 6]. This context dependence could cause the batch and incremental parses to recover differently.

Our goal is not to repair the input, but to recover as much syntactic structure from erroneous code as possible. Kladov calls this *resilient* parsing [8]. LL parsers are a good fit here since the parse stack is an explicit prediction of what must follow.

An error is detected when the terminal at the top of the stack does not match the lookahead, or when the corresponding entry $M'[A, a]$ in the base table is empty [1].

We make each of these cases recoverable. An unmatched terminal on the P-stack is popped and attached as a *missing node*. The remaining cases are handled by filling the empty terminal entries of each table row with recovery actions, in the following order:

(1) *ε-productions*: empty cells for a nonterminal that derives the empty string are filled with its ε-production, so an optional construct parses as empty rather than an error [1]. We skip this rule where popping the nonterminal could terminate an enclosing construct, such as a top-level definition list.
(2) *Synchronizing sets*: a nonterminal is popped and attached as missing at any lookahead in its *synchronizing set*, for which we use $\text{FOLLOW}(A) \cup \{\texttt{var}, \texttt{fun}, \texttt{typ}, \$\}$ with a few language-specific modifications. The added definition starters allow recovery at the next definition.
(3) Cells left empty represent *deletion*: the lookahead is skipped, but kept in the tree as an *unexpected node*, so no token is lost. Repeated application of this rule until a synchronizing token is reached is equivalent to *panic-mode recovery* [1].

All three rules fill only terminal cells, leaving the nonterminal columns unchanged. If the lookahead is a subtree whose table entry is empty, the parser first undoes that subtree, so recovery always operates on terminal lookahead.

Divide needs to be extended to handle recovery artifacts:

(1) the X-sequence represents the *expected* structure: missing nodes are collected while unexpected nodes are skipped,

(2) the Z-subtrees hold the *actual input*: unexpected nodes are collected while missing nodes are discarded.

Missing nodes have empty yield and unexpected nodes retain their token, so the syntax tree still covers the whole document. Since recovery actions depend only on the stack top and lookahead, the argument of Theorem 2.5 extends to syntactically invalid input. Algorithm 1 shows the resulting main loop.

Input: parse stack P and input stack I produced by divide: P holds the X-sequence, I holds y', the Z-subtrees, and \$
Result: a syntax tree for $xy'z$\$

```
while P is not empty do
    X ← top(P);   Y ← top(I);
    if X = root(Y) then                          // match / reattach
        attach Y as normal; pop P, I
    else if X = $ then                           // trailing input
        undo Y; attach unexpected nodes for its leaves; pop I
    else if M'[X, root(Y)] is a production p then   // expand
        pop P; push rhs(p) onto P
    else if Y is a subtree then                  // undo
        pop I; push Y's children with nonempty yield onto I
    else if X is a terminal then                 // insert missing
        attach missing X; pop P
    else if M'[X, root(Y)] = SYNC then           // sync
        attach missing X; pop P
    else                                         // delete
        attach unexpected node for Y; pop I
    end
end
```

Algorithm 1: Incremental parse with table-driven recovery.

4 IMPLEMENTATION AND EVALUATION

We implemented the resulting parser[1] in Java and generated its lexer with JFlex [7]. We evaluated it on synthetic Prev programs ranging from 3 to 10 000 source lines, generated by combining functions from 5 templates. For each size, we generated a base program and a sequential trace of 64 line-local edits.

Each edit inserted, deleted, or replaced a single term on a randomly selected line. New terms were drawn from a fixed set of short expressions, identifiers and integer literals. The same base program and edit trace were used for every configuration.

Measurements were taken on an AMD Ryzen 7 7700X with 16 GiB of memory under Ubuntu 24.04 and OpenJDK 25.0.3, after JVM warmup. We report the mean over 3 independent runs.

Table 1 compares incremental parsing against full reparsing using the same lexer and parser, and against an equivalent LL(1) grammar parsed using ANTLR 4.13.2 [2]. For each edit, the full-reparse and ANTLR configurations process the complete document, while the incremental configuration builds on the existing syntax tree.

Incremental time per edit remains nearly flat, rising from 4.7 µs at 3 lines to 29 µs at 10 000 lines, while both baselines grow approximately linearly with the document size over the tested range. The three configurations differ by only a few microseconds on the smallest programs, but at 10 000 lines the speedups reach about 150× over a full reparse and 170× over ANTLR.

[1] https://github.com/denbal2292/resilient-ll1

Table 1: Mean processing time in µs/edit, rounded to two significant figures. Lower is better.

	Our parser		Baseline
Source lines	Incremental parse [µs]	Full reparse [µs]	ANTLR 4.13.2 [µs]
3	**4.7**	6.8	5.1
10	**4.6**	9.3	8.0
25	**4.8**	15	14
100	**5.2**	46	49
1 000	**6.9**	420	480
10 000	**29**	4 400	4 900

This workload represents a favorable case for incremental parsing since each edit is confined to a single line and changes only one term, so most of the existing syntax tree can be reused. The results should therefore be understood as showing the potential benefit of incremental parsing under highly localized edits, rather than its performance on arbitrary program changes.

5 CONCLUSIONS

We have described an incremental LL(1) parser that produces a usable syntax tree for incomplete input and reuses it for the next edit. Because recovery actions are embedded in the parse table, every step still consults only the top of the stack and the lookahead symbol, so batch and incremental parses recover identically. With an incremental lexer and an appropriate tree representation, this parser could serve as a basis for a language server.

Our measurements show that the advantage of using an incremental parser grows with the document size. Incremental parse times stay in the tens of µs for small edits even on large programs, which is fast enough to support reparsing on every keystroke.

ACKNOWLEDGMENTS

I thank Asst. Prof. Boštjan Slivnik for his guidance and comments on this work.

REFERENCES

[1] Alfred V. Aho, Monica S. Lam, Ravi Sethi, and Jeffrey D. Ullman. 2006. *Compilers: Principles, Techniques, and Tools* (2nd ed.). Pearson.
[2] ANTLR Project. 2024. ANTLR: ANother Tool for Language Recognition. https://www.antlr.org/. Version 4.13.2.
[3] Andrew W. Appel and Jens Palsberg. 2002. *Modern Compiler Implementation in Java* (2nd ed.). Cambridge University Press.
[4] Michael G. Burke and Gerald A. Fisher. 1987. A practical method for LR and LL syntactic error diagnosis and recovery. *ACM Trans. Program. Lang. Syst.* 9, 2 (March 1987), 164–197. https://doi.org/10.1145/22719.22720
[5] Lukas Diekmann. 2019. *Editing composed languages.* Ph. D. Dissertation. King's College London.
[6] C. N. Fischer, D. R. Milton, and S. B. Quiring. 1980. Efficient LL(1) error correction and recovery using only insertions. *Acta Inf.* 13, 2 (Feb. 1980), 141–154. https://doi.org/10.1007/BF00263990
[7] JFlex Project. 2023. JFlex: The Fast Scanner Generator for Java. https://www.jflex.de/. Version 1.9.1.
[8] Aleksey Kladov. 2023. Resilient LL Parsing Tutorial. https://matklad.github.io/2023/05/21/resilient-ll-parsing-tutorial.html. Accessed: 21 July 2026.
[9] Warren X. Li. 1995. A Simple and Efficient Incremental LL(1) parsing. In *Proceedings of the 22nd Seminar on Current Trends in Theory and Practice of Informatics (SOFSEM '95).* Springer-Verlag, Berlin, Heidelberg, 399–404.
[10] Microsoft Corporation. 2026. Language Server Protocol. https://microsoft.github.io/language-server-protocol/. Accessed: 21 July 2026.
[11] Tim A. Wagner and Susan L. Graham. 1998. Efficient and flexible incremental parsing. *ACM Transactions on Programming Languages and Systems* 20, 5 (Sept. 1998), 980–1013. https://doi.org/10.1145/293677.293678

 DOI: https://doi.org/10.26493/scores26.09

Marker Diagrams for String Rewriting Systems

Janez Ignacij Jereb
jj21232@student.uni-lj.si
Faculty of Mathematics and Physics,
University of Ljubljana
SI-1000 Ljubljana, Slovenia

ABSTRACT

This paper introduces a novel graphical formalism for programming and verifying string rewriting systems. The formalism uses distinguished subwords, called markers, to identify situations within a computation. These situations and the transitions between them form marker graphs, which provide an explicit execution structure for string rewriting systems. The semantics of marker graphs and a proof method for establishing invariants are defined. We also define induced marker graphs and a reduction procedure for removing unreachable situations. Finally, we present marker diagrams, a compact graphical representation of marker graphs based on Euler diagrams. The resulting diagrams provide an intuitive representation of the structure and execution of string rewriting programs while supporting the construction of correctness proofs.

KEYWORDS

String rewriting, Euler diagrams, invariants, program correctness

1 INTRODUCTION

Most work on string rewriting systems (SRSs) has focused on confluence, termination, and complexity [2]. These systems are often treated as externally given computational systems to be analyzed rather than as a programming model. In this paper, we develop a graphical formalism intended to ease *formally correct programming* of string rewriting systems.

Correctness proofs for SRSs commonly rely on identifying an invariant and showing that it is preserved by every rule. We introduce special words, called *markers*, to decompose a computation into distinct situations, each with its own invariants. This allows a program to be verified using several smaller invariants rather than a single monolithic invariant.

We represent this structure using *marker graphs*, which capture the situations and transitions induced by the markers. We further introduce *marker diagrams*, a graphical representation of these graphs based on Euler diagrams. The diagrams provide a geometric intuition for the structure of the computation and a new perspective for analyzing such systems.

2 STRING REWRITING SYSTEMS

String rewriting systems provide a Turing-complete model of computation [2].

Formally, a *string rewriting system* is a pair (Σ, R) where Σ is a finite set of symbols and $R \subseteq \Sigma^* \times \Sigma^*$ is a finite set of rewrite rules. Here, Σ^* represents the set of all finite strings created from the alphabet Σ. This set defines the relation

$$sxt \xrightarrow{R} syt :\Leftrightarrow (x, y) \in R,$$

where $s, t \in \Sigma^*$ are arbitrary strings. In other words, a rule may be applied by replacing an occurrence of x by y within an arbitrary context. We use $\square$ to denote the boundary of a string. This is a syntactic convenience that allows us to express rules that apply only at the beginning or end of a string. So the rule $(\square x, y)$ can be applied only when x occurs at the beginning of a string. We will usually leave the set of symbols implicit.

3 MARKER GRAPHS

Unlike most other models of computation, string rewriting systems do not explicitly provide an execution graph. Nevertheless, we can introduce such structure by identifying *markers*: special subwords that indicate the current situation of the computation. Markers should not occur in the input and are used exclusively to structure the computation.

Markers are necessary for implementing certain non-trivial programs. To illustrate this, consider a program that accepts any input w and returns ww. Since the input may have arbitrary size, the program must perform multiple rewriting steps to produce the final result. Thus, for some input w_0, we obtain a trace $w_0, w_1, w_2, \ldots, w_n$, where $w_n = w_0 w_0$. Now consider an intermediate string w_i with $1 < i < n$. Since the program accepts every string as input, we may use w_i as a new input. It should then produce $w_i w_i$. However, continuing the original computation from w_i produces $w_0 w_0$ instead, yielding a contradiction.

This argument generalizes to any program that requires multiple rewriting steps and computes an injective function. Injectivity ensures that distinct inputs cannot legitimately produce the same output. Markers provide the additional state information needed to distinguish an intermediate state from a valid input, allowing such computations to be implemented.

3.1 Marker Graphs for String Rewriting Systems

We now use markers to formally introduce a graph structure that represents an execution graph of a string rewriting system.

We first define the notion of subwords. We write $u \sqsubseteq w$ if and only if there exist strings x and y such that $w = xuy$. Thus, $u \sqsubseteq w$ means that u occurs as a contiguous subword of w.

Let $M \subseteq \Sigma^*$ be an arbitrary finite set of *markers* and define

$$\underline{M}(w) := \{m \in M \mid m \sqsubseteq w\},$$

as the set of markers that occur in w. A marker represents a subword of a string that marks the current *situation* of the computation. The situation of the computation is just the set of markers that occur in the current string.

We use situations for the nodes of the execution graph. An edge between two situations corresponds to a possible transition between those situations, and are labeled with the corresponding rewriting

rule. Together they form a directed labeled graph, called the *marker graph*.

A single rewriting rule may give rise to multiple edges, and a given string rewriting system may admit multiple different marker graphs. We will later define conditions that determine when a marker graph is well-formed.

Definition 3.1. A tuple (M, T, R) is a *marker graph* if R is a string rewriting system, M is a finite set of markers and $T \subseteq 2^M \times R \times 2^M$ is a finite set of edges between situations.

Example 3.2. Let us define a string rewriting system R which computes the parity of a string of ones. Its set of markers $M = \{O, E\}$ represent the parity of the current count of the program.

$$R = \{\ (\square 1,\ \square O),\quad (O1,\ E),\quad (E1,\ O)\ \}$$

$$T = \{\ (\{\}, (\square 1, \square O), \{O\}),\quad (\{O\}, (O1, E), \{E\}),\quad (\{E\}, (E1, O), \{O\})\ \}$$

A marker graph is shown in the figure below:

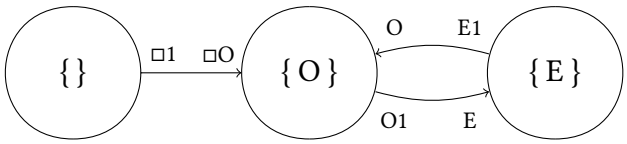


Figure 1: A marker graphs for a SRS that computes parity

3.2 Semantics of marker graphs

We now define the semantics of a marker graph. The semantics are defined in terms of traces of the underlying string rewriting system. A *trace* of a string rewriting system R is a sequence $\tau = (\tau_1, \tau_2, \ldots, \tau_n)$ of strings such that $\tau_i \xrightarrow{R} \tau_{i+1}$ for every $1 \le i < n$.

A marker graph is *well-formed* with respect to a string rewriting system R if, for every trace τ of R and for every $1 \le i < n$, there exists an edge $(\underline{M}(\tau_i),\ e,\ \underline{M}(\tau_{i+1})) \in T$ such that $\tau_i \xrightarrow{\{e\}} \tau_{i+1}$.

Thus, for each trace, we can follow the corresponding path through the marker graph and recover the sequence of transitions of the computation. An example of a well-formed graph is shown in Figure 1. If we removed any edge or vertex from it, the graph would no longer be well-formed.

We next define the operator $\models^\tau$, which specifies when a well-formed marker graph satisfies an invariant along the trace τ. By universally quantifying over traces we get the desired notion of validity.

Definition 3.3. Let $D = (M, T, R)$ be a well-formed marker graph for an SRS R, let $P : 2^M \times \Sigma^* \to \{\bot, \top\}$ be a predicate, and let τ be a trace of R. We write $D \models^\tau P$ if and only if, for every $1 \le i < n$,

$$P(\underline{M}(\tau_i), \tau_i) \Rightarrow P(\underline{M}(\tau_{i+1}), \tau_{i+1}).$$

Definition 3.4. We write $D \models P$ if and only if, for every trace τ of R, $D \models^\tau P$.

We call the predicate P that satisfies $D \models P$ an *invariant* of the marker graph D. The first argument of the invariant is used to identify the current situation. It is mostly used to suggest defining predicates by situations but it could be safely omitted from the definition.

3.3 Proof method

We can now formalize the proof method used for marker graphs. For each edge, we show that if a word is in the source situation and satisfies the invariant, then applying the rewrite rule associated with the edge produces a word in the target situation that satisfies the invariant.

Definition 3.5. Let $D = (M, T, R)$ be a marker graph and let $P : 2^M \times \Sigma^* \to \{\bot, \top\}$ be a predicate. We write $D \vdash P$ if and only if, for every edge $(A, (u, v), B) \in T$, and for every word $x, y \in \Sigma^*$ satisfying $\underline{M}(x) = A,\ \underline{M}(y) = B$, it holds that

$$x \xrightarrow{\{(u,v)\}} y \implies (P(A, x) \Rightarrow P(B, y)).$$

The proof procedure is *sound*: if every edge of a well-formed marker graph preserves the predicate, then the predicate is preserved along every trace. Completeness, however, does not follow immediately. Actual traces may restrict which words can occur as intermediate states, whereas the edge-based proof considers all words belonging to the corresponding situations.

The logic could potentially be extended to obtain completeness, by introducing ghost variables and imperative programs operating on them, translating the additional information into a restricted form of the invariant-oriented diagrams of Back and Preoteasa [1].

Lemma 3.6 (Soundness). *Let $D = (M, T, R)$ be a well-formed marker graph for R. If $D \vdash P$, then $D \models P$.*

Proof. Let τ be an arbitrary trace of R. We must show that

$$P(\underline{M}(\tau_i), \tau_i) \Rightarrow P(\underline{M}(\tau_{i+1}), \tau_{i+1})$$

for every $1 \le i < n$.

Since D is well-formed, for each such i there exists an edge

$$(\underline{M}(\tau_i),\ (u, v),\ \underline{M}(\tau_{i+1})) \in T$$

such that $\tau_i \xrightarrow{\{(u,v)\}} \tau_{i+1}$. By the definition of $\vdash$, applied to this edge with $x = \tau_i$ and $y = \tau_{i+1}$, we obtain

$$P(\underline{M}(\tau_i), \tau_i) \Rightarrow P(\underline{M}(\tau_{i+1}), \tau_{i+1}).$$

Therefore $D \models^\tau P$. Since τ was arbitrary, it follows that $D \models P$. □

Example 3.7. Continuing with the previous example, we will for now skip the proof that the marker graph is well-formed.

Let w_s be the starting word. We can define an invariant P_{w_s} by

$$P_{w_s}(\emptyset, 1^n) = \top,$$
$$P_{w_s}(\{O\}, O1^n) = \text{odd}(|w_s| - n),$$
$$P_{w_s}(\{E\}, E1^n) = \text{even}(|w_s| - n).$$

Here, $\text{odd}(n)$ and $\text{even}(n)$ denote the propositions that n is odd and even, respectively. We can prove this invariant by considering each of the three edges of the marker graph.

For the first edge, $(\emptyset, (\square 1, \square O), \{O\})$, we consider a word 1^n and its successor $O1^{n-1}$. Since this transition can occur only when the

starting word is a string of ones of length $n = |w_s|$, we obtain $P_{w_s}(\emptyset, 1^n) = \top$ and

$$\begin{aligned} P_{w_s}(\{O\}, O1^{n-1}) &= \text{odd}(|w_s| - (n-1)) \\ &= \text{odd}(n - (n-1)) \\ &= \text{odd}(1) = \top. \end{aligned}$$

For the second edge, $(\{O\}, (O1, E), \{E\})$, consider a transition $O1^n \longrightarrow E1^{n-1}$. If the invariant holds before the transition, then $\text{odd}(|w_s| - n)$ holds. Consequently, $|w_s| - n$ is odd, and therefore $|w_s| - (n-1) = (|w_s| - n) + 1$ is even. Hence

$$P_{w_s}(\{E\}, E1^{n-1}) = \text{even}(|w_s| - n + 1) = \top.$$

We use a similar argument for the third transition.

We have proven $(M, T, R) \vdash P_{w_s}$ and so by soundness $(M, T, R) \models P_{w_s}$. Thus P_{w_s} is an invariant of every trace of the SRS.

4 CONSTRUCTING MARKER GRAPHS

4.1 Induced marker graphs

While the full graph is well-formed by construction, we are interested in more compact well-formed marker graphs. For every string rewriting system and set of markers, we can construct a canonical well-formed marker graph containing all transitions of the system.

Definition 4.1. An *induced marker graph* (M, T, R) is defined by a set of markers $M \subseteq \Sigma^*$, an SRS R, and

$$T := \left\{ (\underline{M}(x), (u, v), \underline{M}(y)) \mid x \xrightarrow{\{(u,v)\}} y,\ (u, v) \in R \right\}.$$

The well-formedness of the induced marker graph follows immediately from the definition.

The induced marker graph is not necessarily convenient to compute directly, as its definition quantifies over arbitrary strings. We therefore restrict our attention to a class of string rewriting systems for which the induced graph can be constructed in a simpler way.

Definition 4.2. A SRS R is *non-overlapping* with respect to a set of markers M if for every rule $(u, v) \in R$, every marker $m \in M$, and all strings $x, y \in \Sigma^*$, $\underline{M}(xuy) = \underline{M}(x) \cup \underline{M}(u) \cup \underline{M}(y)$, and likewise for xvy.

In simpler terms, the interaction between the rewriting rules and their context should not introduce new markers. This condition is automatically satisfied when all markers are single symbols.

For non-overlapping SRSs, the induced marker graph can be constructed directly from the rules and the set of markers.

Definition 4.3. The *simple induced marker graph* (M, T, R) for a non-overlapping rewriting system R is

$$T := \left\{ (S \cup \underline{M}(u), (u, v), S \cup \underline{M}(v)) \mid (u, v) \in R,\ S \subseteq M \right\}.$$

Lemma 4.4. *For every non-overlapping SRS R with respect to a set of markers M, the simple induced marker graph (M, T, R) is well-formed.*

Proof. Consider an arbitrary transition $xux' \xrightarrow{\{(u,v)\}} xvx'$ in a trace of R, where $(u, v) \in R$. Let $S := \underline{M}(x) \cup \underline{M}(x')$. Since R is non-overlapping with respect to M, every marker occurring in xux' must occur entirely within one of x, u, or x'. Hence

$$\underline{M}(xux') = \underline{M}(x) \cup \underline{M}(u) \cup \underline{M}(x') = S \cup \underline{M}(u).$$

Similarly, $\underline{M}(xvx') = S \cup \underline{M}(v)$. Therefore,

$$\left(\underline{M}(xux'), (u, v), \underline{M}(xvx')\right) = \left(S \cup \underline{M}(u), (u, v), S \cup \underline{M}(v)\right) \in T.$$

Thus every transition occurring in a trace is represented by an edge of the simple induced marker graph, so the graph is well-formed. □

Example 4.5. For the string rewriting system R which returns the parity of a string of ones we can define its induced marker graph.

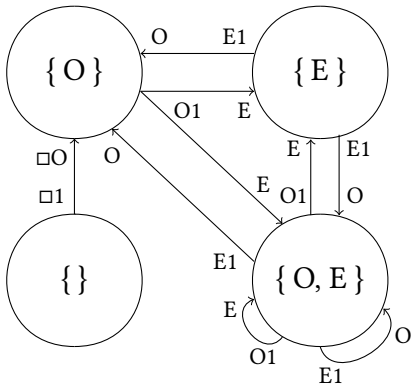


Figure 2: The induced marker graph for a SRS computing parity

4.2 Marker graph reduction

A drawback of (simple) induced marker graphs is their size. In particular, they may contain situations that can never occur during an execution. We can reduce the size of the graph by removing unreachable situations.

To perform this reachability analysis we first introduce *seeded string rewriting systems*. A seeded SRS has a distinguished set of *seed rules*. Each seed rule is of the form $\square x_i \longrightarrow \square \dagger y_i$, where x_i contains no markers, all seed rules introduce the same number of markers, and no other rule contains the symbol †.

The seed rules determine the initial number of markers. We associate with a seeded SRS a vector of marker counts of dimension $|M|$, where each component corresponds to a marker. The initial vector is obtained from the seed rule set and records the number of occurrences of each marker introduced by the seed rule.

Each non-seed rewriting rule induces a vector describing its effect on the marker counts. Thus, the evolution of marker counts can be represented as a vector addition system. We can use the Karp–Miller procedure [3] to determine the reachable marker-count vectors.

Since marker graphs record only whether a marker is present, rather than its exact multiplicity, we can modify the Karp–Miller procedure to track only zero and nonzero counts.

Example 4.6. Consider again the parity SRS, now written as a seeded SRS:

$$R = \{\ (\square 1,\ \square \dagger O),\ \ (O1,\ E),\ \ (E1,\ O)\ \}.$$

The non-seed rules induce the marker-count vectors

$$(O1,\ E) \rightsquigarrow (-1,\ 1), \quad (E1,\ O) \rightsquigarrow (1,\ -1).$$

 DOI: https://doi.org/10.26493/scores26.09

The seed rule produces the initial vector $(1, 0)$.

Applying the Karp–Miller procedure, we find that the only unreachable situation is $\{O, E\}$. It can therefore be removed from the marker graph from Figure 2 to obtain the marker graph in Figure 1.

5 MARKER DIAGRAMS

We can now introduce the graphical representation of marker graphs, which we call *marker diagrams*. A marker diagram is not a new mathematical object, but a graphical representation of a marker graph. The marker graph contains the formal information about situations and transitions, while the marker diagram provides a compact and intuitive way of displaying this information.

Marker diagrams use Euler diagrams [4] augmented with arrows. Euler diagrams are particularly well suited for this purpose because they provide an intuitive way of representing sets and their intersections. Moreover, Venn diagrams can represent all subsets of a finite set [5], while Euler diagrams allow us to omit unnecessary regions and obtain more readable representations.

The regions of an Euler diagram represent situations, while arrows represent transitions between situations. Each region corresponds to a set of markers: a marker is present in a situation precisely when the corresponding region lies inside the circle or oval associated with that marker.

An arrow between two regions represents an edge of the marker graph. An arrow from or to a circle or oval represents multiple edges, one for each corresponding region. When multiple arrows are drawn together, they are understood to use the same rewrite rule, and the drawing represents all corresponding edges of the marker graph. For aesthetic reasons, arrows with the same augmentation are joined whenever possible.

We also use variables to reduce the amount of writing and to make diagrams more readable and general. Before presenting a diagram, we specify the sets over which its variables range, together with any additional constraints on them.

To illustrate the diagrammatic representation, consider an SRS that checks whether a string is a palindrome. The general SRS is

$$x, y, z \in \Sigma^* \setminus \{\lceil, \rceil\}, \qquad \square x \to \square\lceil x, \qquad x\square \to x\rceil\square,$$
$$\lceil xx\rceil \to, \qquad \lceil xyz \to y\lceil xz, \qquad xyz\rceil \to xz\rceil y.$$

Its operation can be seen from the following trace:

$$abba \to \lceil abba \to b\lceil aba \to b\lceil aba\rceil \to$$
$$\to b\lceil aa\rceil b \to \lceil b\lceil aa\rceil b \to \lceil bb \to \lceil bb\rceil \to .$$

In Figure 3, we give a marker diagram representing a well-formed marker graph. The markers are $M = \{\lceil, \rceil\}$. For $x, y, z \in \Sigma^* \setminus M$:

The diagram uses the geometry of the Euler diagram to represent several situations simultaneously. For example, the middle arrow originating in the region corresponding to $\{\lceil, \rceil\}$ and bifurcating towards $\emptyset$, $\{\lceil\}$, $\{\rceil\}$, and $\{\lceil, \rceil\}$ represents four distinct edges of the marker graph.

Similarly, the arrow on the left corresponding to the rule $\lceil xyz \to y\lceil xz$ is drawn from the circle representing $\lceil$ back to itself. It therefore represents four edges, corresponding to all combinations of source and target situations containing $\lceil$:

$$\{\lceil\} \to \{\lceil\}, \quad \{\lceil\} \to \{\lceil, \rceil\}, \quad \{\lceil, \rceil\} \to \{\lceil\}, \quad \{\lceil, \rceil\} \to \{\lceil, \rceil\}.$$

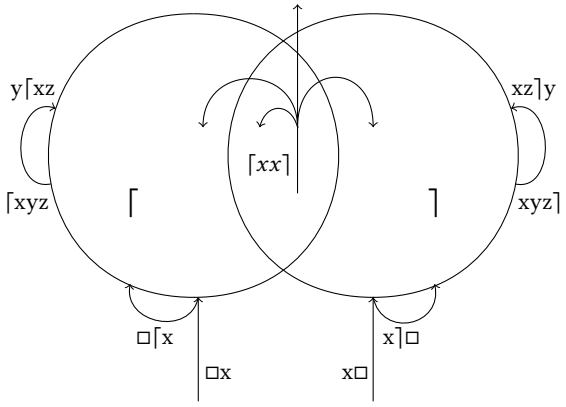


Figure 3: Palindrome testing SRS

The bottom-left arrow similarly represents six edges of the marker graph: three edges from $\emptyset$ to each situation containing $\lceil$, and three edges between situations containing $\lceil$.

This notation allows us to represent the marker graph very compactly. In this example, five arrows, some of which bifurcate or represent multiple edges, suffice to represent 24 edges of the underlying marker graph.

6 CONCLUSIONS

We introduced a graphical formalism for programming and verifying string rewriting systems. By introducing *markers*, we can distinguish different situations of a computation and represent them as a *marker graph*. We defined the semantics of marker graphs and a local proof method for invariants, and proved its soundness.

We further introduced induced marker graphs and a reachability-based reduction procedure for removing situations that cannot occur during an execution. Finally, we presented *marker diagrams*, a compact representation of marker graphs based on Euler diagrams. These diagrams provide an intuitive view of the structure and execution of string rewriting programs while retaining the formal structure needed for verification.

Future work includes software support, exploring the inclusion of marker multiplicity in the formalism, developing a complete proof system, and improving graph reduction.

REFERENCES

[1] Ralph-Johan Back and Viorel Preoteasa. 2011. Semantics and proof rules of invariant based programs. In *Proceedings of the 2011 ACM Symposium on Applied Computing*. 1658–1665.

[2] Ronald V. Book and Friedrich Otto. 1993. String-rewriting systems. In *String-Rewriting Systems*. Springer, 35–64.

[3] Richard M. Karp and Raymond E. Miller. 1969. Parallel program schemata. *Journal of Computer and system Sciences* 3, 2 (1969), 147–195.

[4] Peter Rodgers. 2014. A survey of Euler diagrams. *Journal of Visual Languages & Computing* 25, 3 (2014), 134–155.

[5] J. Venn. 1880. I. On the diagrammatic and mechanical representation of propositions and reasonings. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 10, 59 (1880), 1–18. https://doi.org/10.1080/14786448008626877 arXiv:https://doi.org/10.1080/14786448008626877

 DOI: https://doi.org/10.26493/scores26.11

Encoding Graphs without Gradients

Matic Požar
matic.pozar@upr.si
University of Primorska
Faculty of Mathematics, Natural Sciences and Information Technologies ,
Glagoljaška 8
SI-6000 Koper, Slovenia

ABSTRACT

Graph Autoencoders are commonly used for learning representations that are subsequently used for downstream tasks such as link prediction. Most canonical methods employ iterative updates with gradient descent to refine their representations, which poses limitations on scalability, interpretability, and generalizability. We develop a deterministic graph kernel, called EGG, that outperforms linear and graph-convolutional autoencoders on Cora, Citeseer, and Pubmed datasets. Furthermore, our kernel exhibits favorable asymptotic runtime, which is reflected in the results of our experiments, where we see speedups of over two orders of magnitude.

KEYWORDS

graph autoencoder, link prediction, graph kernel, spectral methods

1 INTRODUCTION AND MOTIVATION

Graph Structured Learning (GSL) has received significant attention in recent years [13]. One of the most important objectives of GSL is link prediction, where the model is asked to identify edges that are likely missing from the given graph. Link prediction has applications in numerous fields, such as recommender systems [12], drug discovery [15], and knowledge graphs [2]. Most commonly, link prediction is implemented with a subdomain of GSL called Graph Autoencoders (GAE) [6] using Convolutional Graph Neural Networks (GCN) [7] with binary cross-entropy (BCE) loss. More recently Salha et al. [10] empirically demonstrated that simple linear encoders can match the performance of more complex counterparts in a wide range of domains.

This motivated us to pose the central research question of this work: can GAEs be modeled analytically, while maintaining their effectiveness? One of the most noteworthy success stories of analytical modeling in GSL is NetMF [9], which successfully unified several popular network embedding methodologies [3, 8, 11] under the framework of implicit matrix factorization, thus producing both better empirical results as well as deepening the understanding of said methods.

Our main contributions are:

- We derive a finite-time analytical approximation of the optimization dynamics of Linear GAE.
- We show that these dynamics induce a deterministic spectral graph kernel requiring no gradient-based training.
- We introduce a normalized positive-spectrum formulation that transfers across graphs of different spectral scales.
- On three citation-network benchmarks, EGG outperforms Linear and GCN autoencoders in the featureless setting and remains competitive or superior with node features, while substantially reducing runtime.

2 FROM LINEAR GAE TO A DETERMINISTIC KERNEL

In this section we shall describe the derivation of our model. Following [10] we have

$$S = \tilde{A} = D^{-1/2}(A + I)D^{-1/2},\ Z = SW,\ X = ZZ^{\top},$$

where $A \in \mathbb{R}^{n\times n}$ is the adjacency matrix of a graph with n nodes, I is the identity matrix, D is the diagonal degree matrix of $A + I$, $W \in \mathbb{R}^{n\times d}$ is the weight matrix, $Z \in \mathbb{R}^{n\times d}$ is the learned representation matrix, and $X \in \mathbb{R}^{n\times n}$ serves as the matrix of predicted edge scores, with probabilities $\sigma(X)$.

2.1 Quadratic surrogate and optimization dynamics

The binary cross-entropy objective is poorly suited to analytical treatment because the sigmoid-induced nonlinearity prevents a closed-form solution. That is why we decided to replace it in the loss function with a suitable surrogate. A direct squared-error replacement substantially degraded link-prediction performance in preliminary experiments, whereas the second-order Taylor approximation of BCE retained performance close to the original objective.

For a single logit x and binary target $y \in \{0, 1\}$, the binary cross-entropy loss can be written as $\ell(x, y) = \log(1+e^x) - yx$. Expanding around $x = 0$ gives $\ell(x, y) = \log 2 + (0.5 - y)\, x + x^2/8 + O(x^3)$. Completing the square in the quadratic terms yields

$$\ell(x, y) = \frac{1}{8}\,[x - (4y - 2)]^2 + C_y + O(x^3),$$

where C_y is independent of x. Thus, up to an additive constant and a positive multiplicative factor, minimizing the second-order approximation of BCE is equivalent to minimizing the squared error between the predicted logit x and the transformed target $4y - 2$.

Applying this entrywise to the decoder score matrix $X = ZZ^{\top}$ gives the quadratic surrogate

$$\mathcal{L} = \frac{1}{n^2}\left\|M - ZZ^{\top}\right\|_F^2, \qquad M = 4Y - 2J,$$

where $Y = A + I$ is the binary reconstruction target including self-loops and J is the all-ones matrix. Consequently, entries corresponding to edges and self-loops have target value +2, whereas nonedges have target value −2. The matrix M therefore arises directly from the second-order Taylor approximation of the BCE objective rather than being introduced as an independent design choice. Although the resulting matrix-factorization problem admits a spectral optimum, we observed that lower reconstruction loss

Table 1: Cora ablation of the derivation. Results are mean ± sample standard deviation across two random edge splits.

Variant	AUC (%)
Linear AE + BCE	81.73 ± 0.15
Taylor-BCE GD	80.66 ± 0.97
Exact spectral optimum	75.30 ± 0.18
Direct Z-optimization	74.74 ± 2.46
W-optimization	80.39 ± 0.63
EGG	88.31 ± 0.14

does not translate into better link prediction. This motivates approximating finite-time training rather than solving the surrogate to convergence.

Substituting $Z = SW$ gives

$$\mathcal{L}(W) = \frac{1}{n^2} \left\| M - SWW^\top S \right\|_F^2 .$$

Since M and S are symmetric, differentiating with respect to W yields

$$\begin{aligned} \nabla_W \mathcal{L} &= \frac{4}{n^2} S \left(SWW^\top S - M \right) SW \\ &= \frac{4}{n^2} \left[S^2 W (W^\top S^2 W) - SMSW \right] \end{aligned} \tag{1}$$

2.2 Ablation of the derivation

To examine which parts of the Linear GAE training procedure are responsible for its predictive performance, we performed a series of ablations on Cora, summarized in Table 1. Replacing BCE directly with squared error led to a substantial decrease in link-prediction accuracy, whereas the second-order Taylor approximation retained performance close to BCE. Interestingly, solving the resulting quadratic factorization problem to its spectral optimum further decreased link-prediction accuracy, despite achieving a lower reconstruction loss. This indicates that the finite optimization trajectory acts as an important form of implicit regularization.

We further compared direct optimization of the embeddings Z with optimization of the encoder weights W. The latter consistently performed better, supporting the importance of the S^2-preconditioned optimization geometry induced by $Z = SW$. Finally, during finite-time training the cubic and linear components of Eq. 1 become strongly aligned, motivating the linearized dynamics used by EGG.

To further examine the relationship between reconstruction quality and link prediction, we compute the exact rank-r PSD optimum of the quadratic surrogate for increasing r. As shown in Fig. 1, the reconstruction objective decreases monotonically with rank, whereas validation AUC reaches its maximum at only $r = 20$ and subsequently deteriorates. Thus, improved optimization of the reconstruction objective does not necessarily translate into improved downstream performance, motivating our focus on finite-time rather than converged dynamics.

2.3 Finite-time linearization

The gradient in Equation 1 consists of a term linear in W, $SMSW$, and a cubic term, $S^2W(W^\top S^2 W)$. Empirically, the nonlinear term can become comparable in norm to the linear term, while their

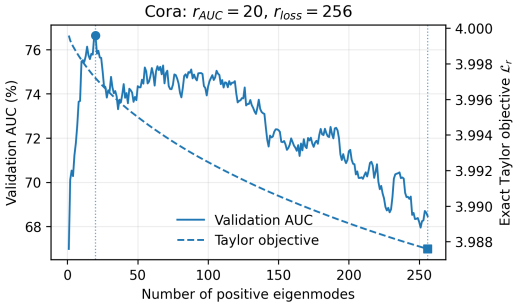


Figure 1: Exact Taylor reconstruction versus link-prediction performance on Cora. Increasing the number of positive eigenmodes monotonically decreases the exact quadratic reconstruction objective, while validation AUC peaks at $r = 20$, far before the highest evaluated rank $r = 256$. This illustrates the mismatch between reconstruction optimality and downstream link-prediction performance.

cosine similarity approaches one during training. For example, at the validation-AUC-selected checkpoint, the linear and cubic gradient components have cosine similarity 0.959 ± 0.021 across ten Cora splits; complete trajectory diagnostics are available in the accompanying repository. This suggests that the nonlinear component primarily regulates the magnitude of the update rather than substantially altering its direction. We therefore approximate the finite-time dynamics by retaining only the linear component $\nabla_W \mathcal{L} \approx -\frac{4}{n^2} SMSW$. Using the standard gradient descent updates with $\alpha = \frac{4\eta}{n^2}$, where η is the learning rate, one obtains $W_{t+1} = (I + \alpha SMS)W_t$ and by iteration $W_t = (I + \alpha SMS)^t W_0$ which gives

$$Z_t = S(I + \alpha SMS)^t W_0, \text{ and } X_t = S(I + \alpha SMS)^t W_0 W_0^\top (I + \alpha SMS)^t S.$$

Marginalizing the isotropic initialization gives

$$\mathbb{E}\left[W_0 W_0^\top \right] \propto I.$$

Finally, we arrive at our deterministic finite-time link-prediction kernel

$$K_t = S(I + \alpha SMS)^{2t} S.$$

2.4 Normalized spectral kernel

The parameter α in the preceding formulation is not directly comparable across graphs, since the evolution of each spectral component depends on the product $\alpha\lambda_i$, where λ_i is an eigenvalue of SMS. Consequently, a fixed value of α can induce very different amounts of amplification on graphs with different spectral scales. We therefore normalize the step size by the largest positive eigenvalue,

$$\alpha = \frac{\gamma}{\lambda^+_{\max}(SMS)},$$

where $\gamma > 0$ is a dimensionless parameter.

Let $SMS = V\Lambda V^\top$ be the eigendecomposition of SMS. The corresponding spectral filter is

$$g_t(\lambda_i) = \left(1 + \gamma \frac{\lambda_i}{\lambda^+_{\max}} \right)^{2t} .$$

Since the linearized dynamics amplify the positive spectral modes, we retain the r largest positive eigenvalues and their corresponding eigenvectors V_r. This yields the truncated kernel

$$K_t \approx S^2 + SV_r \operatorname{diag}\left[\left(1+\gamma\frac{\lambda_i}{\lambda^+_{\max}}\right)^{2t} - 1\right] V_r^\top S.$$

The effective amount of spectral evolution is governed primarily by $\tau = 2t\gamma$. Indeed, for sufficiently small γ,

$$\left(1+\gamma\frac{\lambda_i}{\lambda^+_{\max}}\right)^{2t} \approx \exp\left(\tau\frac{\lambda_i}{\lambda^+_{\max}}\right),$$

providing a dimensionless interpretation of the finite training horizon.

2.5 Node features

For graphs with node features F, setting $H = SF$ gives the analogous feature-space operator $B_F = F^\top SMSF$. The resulting kernel is

$$K_t^{(F)} = SF\left(I + \frac{\gamma}{\lambda^+_{\max}(B_F)}B_F\right)^{2t} F^\top S,$$

with the same positive-spectrum truncation used in practice.

2.6 Practical computation of EGG

Starting from the training graph, we construct the normalized adjacency matrix S. The dense matrix $M = 4Y - 2J$ need not be formed explicitly, since for any vector x,

$$SMSx = S\left(4Y(Sx) - 2\mathbf{1}\left(\mathbf{1}^\top Sx\right)\right).$$

Thus, matrix-vector products required by the iterative eigensolver can be evaluated using sparse graph operations. We retain the r largest positive eigenpairs of SMS, form $U = SV_r$, and define

$$g_i = \left(1+\gamma\frac{\lambda_i}{\lambda^+_{\max}}\right)^{2t}.$$

For a node pair (u, v), the truncated kernel is

$$K_t(u,v) = (S^2)_{uv} + \sum_{i=1}^{r}(g_i - 1)U_{ui}U_{vi}.$$

The reported implementation explicitly computes S^2. Its cost therefore depends on the number of nonzero entries in S^2, which can be substantially larger than in S. This is an implementation limitation rather than a requirement of the formulation: since S is symmetric,

$$(S^2)_{uv} = \langle S_{u,:}, S_{v,:}\rangle,$$

the baseline term can instead be evaluated only for queried pairs using a sparse row inner product. After computing U, such a memory-efficient implementation scores a pair in

$$O(d_S(u) + d_S(v) + r)$$

time without materializing either S^2 or the full dense kernel.

3 EXPERIMENTS AND RESULTS

All experiments were run on a Dell Precision 7710 workstation with an Intel Core i7-6820HQ processor (8 logical CPUs), 16 GB of RAM, and a 256.1 GB disk, running Ubuntu 24.04.4 LTS.

We evaluate on Cora, Citeseer, and Pubmed [14] using 10 random edge splits, reserving 5% of edges for validation and 10% for testing, with an equal number of sampled nonedges. We consider both featureless and feature-aware settings. Hyperparameters and training epochs are selected exclusively on validation AUC. We compare against Linear AE [10] and two- and three-layer GCN autoencoders [6] using the same splits. Classical link predictors including PPR [4] and Katz [5] were additionally evaluated but consistently underperformed and are omitted from Table 2.

All autoencoders use a 16-dimensional latent representation and are trained with Adam with learning rate 0.01 for at most 250 epochs. The GCN models use 32-dimensional hidden layers and no dropout. The epoch maximizing validation AUC is selected independently for each split. For EGG, we retain the largest 256 positive eigenmodes and select $\gamma \in \{0.01, 0.02, 0.05, 0.1\}$ and $t \in \{0, 5, \ldots, 300\}$ using validation AUC. All experiments are performed on CPU. Source code and scripts for reproducing all experiments are publicly available at `https://github.com/Matic054/EGG`.

In the featureless setting EGG outperformed every method on all 30 splits. In particular, on the Pubmed dataset EGG achieved a mean AUC of 89.45, while the second best, a 3-layer GCN, obtained a mean AUC of 82.92.

When features were available, EGG showed a strong gain on Citeseer and Pubmed, while being essentially tied with the Linear AE in terms of AUC and slightly behind in AP.

As a robustness check, we repeated the experiments using degree-corrected negative sampling [1]. In the featureless setting, EGG retained a clear advantage, achieving AUCs of 89.83, 88.08, and 88.23 on Cora, Citeseer, and Pubmed, compared with 84.13, 78.67, and 80.57 for the strongest learned baseline, respectively. With node features the results were mixed: EGG remained strongest on Citeseer, but underperformed the learned models on Cora and Pubmed, suggesting that the featureless structural kernel is particularly robust to controlling for endpoint-degree effects.

In terms of runtime, EGG has a clear advantage in all settings. The likely explanation is that the inner-product decoder used by the learned autoencoders requires evaluating $ZZ^\top$, resulting in $O(n^2d)$ work per training epoch, or $O(Tn^2d)$ over T epochs. In contrast, EGG uses a truncated eigendecomposition of the implicitly represented operator SMS, whose matrix-vector products can be evaluated in $O(m+n)$ sparse work. The current implementation explicitly forms S^2, so its runtime and memory additionally depend on the number of nonzero entries in S^2. Despite this, EGG is substantially faster in our experiments. This difference is particularly visible on Pubmed, the largest dataset: in the featureless setting, the smallest mean runtime among the learned methods is 1268.53 s, compared with 29.22 s for EGG. With node features, the corresponding runtimes are 1313.38 s and 11.29 s.

Reported EGG runtimes include validation-based selection over (γ, t); this search can constitute a substantial fraction of the total runtime, although candidate configurations are independent and can be evaluated in parallel.

Table 2: Link prediction on Cora, Citeseer, and Pubmed. Results are mean ± sample standard deviation across ten random edge splits. AUC and AP are in percent; runtime is wall-clock seconds.

Model	Cora			Citeseer			Pubmed		
	AUC (%)	AP (%)	Time (s)	AUC (%)	AP (%)	Time (s)	AUC (%)	AP (%)	Time (s)
Without node features									
Linear AE	83.80 ± 1.18	88.15 ± 0.93	26.68 ± 1.47	76.85 ± 1.33	83.11 ± 1.18	47.43 ± 0.05	81.50 ± 0.50	87.32 ± 0.37	1268.53 ± 15.18
2-layer GCN AE	85.56 ± 1.40	89.07 ± 1.04	26.50 ± 0.51	77.99 ± 1.94	83.20 ± 1.93	47.99 ± 0.15	81.93 ± 0.46	86.84 ± 1.46	1274.14 ± 19.86
3-layer GCN AE	85.44 ± 1.19	88.31 ± 1.05	26.71 ± 0.80	79.13 ± 1.75	82.56 ± 1.72	48.47 ± 0.08	82.92 ± 0.53	87.64 ± 0.35	1280.39 ± 20.75
EGG	**88.55 ± 0.82**	**91.43 ± 0.73**	**1.82 ± 0.21**	**83.11 ± 1.85**	**87.11 ± 1.44**	**2.54 ± 0.45**	**89.45 ± 0.43**	**91.61 ± 0.35**	**29.22 ± 0.43**
With node features									
Linear AE	92.41 ± 0.51	**93.30 ± 0.71**	27.53 ± 0.92	93.86 ± 1.08	94.14 ± 1.17	50.82 ± 0.09	96.26 ± 0.23	96.18 ± 0.26	1313.38 ± 14.31
2-layer GCN AE	91.71 ± 0.89	92.47 ± 0.99	28.13 ± 1.07	89.80 ± 1.20	90.43 ± 1.50	51.42 ± 0.08	96.38 ± 0.23	96.55 ± 0.24	1318.75 ± 15.18
3-layer GCN AE	90.35 ± 0.96	91.53 ± 0.75	28.22 ± 0.54	89.65 ± 1.66	90.69 ± 1.74	51.92 ± 0.05	95.30 ± 0.35	95.81 ± 0.26	1325.29 ± 15.19
EGG	**92.54 ± 0.63**	92.81 ± 0.59	**1.77 ± 0.02**	**96.21 ± 0.59**	**96.58 ± 0.49**	**3.20 ± 0.09**	**97.42 ± 0.13**	**97.01 ± 0.22**	**11.29 ± 0.41**

4 DISCUSSION AND CONCLUSION

Our results suggest several broader conclusions about graph autoencoders. First, the finite-time optimization trajectory can be more relevant for link prediction than the exact optimum of the reconstruction objective: although the quadratic surrogate admits a spectral optimum, solving it directly does not reproduce the predictive performance of finite-time training. Second, the parameterization $Z = SW$ induces a non-trivial optimization geometry through the S^2 preconditioner, showing that even a simple linear encoder affects how representations evolve during training. Finally, these dynamics can be approximated by a deterministic spectral filter, leading to EGG, which requires no gradient-based optimization and provides an explicit interpretation of how different spectral components of the graph are amplified. Besides its computational advantages, this gives EGG a degree of interpretability that is harder to obtain from iteratively trained graph neural networks.

There are several limitations to the present study. Our experiments are restricted to the classical Cora, Citeseer, and Pubmed benchmarks, the derivation relies on a quadratic approximation of BCE and a linearization of the resulting dynamics, and the restriction to the positive spectrum is currently empirically motivated rather than theoretically justified. Moreover, all runtime experiments were performed without CUDA acceleration, so the reported runtimes characterize CPU execution and should not be interpreted as hardware-independent. We also do not claim state-of-the-art performance against substantially more complex modern link-prediction architectures. Future work will therefore consider a broader range of graphs and link-prediction benchmarks, investigate the theoretical role of positive-spectrum truncation, and extend finite-time analytical dynamics to nonlinear graph encoders. It would also be interesting to evaluate EGG representations on other downstream tasks, such as node classification, and more generally to investigate whether the same pipeline—gradient updates, finite-time recurrence, and closed-form approximation—can yield interpretable analytical models in other learning domains.

REFERENCES

[1] Rachith Aiyappa, Xin Wang, Munjung Kim, Ozgur Can Seckin, Yong-Yeol Ahn, and Sadamori Kojaku. 2025. Implicit degree bias in the link prediction task. In *Proceedings of the 42nd International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 267)*, Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (Eds.). PMLR, 874–908. https://proceedings.mlr.press/v267/aiyappa25a.html

[2] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-Relational Data. In *Advances in Neural Information Processing Systems*, Vol. 26. 2787–2795.

[3] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable Feature Learning for Networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Association for Computing Machinery, 855–864.

[4] Glen Jeh and Jennifer Widom. 2003. Scaling Personalized Web Search. In *Proceedings of the 12th International Conference on World Wide Web*. Association for Computing Machinery, 271–279.

[5] Leo Katz. 1953. A New Status Index Derived from Sociometric Analysis. *Psychometrika* 18, 1 (1953), 39–43.

[6] Thomas N. Kipf and Max Welling. 2016. Variational Graph Auto-Encoders. In *NIPS Workshop on Bayesian Deep Learning*.

[7] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations*.

[8] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: Online Learning of Social Representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Association for Computing Machinery, 701–710.

[9] Jiezhong Qiu, Yuxiao Dong, Hao Ma, Jian Li, Kuansan Wang, and Jie Tang. 2018. Network Embedding as Matrix Factorization: Unifying DeepWalk, LINE, PTE, and node2vec. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. Association for Computing Machinery, 459–467.

[10] Guillaume Salha, Romain Hennequin, and Michalis Vazirgiannis. 2021. Simple and Effective Graph Autoencoders with One-Hop Linear Models. In *Machine Learning and Knowledge Discovery in Databases*. Springer International Publishing, Cham, 319–334.

[11] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. LINE: Large-Scale Information Network Embedding. In *Proceedings of the 24th International Conference on World Wide Web*. 1067–1077.

[12] Rianne van den Berg, Thomas N. Kipf, and Max Welling. 2017. Graph Convolutional Matrix Completion. *arXiv preprint arXiv:1706.02263* (2017).

[13] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S. Yu. 2021. A Comprehensive Survey on Graph Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems* 32, 1 (2021), 4–24.

[14] Zhilin Yang, William W. Cohen, and Ruslan Salakhutdinov. 2016. Revisiting Semi-Supervised Learning with Graph Embeddings. In *Proceedings of the 33rd International Conference on Machine Learning*. 40–48.

[15] Marinka Zitnik, Monica Agrawal, and Jure Leskovec. 2018. Modeling Polypharmacy Side Effects with Graph Convolutional Networks. *Bioinformatics* 34, 13 (2018), i457–i466.

 DOI: https://doi.org/10.26493/scores26.14

Public Transport Anchored Flow Termini Coverage Model for Dock Placement in Bicycle Sharing Systems

Dmytro Tupkalenko
University of Primorska
Koper, Slovenia
89252101@student.upr.si

Daniil Baldouski (co-Mentor)
University of Primorska
Koper, Slovenia
daniil.baldouski@iam.upr.si

Miklós Krész (Mentor)
University of Primorska
Koper, Slovenia
miklos.kresz@iam.upr.si

ABSTRACT

Bicycle sharing systems can be planned to serve different roles in a city's public transport: as a standalone network, as a first/last-mile extension of other transit modes, or as some mixture of both. The mixed-integer linear programming formulation formulation for dock location presented in this paper gives planners direct control over these roles through weights assigned to different types of coverage achieved. The formulation is also empirically evaluated to assess how well it achieves the intended behavior.

KEYWORDS

bicycle sharing systems, facility location, public transport integration, mixed-integer linear programming

1 INTRODUCTION

Bicycle sharing systems (BSS) have become an essential part of urban transport planning, but their effectiveness depends strongly on dock (station) locations and how well they integrate with public transport (PT). Depending on the planning objective, BSS may serve as a self-contained network, complement PT for first- and last-mile travel, or fulfill both roles simultaneously. Thus, station placement should account not only for how much demand is covered, but also for which types of trips the network serves.

In this paper, we introduce the **Public Transport Anchored Flow Termini Coverage Model (PT-FTCM)**, a mixed-integer linear programming (MILP) model for BSS station placement that represents demand as origin–destination (OD) flows and integrates BSS–PT interactions into the location decision through a tiered objective. The model evaluates each flow across three coverage tiers: direct, PT-hop, and joint coverage, with tunable weights that allow planners to trade them off and thereby control the resulting network structure. We present the complete formulation and evaluate its behavior using open real-world data from Washington, D.C.

2 RELATED WORK

BSS station location builds on the facility location problem: choosing a subset of candidate sites, under a budget constraint, to maximize served demand — see [1] for a review. We focus on two design dimensions: how demand is represented, and how the network relates to PT.

Demand may be represented through discrete points (residences, workplaces, etc.), each considered covered once an open station falls within its service radius. This traces back to the maximal covering location problem (MCLP) [2], later applied to BSS siting [3, 9]. Alternatively, demand can be represented as origin–destination flows, where a trip counts as covered only if stations are open near both its origin and destination. Lin and Yang [7] were, to our knowledge, the first to apply this idea to BSS, as part of their model for determining station locations, bicycle lanes, and demand routing. Frade and Ribeiro [5] embed flow-based coverage within their system-design model that also accounts for station capacity and rebalancing. Liu et al. [8] later recast this concept into a simpler formulation, the Flow Termini Coverage Model (FTCM), showing that it achieves higher coverage than point-based MCLP.

This flow-based view, however, comes with a practical cost: it requires an OD flow matrix, which is harder to obtain than point demand, as the latter can be derived by aggregating flows. We nevertheless argue that this cost is justified by the more realistic picture it provides. Moreover, the underlying data gap is shrinking as recent work addresses the generation of OD flows, e.g., [6].

A separate branch of the literature addresses BSS–PT integration, generally by folding a transit-related measure into the objective. Wang et al. [10] determine the minimum number of BSS stations needed to satisfy a connectivity-based accessibility condition with the bus network. Lin and Yang [7] describe designing stations to link commuters from transit stops to their destinations. Fan and Harper [4] compute a per-block transit score, then trade it off against demand coverage via a planner-controlled weight.

What is missing is a flow-termini model where a trip's coverage through PT is tracked as its own outcome, distinct from direct BSS coverage, so that a planner can weigh the two against each other directly and not through some proxy — PT-FTCM is such a model.

3 MODEL FORMULATION

3.1 Parameters

Table 1 presents the model parameters in two groups. The first concerns spatial and temporal feasibility. The second consists of the control parameters w_1 and w_2, which determine the relative value of each coverage type introduced in the next subsection.

Table 1: Model parameters.

Param.	Unit / Domain	Description
$R^{\text{walk}}_{\text{bss}}$	(m)	Max. walking radius to/from BSS station
$R^{\text{walk}}_{\text{pt}}$	(m)	Max. walking radius from BSS station to PT stop
$R^{\text{ride}}_{\text{min}}$	(m)	Min. cycling radius for any cycling trip
$R^{\text{ride}}_{\text{dir}}$	(m)	Max. cycling radius for direct BSS-to-BSS trip
$R^{\text{ride}}_{\text{pt}}$	(m)	Max. cycling radius to PT-adjacent BSS station
$D^{\text{OD}}_{\text{min}}$	(m)	Min. origin–destination distance for an OD pair
$T^{\text{pt}}_{\text{max}}$	(min)	Max. in-vehicle PT time between two stops
α	$[0, 1]$	Proportion of active candidate stations to open
$w_1,\ w_2$	$(w_2 \geq w_1 \geq 1)$	**Control:** tier coverage weights

3.2 Coverage Tiers

For a given OD flow q with o_q and d_q being the origin and destination, respectively, we define the tier t coverage as:

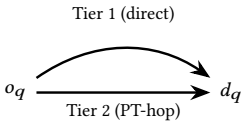
- **Tier 1 — Direct BSS coverage:**

$$o_q \xrightarrow{R^{\text{walk}}_{\text{bss}}} j \xrightarrow{[R^{\text{ride}}_{\text{min}}, R^{\text{ride}}_{\text{dir}}]} m \xrightarrow{R^{\text{walk}}_{\text{bss}}} d_q$$

  A trip is directly covered if the user walks to an open BSS station j, cycles to an open BSS station m within the permitted cycling distance interval, and walks to d_q.
- **Tier 2 — PT-hop coverage:**

$$o_q \xrightarrow{R^{\text{walk}}_{\text{bss}}} j' \xrightarrow{[R^{\text{ride}}_{\text{min}}, R^{\text{ride}}_{\text{pt}}]} i \xrightarrow{R^{\text{walk}}_{\text{pt}}} k \xrightarrow{\text{PT}} k' \xrightarrow{R^{\text{walk}}_{\text{bss}}} d_q$$

  A trip achieves PT-hop coverage if the user walks to an open BSS origin station j', cycles to an open PT-adjacent BSS station i, walks to a PT station k, rides PT transit to k', and walks to d_q.
- **Tier 3 — Joint coverage:**



  A trip is jointly covered if it satisfies both Tier 1 and Tier 2.

3.3 Input and Derived Sets

- $\bar{J}$ — initial set of candidate BSS station locations j.
- K — set of existing PT stops k.
- $\bar{Q}$ — initial set of OD trip pairs q:

$$\bar{Q} = \left\{q = (o_q, d_q) \mid F_q > 0,\ d(o_q, d_q) \geq D^{\text{OD}}_{\text{min}}\right\}$$

Feasibility structures:

- $N^o_q,\ N^d_q \subseteq \bar{J}$ — stations within $R^{\text{walk}}_{\text{bss}}$ of q's origin/destination:

$$N^{[o/d]}_q = \left\{j \in \bar{J} \mid d(j, [o_q/d_q]) \leq R^{\text{walk}}_{\text{bss}}\right\}$$

- P^{dir}_q — feasible direct station pairs for trip q:

$$P^{\text{dir}}_q = \left\{(j,m) \mid j \in N^o_q,\ m \in N^d_q,\ R^{\text{ride}}_{\text{min}} \leq d(j,m) \leq R^{\text{ride}}_{\text{dir}}\right\}$$

- $C_{kk'} \in \{0,1\}$ — 1 if k' is reachable from k within $T^{\text{pt}}_{\text{max}}$ via PT.
- $M_{q,j} \subseteq \bar{J}$ — stations cyclable from j that connect to d_q via PT:

$$M_{q,j} = \Big\{i \in \bar{J} \mid\ R^{\text{ride}}_{\text{min}} \leq d(i,j) \leq R^{\text{ride}}_{\text{pt}}\ \wedge\ \exists k, k' \in K :\ d(i,k) \leq R^{\text{walk}}_{\text{pt}}\ \wedge\ C_{kk'} = 1\ \wedge\ d(k', d_q) \leq R^{\text{walk}}_{\text{bss}}\Big\}$$

- $S^{\text{pt}}_q \subseteq N^o_q$ — origin stations with nonempty intermediaries:

$$S^{\text{pt}}_q = \left\{j \in N^o_q \mid M_{q,j} \neq \emptyset\right\}$$

Active sets used by the model:

- $Q \subseteq \bar{Q},\ J \subseteq \bar{J}$ — active OD trip pairs and active candidate stations appearing in at least one coverage structure:

$$Q = \left\{q \in \bar{Q} \mid P^{\text{dir}}_q \neq \emptyset\ \vee\ S^{\text{pt}}_q \neq \emptyset\right\}$$

$$J = \bigcup_{q \in Q} \bigcup_{(j,m) \in P^{\text{dir}}_q} \{j, m\} \cup \bigcup_{q \in Q} S^{\text{pt}}_q \cup \bigcup_{q \in Q} \bigcup_{j \in S^{\text{pt}}_q} M_{q,j}$$

- $E^1_q,\ E^2_q$ — station-edge sets representing feasible pairs:

$$E^1_q = \left\{\{j,m\} \mid (j,m) \in P^{\text{dir}}_q\right\},\ E^2_q = \left\{\{j,i\} \mid j \in S^{\text{pt}}_q,\ i \in M_{q,j}\right\}$$

- $\sigma_1(q),\ \sigma_2(q),\ \sigma_3(q)$ — coverage signatures of OD pair q:

$$\sigma_1(q) = E^1_q, \qquad \sigma_2(q) = E^2_q, \qquad \sigma_3(q) = (E^1_q, E^2_q)$$

- Q_1, Q_2, Q_3 — sets of unique representative OD pairs, one per distinct Tier 1, Tier 2, and Tier 3 signature, where a pair is Tier 1 (resp. Tier 2) feasible iff $E^1_q \neq \emptyset$ (resp. $E^2_q \neq \emptyset$), and Tier 3 feasible iff both hold. The maps $\rho_t(q) \in Q_t$ ($t \in \{1,2,3\}$) and $\pi_1(r) \in Q_1$, $\pi_2(r) \in Q_2$ ($r \in Q_3$) are defined only where the corresponding tier is feasible. $E^t_r \coloneqq E^t_q$ for any q with $\rho_t(q) = r$
- F^1_r, F^2_r, F^3_r — aggregated flows of representative r:

$$F^1_r = \sum_{\substack{q \in Q \\ \rho_1(q) = r}} F_q, \qquad F^2_r = \sum_{\substack{q \in Q \\ \rho_2(q) = r}} F_q, \qquad F^3_r = \sum_{\substack{q \in Q \\ \rho_3(q) = r}} F_q$$

- $C^1_r,\ C^2_r$ — families of all inclusion-minimal vertex covers for edge sets E^1_r and E^2_r. Each $C \in C^t_r$ satisfies:

$$C \cap e \neq \emptyset \qquad \forall e \in E^t_r$$

Lemma 3.1 (Cover certificate). *Let E be a set of unordered pairs over a vertex set V, let $\mathcal{C}$ be the family of all inclusion-minimal vertex covers of E, and let $X \subseteq V$. Then X contains both endpoints of at least one edge of E if and only if $X \cap C \neq \emptyset$ for every $C \in \mathcal{C}$.*

Proof. No edge of E is fully contained in X if and only if X is an independent set, which holds if and only if every edge has an endpoint outside X, or equivalently, if and only if $V \setminus X$ is a vertex cover of E. Because any vertex cover contains an inclusion-minimal one, $V \setminus X$ is a cover if and only if some $C \in \mathcal{C}$ satisfies $C \subseteq V \setminus X$, meaning $C \cap X = \emptyset$. Negating both sides yields the claim. □

3.4 Decision Variables

- $x_j \in \{0,1\}$: 1 if station $j \in J$ is opened.
- $y^t_r \in [0,1],\ \forall t \in \{1,2,3\},\ \forall r \in Q_t$: 1 if r achieves Tier t.

3.5 Model

$$\max \sum_{r \in Q_1} F^1_r y^1_r + (w_1 - 1) \sum_{r \in Q_2} F^2_r y^2_r + (w_2 - w_1) \sum_{r \in Q_3} F^3_r y^3_r$$

Budget:

$$\sum_{j \in J} x_j = \lfloor \alpha \cdot |J| \rfloor \tag{1}$$

Coverage:

$$y^1_r \leq \sum_{j \in C} x_j \qquad \forall r \in Q_1,\ \forall C \in C^1_r \tag{2}$$

$$y^2_r \leq \sum_{j \in C} x_j \qquad \forall r \in Q_2,\ \forall C \in C^2_r \tag{3}$$

$$y^3_r \leq y^1_{\pi_1(r)} \qquad \forall r \in Q_3 \tag{4}$$

$$y^3_r \leq y^2_{\pi_2(r)} \qquad \forall r \in Q_3 \tag{5}$$

Note: The variables y^1_r, y^2_r, y^3_r are continuous relaxations. Their objective coefficients are nonnegative (Table 1) and strictly positive exactly when the corresponding tier is valued ($w_1 > 1$ for Tier 2, $w_2 > w_1$ for Tier 3); only then are they forced to be binary at optimality. With $X = \{j \in J \mid x_j = 1\}$ and $E = E^t_r$, Lemma 3.1 states that r attains Tier t exactly when the opened stations hit every cover in C^t_r, as enforced by (2) and (3).

3.6 Policy Weight Configurations

Table 2 lists the possible model configurations. Each row reports the objective value of one unit of flow achieving that coverage pattern at optimality; a jointly covered trip collects 1, $w_1 - 1$, and $w_2 - w_1$ from the three tiers, totaling w_2.

Table 2: Policy configurations and per-trip values.

Case	(w_1, w_2)	Direct	PT-hop	Joint
1. Direct-Only	1, 1	1	0	1
2. Discounted PT-hop	w, w, $w \in (1, 2)$	1	$w-1$	w
3. Equal Weighted	2, 2	1	1	2
4. Prioritized PT-hop	w, w, $w > 2$	1	$w-1$	w
5. Full Tiered	$w_2 > w_1 \geq 1$	1	$w_1 - 1$	w_2

4 MODEL EVALUATION

Since a controlled experiment is infeasible (only one configuration can be deployed in an ever-changing city), we evaluate observationally, examining *parameter responsiveness*: whether observed-trip metrics move in the expected direction as w_1 and w_2 vary.

4.1 Setup

Instance. We evaluate the model on data from Washington, D.C., using the sources in Table 3. Spatial and temporal parameters are fixed at $R^{\text{walk}}_{\text{bss}} = 300\,\text{m}$, $R^{\text{walk}}_{\text{pt}} = 30\,\text{m}$, $R^{\text{ride}}_{\text{min}} = 300\,\text{m}$, $R^{\text{ride}}_{\text{dir}} = 5\,\text{km}$, $R^{\text{ride}}_{\text{pt}} = 2\,\text{km}$, $D^{\text{OD}}_{\text{min}} = 1\,\text{km}$, $T^{\text{pt}}_{\text{max}} = 20\,\text{min}$, and the budget parameter is $\alpha = 0.15$. Stations, transit stops, and both endpoints of every origin–destination pair are restricted to the city boundary (2023 TIGER/Line place shapefile[1]), and all distances are computed as great-circle (haversine) distances. Trip records with missing station identifiers or coordinates are discarded, transit stops sharing a parent station are merged, and $C_{kk'}$ links stops served by the same scheduled trip within $T^{\text{pt}}_{\text{max}}$ without transfers. After feasibility pruning, 371 candidate stations and 71,873 of the 167,039 origin–destination pairs remain active. These aggregate into 47,975, 10,230, and 27,969 representatives for Tiers 1, 2 and 3, respectively.

Table 3: Data sources.

Item	Source	Detail
J	Cap. Bikeshare[2]	Active stations, June 2023
K	WMATA GTFS[3]	Metrorail + bus stops
Q, F_q	LODES 8.1[4]	Home-to-work flows for 2023
$\mathcal{T}$	Cap. Bikeshare	Observed trip records, June 2023

[1]U.S. Census Bureau, *2023 TIGER/Line Shapefiles: Place boundaries, District of Columbia.* https://www2.census.gov/geo/tiger/TIGER2023/PLACE/tl_2023_11_place.zip. Accessed 20 July 2026.
[2]Capital Bikeshare, *System data: June 2023 trip history.* https://s3.amazonaws.com/capitalbikeshare-data/index.html. Accessed 20 July 2026.
[3]Washington Metropolitan Area Transit Authority, *General Transit Feed Specification (GTFS) static feed*, 2023, via https://catalog.data.gov/dataset/2023-ntd-annual-data-general-transit-feed-specification-gtfs-weblinks. Accessed 20 July 2026.
[4]U.S. Census Bureau, *LEHD Origin–Destination Employment Statistics (LODES), Version 8.1*, District of Columbia, 2023. https://lehd.ces.census.gov/data/lodes/LODES8/dc/. Accessed 20 July 2026.

Procedure. For each considered configuration of (w_1, w_2), the model is solved on J, K, Q, F_q to obtain an optimal station set S^*. The resulting solution is then evaluated using the metrics defined in Section 4.2 on the trip log $\mathcal{T}$ restricted to S^*, yielding an out-of-sample validation structure: the optimizer sees only external demand (not derived from BSS trip data), while performance is measured on actual cycling data from the log. This separation, however, is not complete: the candidate station set J that the model optimizes over must coincide with the stations appearing in the historical trip log, since observed trip behavior exists only for these stations. Thus, the evaluation should be understood as testing PT-FTCM as a reallocation model over an already-deployed, presumably reasonably well-optimized network, rather than as a siting exercise over an unconstrained candidate pool. Given this, the experiments are designed to observe whether the model's behavior changes in the expected direction across policy configurations, not to establish that the resulting station sets outperform the existing network in absolute terms.

Computational Environment. The model is implemented in Python and solved with **Gurobi 13.0.2**, using a MIP gap of 0.1%. Everything was run on a MacBook Air with an Apple M5 chip and 16 GB of unified memory. Solve times per weight configuration ranged from 7 to 89 minutes (36 minutes avg.), for a total run of about 19 hours.

4.2 Evaluation Metrics

Let $\mathcal{T}$ be the trip log restricted to candidate stations J, $\mathcal{T}(S^*) = \{t \in \mathcal{T} \mid o_t, d_t \in S^*\}$ the trips covered by the open set S^*, and $\mathcal{T}_{\text{pt}}(S^*) \subseteq \mathcal{T}(S^*)$ those with at least one endpoint within $R^{\text{walk}}_{\text{pt}}$ of a PT stop. We report:

$$\eta_{\text{cov}} = \frac{|\mathcal{T}(S^*)|}{|\mathcal{T}|}, \qquad \eta^{\text{rel}}_{\text{pt}} = \frac{|\mathcal{T}_{\text{pt}}(S^*)|}{|\mathcal{T}(S^*)|},$$

$$\eta^{\text{abs}}_{\text{pt}} = \eta_{\text{cov}} \cdot \eta^{\text{rel}}_{\text{pt}} = \frac{|\mathcal{T}_{\text{pt}}(S^*)|}{|\mathcal{T}|}, \quad \eta_{\text{direct}} = \eta_{\text{cov}} - \eta^{\text{abs}}_{\text{pt}}.$$

Here η_{cov} is overall network coverage, $\eta^{\text{rel}}_{\text{pt}}$ is the relative PT-adjacent share among covered trips, and $\eta^{\text{abs}}_{\text{pt}}$ and η_{direct} split η_{cov} into PT-adjacent and non-PT-adjacent trips.

4.3 Experiments and Results

Equal-weight experiment. The first experiment varies a common weight $w = w_1 = w_2$ over $\mathcal{G}_{\text{equal}} = \{(w, w) \mid w \in \{1.0, 1.2, \ldots, 2.8\}\}$, covering Case 1 ($w = 1$), Case 2 ($1 < w < 2$), Case 3 ($w = 2$), and Case 4 ($w > 2$) in turn. Since $w_2 - w_1 = 0$, the Tier 3 coefficient vanishes entirely, isolating the direct/PT-hop trade-off: as w grows, we expect $\eta^{\text{rel}}_{\text{pt}}$ to rise and η_{direct} to fall accordingly.

Joint-weight experiment. The second experiment fixes $w_1 \in \{1.0, 1.5, 2.0, 2.5\}$ and varies $w_2 > w_1$ over $\mathcal{G}_{\text{joint}} = \{(w_1, w_2) \mid w_1 \in \{1.0, 1.5, 2.0, 2.5\},\ w_2 \in \{1.2, 1.4, \ldots, 2.8\},\ w_2 > w_1\}$, covering Case 5 throughout. Since $w_2 - w_1 > 0$, the Tier 3 coefficient is active, isolating the reward for joint coverage specifically. As w_2 grows, joint coverage is increasingly valued; since it also requires a direct coverage, we expect $\eta^{\text{abs}}_{\text{pt}}$ to rise without any corresponding loss in η_{direct}.

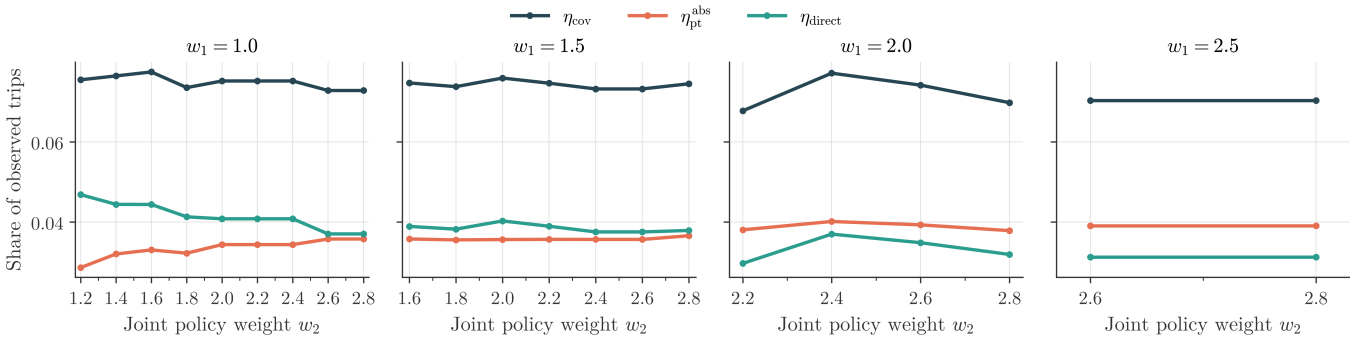


Figure 1: Joint-weight experiment.

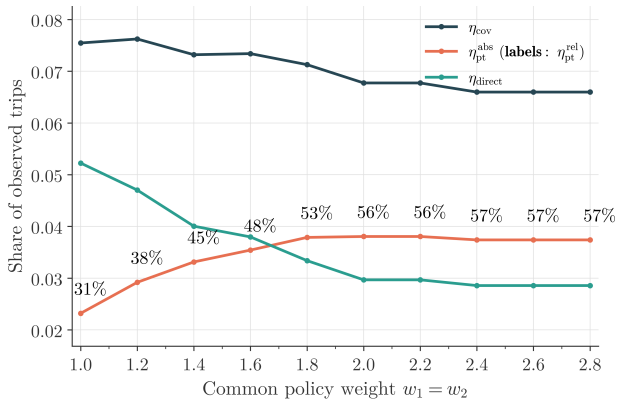


Figure 2: Equal-weight experiment.

4.4 Results Discussion

The equal-weight experiment (Fig. 2) matches expectations: as w rises from 1.0 to 2.8, $\eta_{\text{pt}}^{\text{rel}}$ climbs monotonically from 0.31 to 0.57 while η_{direct} falls from 0.052 to 0.029, confirming the intended trade-off between direct and PT-hop coverage.

The joint-weight experiment (Fig. 1) is less definitive: only at $w_1 = 1.0$ does $\eta_{\text{pt}}^{\text{abs}}$ rise with w_2 as expected, but even there η_{direct} declines alongside it rather than holding stable; for $w_1 \geq 1.5$ both metrics are largely flat or non-monotonic. We attribute this weak local responsiveness to the instance — likely the PT-associated and direct-coverage station sets barely overlap here, limiting how the joint bonus can be realized — though confirming this needs further experiments across multiple cities. Still, the joint-weight grid shows a global result the equal-weight configurations do not: the highest $\eta_{\text{pt}}^{\text{abs}}$ across all 32 configurations occurs at a joint point, $(w_1, w_2) = (2.0, 2.4)$ with a value of 0.0401, exceeding every equal-weight configuration, including the most PT-favoring one. Decoupling w_1 and w_2 thus reaches trade-offs a single weight cannot — corresponding to one of the goals of the third objective component.

5 CONCLUSION

This paper presented PT-FTCM, a MILP model for BSS facility location that integrates PT connectivity through a tiered, policy-parameterized objective — paper's primary contribution. Experiments give preliminary partial support for the model's behavior. Results, however, rely on a single city and we thus treat them as an initial signal rather than a conclusion; further experiments are needed before stronger claims can be made. The code is available[5].

ACKNOWLEDGMENTS

This work has been supported by the European Commission's Horizon Europe Research and Innovation programme through the Marie Sklodowska-Curie Actions Staff Exchanges (MSCA-SE) under Grant Agreement no. 101182819 (COVER: (C)ombinatorial (O)ptimization for (V)ersatile Applications to (E)merging u(R)ban Problems). The authors also acknowledge the support of the Slovenian Research and Innovation Agency (ARIS) through grant J1-50000.

REFERENCES

[1] Mohammad Sadegh Bahadori, Alexandre Gonçalves, and Filipe Moura. 2021. A Systematic Review of Station Location Techniques for Bicycle-Sharing Systems Planning and Operation. *ISPRS International Journal of Geo-Information* 10 (08 2021), 554. https://doi.org/10.3390/ijgi10080554

[2] RICHARD CHURCH and CHARLES R. VELLE. 1974. THE MAXIMAL COVERING LOCATION PROBLEM. *Papers in Regional Science* 32, 1 (1974), 101–118. https://doi.org/10.1111/j.1435-5597.1974.tb00902.x

[3] Lindsey Conrow, Alan Murray, and Heather Fischer. 2018. An optimization approach for equitable bicycle share station siting. *Journal of Transport Geography* 69 (05 2018), 163–170. https://doi.org/10.1016/j.jtrangeo.2018.04.023

[4] Zhufeng Fan and Corey Harper. 2024. Taking a multimodal approach to equitable bike share station siting. *Journal of Transport Geography* 115 (02 2024), 103814. https://doi.org/10.1016/j.jtrangeo.2024.103814

[5] Inês Frade and Anabela Ribeiro. 2015. Bike-sharing stations: A maximal covering location approach. *Transportation Research Part A General* 82 (12 2015), 216–227. https://doi.org/10.1016/j.tra.2015.09.014

[6] Yuebing Liang, Zhan Zhao, and Chris Webster. 2024. Generating sparse origin–destination flows on shared mobility networks using probabilistic graph neural networks. *Sustainable Cities and Society* 114 (08 2024), 105777. https://doi.org/10.1016/j.scs.2024.105777

[7] Jenn-Rong Lin and Ta-Hui Yang. 2011. Strategic design of public bicycle sharing systems with service level constraints. *Transportation Research Part E: Logistics and Transportation Review* 47 (03 2011), 284–294. https://doi.org/10.1016/j.tre.2010.09.004

[8] Hung-Chi Liu, Daoqin Tong, and Michael Kuby. 2024. The Flow Termini Coverage Model for Locating Bike-Sharing Stations. *Annals of the American Association of Geographers* 114, 8 (2024), 1668–1680. https://doi.org/10.1080/24694452.2024.2313495

[9] Suleyman Mete, Zeynel Abidin Çil, and Eren Özceylan. 2018. Location and Coverage Analysis of Bike- Sharing Stations in University Campus. *Business Systems Research Journal* 9 (07 2018), 80–95. https://doi.org/10.2478/bsrj-2018-0021

[10] L. Wang, C. Li, M. Z. Q. Chen, Q.-G. Wang, and F. Tao. 2018. Connectivity-based accessibility for public bicycle sharing systems. *IEEE Transactions on Automation Science and Engineering* 15, 4 (2018), 1521–1532. https://doi.org/10.1109/TASE.2018.2868471

[5] https://github.com/tupkalenkodi/PT-FTCM

 DOI: https://doi.org/10.26493/scores26.17

Two-Step Relaxation for All-Pairs Shortest Paths in Sparse Graphs

Dani Zugan
dani.zugan@fri.uni-lj.si
University of Ljubljana,
Faculty of Computer and Information Science
Večna pot 113
SI-1000 Ljubljana, Slovenia

ABSTRACT

The all-pairs shortest path problem is a fundamental problem in graph algorithms, with applications in telecommunications, logistics, and network analysis. In this paper, we study this problem in directed graphs and introduce a new relaxation based approach. The method decomposes the graph into strongly connected components, computes shortest paths within each component, and then determines shortest paths between different components. Unlike earlier component based methods that propagate distances directly through outgoing edges, the proposed approach uses a two-step initial-terminal relaxation scheme. Here, an initial vertex is the starting vertex of an outgoing edge of a component, and a terminal vertex is its ending vertex outside that component. In the first step, distances are propagated through these vertices; in the second step, the resulting values are extended through subsequent components. We prove that this decomposition correctly computes inter-component shortest path distances once shortest paths within the components are known. We also derive the running-time bound of the algorithm. In the worst case, it matches the cubic complexity of the classical Floyd–Warshall algorithm, while for graphs with small sets of initial and terminal vertices it can substantially reduce the number of required relaxations.

KEYWORDS

all-pairs shortest path, strongly connected components, Floyd-Warshall algorithm.

1 INTRODUCTION

Many real-world problems in bioinformatics, logistics, telecommunications, and related domains are modeled as graphs, where the main task is to find shortest paths between pairs of nodes [1]. The shortest path problem appears in two variants: finding shortest paths from a single source to all other nodes (single-source shortest path, SSSP), and finding shortest paths for all node pairs (all-pairs shortest path, APSP).

Two conceptually different strategies exist for APSP. The first reduces APSP to repeated SSSP computations—running an SSSP algorithm from every node and aggregating the results. The second constructs shortest paths for all pairs simultaneously. For graphs with non-negative edge weights, the repeated SSSP strategy uses Dijkstra-type algorithms. A classical representative of the simultaneous strategy is the Floyd–Warshall algorithm [7, 18], based on dynamic programming.

Despite their different viewpoints, both strategies rely on the same operation: *relaxation*. Since the overall running time depends on the number of relaxation attempts, repeated SSSP approaches tend to be preferable on sparse graphs, whereas Floyd–Warshall-like methods are well-suited for dense graphs.

The component-based framework used in this paper originates from [19]. In that work, the graph is first decomposed into strongly connected components (SCCs), and APSP is solved independently within each SCC, which is possible because all vertex pairs inside an SCC are mutually reachable.

The remaining task is to compute shortest paths between vertices that belong to different components. In [19], this is done by propagating distances across components through outgoing edges, using a reachability observation: if an arbitrarily chosen vertex in a source component cannot reach an arbitrarily chosen vertex in a destination component, then relaxations between these two components can be omitted. Our work adopts the same SCC decomposition and the same reachability viewpoint, but replaces the edge-based inter-component propagation of [19] with a computation based on classical Floyd–Warshall relaxations.

The goal of this paper is to introduce an SCC-guided approach to the APSP problem, where relaxations are performed only over relevant vertex triples.

2 RELATED WORK

The APSP problem has been studied extensively in the graph algorithms literature. The classical dynamic-programming solution is the Floyd–Warshall algorithm [7, 18], which runs in $O(n^3)$ time. A different approach is to solve APSP through repeated single-source shortest path computations. For graphs with non-negative edge weights, this is typically done with Dijkstra's algorithm [6], while Johnson's algorithm [10] extends this approach to graphs without negative cycles.

Several improvements of Floyd–Warshall aim to reduce the number of unsuccessful relaxations. In [3], the authors introduce the Tree and Hourglass data structures and obtain an expected running time of $O(n^2 \log^2 n)$ for complete directed graphs with edge weights drawn uniformly from $[0, 1]$. Similarly, [12] proposes the Smart-Force technique and a hybrid variant that processes only selected relaxations likely to improve distances.

Other work exploits sparsity or structural restrictions. The Rectangular Algorithm [2] skips inactive regions of the distance matrix, while [17] improves Floyd–Warshall on disconnected and very sparse graphs by maintaining reachability information. For sparse directed graphs, Pettie [13] improves the worst-case bound over Johnson's algorithm. A particularly relevant direction for our setting is APSP on graphs of low treewidth. For a vertex ordering d, Planken, de Weerdt, and van der Krogt [14] propose DPC-based APSP algorithms running in $O(n^2 w_d)$ time, where w_d denotes the

graph width induced by d. This results in $O(n^2)$ time for graphs of constant treewidth and $O(nm)$ time for chordal graphs.

A closely related line of work is based on strongly connected components (SCCs). SCC decomposition can be computed in linear time by Tarjan's algorithm [15], and its structural depth-first-search foundations are revisited in [16]. Building on this idea, [19] first solves shortest paths inside each component and then propagates distances between components by relaxing via outgoing edges from components.

For sparse graphs, repeated single-source methods remain a strong baseline. Their performance depends heavily on the priority queue used inside Dijkstra's algorithm. Fibonacci heaps give the classical theoretical bound [9], while pairing heaps are often competitive in practice [8].

Finally, for complete directed graphs with independent edge weights drawn uniformly from $[0, 1]$, sharper expected bounds are known. The Hidden Paths algorithm [11] achieves expected running time $O(n^2 \log n)$, and this bound was later matched by Demetrescu and Italiano, who also provided an experimental analysis of APSP algorithms [4, 5].

3 PRELIMINARIES

A *directed graph* is denoted by $G = (V, E)$, where V is a finite, nonempty set of vertices and $E \subseteq V \times V$ is a set of directed edges. We write $V = \{v_1, \ldots, v_n\}$.

An *edge* $e = (u, v) \in E$ is directed *from* u *to* v. The vertex u is called the *initial vertex* of e, denoted by $\mathrm{ini}(e) = u$, while v is called the *terminal vertex* of e, denoted by $\mathrm{ter}(e) = v$.

A *path* in G is a sequence of pairwise distinct vertices $P = (v_1, \ldots, v_k)$ such that $(v_i, v_{i+1}) \in E$ for all $1 \le i < k$. We say that v is *reachable* from u if there exists a path starting at u and ending at v.

If $P = (v_1, \ldots, v_k)$ is a path and $1 \le a \le b \le k$, then $(v_a, \ldots, v_b)$ is called a *subpath* of P. We denote this subpath by $v_a \rightsquigarrow v_b$.

A graph $G' = (V', E')$ is a *subgraph* of G if $V' \subseteq V$ and $E' \subseteq E$.

A *strongly connected component (SCC)* of a directed graph $G = (V, E)$ is a maximal set of vertices $C \subseteq V$ such that every two vertices in C are mutually reachable.

A directed graph can be partitioned into t strongly connected components $C_1, C_2, \ldots, C_t$, ordered topologically so that there are no edges from C_j to C_i for $j > i$. Further, we denote the union $C_i \cup C_{i+1} \cup \cdots \cup C_t$ by $C^{(i\ldots t)}$.

Given an SCC $C_i \subseteq V$, the set of *outgoing edges* from C_i is defined as $\mathrm{Out}(C_i) = \{(u, v) \in E \mid u \in C_i,\ v \notin C_i\}$ and the set of *incoming edges* to C_i is defined as $In(C_i) = \{(u, v) \in E \mid u \notin C_i,\ v \in C_i\}$.

For an SCC C_i, we denote by $I_i = \{ini(e) \mid e \in \mathrm{Out}(C_i)\}$ the set of *initial vertices* of outgoing edges, and by $T_i = \{ter(e) \mid e \in \mathrm{Out}(C_i)\}$ the set of *terminal vertices* of outgoing edges.

Note that distinct edges in $\mathrm{Out}(C_i)$ may share the same initial vertex or the same terminal vertex. Thus, multiple outgoing edges may originate at a single vertex in I_i, and multiple outgoing edges may enter the same vertex in T_i.

For vertex $v \in C^{(i+1\ldots t)}$ we define $\mathrm{In}_i(v) = \{e \in \mathrm{Out}(C_i) \mid \mathrm{ter}(e) = v\}$.

A *weighted graph* is a directed graph $G = (V, E)$ with a *weight function* $w\colon E \to \mathbb{R}$ that assigns each directed edge $e \in E$ a weight $w(e)$. Any such weight function extends to a path $P = (v_1, v_2, \ldots, v_k)$ by $w(P) = \sum_{i=1}^{k-1} w(v_i, v_{i+1})$.

A *shortest path* from s to d is a path of minimum weight among all paths from s to d. The *distance* between two vertices s and d, denoted by $D_{s,d}$, is the weight of a shortest path from s to d in G.

For a component C_i, we write $D^{(i)}_{u,v}$ for the distance from u to v restricted to paths whose vertices lie entirely in C_i. Similarly, $D^{(i\ldots t)}_{u,v}$ denotes the distance from u to v restricted to paths whose vertices lie in $C^{(i\ldots t)}$.

For simplicity, we will refer to directed graphs simply as *graphs* throughout the rest of this paper.

4 OUR SOLUTION

We propose an alternative algorithm for computing all-pairs shortest paths in graphs, building directly on the SCC-based solution of [19]. The algorithm of [19] presented in Algorithm 1 computes inter-component shortest paths by iterating over all edges in $\mathrm{Out}(C_i)$ directly. This is efficient when the number of outgoing edges is small, but the approach becomes costly when $|\mathrm{Out}(C_i)|$ is large: in the worst case, every vertex in C_i has an outgoing edge to every vertex in the next component, causing the number of relaxation attempts to grow as $O(|C_i|^2)$ per component pair, which can push the overall complexity above $O(n^3)$.

```
1  for i = 1 to t do
2  |  (D^(i)_{s,d})_{s,d∈C_i} ← APSP(C_i)
3  end
4  for i = t − 1 to 1 do
5  |  for each node d ∈ C^(i+1...t) do
6  |  |  for each node s ∈ C_i do
7  |  |  |  D^(i...t)_{s,d} ← ∞;
8  |  |  |  for each edge e ∈ Out(C_i) do
9  |  |  |  |  D^(i...t)_{s,d} ←
   |  |  |  |    min(D^(i...t)_{s,d}, D^(i)_{s,ini(e)} + w(e) + D^(i+1...t)_{ter(e),d})
10 |  |  |  end
11 |  |  end
12 |  end
13 end
```

Algorithm 1: APSP algorithm in [19]

Our solution modifies line 9 of Algorithm 1. Instead of relaxing through each edge $e \in \mathrm{Out}(C_i)$ in a single step as $D^{(i)}_{s,\mathrm{ini}(e)} + w(e) + D^{(i+1\ldots t)}_{\mathrm{ter}(e),d}$, we decompose this into two vertex relaxations: first through $\mathrm{ini}(e) \in I_i$, which contains all initial vertices of $\mathrm{Out}(C_i)$, then through $\mathrm{ter}(e) \in T_i$, which contains all terminal vertices of $\mathrm{Out}(C_i)$. This reduces the inter-component work when $|I_i|$ and $|T_i|$ are small relative to $|\mathrm{Out}(C_i)|$.

4.1 Reducing the Floyd–Warshall Relaxation

The roles of I_i and T_i are structurally different. Initial vertices route paths from vertices in C_i to the vertices T_i (bridge step), terminal vertices then propagate those paths into $C^{(i+1\ldots t)}$ (reduced-relaxation step).

In the notation below, the left subscripts I and T indicate relaxation through the initial vertex set I_i and the terminal vertex set T_i, respectively. The superscript $(i)+$ denotes a distance obtained by following a path within C_i and then one outgoing edge from C_i.

Given distances $D^{(i)}_{s,d}$ and $D^{(i+1\ldots t)}_{s,d}$ between vertex pairs within C_i and within $C^{(i+1\ldots t)}$, respectively, we propose the following two-step decomposition:

Bridge step. For each $s \in C_i$ and $h \in T_i$, let

$$ {}_I D^{(i)+}_{s,h} = \min_{e \in \mathrm{In}_i(h)} \left(D^{(i)}_{s,\mathrm{ini}(e)} + w(e) \right). $$

Reduced-relaxation step. For each $s \in C_i$ and $d \in C^{(i+1\ldots t)}$, let

$$ {}_T D^{(i\ldots t)}_{s,d} = \min_{h \in T_i} \left({}_I D^{(i)+}_{s,h} + D^{(i+1\ldots t)}_{h,d} \right). $$

Theorem 4.1. *With notation above, for each $i \in \{1, \ldots, t\}$, each $s \in C_i$ and $d \in C^{(i+1\ldots t)}$ we have $D^{(i\ldots t)}_{s,d} = {}_T D^{(i\ldots t)}_{s,d}$.*

Proof. We prove the theorem by contradiction. Assume that for some $s \in C_i$ and some $d \in C^{(i+1\ldots t)}$

$$ D^{(i\ldots t)}_{s,d} < {}_T D^{(i\ldots t)}_{s,d}. $$

Let P be a path from s to d with $w(P) = D^{(i\ldots t)}_{s,d}$. Then:

$$ w(P) = D^{(i\ldots t)}_{s,d} < {}_T D^{(i\ldots t)}_{s,d} = \min_{h \in T_i} \left({}_I D^{(i)+}_{s,h} + D^{(i+1\ldots t)}_{h,d} \right). $$

Since $s \in C_i$ and $d \notin C_i$, the path P must leave component C_i. Therefore, there exists at least one edge $(u, v) \in \mathrm{Out}(C_i)$ along this path. The path P thus decomposes into two subpaths connected by (u, v):

$$ P = s \rightsquigarrow u, v \rightsquigarrow d. $$

Since the subpaths $s \rightsquigarrow u$ and $v \rightsquigarrow d$ are no shorter than the corresponding shortest paths, we have:

$$ w(P) \geq D^{(i)}_{s,u} + w(u, v) + D^{(i+1\ldots t)}_{v,d}. $$

By the definition of ${}_I D^{(i)+}_{s,v}$:

$$ D^{(i)}_{s,u} + w(u, v) \geq \min_{e \in \mathrm{In}_i(h)} \left(D^{(i)}_{s,\mathrm{ini}(e)} + w(e) \right) = {}_I D^{(i)+}_{s,v}, $$

and thus $w(P) \geq {}_I D^{(i)+}_{s,v} + D^{(i+1\ldots t)}_{v,d}$. Moreover, by the definition of ${}_T D^{(i\ldots t)}_{s,d}$:

$$ D^{(i\ldots t)}_{s,d} = w(P) \geq {}_I D^{(i)+}_{s,v} + D^{(i+1\ldots t)}_{v,d} \geq \min_{h \in T_i} \left({}_I D^{(i)+}_{s,h} + D^{(i+1\ldots t)}_{h,d} \right) = {}_T D^{(i\ldots t)}_{s,d}, $$

a contradiction. □

4.2 Algorithm

Algorithm 2 implements the above two-step relaxation for all components, processing them in reverse topological order so that distances within $C^{(i+1\ldots t)}$ are available when component C_i is processed.

In the reduced-relaxation step, one could in principle iterate over all vertices in $C^{(i+1\ldots t)}$. However, for a fixed head vertex $h \in T_i$, only vertices reachable from h can lead to a finite value of

$$ {}_I D^{(i)+}_{s,h} + D^{(i+1\ldots t)}_{h,d}, $$

```
1  Tarjan(G);
   /* computes SCCs C_1, ..., C_t and their topological
      order                                              */
2  for i = 1 to t do
3  |  Solve APSP inside C_i;
4  end
5  for i = t − 1 to 1 do
6  |  D^(i...t)_{s,d} ← ∞   ∀s ∈ C_i, d ∈ C^(i+1...t);
   |  /* Bridge step                                     */
7  |  for each edge e ∈ Out(C_i) do
8  |  |  for each node s ∈ C_i do
9  |  |  |  D^(i...t)_{s,ter(e)} ←
   |  |  |    min(D^(i...t)_{s,ter(e)}, D^(i)_{s,ini(e)} + w(ini(e), ter(e)));
10 |  |  end
11 |  end
   |  /* Reduced-relaxation step                        */
12 |  for each node h ∈ T_i do
13 |  |  for each node s ∈ C_i do
14 |  |  |  for each node d reachable from h in C^(i+1...t) do
15 |  |  |  |  D^(i...t)_{s,d} ← min(D^(i...t)_{s,d}, D^(i...t)_{s,h} + D^(i+1...t)_{h,d});
16 |  |  |  end
17 |  |  end
18 |  end
19 end
```

Algorithm 2: Our Two-Step relaxation APSP algorithm

and therefore only those vertices need to be considered.

This observation can be combined with the Tree algorithm of [3]. For each head vertex h, the predecessor information obtained during the APSP computation on $C^{(i+1\ldots t)}$ defines a shortest-path tree rooted at h containing the vertices reachable from h. Instead of relaxing all reachable vertices independently, we traverse this tree in depth-first order.

The Tree algorithm provides an additional reduction beyond reachability filtering. If relaxation through h improves the current distance to a vertex d, the traversal continues with the next vertex in DFS order. If the relaxation does not improve $D_{s,d}$, the entire subtree rooted at d can be skipped, since relaxation through h cannot improve the distance to any descendant of d in the shortest-path tree.

Thus, the tree-based reduced-relaxation step avoids both unreachable vertices and relaxation attempts that are guaranteed not to improve the current distances. In the worst case, all reachable vertices are still processed, but when unsuccessful relaxations occur near the roots of large subtrees, a substantial number of relaxation attempts can be avoided.

4.3 Time Complexity

Computing SCCs with Tarjan's algorithm takes $O(|V| + |E|)$ time. Solving APSP within each component C_i takes $O(|C_i|^3)$ time. For the bridge step, the algorithm considers every source $s \in C_i$ and every outgoing edge $(k, h) \in \mathrm{Out}(C_i)$, resulting in $O(|C_i| \cdot |\mathrm{Out}(C_i)|)$

relaxations. The reduced-relaxation step performs at most $O(|T_i| \cdot |C_i| \cdot |C^{(i+1\ldots t)}|)$ relaxations. Incorporating the Tree algorithm does not change this asymptotic bound, since in the worst case all vertices in $C^{(i+1\ldots t)}$ may still need to be processed.

Summing the computations within and between all components, the total running time is

$$O\left(\sum_{i=1}^{t} |C_i|^3 + \sum_{i=1}^{t-1} |C_i| \cdot \left(|\mathrm{Out}(C_i)| + |T_i| \cdot |C^{(i+1\ldots t)}|\right)\right). \quad (1)$$

To see that this is $O(n^3)$ in the worst case, we first note that $\sum_{i=1}^{t} |C_i| = n$. Consequently this gives $\sum_{i=1}^{t} |C_i|^3 \leq O(n^3)$, with this bound tight (up to constants) when $t = 1$, i.e., when G consists of a single SCC and the method degenerates to plain Floyd–Warshall.

For the second sum, each term is bounded using $|T_i| \leq |C_i| \leq n$, $|C^{(i+1\ldots t)}| \leq n$, and $|\mathrm{Out}(C_i)| \leq |C_i| \cdot |C^{(i+1\ldots t)}| \leq n^2$, giving

$$|C_i|\left(|\mathrm{Out}(C_i)| + |T_i| \cdot |C^{(i+1\ldots t)}|\right) = O(|C_i| \cdot n^2).$$

Summing over i and again using $\sum_i |C_i| \leq n$ gives:

$$\sum_{i=1}^{t-1} |C_i|\left(|\mathrm{Out}(C_i)| + |T_i| \cdot |C^{(i+1\ldots t)}|\right) = O(n^3)$$

.

Combining both parts, the total running time is $O(n^3)$ in the worst case, matching the classical Floyd–Warshall algorithm. For strongly disconnected graphs where the components, the sets T_i, and the sets of outgoing edges are small, the number of inter-component relaxations is substantially reduced.

The two orderings of the steps are not fixed: the scheme above first bridges through I_i within C_i and only then propagates through T_i into $C^{(i+1\ldots t)}$, so the reduced-relaxation step is governed by $|T_i|$ and $|C_i|$. By symmetry, the same computation can be carried out with the steps reversed. The propagation into $C^{(i+1\ldots t)}$ is performed first: for each $k \in I_i$ and $d \in C^{(i+1\ldots t)}$, combining every outgoing edge originating at k with the distance $D^{(i+1\ldots t)}_{\mathrm{ter}(e),d}$. The bridging through $D^{(i)}_{s,k}$ is then applied afterwards, for each $s \in C_i$. In this variant shortest paths from each component cost

$$O(|C^{(i+1\ldots t)}| \cdot (|\mathrm{Out}(C_i)| + |C_i| \cdot |I_i|))$$

instead of

$$O(|C_i| \cdot (|\mathrm{Out}(C_i)| + |T_i| \cdot |C^{(i+1\ldots t)}|).$$

Since both orderings compute the same inter-component distances, the smaller of the two sets can always be chosen.

5 CONCLUSIONS

We presented a new SCC-guided approach to the APSP problem. The method builds on the decomposition of the graph into strongly connected components and replaces direct edge-based inside the component propagation with a two-step initial-terminal relaxation scheme. The main result of the paper is a formal proof that this decomposition is correct: once APSP has been solved inside a component and in the subgraph induced by later components, the distances between them can be computed through the proposed two-step relaxation process.

The resulting algorithm preserves the worst-case $O(n^3)$ running time of Floyd–Warshall, while offering a more refined structural view of inter-component computation. In particular, the number of relaxations depends on the sizes of the boundary sets I_i and T_i rather than directly on all outgoing edges. This makes the method effective for strongly disconnected or structurally sparse graphs, with many outgoing edges.

As future work, we plan to complement this theoretical analysis with an empirical evaluation of the proposed algorithm. This includes implementing the two-step relaxation scheme and comparing its running time against the classical Floyd–Warshall algorithm, and other algorithms mentioned in Section 2, across graphs of varying density, SCC structure, and boundary-set sizes. Such experiments would show how closely practical performance tracks the derived worst-case bound, and to what extent the reduction in relaxations produces measurable speedups.

ACKNOWLEDGMENTS

This work was partially supported by ARIS grant P2-0359.

REFERENCES

[1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. 1993. *Network flows: Theory, Algorithms and Applications.* Prentice-Hall, Inc.
[2] A. Aini and A. Salehipour. 2012. Speeding up the Floyd–Warshall algorithm for the cycled shortest path problem. *Applied Mathematics Letters* 25 (2012), 1–5.
[3] Andrej Brodnik, Marko Grgurovič, and Rok Požar. 2021. Modifications of the Floyd–Warshall algorithm with nearly quadratic expected-time. *Ars Mathematica Contemporanea* 22 (2021), 1–22.
[4] C. Demetrescu and G. F. Italiano. 2004. A new approach to dynamic all pairs shortest paths. *J. ACM* 51 (2004), 968–992.
[5] C. Demetrescu and G. F. Italiano. 2006. Experimental analysis of dynamic all pairs shortest path algorithms. *ACM Transactions on Algorithms* 2 (2006), 578–601.
[6] E. W. Dijkstra. 1959. A note on two problems in connexion with graphs. *Numer. Math.* 1 (1959), 269–271.
[7] R. W. Floyd. 1962. Algorithm 97: Shortest path. *Commun. ACM* 5 (1962), 345.
[8] M. L. Fredman, R. Sedgewick, D. D. Sleator, and R. E. Tarjan. 1986. The pairing heap: A new form of self-adjusting heap. *Algorithmica* 1 (1986), 111–129.
[9] M. L. Fredman and R.E. Tarjan. 1987. Fibonacci heaps and their uses in improved network optimization algorithms. *J. ACM* 34 (1987), 596–615. https://doi.org/10.1145/28869.28874
[10] D. B. Johnson. 1977. Efficient algorithms for shortest paths in sparse networks. *J. ACM* 24 (1977), 1–13.
[11] D. Karger, D. Koller, and S. J. Phillips. 1993. Finding the hidden path: time bounds for all-pairs shortest paths. *SIAM J. Comput.* 22 (1993), 1199–1217.
[12] Giuseppe Lancia and Marcello Dalpasso. 2025. Speeding Up Floyd–Warshall's Algorithm to Compute All-Pairs Shortest Paths and the Transitive Closure of a Graph. *Algorithms* 18, 9 (2025), 560.
[13] Seth Pettie. 2004. A new approach to all-pairs shortest paths on real-weighted graphs. *Theoretical Computer Science* 312 (2004), 47–74. https://doi.org/10.1016/S0304-3975(03)00402-X
[14] Léon Planken, Mathijs de Weerdt, and Roman van der Krogt. 2012. Computing all-pairs shortest paths by leveraging low treewidth. *J. Artif. Int. Res.* 43 (Jan. 2012), 353–388.
[15] R. E. Tarjan. 1972. Depth-first search and linear graph algorithms. *SIAM J. Comput.* 1 (1972), 146–160.
[16] R. E. Tarjan and U. Zwick. 2024. Finding strong components using depth-first search. *European Journal of Combinatorics* 119 (2024).
[17] I. H. Toroslu. 2023. The Floyd–Warshall all-pairs shortest paths algorithm for disconnected and very sparse graphs. *Software: Practice and Experience* 53 (2023), 1287–1303.
[18] S. Warshall. 1962. A theorem on boolean matrices. *J. ACM* 9 (1962), 11–12.
[19] Dani Zugan, Rok Požar, and Andrej Brodnik. 2025. Floyd–Warshall Algorithm for Sparse Graphs. *Algorithms* 18, 12 (2025), 766. https://doi.org/10.3390/a18120766

Applied Machine Learning, Computer Vision and Signal Processing

Attack Detection on Graph Neural Networks

Diana Ioana Catana, Caio Simon, Đorđe Stevanović and Pietro Sestito

https://doi.org/10.26493/scores26.10

Deep Learning-based Axle Detection from B-WIM strain signals

David Chonev and Domen Šoberl

https://doi.org/10.26493/scores26.12

Toward Accessible Melanoma Screening on Mobile Devices: A Three-Tier Feature-Fusion CAD Prototype

Vuk Ignjatović, Nedim Šišić and Peter Rogelj

https://doi.org/10.26493/scores26.16

Cepstral Features for EEG Classification: Comparison with Spectral Features

Luka Matič and Peter Rogelj

https://doi.org/10.26493/scores26.19

 DOI: https://doi.org/10.26493/scores26

 DOI: https://doi.org/10.26493/scores26.12

Deep Learning-based Axle Detection from B-WIM strain signals

David Chonev
89221037@student.upr.si
Faculty of Mathematics, Natural Sciences
and Information Technologies
University of Primorska
Glagoljaška 8
SI-6000 Koper, Slovenia

Domen Šoberl
domen.soberl@famnit.upr.si
Faculty of Mathematics, Natural Sciences
and Information Technologies
University of Primorska
Glagoljaška 8
SI-6000 Koper, Slovenia

ABSTRACT

Bridge weigh-in-motion (B-WIM) systems estimate weights of heavy vehicles that drive over a monitored bridge. Sensors positioned below the bridge capture the strain response of the bridge, and a prerequisite for a reliable weight estimation is an accurate detection of individual axles within the signal. Existing practice of detecting axles relies on site-specific hand-tuned signal processing heuristics that can achieve a high detection accuracy, but require manual adaptation to each B-WIM site. In this paper, we address the problem of automating the fine-tuning of axle detectors by proposing and evaluating a machine learning approach. We use a classical signal peak detector as the baseline method, which misses 11.6% of axles in our dataset. The two proposed deep neural network architectures, one based on 1D U-Net and the other on temporal convolutional networks (TCN), almost entirely eliminate missed axles (recall > 0.999) while keeping false axle detection at a similarly low level as the baseline detector (≈ 0.997 precision). Timing precision of the proposed detectors improves by a factor of four in comparison to the baseline (MATE: 0.22–0.23 vs. 0.87). The results are promising for a future on-site evaluation and deployment.

KEYWORDS

weigh-in-motion, B-WIM, axle detection, temporal convolutional network, U-Net, time series

1 INTRODUCTION

Heavy goods vehicles (HGVs) exert a disproportionate impact on road and bridge infrastructure. Overloaded vehicles seriously increase the risks for road users and have severe impacts on the durability of the infrastructure [2]. Enforcing weight regulations continuously and without disrupting traffic requires a scalable monitoring approach.

Bridge weigh-in-motion (B-WIM) systems address this need by instrumenting existing bridges as dynamic scales [5]. Strain transducers installed on the underside of a bridge record its structural response as each vehicle crosses, enabling the estimation of gross vehicle weight (GVW) and individual axle loads. Modern installations increasingly adopt the free-of-axle detector (FAD) approach, in which axle timing is inferred from strain sensors mounted underneath the bridge [9, 11]. This arrangement protects the sensors from direct vehicle impact and conceals them from drivers, thereby reducing the potential influence of the measurement system on driving behavior. Accurate axle detection is essential for reliable vehicle weight estimation: the measured strain is modeled as a superposition of individual axle contributions according to the bridge influence line, and errors in the detected axle configuration can consequently compromise both axle-load and gross-weight estimates [11].

Figure 1 shows an example of a strain-based B-WIM signal used to detect individual axles of a passing HGV. Current B-WIM practice of detecting axles uses site-specific heuristics that require time-consuming manual fine-tuning for each deployed B-WIM site, which do not generalize across vehicle speed ranges and site-specific sensor deployments. The basic idea is straightforward: a tire driving over an influence line introduces a peak in the signal. The problem arises when axles are grouped close together. Instead of individual peaks, such an axle group often manifests as a single peak with certain anomalies, which is the case with the second and the third axle in the figure. Relying only on detecting peaks fails to detect such axles. On the other hand, dynamic oscillations may produce secondary spurious peaks that can mistakenly be associated with an axle. We therefore recognize two types of errors in axle detection: a *missed axle* (false negative) caused by closely spaced axle groups, and a *ghost axle* (false positive) caused by secondary oscillations.

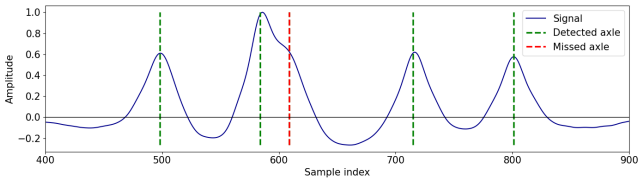


Figure 1: B-WIM strain-based signal

This paper addresses the problem of fully automating axle detection due to the *missed* and *ghost* axle phenomena. The naive approach of inferring axles from peaks in the strain signal is evaluated and taken as a baseline. Two deep learning architectures are then proposed and evaluated separately in comparison with the peak detection method: one based on 1D U-Net and the other on temporal convolutional networks (TCN). We hypothesize that deep learning can learn to correctly interpret peak anomalies as individual axles.

The contributions of this paper are: (i) we measure the prevalence of *missed* and *ghost* axles due to peak anomalies in bridge-strain signals; (ii) we measure the mean absolute timing error (MATE) of axle detection from peaks; and (iii) we propose two neural network architectures adapted to bridge-strain signals that significantly improve in all measured performances.

 DOI: https://doi.org/10.26493/scores26.12

2 RELATED WORK

Žnidarič et al. [11] improved B-WIM accuracy by advanced signal conditioning, including a centered moving average with a triangular window and Savitzky-Golay smoothing [7]. They also implemented an axle reconstruction algorithm that uses predefined rules to search for and correct potentially missing axles. The parameters are calibrated manually by a qualified engineer and must be re-tuned for each new site, speed regime, and sensor configuration.

Szinyéri et al. [10] proposed BwimNet, an encoder–decoder network that estimates GVW from strain-gauge signals, accepting axle positions as an input rather than inferring them, demonstrating that deep learning can extract load information from raw strain data.

Lorenzen et al. [3] applied a fully convolutional network to wavelet transforms of bridge acceleration data for virtual axle detection on a railway bridge. Their model correctly detected 95% of axles, where 90% of those were detected with a maximum spatial error of 20 cm. Their approach targets railway rather than road traffic.

Šoberl et al. [8] improved axle configuration inference in B-WIM by fusing a roadside camera classifier with the strain heuristic. Their approach did not aim to improve the accuracy of the existing strain-based axle detector, but rather eliminate unreliable estimations, which were determined by introducing uncertainty quantification through a camera-based deep learning method. Their hybrid approach substantially reduced ghost and missed axles within the reliable subset of outputs. In contrast, this paper researches whether the strain data alone, processed by a learned detector, can achieve comparable improvements without additional hardware.

3 DATA

The dataset consists of 32,141 heavy good vehicles (HGVs) bridge-crossing events collected by an operational B-WIM installation managed by the Slovenian National Building and Civil Engineering Institute (ZAG). Each crossing event is represented as a time series of a certain duration, which is derived by time-aligning and summing 12 raw strain signals as described in [11]. All signals were sampled at 512 Hz.

None of the signals in the dataset exceeded the length of 1300 samples (or about 2.54 seconds), hence we padded all the signals with zeroes to the common length of 1300. This choice was arbitrary and could be different with a different dataset or sampling rate.

Ground-truth labels are binary pulse vectors of length 1300, with value 1 at each true axle position. Axle counts per vehicle range from 2 to 11, with the total count of 145,592 labeled axles. Labels were generated automatically using B-WIM axle detection heuristics [11], hence our models were trained in a weakly supervised setting.

The dataset was restricted to signals exhibiting no apparent anomalies, excessive noise, unexpected interference, or other erroneous behavior that typically appears from multiple vehicle presence on the bridge. Such erroneous bridge-crossings are automatically determined and flagged by B-WIM. However, a subset of a few thousand critical cases was manually checked and approved for inclusion by the experts at ZAG.

We divided the dataset into a fixed held-out test set of 3,215 bridge-crossing events (14,574 individual axles) and a training set of 28,926 bridge-crossing events (131,018 individual axles). This corresponds to an approximately 90:10 training/test split in terms of both crossing events and individual axles. Given the stable cross-validation performance reported in the Results section, we do not expect a moderate reduction in the training-set size to substantially affect performance.

4 METHODS

We frame strain-based B-WIM axle detection as a 1D sequence-to-sequence translation, where the size of the output vector matches the size n of the input vector. Each output value corresponds to a probability of an axle being present at its corresponding position within the strain signal. We evaluate three methods that can do such translations: Peak-detection algorithm (as a baseline method), 1D U-Net and Temporal Convolutional Network (TCN).

4.1 Peak-Detection Baseline

Each strain signal is first smoothed using the Savitzky–Golay filter [7] (a sliding second-order polynomial fit) to suppress high-frequency noise while preserving the peak shape. Axle positions are identified as local maxima exceeding a height threshold, separated by a minimum distance, and having a minimum prominence. The height and prominence thresholds are expressed as fractions of the signal's amplitude range so that they adapt to each signal's scale. Height threshold, minimum peak distance and prominence were fine-tuned by grid search on the training data. The following values were found to maximize the performance: height threshold = 0.15 × amplitude range; minimum peak distance = 20 samples; prominence = 0.05 × amplitude range.

4.2 1D U-Net

The 1D U-Net is an adaptation of the U-Net architecture [6] to one-dimensional sequential data, such as time series and sensor signals. It follows a symmetric encoder–decoder structure in which the encoder progressively extracts increasingly abstract features while reducing the temporal resolution, and the decoder reconstructs a high-resolution output through successive upsampling operations. Skip connections between corresponding encoder and decoder stages transfer fine-grained temporal information directly to the decoding path, mitigating information loss caused by downsampling and facilitating precise localization of signal features.

Our implementation consists of 4 encoding levels. Each level applies two convolution blocks [Conv1D → BatchNorm → ReLU] and then halves the temporal resolution with max pooling. The channel count doubles at each level and goes from 32 to 256 within the encoder. A bottleneck block with dropout rate 0.2 sits at the lowest resolution. The decoder mirrors the encoder across 4 levels, with each level upsampling the feature map by a factor of two, concatenating it with the corresponding encoder feature map via a skip connection, and processing the result through an additional double-convolution block. A final 1×1 convolution maps the 32 decoder channels to a single output channel, producing one logit per sample. The network has 2,620,289 trainable parameters.

4.3 Temporal Convolutional Network (TCN)

A Temporal Convolutional Network (TCN) is a convolutional architecture designed for modeling sequential data and provides an alternative to recurrent neural networks for time-series processing [1]. TCNs employ one-dimensional causal convolutions to prevent future information leakage, and dilated convolutions to progressively expand the receptive field without a proportional increase in computational cost. Residual connections that bypass a set of convolutional layers are used to stabilize training and avoid the vanishing gradient problem.

Our implementation consists of 7 dilated blocks, each containing two dilated convolutional sequences [Conv1D → BatchNorm → ReLU → Dropout], and a residual connection from the previous block. The first block starts with dilation 1, which doubles at each successive block, ending with dilation 64 at block 7. All layers use 64 filters of size 5. Dropout rate of 0.2 is used in all blocks. Their combined receptive field is 1,017 samples, which spans most of the signal. The final 1×1 convolution produces one logit per sample. The network has 269,441 trainable parameters, roughly ten times fewer than our 1D U-Net implementation.

The architectural comparison between our two deep learning architectures is given in Table 1.

Table 1: Architecture comparison.

Property	1D U-Net	TCN
Depth	4 encoder levels	7 dilated blocks
Channels	32–256	64 (fixed)
Kernel size	3 (fixed)	5 (fixed)
Dropout rate	0.2 (bottleneck)	0.2 (every block)
Skip connections	concatenative	additive
Parameters	2,620,289	269,441

4.4 Training and Evaluation

Both neural network models were trained using *binary cross-entropy* loss and AdamW optimization [4] with learning rate 10^{-3} and weight decay 10^{-4}. Gradient norms were clipped to a maximum $L_2 = 1.0$. Because the two classes (*axle*, *no axle*) are highly imbalanced (between 2 and 11 axles per 1300 signal samples), the loss weight of the positive class (*axle*) was increased by the ratio of negative to positive samples. Training was conducted in batches of 64 instances for 50 epochs.

To evaluate the three methods we use *precision*, *recall*, and MATE (mean absolute timing error). A positive axle prediction is considered a *true positive* (TP), if an actual axle that has not been associated with another position lies within ±5 sample tolerance. If there is no such true axle, the prediction is considered *false positive* (FP). If a true axle has not been detected within the tolerance, it is considered a *false negative* (FN). Precision and recall are then associated with the ability to minimize *ghost axles* and *missed axles* respectively:

$$\text{Precision} = \frac{TP}{TP + FP}, \quad \text{Recall} = \frac{TP}{TP + FN}. \tag{1}$$

Note that classification accuracy is of limited relevance here due to the pronounced class imbalance and was therefore not measured.

The above metrics do not capture temporal errors, as long as the predictions are made within the given ±5 sample tolerance. The mean absolute temporal error is measured as:

$$\text{MATE} = \frac{1}{|TP|} \sum_{i \in TP} |\hat{t}_i - t_i|, \tag{2}$$

where $\hat{t}_i$ and t_i are the predicted and the matched ground-truth axle positions of the i-th true positive sample. A lower MATE is better.

5 RESULTS

Table 2 shows the results obtained through a 10-fold cross validation conducted on the training set of 28,926 bridge-crossing events. The given values are the means of the 10 repetitions, together with standard deviation. Since the peak-detection method is not trainable, it was run only once on all instances. Both neural architectures significantly outperformed the peak-detection baseline method, while the differences in their performance are negligible.

Table 2: Performance on the training set measured using 10-fold cross validation.

Method	Precision	Recall	MATE
Peak detection	0.9967	0.8802	0.866
1D U-Net	0.9966 ± 0.0008	0.9999 ± 0.0001	0.220 ± 0.024
TCN	0.9980 ± 0.0004	0.9999 ± 0.0001	0.231 ± 0.005

The small standard deviations across cross-validation folds indicate a stable performance of both neural architectures. Neither model depends on a favorable data split or exhibits signs of overfitting.

For a more intuitive assessment of detection performance, Table 3 reports the actual numbers of *ghost* and *missed* axles on the held-out test set, comprising 3,215 bridge-crossing events and 14,574 individual axles. Both neural architectures detected all labeled axles, whereas the peak detection method missed 1,686. The precision, recall, and MATE values are consistent with those obtained through cross-validation (Table 2). Ghost axle counts are broadly comparable to the baseline, with 1D U-Net producing somewhat more (58 vs. 40) and TCN slightly fewer (37 vs. 40), indicating no consistent improvement in precision across the two architectures.

Table 3: Performance of the held-out testing set.

Method	Ghost axles	Missed axles	Precision	Recall	MATE
Peak detection	40	1686	0.9969	0.8843	0.869
1D U-Net	58	0	0.9960	1.0000	0.197
TCN	37	0	0.9975	1.0000	0.228

Neural networks also significantly reduce MATE, which means that they detect axles more precisely within the signal than the peak-based detector. Assuming 90 km/h average speed of trucks when crossing the bridge, one signal sample spans about 5 cm of the truck's length. This means that the peak-detector would on average make a 4.2 cm spatial error, and the trained networks only about 1.1 cm.

6 DISCUSSION

Sensitivity to Temporal Tolerance. All axle-level metrics use a matching tolerance of ±5 samples, which is approximately 25 cm tolerance. The choice of this tolerance was arbitrary, considering that within our dataset, the minimal spacing between axles was 23 samples, and that the diameter of a truck tire is typically 108 cm. Tightening this axle matching window would reduce TP counts for all three methods, most severely for the baseline (whose MATE of 0.87 places a non-negligible fraction of true positives outside the tighter window), while having a much lower effect on the neural models (MATE < 0.25). This implies that the F_1 advantage of the neural models would grow substantially under stricter evaluation criteria.

Training on New Datasets. Certain architectural adaptations may be required for datasets from different B-WIM deployments. Vehicle speed and sampling rate may affect the required input and output vector size, which was 1300 in our case. Adjusting this size should have minimal impact on the core architecture, training time, stability, or performance, while substantially lower temporal resolution or axle tolerance may warrant a scaled-down architecture with fewer trainable parameters.

A different sampling rate may also alter class imbalance by changing the number of negative samples while leaving the average number of axles per crossing event unchanged. The positive-to-negative sample ratio should therefore be evaluated for each new dataset and the loss weights adjusted accordingly before training.

Suggested Architecture. While both neural architectures exhibit similar end-results, with a marginal preference towards TCN, it is also reasonable to consider the size of the model. With TCN being roughly 10 times smaller than 1D U-Net, the choice of TCN over 1D U-Net for axle detection is obvious. Not only is the footprint of a model much smaller, we have also observed roughly two times faster and visibly more stable training when using TCN. Other architectural solutions might be possible to reduce the model's footprint, but regarding axle detection performance, there is very little room left for improvements.

6.1 Limitations

We recognize three important limitations of this research work: (i) All the data was obtained from a single B-WIM site. No external cross-site validation dataset has been used to determine generalizability and transferability of the learned models; (ii) the dataset has been cleaned of erroneous measurements and only contains strain signals of single vehicle presence. The results presented in this paper may be regarded as representative of near-ideal laboratory conditions, whereas on-site testing may yield lower precision and recall; and (iii) the reference axle positions were generated by the operational B-WIM system rather than independently annotated by experts, although a small subset of the labels was manually verified. The reported performance should therefore be interpreted primarily as agreement with the existing B-WIM heuristic rather than as performance against independent ground truth. Some of the 37–58 detections classified as ghost axles may correspond to real axles missed by the existing heuristic and would thus constitute false positives only with respect to the weak labels. Expert examination of these cases and validation against independently annotated data are needed to determine the actual improvement of the proposed models over the existing heuristic approach.

7 CONCLUSION

We proposed two deep learning architectures for automated axle detection in B-WIM strain signals: a 1D U-Net and a TCN, replacing hand-tuned, site-specific heuristics with a data-driven approach. The models were trained under weak supervision and showed stable performance across cross-validation folds. Both models achieved near-perfect recall with respect to the available labels (0.999 vs. 0.884 for the baseline), reduced the mean axle timing error by approximately a factor of four (MATE 0.22–0.23 vs. 0.87), and maintained high precision (≈ 0.997). These results demonstrate the potential of deep learning for robust axle detection in B-WIM strain signals, while further validation against independently annotated data and across different B-WIM sites remains necessary.

ACKNOWLEDGMENTS

This research was conducted as part of the BrAId project, funded by the Slovenian Research and Innovation Agency (ARIS) under grant J7-50096. The authors gratefully acknowledge the Slovenian National Building and Civil Engineering Institute (ZAG) for providing access to the data used in this study. The data are not publicly available due to confidentiality restrictions.

REFERENCES

[1] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. 2018. An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling. https://doi.org/10.48550/arXiv.1803.01271 arXiv:1803.01271 [cs.LG]

[2] Bernard Jacob and Véronique Feypell-de La Beaumelle. 2010. Improving Truck Safety: Potential of Weigh-in-Motion Technology. *IATSS Research* 34, 1 (July 2010), 9–15. https://doi.org/10.1016/j.iatssr.2010.06.003

[3] Steven Robert Lorenzen, Henrik Riedel, Maximilian Michael Rupp, Leon Schmeiser, Hagen Berthold, Andrei Firus, and Jens Schneider. 2022. Virtual Axle Detector Based on Analysis of Bridge Acceleration Measurements by Fully Convolutional Network. *Sensors* 22, 22 (Nov. 2022), 8963. https://doi.org/10.3390/s22228963

[4] Ilya Loshchilov and Frank Hutter. 2019. Decoupled Weight Decay Regularization. https://doi.org/10.48550/arXiv.1711.05101 arXiv:1711.05101 [cs.LG]

[5] Fred Moses. 1979. Weigh-in-Motion System Using Instrumented Bridges. *Transportation Engineering Journal of ASCE* 105, 3 (May 1979), 233–249. https://doi.org/10.1061/TPEJAN.0000783

[6] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi (Eds.). Vol. 9351. Springer International Publishing, Cham, 234–241. https://doi.org/10.1007/978-3-319-24574-4_28

[7] Abraham. Savitzky and M. J. E. Golay. 1964. Smoothing and Differentiation of Data by Simplified Least Squares Procedures. *Analytical Chemistry* 36, 8 (July 1964), 1627–1639. https://doi.org/10.1021/ac60214a047

[8] Domen Šoberl, Jan Kalin, Andrej Anžlin, Maja Kreslin, Klen Čopič Pucihar, Matjaž Kljun, Doron Hekič, and Aleš Žnidarič. 2025. Enhanced Precision in Axle Configuration Inference for Bridge Weigh-in-motion Systems Using Computer Vision and Deep Learning. *Computer-Aided Civil and Infrastructure Engineering* 40, 30 (Dec. 2025), 6201–6216. https://doi.org/10.1111/mice.70144

[9] Mohhammad Sujon and Fei Dai. 2021. Application of Weigh-in-Motion Technologies for Pavement and Bridge Response Monitoring: State-of-the-art Review. *Automation in Construction* 130 (Oct. 2021), 103844. https://doi.org/10.1016/j.autcon.2021.103844

[10] Bence Szinyéri, Bence Kővári, István Völgyi, Dénes Kollár, and Attila László Joó. 2023. A Strain Gauge-Based Bridge Weigh-In-Motion System Using Deep Learning. *Engineering Structures* 277 (Feb. 2023), 115472. https://doi.org/10.1016/j.engstruct.2022.115472

[11] Aleš Žnidarič, Jan Kalin, and Maja Kreslin. 2018. Improved Accuracy and Robustness of Bridge Weigh-in-Motion Systems. *Structure and Infrastructure Engineering* 14, 4 (April 2018), 412–424. https://doi.org/10.1080/15732479.2017.1406958

 DOI: https://doi.org/10.26493/scores26.16

Toward Accessible Skin Cancer Screening on Mobile Devices: A Three-Tier Feature-Fusion Framework Prototype

Vuk Ignjatović, Nedim Šišić, and Peter Rogelj
vuk.ignjatovic05@gmail.com
Faculty of Mathematics, Natural Sciences
and Information Technologies (FAMNIT),
University of Primorska
Koper, Slovenia

Abstract

Skin cancer is among the most dangerous cancers when it is not detected early, yet early detection depends on access to dermatological expertise that is scarce in many regions. Camera-equipped smartphones, by contrast, are nearly ubiquitous. We present the design and current implementation of a mobile application prototype intended to make dermoscopic screening more accessible. The system follows a three-tier architecture: a cross-platform React Native client for image capture and metadata entry, a FastAPI server, and a machine-learning pipeline that reimplements a published handcrafted-plus-deep feature-fusion method. Handcrafted descriptors encoding the asymmetry, border and colour (ABC) cues of the clinical ABCD dermoscopy rule, together with gray-level texture features, are fused with deep features from a pre-trained MobileNetV2 backbone, reduced by mutual-information selection, and passed to a shallow classifier that decides benign vs. malignant. We target the HAM10000 dermatoscopy dataset, collapsing its seven diagnostic classes into these two categories and mitigating the resulting imbalance with SMOTE. The work is a prototype framework: several components are designed but not yet connected end to end, integrating the trained model with the serving layer being the primary remaining obstacle. The presented architecture serves as a reproducible reference design for end-to-end mobile medical-imaging applications.

Keywords

skin cancer, melanoma detection, computer-aided detection, feature fusion, mutual information, dermoscopy, MobileNetV2, HAM10000, mobile health

1 Introduction

Expert dermoscopic assessment is unevenly distributed across the world, and in rural or low-resource settings a patient may have no convenient path to a specialist. Smartphones with capable cameras, however, are widely available, so software running on commodity mobile hardware can act as a first-line triage aid—a decision-support tool, not a replacement for a clinician.

Deep learning has driven strong results in skin-lesion analysis, most visibly the dermatologist-level classifier of Esteva et al. [3]. Such models, however, require large curated datasets and considerable compute, and they behave as opaque predictors. A complementary line of work fuses *handcrafted* descriptors—quantitative encodings of the clinical ABCD rule [6]—with *deep* features from a pre-trained convolutional network, combining clinical interpretability with the representational power of transfer learning. The method of Almaraz-Damian et al. [1] is representative: it selects the most informative fused features by a mutual-information (MI) criterion and reports an accuracy of about 0.80 on a balanced version of the problem. A distinguishing aim of our design is to fuse the patient metadata our questionnaire already collects—age, sex, and lesion site—with the image features, so the classifier can exploit their correlation with malignancy.

Our contributions are:

- an open, three-tier architecture (mobile client, ingest server, ML pipeline) for dermoscopic screening on consumer devices (Section 3);
- a from-scratch reimplementation of a handcrafted (ABC + GLCM texture) and deep (MobileNetV2) feature-fusion pipeline with MI-based selection, targeting HAM10000 and addressing its class imbalance (Section 6); and
- a candid analysis of the integration gaps between a working research pipeline and a deployable system, offered as practical lessons for other students building end-to-end medical-imaging applications (Section 7).

We stress at the outset that the prototype is not a validated medical device and produces no clinical claims.

2 Background and Related Work

Dermoscopy and the ABCD rule. Clinical dermoscopy summarises a lesion's malignancy risk through the ABCD rule—*A*symmetry, *B*order irregularity, *C*olour variegation, and *D*ermoscopic structures [6]. Our handcrafted stage quantifies the asymmetry, border and colour components numerically from the segmented lesion; the dermoscopic-structures term (D), which requires identifying specific local patterns, is not extracted in the current implementation, so we refer to the implemented subset as ABC.

Feature fusion. Handcrafted descriptors offer interpretability, while deep features capture more complex image patterns at the cost of reduced interpretability. Almaraz-Damian et al. [1] fuse the two and use mutual information to keep only the features most predictive of the label—a single feature subset chosen once on the training data and thereafter applied identically to every input. Our pipeline follows this design and reuses the two information-measure-of-correlation texture terms as the MI signal source.

Transfer learning. Training a deep network from scratch is impractical on modest hardware and small datasets [5]. We instead use

 DOI: https://doi.org/10.26493/scores26.16

MobileNetV2 [8], a lightweight architecture designed for mobile inference, as a *fixed* feature extractor with ImageNet weights.

Dataset. HAM10000 [10] contains 10,015 dermatoscopic images across seven diagnostic classes with per-image metadata (age, sex, localisation). For this prototype we collapse the seven classes into a single binary benign-vs-malignant label. The set is deliberately realistic: images vary in scale and illumination, some metadata is missing, and the classes are severely imbalanced—melanocytic nevi dominate while melanoma is a minority—which we address with SMOTE [2].

3 System Architecture

The prototype is organised as three loosely coupled tiers communicating over HTTP (Figure 1): a mobile client, an ingest server, and the machine-learning pipeline. The separation lets the resource-intensive work run on a server rather than the phone and keeps the three codebases independently deployable. The next three sections describe each tier as it is currently implemented; Figure 1 also marks the one link that is designed but not yet in place—the call from the server into the model.

4 Mobile Client

The client is built with Expo (SDK 50) and React Native 0.73. Navigation uses React Navigation's stack navigator, and screen state is held entirely in memory for the session through three React Context providers; there is no Redux and no local persistence.

Screen flow. A single stack navigator drives a linear, wizard-style flow tracked by a five-step progress bar: a welcome screen, camera- and gallery-permission requests, photo capture or selection, a metadata form, and a result screen.

State and capture. *PermissionsContext* wraps the Expo permission requests and records the camera and gallery grants. *ImageContext* exposes `captureImage(isCamera)`, which launches the camera or library through `expo-image-picker` with editing enabled, a 1:1 aspect ratio and full quality, and retains both the base64 payload and the local file URI. *FormContext* holds the questionnaire—age, sex, and lesion localisation, the latter a body-site dropdown (abdomen, back, trunk, upper and lower extremity)—and owns the network call.

Upload. On submission, `sendData()` assembles a `multipart/form-data` body containing the metadata fields and the photo as a file part carrying its local URI, MIME type and filename, then POSTs it to the server's `/predict` route with axios. The user interface is complete; rendering the returned prediction on the result screen is the one client-side item still open (Section 7).

5 Ingest Server

The server is a single-file FastAPI application served by Uvicorn, with permissive CORS. It exposes three routes: a `GET /` health check, a `POST /user-data` debug echo, and the main `POST /predict`.

The `/predict` *endpoint.* It accepts `age`, `gender` and `localization` as form fields and the image as an `UploadFile`. The handler writes the uploaded bytes to disk and calls a helper, `image_to_array()`, which opens the image with Pillow, resizes it to 450×600, converts it to a NumPy array and normalises by dividing by 255. Writing to disk serves logging and debugging rather than necessity—the array could be passed to the model in memory, and the resize/normalise step could equally run on the client—so we keep this preprocessing on the server only to centralise image handling while the pipeline is under development. In the current build the endpoint returns the received image rather than a prediction, since no classifier is loaded in the serving path—the integration gap highlighted in Figure 1.

Deployment. The server and model ship Docker and Compose files based on Docker's Python 3.11-slim template with a non-root user. The server listens on port 7999 and the model image on 8000; reconciling these ports with the client's base URL is part of the remaining deployment work (Section 7).

Table 1: Handcrafted feature groups (the ABC components of the ABCD rule plus GLCM texture).

Group	Descriptors
Shape / geometry	area, perimeter, circularity, diameter, eccentricity
Asymmetry (A)	symmetry at 0° and 90°, averaged
Border (B)	perimeter / convex-hull perimeter
Colour (C)	min, max, mean, variance of blue and b^* channels
Texture (GLCM)	ASM, contrast, correlation, entropy, and two mutual-information (IMC) terms

6 Feature-Fusion Pipeline

The machine-learning tier turns a raw dermatoscopic image into a prediction in five stages: preprocessing and segmentation, handcrafted feature extraction, deep feature extraction, MI-based fusion and selection, and classification. Class balancing by SMOTE is applied to the training data beforehand, as detailed at the end of this section.

6.1 Preprocessing and segmentation

Each image is Gaussian-blurred and converted to the CIELAB colour space; a grayscale copy is thresholded by Otsu's method to obtain a binary mask. The largest contour yields a bounding box, and a padded square region of interest (ROI) around the lesion is cropped. A quality gate rejects empty or saturated crops so that degenerate inputs do not reach the classifier. The stage outputs the cropped ROI and its binary mask.

6.2 Handcrafted features (ABC + texture)

From the ROI and mask we compute a vector of clinically motivated descriptors, grouped as in Table 1. Shape and asymmetry come from image moments and mask flips/rotations; border irregularity is the ratio of the lesion perimeter to its convex-hull perimeter; colour statistics are drawn from the RGB blue and LAB b^* channels. Texture is described by Haralick gray-level co-occurrence matrix (GLCM) features [4] computed over four orientations, including the two information-measure-of-correlation terms that supply the mutual-information signal used later.

 DOI: https://doi.org/10.26493/scores26.16

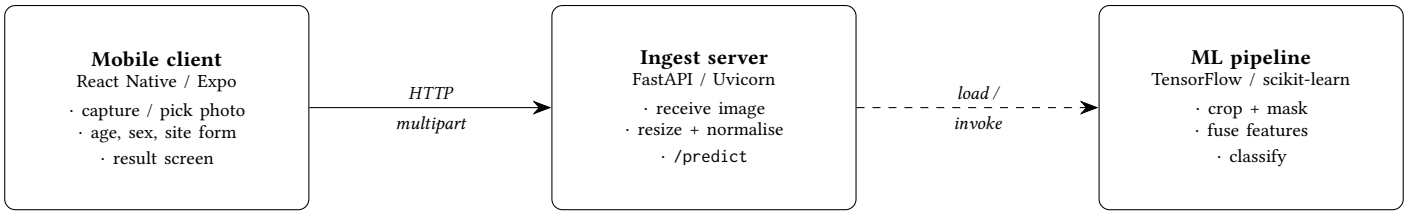


Figure 1: Three-tier architecture. The client→server link is implemented; the server→model link (dashed) is the main integration gap discussed in Section 7.

6.3 Deep features

The ROI is passed through MobileNetV2 [8] with ImageNet weights and no classification head, producing a $7 \times 7 \times 1280$ activation tensor used as a fixed deep-feature descriptor. The backbone is frozen rather than fine-tuned: this avoids backpropagating through its parameters during training, lowering the memory and compute the training loop requires, and it guards against overfitting a large backbone on limited data.

6.4 Fusion, selection, and classification

For each image the handcrafted vector $\mathbf{h}$ and the flattened deep vector $\mathbf{d}$ are concatenated into a fused vector $\mathbf{f} = [\mathbf{h}; \mathbf{d}]$. Because $\mathbf{f}$ is high-dimensional and partly redundant, we keep the k features whose mutual information $I(f_i; y)$ with the label y is largest, where

$$I(f_i; y) = \sum_{f_i} \sum_{y} p(f_i, y) \log \frac{p(f_i, y)}{p(f_i)\, p(y)}, \tag{1}$$

with $k = 100$ in our configuration, implemented via scikit-learn's `SelectKBest` [7]. The selection is fit once on the training partition and the same indices are applied to every image at inference, so the classifier always receives inputs of consistent meaning and dimension. The reduced vectors feed three classifiers compared under 5-fold grid search: logistic regression, a support vector machine with an RBF kernel, and a relevance vector classifier [9].

6.5 Class imbalance and evaluation metric

In HAM10000, benign lesions vastly outnumber malignant ones, so a classifier trained on raw counts is biased toward predicting "benign." We rebalance with SMOTE [2], which synthesises new minority-class samples rather than duplicating them. Crucially, oversampling is applied only to the training partition: the validation and test partitions contain no synthetic samples, so balancing cannot leak into evaluation and inflate the reported recall. For a screening task, raw accuracy is misleading; precision and recall matter more, since a missed melanoma (false negative) is far costlier than a false alarm:

$$\text{Precision} = \frac{TP}{TP + FP}, \qquad \text{Recall} = \frac{TP}{TP + FN}. \tag{2}$$

Evaluation is therefore planned as a full precision/recall report under a cross-validation split rather than a single accuracy figure; producing it is the immediate goal (Section 9).

Table 2: Implemented vs. planned functionality.

Component	Status
Client capture, form, upload	Implemented
Server ingest + preprocessing	Implemented
Handcrafted + deep feature code	Implemented (needs debugging)
Model training on HAM10000	Planned
Server→model inference	Planned
Client rendering of prediction	Planned
On-device (edge) inference	Planned

7 Implementation Status

Table 2 separates the components that are implemented from those that are planned. The methodology is fully expressed in code, but the tiers are not yet joined end to end: the pipeline still contains defects in the handcrafted-feature stage (undefined intermediate terms and mismatched function signatures) that must be resolved before it trains, and the trained model is not yet serialised or served. The remaining work is primarily integration rather than methodology—fix and train the pipeline, persist the model, load it behind `/predict`, and have the client render the returned label—together with reconciling development-only settings (a hardcoded API address and sex field, the client/server port mismatch, and absolute paths in the model container).

8 Ethical and Safety Considerations

Because the application targets a medical-screening task, several non-technical safeguards are part of the design rather than optional extras. The tool is positioned as *triage support*: every result is presented with a disclaimer that it is not a diagnosis and does not replace a dermatologist, and the interface is deliberately conservative about implying certainty. The training data also carries known biases—HAM10000 is drawn from two largely fair-skinned populations [10], so a model fitted to it must not be assumed to generalise across skin tones, and any real deployment would require evaluation on more representative data. A screening aid that *under*-triages is especially dangerous, which is why we optimise and report recall for the malignant class rather than overall accuracy, deliberately accepting more false alarms to reduce missed melanomas.

9 Conclusion and Future Work

We described a three-tier mobile prototype for skin-cancer screening that fuses handcrafted ABC descriptors with MobileNetV2 deep

features, reduces them by mutual information, and classifies benign vs. malignant on HAM10000. The core components exist within a single system; their end-to-end integration is the key remaining step.

Future work will connect the pipeline end to end and, once it trains, report melanoma recall under a cross-validation split, testing against the ≈ 0.80 balanced accuracy of the reference method [1]. We also plan to (i) fuse the patient metadata the form already collects into the model input, exploiting the additional signal present in HAM10000, and (ii) extend the binary classifier to a multi-class model that reports per-disease probabilities.

Edge deployment. A further goal is to move inference from the server onto the handset, so that images never leave the device—easing GDPR obligations, protecting user data, and enabling use where connectivity is poor. MobileNetV2 is already a mobile-scale backbone (about 3.4M parameters) [8], and compressing the classifier—for example with a ternary-weight network—is a promising route to a model small enough to run entirely on-device. Throughout, the tool is intended as triage support that defers to a clinician, never as a diagnosis.

References

[1] Jose-Agustin Almaraz-Damian, Volodymyr Ponomaryov, Sergiy Sadovnychiy, and Heydy Castillejos-Fernandez. 2020. Melanoma and Nevus Skin Lesion Classification Using Handcraft and Deep Learning Feature Fusion via Mutual Information Measures. *Entropy* 22, 4 (2020), 484. doi:10.3390/e22040484

[2] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. 2002. SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research* 16 (2002), 321–357.

[3] Andre Esteva, Brett Kuprel, Roberto A. Novoa, Justin Ko, Susan M. Swetter, Helen M. Blau, and Sebastian Thrun. 2017. Dermatologist-Level Classification of Skin Cancer with Deep Neural Networks. *Nature* 542, 7639 (2017), 115–118.

[4] Robert M. Haralick, K. Shanmugam, and Its'hak Dinstein. 1973. Textural Features for Image Classification. *IEEE Transactions on Systems, Man, and Cybernetics* SMC-3, 6 (1973), 610–621.

[5] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep Learning. *Nature* 521, 7553 (2015), 436–444.

[6] Franz Nachbar, Wilhelm Stolz, Tanja Merkle, Armand B. Cognetta, Thomas Vogt, Michael Landthaler, Peter Bilek, Otto Braun-Falco, and Gerd Plewig. 1994. The ABCD Rule of Dermatoscopy: High Prospective Value in the Diagnosis of Doubtful Melanocytic Skin Lesions. *Journal of the American Academy of Dermatology* 30, 4 (1994), 551–559.

[7] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[8] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. 2018. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 4510–4520.

[9] Michael E. Tipping. 2001. Sparse Bayesian Learning and the Relevance Vector Machine. *Journal of Machine Learning Research* 1 (2001), 211–244.

[10] Philipp Tschandl, Cliff Rosendahl, and Harald Kittler. 2018. The HAM10000 Dataset, a Large Collection of Multi-Source Dermatoscopic Images of Common Pigmented Skin Lesions. *Scientific Data* 5 (2018), 180161. doi:10.1038/sdata.2018.161

 DOI: https://doi.org/10.26493/scores26.19

Cepstral Features for EEG Classification: Comparison with Spectral Features

Luka Matič
Faculty of Mathematics, Natural Sciences and Information Technologies (FAMNIT)
University of Primorska
Koper, Slovenia
89252128@student.upr.si

Peter Rogelj
Faculty of Mathematics, Natural Sciences and Information Technologies (FAMNIT)
University of Primorska
Koper, Slovenia
peter.rogelj@upr.si

Abstract

Electroencephalography (EEG) signal classification is a foundational task across a wide range of brain-activity-related applications and the choice of feature representation strongly influences classifier performance. This paper compares a spectral feature extraction method (Welch power spectral density over a five-channel sensorimotor montage) against a cepstral feature pipeline (triangular filterbank, logarithmic compression, discrete cosine transform and multivariate Gaussian probability density modeling) on two binary EEG classification tasks drawn from the PhysioNet EEG Motor Movement/Imagery Dataset: eyes-open versus eyes-closed resting state and movement versus rest (combining executed and imagined unilateral fist movement). Within 5-fold cross-validation, features were selected in each fold using Minimum Redundancy Maximum Relevance (MRMR) to avoid data leakage and Support Vector Machine (SVM) and Linear Discriminant Analysis (LDA) classifiers were trained and evaluated, with statistical significance of performance differences assessed using ANOVA. Results show that cepstral features combined with LDA provide a statistically significant improvement of approximately 4% over spectral features for the eyes-state task, while classification of the movement task remains substantially harder (52–55% accuracy) due to high inter-subject variability.

EEG, cepstral analysis, spectral analysis, CSP, MRMR, motor imagery, feature extraction

1 Introduction

Decoding mental states from electroencephalography (EEG) signals underlies a range of brain-activity-based applications, from brain-computer interfaces (BCIs) to clinical monitoring and cognitive research and a central design choice in any EEG classification pipeline is the feature representation used to translate raw, noisy multichannel recordings into a compact, discriminative form. Spectral features characterize frequency content, typically via power spectral density (PSD) or spatial filters such as Common Spatial Patterns (CSP) and are well established due to their direct link to known EEG rhythms (alpha, beta, mu). Cepstral features, by contrast, originate from speech processing and apply a further transformation to the spectral envelope – a filterbank, logarithmic compression and discrete cosine transform (DCT) – to capture smoothed, decorrelated spectral shape information analogous to Mel-Frequency Cepstral Coefficients (MFCCs).

While cepstral representations have shown promise in speech and audio classification, their relative merit for EEG classification is less consistently established across tasks that vary in difficulty. This raises a natural question: does the additional transformation in cepstral analysis yield a measurable advantage over spectral features and does this advantage hold across different EEG classification tasks?

This paper addresses this question empirically through two binary classification tasks performed on the PhysioNet EEG Motor Movement/Imagery Dataset (EEGMMIDB): (1) discriminating eyes-open from eyes-closed resting-state EEG and (2) discriminating rest from movement, combining executed and imagined unilateral fist movement. For each task, spectral and cepstral feature pipelines are constructed, features are selected using Minimum Redundancy Maximum Relevance (MRMR) within cross-validation folds to prevent information leakage and two classifiers, Support Vector Machine (SVM) and Linear Discriminant Analysis (LDA), are trained and evaluated using 5-fold cross-validation. Statistical significance of performance differences is assessed using analysis of variance (ANOVA).

The remainder of this paper is organized as follows. Section II describes the dataset and task definitions. Section III details the feature extraction, feature selection and classification methodology. Section IV presents classification results for both tasks. Section V discusses the implications of these results and Section VI concludes the paper.

2 Dataset and Task Definitions

2.1 Dataset

All experiments in this study use the PhysioNet EEG Motor Movement/Imagery Dataset (EEGMMIDB). The dataset consists of over 1500 one- and two-minute EEG recordings collected from 109 healthy subjects, each recorded using a 64-channel EEG montage sampled at 160 Hz according to the international 10-10 electrode placement system. For each subject, the dataset provides 14 experimental runs, each corresponding to a distinct combination of task type (baseline, real movement, or motor imagery) and limb (fists or feet). Recordings were preprocessed and epoched using EEGLAB's `pop_epoch` function. Rather than using the full 64-channel montage, analysis in this study was restricted to a five-channel sensorimotor subset (C3, Cz, C4, CP3, CP4), selected for their established relevance to motor cortex activity and to keep feature dimensionality manageable.

 DOI: https://doi.org/10.26493/scores26.19

2.2 Task 1: Eyes-Open versus Eyes-Closed

The first classification task uses runs R01 (eyes open, baseline) and R02 (eyes closed, baseline) for each subject, with each continuous run segmented into non-overlapping 4-second epochs for feature extraction. This is a resting-state task with no motor component and it is expected to be comparatively easier to classify, since eyes-closed recordings typically exhibit a strong and well-documented increase in occipital alpha-band (8–12 Hz) power relative to eyes-open recordings. This task therefore serves as a sanity-check baseline against which the more challenging movement task can be compared.

2.3 Task 2: Motor Imagery

The second classification task is a binary discrimination between resting-state segments (labeled T0) and movement segments in which the subject either physically performs or imagines performing a left- or right-fist movement (labeled T1 and T2 respectively, here pooled together as a single "movement" class). This task draws on runs R03, R04, R07, R08, R11 and R12 of the EEGMMIDB protocol, which correspond to the unilateral fist condition and include both real-movement runs (R03, R07, R11) and motor-imagery runs (R04, R08, R12). Combining executed and imagined movement in this way follows directly from the pipeline structure used in this study and is noted here explicitly since it distinguishes this task from a pure motor-imagery formulation. Unlike Task 1, epochs for this task were extracted as fixed 2-second windows locked to the onset of each T0, T1, or T2 event marker, rather than by segmenting continuous recordings. Classification on this task is substantially harder than the eyes-state task, since the underlying event-related desynchronization/synchronization (ERD/ERS) patterns in the mu (8–12 Hz) and beta (13–30 Hz) bands are subtler, more spatially localized and known to vary considerably across subjects.

3 Methods

This section describes the exact implementation of the two feature extraction pipelines, the feature selection procedure, the classifiers and the validation methodology, as applied in this study.

3.1 Spectral Features

For each 4-second epoch (Task 1) or 2-second event-locked epoch (Task 2), the power spectral density (PSD) of each of the five selected channels (C3, Cz, C4, CP3, CP4) was estimated in MATLAB using `pwelch` with default window, overlap and FFT length settings which measures how much signal power is present at each frequency. The resulting PSD was then reduced to a feature vector by computing the mean power within a series of twelve 1 Hz-wide bins spaced every 2 Hz across the 8–30 Hz range (bin centers at 8, 10, 12, …, 30 Hz), i.e., for each bin edge f_b, the feature was mean$\{P(f) : f_b \le f < f_b + 1\}$. This spacing was chosen to keep the spectral feature count comparable in size to the cepstral feature (60 versus 65 features per epoch), so that differences in classification performance would reflect the feature representations themselves rather than a large disparity in feature-vector dimensionality. This was repeated independently for each of the five channels, yielding $12 \times 5 = 60$ spectral features per epoch.

3.2 Cepstral Features

The cepstral feature pipeline used in this study follows the filterbank–log–DCT–MVGPDF method described by Bhattacharyya and Mukul. For each epoch, the following steps were applied independently to each of the five channels:

(1) **Framing**: Each epoch was divided into `NUM_FRAMES` = 7 overlapping frames with `OVERLAP` = 0.5 (50%), using a frame length computed as win_len$/(7 \times 0.5 + 0.5)$ samples and a hop size of half the frame length. Each frame was windowed with a Hamming window before further processing. A standard tapering function that smooths the edges of each frame to reduce artifacts before taking its Fourier transform.

(2) **Power spectrum**: For each windowed frame, an FFT of length NFFT $= 2^{\lceil \log_2(\text{WIN\_SEC} \times f_s) \rceil}$ was computed (with `WIN_SEC` = 4) and the squared magnitude of the first half of the spectrum (up to the Nyquist frequency) was retained as the frame's power spectrum, i.e., how much signal energy is present at each frequency.

(3) **Triangular filterbank**: A bank of `NUM_FILT` = 20 overlapping triangular filters was constructed with center frequencies evenly spaced between `FREQ_LO` = 8 Hz and `FREQ_HI` = 30 Hz. Each filter's weight rises linearly from 0 to 1 between its left edge and center frequency, then falls linearly from 1 back to 0 between its center and right edge, forming a triangular shape; this smooths and pools energy from neighboring frequencies into 20 broader bands rather than treating each frequency independently. The power spectrum of each frame was projected onto this filterbank via matrix multiplication to obtain 20 filterbank energies per frame.

(4) **Logarithmic compression**: The natural logarithm of each filterbank energy was taken (with a small additive constant, 10^{-10}, to avoid taking the log of zero). This compresses large energy differences into a smaller, more comparable range, similar to how loudness is perceived logarithmically rather than linearly.

(5) **Discrete Cosine Transform (DCT)**: The 20 log filterbank energies of each frame were transformed via a custom DCT-II matrix into `NUM_CC` = 12 cepstral coefficients, using DCT$[n, m] = \cos(\pi n(m - 0.5)/20)$ for coefficient index $n = 1, \ldots, 12$ and filter index $m = 1, \ldots, 20$. This step reduces 20 correlated filterbank values down to 12 largely independent coefficients, concentrating the most informative shape of the spectrum into fewer numbers.

(6) **Multivariate Gaussian probability density function (MVGPDF) modeling**: Across the 7 frames of an epoch, the mean vector μ (12-dimensional) and regularized covariance matrix $\Sigma = \text{cov}(\text{CC}) + 10^{-6} I$ of the frame-wise cepstral coefficient vectors were computed. The multivariate Gaussian density was then evaluated at each frame's coefficient vector relative to this fitted mean and covariance, producing one density value per frame that summarizes how typical or unusual that frame's coefficients are relative to the epoch as a whole.

The final cepstral feature vector for each channel consisted of the mean of the 7 per-frame MVGPDF density values together with the

12-dimensional mean cepstral coefficient vector μ, i.e., 13 features per channel, giving $13 \times 5 = 65$ cepstral features per epoch in total.

3.3 Feature Selection

Within each of the 5 cross-validation folds, `fscmrmr` (MATLAB's built-in Minimum Redundancy Maximum Relevance implementation) was run separately on the spectral training features and the cepstral training features, using only the training partition of that fold. `fscmrmr` ranks features by how strongly each one relates to the class label while penalizing features that are redundant with ones already selected, so the retained set is both informative and non-repetitive. The top 20 ranked features were retained and the same feature indices selected on the training data were then applied to the corresponding test partition. This fold-local selection was applied independently to each of the two feature sets, so the specific features retained could differ from fold to fold.

3.4 Classifiers

For each fold and each feature type, two classifiers were trained on the selected training features and evaluated on the corresponding test features:

- **SVM**: trained via `fitcsvm` with a linear kernel (`KernelFunction = 'linear'`), automatic kernel scale (`KernelScale = 'auto'`), box constraint $C = 1$ and standardized input features (`Standardize = true`).
- **LDA**: trained via `fitcdiscr` with `DiscrimType = 'linear'` (i.e., a shared-covariance linear discriminant classifier).

This produced four accuracy values per fold: SVM-Spectral, LDA-Spectral, SVM-Cepstral and LDA-Cepstral.

3.5 Validation Procedure and Statistical Testing

All epochs across all subjects were pooled and partitioned into 5 stratified folds using MATLAB's `cvpartition(Y_all, 'KFold', 5)` with a fixed random seed (`rng(42)`) for reproducibility. For each fold, test-set classification accuracy was recorded for all four feature-classifier combinations, yielding a 5×4 accuracy matrix per task. To test whether cepstral features significantly outperformed spectral features, two separate one-way ANOVAs (`anova1`) were run per task: one comparing the 5 SVM-Spectral accuracies against the 5 SVM-Cepstral accuracies and one comparing the 5 LDA-Spectral accuracies against the 5 LDA-Cepstral accuracies. Pairwise mean differences and their confidence intervals were obtained via `multcompare` on the resulting ANOVA statistics.

4 Results

4.1 Eyes-Open versus Eyes-Closed

Table 1 summarizes 5-fold cross-validation accuracy for the eyes-open versus eyes-closed task across both feature types and both classifiers. Cepstral features outperformed spectral features for both classifiers: SVM accuracy increased from 67.19% ± 1.53% (spectral) to 69.08% ± 0.92% (cepstral), a mean improvement of approximately 1.9 percentage points ($p = 0.0450$), while LDA accuracy increased from 65.23% ± 2.69% (spectral) to 69.11% ± 0.97% (cepstral), a mean improvement of approximately 3.9 percentage points ($p = 0.0161$). Both differences are statistically significant at the $\alpha = 0.05$ level. Notably, spectral-feature accuracy was also more variable across folds (particularly for LDA, with a standard deviation of 2.69 percentage points) than cepstral-feature accuracy, which was consistently in the 68–70% range across all folds for both classifiers.

Table 1: 5-Fold CV Accuracy: Eyes-Open vs. Eyes-Closed

Feature Type	Classifier	Accuracy (%)	ANOVA p
Spectral	SVM	67.19 ± 1.53	0.0450
Cepstral	SVM	69.08 ± 0.92	
Spectral	LDA	65.23 ± 2.69	0.0161
Cepstral	LDA	69.11 ± 0.97	

4.2 Movement Task (T0 vs. T1+T2)

For the movement task, a total of 19,690 epochs were extracted, balanced evenly between the rest class (T0: 9,845 epochs) and the combined movement class (T1+T2: 9,845 epochs). Table 2 summarizes 5-fold cross-validation accuracy for this task. As expected, overall accuracy is substantially lower than for the eyes-state task, consistent with the greater difficulty of decoding movement-related activity relative to a resting baseline. Cepstral features again outperformed spectral features for both classifiers: SVM accuracy increased from 52.31% ± 0.50% (spectral) to 55.22% ± 1.43% (cepstral), a mean improvement of approximately 2.9 percentage points ($p = 0.0026$), while LDA accuracy increased from 52.80% ± 0.62% (spectral) to 55.24% ± 1.47% (cepstral), a mean improvement of approximately 2.4 percentage points ($p = 0.0093$). Both differences are statistically significant at the $\alpha = 0.05$ level and in fact the SVM p-value here is smaller than for the eyes-state task, indicating a comparatively more consistent cepstral advantage across folds for this classifier despite the lower absolute accuracy.

Table 2: 5-Fold CV Accuracy: Movement vs. Rest (T0 vs. T1+T2)

Feature Type	Classifier	Accuracy (%)	ANOVA p
Spectral	SVM	52.31 ± 0.50	0.0026
Cepstral	SVM	55.22 ± 1.43	
Spectral	LDA	52.80 ± 0.62	0.0093
Cepstral	LDA	55.24 ± 1.47	

4.3 Summary of Findings

Across both tasks and both classifiers, cepstral features consistently and significantly outperformed spectral features, with all four ANOVA comparisons reaching statistical significance ($p < 0.05$). The magnitude of the cepstral advantage was larger for LDA than for SVM on the eyes-state task (3.9 vs. 1.9 percentage points), whereas the two classifiers showed a more comparable advantage on the movement task (2.4 vs. 2.9 percentage points). Absolute accuracy was substantially higher for the eyes-state task (65–69%) than

for the movement task (52–55%) across all feature-classifier combinations, reflecting the greater physiological separability of the eyes-open/eyes-closed conditions compared to the more subtle and subject-variable signatures of executed and imagined movement.

5 Discussion

5.1 Why Cepstral Features Outperform Spectral Features

The consistent cepstral advantage across both tasks and classifiers suggests the filterbank–log–DCT–MVGPDF pipeline captures structure in the spectral envelope not fully accessible to raw per-bin power estimates. The triangular filterbank pools energy from neighboring frequencies into 20 overlapping bands, likely reducing sensitivity to noise in any single narrow region, while the DCT decorrelates the resulting log filterbank energies, reducing the redundancy MRMR's selection criterion would otherwise need to penalize and concentrates most of the discriminative variance into a few low-order coefficients. This pooling step is functionally similar to the 1 Hz-bin averaging used for spectral features, but with triangular rather than rectangular weighting, finer and overlapping resolution (20 filters versus 12 bins) and continuous rather than gapped coverage of the 8–30 Hz range; part of the cepstral advantage may therefore stem from this finer pooling itself rather than the subsequent log, DCT and MVGPDF steps alone. The same pooling difference may also explain the lower fold-to-fold variability seen for cepstral features, most notably for LDA on the eyes-state task, where the spectral standard deviation (2.69 points) was nearly three times the cepstral value (0.97 points).

5.2 Why Movement Classification Remains Difficult

Movement-task accuracy (52–55%) stayed far below the eyes-state task (65–69%) and only modestly above chance. Unlike the strong, spatially broad occipital alpha-band effect underlying eyes-open/eyes-closed, movement-related ERD/ERS patterns are weaker, more spatially localized and vary considerably across subjects. The problem compounded here since all 109 subjects were pooled into a single cross-subject model rather than subject-specific classifiers. Combining executed movement (R03, R07, R11) with imagined movement (R04, R08, R12) into one class likely adds further within-class heterogeneity, since executed movement typically produces stronger sensorimotor activation than imagined movement.

5.3 Limitations

Several limitations should be noted. First, cross-validation was performed at the epoch level rather than with subject-level splits, so a subject's epochs may appear in both train and test partitions, potentially inflating accuracy relative to fully unseen subjects. Second, analysis used only a five-channel sensorimotor subset (C3, Cz, C4, CP3, CP4) rather than the full 64-channel montage. Third, the two pipelines are not a strictly controlled comparison of "cepstral versus spectral" processing alone: spectral features used rectangular 1 Hz bins spaced 2 Hz apart (chosen to match the cepstral feature count, 60 versus 65), leaving roughly half of the 8–30 Hz range unrepresented, whereas the cepstral filterbank used finer, overlapping and continuous coverage. Part of the cepstral advantage may therefore stem from this more refined frequency pooling itself rather than the subsequent log, DCT and MVGPDF steps; an ablation with matched filterbank/binning configurations would be needed to isolate these effects. Fourth, Task 1 and Task 2 used different epoch lengths (4 s and 2 s, respectively), which may itself affect the comparability of results across tasks.

6 Conclusion

This paper compared spectral (Welch PSD) and cepstral (filterbank–log–DCT–MVGPDF) features for two binary EEG classification tasks on PhysioNet EEGMMIDB: eyes-open versus eyes-closed and movement versus rest. Using MRMR feature selection within cross-validation folds and both SVM and LDA classifiers, cepstral features significantly outperformed spectral features across both tasks (1.9–3.9 percentage-point improvements, ANOVA $p < 0.05$ throughout). The eyes-state task was classified substantially more accurately (65–69%) than the movement task (52–55%), reflecting the stronger physiological basis of the eyes-state contrast.

These results support cepstral features, originally developed for speech processing, as a viable and here superior alternative to spectral features for EEG classification. Future work could use subject-level cross-validation splits, the full 64-channel montage and separate classification of executed versus imagined movement, while more directly aligning the two pipelines through matched frequency sub-bands and a consistent epoch length across tasks.

References

[1] G. Schalk, D. J. McFarland, T. Hinterberger, N. Birbaumer and J. R. Wolpaw. 2004. BCI2000: A general-purpose brain-computer interface (BCI) system. *IEEE Transactions on Biomedical Engineering* 51, 6 (2004), 1034–1043. https://www.physionet.org/content/eegmmidb/1.0.0/

[2] S. Bhattacharyya and A. Mukul. 2016. Cepstrum based feature extraction method for EEG signal classification. *International Journal of Engineering and Technology (IJET)* 8, 1 (2016). https://www.enggjournals.com/ijet/docs/IJET16-08-01-008.pdf

 DOI: https://doi.org/10.26493/scores26.10

Attack Detection on Graph Neural Networks

Diana Ioana Catana
dc05267@student.uni-lj.si
Faculty of Computer and Information Science,
University of Ljubljana
Večna pot 113
SI-1000 Ljubljana, Slovenia

Caio Simon
cs67685@student.uni-lj.si
Faculty of Computer and Information Science,
University of Ljubljana
Večna pot 113
SI-1000 Ljubljana, Slovenia

Đorđe Stevanović
xs19452@student.uni-lj.si
Faculty of Computer and Information Science,
University of Ljubljana
Večna pot 113
SI-1000 Ljubljana, Slovenia

Pietro Sestito
pietro.sestito01@estudiant.upf.edu
Faculty of Computer and Information Science,
University of Ljubljana
Večna pot 113
SI-1000 Ljubljana, Slovenia

ABSTRACT

Graph Neural Networks (GNNs) have become a popular tool in high-stakes applications, yet they remain vulnerable to adversarial structural perturbations that subtly modify the graph topology to degrade model performance. Detecting such attacks has immediate practical importance. This paper proposes a hypothesis testing framework for adversarial perturbation detection that explicitly accounts for expected graph changes by calibrating the null distribution against structured perturbations guided by link-prediction indices, rather than against uniform random change. Four complementary graph comparison measures are used — Portrait Divergence, Graphlet Disagreement, D-measure, and a Degree Distribution likelihood-ratio test — against three adversarial attacks (Nettack, Fast Gradient Attack, and a random baseline) on the CiteSeer and Cora citation networks at different perturbation budgets. Graphlet Disagreement emerges as the most effective detector against gradient-based stealthy attacks, providing consistent detection power across all budgets where the Degree Distribution test — the measure Nettack is specifically designed to preserve — achieves near-zero power. Portrait Divergence, while a strong discriminator overall, offers limited advantage over Graphlet Disagreement for detecting FGA and Nettack specifically.

KEYWORDS

Graph Neural Network, Adversarial Attacks, Graph Attack Detection, Adversarial Machine Learning, Link Prediction, Structural Null Models

1 INTRODUCTION

Graph Neural Networks (GNNs) have become central to tasks such as social network analysis, drug discovery, and fraud detection. Yet GNNs are vulnerable to adversarial structural perturbations [9], in which graph topology is modified to degrade model performance on the targeted nodes. In this work, a hypothesis testing framework is developed for the detection of such attacks, in which graph comparison measures are leveraged to construct a permutation test with greater power than previously proposed methods.

A key contribution is the construction of a structured null distribution calibrated via the Adamic-Adar link prediction index, under which null perturbations are generated by preferentially adding edges between nodes with high-weight common neighbours and removing structurally peripheral edges. The null hypothesis is therefore not "the graph has not changed" but the more demanding "observed changes are consistent with link-prediction-plausible structural dynamics," providing strictly stronger evidence of malicious intent upon rejection.

Three contributions are made: (i) a time-independent detection procedure in which expected graph changes are simulated via link prediction, requiring neither temporal data nor prior calibration; (ii) a systematic empirical comparison of four structural detectors across three attacks and four perturbation budgets, with bootstrapped confidence intervals; and (iii) evidence that Graphlet Disagreement provides the most consistent detection power against gradient-based stealthy attacks, with important caveats discussed in the Results.

Related work

Adversarial attacks on GNNs. Adversarial attacks on neural networks were initially studied in image classification [3, 6] before Zügner et al. [9] extended the framework to GNNs. Their Nettack algorithm operates on a linear surrogate model to produce transferable structural perturbations, targeting nodes that the victim model classifies correctly with high confidence. Crucially, Nettack incorporates explicit stealth constraints: perturbations must preserve the graph's power-law degree distribution and avoid linking feature-dissimilar nodes, properties that the authors argue are easily audited in practice. A naive baseline, RND, is also introduced, which adds edges uniformly at random between nodes of differing labels without any stealth guarantee.

Subsequent work relaxed these constraints in favour of attack power. Fast Gradient Attack (FGA) [4] selects perturbations by inspecting pairwise node-embedding gradients in a GCN, and Dai et al. [5] employ reinforcement learning to the same end. Neither

method incorporates formal stealth constraints, relying instead on a fixed perturbation budget to limit detectability.

Graph comparison measures. The presented work relies on general network science concepts, such as graphlets, graphlet counting through the orbit counting algorithm [7], D-measure [8], and portrait divergence [2]. Link prediction indices such as the Adamic-Adar index [1] are leveraged to model plausible network changes. It dampens the impact of high-degree nodes in the shared neighbourhood of two nodes while increasing the likelihood of linkages when nodes share more obscure common neighbours, making it a well-motivated model for triadic closure in sparse citation networks.

2 METHODS

Dataset and perturbed graphs. Two directed citation graphs are used, CiteSeer and Cora, treated as undirected and unweighted for all structural comparisons. Perturbed graph instances are pregenerated for each attack at budgets $B \in \{1, 3, 5, 10\}$. For each attack-budget combination, perturbations are computed targeting 100 nodes, with the objective of inducing misclassification at each target. The targets are selected according to the Nettack scheme: a quarter are nodes classified with high confidence, a quarter are classified with low confidence, and the rest are randomly sampled.

Baseline Detector. The benchmark uses the degree-distribution test included in the "Nettack" paper [9]. Formally, they use a likelihood ratio approach to test whether two graphs — the original G_{orig} and the new sample G_{new} — come from the same degree distribution. This yields the test statistic $\Lambda(G_{orig}, G_{new})$, which follows a χ^2 distribution with 1 degree of freedom. Within their paper the authors argued that the traditional hypothesis testing framework did not reflect the thought process that most analysts would go through. Rather than being interested in the Type I error that hypothesis testing controls (which would correspond to a benign attack being mistaken for malicious), they argue that a human trying to ascertain whether a network has been tampered with would be more interested in the Type II error rate. That is, whether a truly malicious attack has been missed by the test. While this cannot be easily computed, they rely on the inverse relationship between Type I and Type II errors, such that they set a critical p-value at 0.95, as opposed to the traditional 0.05. They use this as a conservative test to make sure that their algorithm does not create obvious changes to the graph. Thus they only accept perturbations where

$$\Lambda(G_{orig}, G_{new}) < \tau \approx 0.004 \tag{1}$$

where the threshold τ comes from the 5th percentile of the $\chi^2(1)$ distribution.

Graph comparison tools. Portrait Divergence [2] summarises a graph through its *portrait* matrix $\mathbf{B}$, where B_{lk} counts the number of nodes with exactly k other nodes at shortest-path distance l. The Jensen-Shannon divergence between the row-normalised portraits of two graphs yields a score in $[0, 1]$, with higher values indicating greater structural dissimilarity. Therefore, higher scores are treated as evidence of attack.

Graphlet Disagreement runs the ORCA binary [7] on size-4 graphlets to obtain per-node orbit counts for each graph. Each node's orbit-count vector is normalised into a distribution over the 15 size-4 orbits, then reweighted by the inverse frequency of each orbit across the graph, so that common orbits associated with hub nodes contribute less than rare, locally-informative ones. The disagreement score is the average Euclidean distance between the reweighted orbit distributions of matched nodes across the two graphs. Lower values indicate a more similar local subgraph structure.

The D-measure [8] compares graphs through the global distribution of shortest-path lengths. For each node i, we define a probability distribution P_i where $P_i[j]$ is the fraction of nodes at shortest-path distance j from i. Let $\mu = \frac{1}{n}\sum_i P_i$ be the mean profile over all nodes. The within-graph *node dispersion* is $\delta = \mathrm{JS}(\{P_i\}, \mu)/\log d$, where JS is the Jensen–Shannon divergence and d is the graph diameter. The *node distance* between two graphs G_1, G_2 with mean profiles μ_1, μ_2 is $\Delta = \mathrm{JS}(\mu_1, \mu_2)/\log 2$. The simplified D-measure is then

$$\mathcal{D}(G_1, G_2) = \frac{\sqrt{\Delta} + \left|\sqrt{\delta_1} - \sqrt{\delta_2}\right|}{2}, \tag{2}$$

where δ_1, δ_2 are the within-graph dispersions of G_1 and G_2 respectively, each computed using that graph's own diameter d for normalisation. Higher values of $\mathcal{D}$ indicate greater structural dissimilarity. BFS profiles are approximated by sampling $k = 300$ source nodes per graph to reduce computation. Higher scores are treated as evidence of attack.

Link predictor function. In order to select a good baseline link predictor for the structured null perturbations, three neighbourhood-based indices are evaluated on both Cora and CiteSeer. The indices considered were Adamic-Adar, Preferential Attachment, and Jaccard Coefficient. For each dataset,15% of edges are held-out as positive test pairs, and an equal number of non-edges are sampled as negatives. Each index was then scored on the remaining training graph, and performance was measured by AUC-ROC and Average Precision. Adamic-Adar and Jaccard performed similarly on held-out AUC-ROC and Average Precision, while Preferential Attachment was less competitive (Table 1); Adamic-Adar was retained as the implementation choice for its interpretability and standard use in triadic-closure modelling, rewarding pairs of nodes that share common neighbours with low degree, with the similarly-performing alternative (Jaccard) left as a natural robustness check for future work.

Table 1: Performance of classical link prediction scorers across datasets.

Dataset	Scorer	AUC-ROC	Avg. Precision
Cora	Adamic-Adar	0.6948	0.6952
	Preferential Attachment	0.6285	0.6815
	Jaccard	0.6920	0.6786
CiteSeer	Adamic-Adar	0.6996	0.7008
	Preferential Attachment	0.6143	0.6925
	Jaccard	0.6987	0.6972

Null distribution calibration. For each budget B, the median number of edge modifications between the original and the perturbed graph instances is computed via symmetric edge difference. Thus,

for each budget, $N_{\text{null}} = 100$ null bootstraps are created by applying structured perturbations in accordance with the Adamic-Adar index. Candidate edges for deletion are ranked by low Adamic-Adar score (structurally peripheral), and candidate non-edges for addition are ranked by high score (plausible future links), with selection probabilities proportional to these rankings. The Adamic-Adar index between nodes u and v is $\text{AA}(u, v) = \sum_{w \in \Gamma_u \cap \Gamma_v} (\log |N(w)|)^{-1}$, where Γ_u denotes the neighbourhood of node u.

Power estimation and confidence intervals. For Portrait Divergence and D-Measure, the detection threshold at significance level α is set at the $(1 - \alpha)$-quantile. For Graphlet disagreement, this is reversed to test against the α-quantile of the empirical null distribution. This reversal is used because Graphlet Disagreement tends to be lower for attacked graphs than graphs generated with the Adamic-Adar index. For the Degree Distribution test, the threshold is the $(1 - \alpha)$-quantile of $\chi^2(1)$. Power is estimated as the fraction of attack graph instances whose score exceeds the threshold. Bootstrap confidence intervals are computed over $B_{\text{CI}} = 500$ resampled replicates of both the null and attack score arrays, with 95% intervals given by the 2.5th and 97.5th percentiles of the resulting power distribution.

The full code is available on the project repository: https://github.com/S-Caio/ina_project/tree/main.

3 RESULTS

Attack success on victim models. Table 2 reports mean ASR (Attack Success Rate) across budgets (per-budget variance is negligible; max range 0.19), revealing broadly comparable effectiveness across all three attacks on both datasets. GCN is the most vulnerable victim, likely reflecting its use as the surrogate model; GAT is the most robust, with ASR as low as 0.34 on CiteSeer. The parity between attacks underscores a fundamental asymmetry: the sole advantage of Nettack and FGA over RND is structural evasion, not attack power.

Table 2: Mean ASR across $B \in \{1, 3, 5, 10\}$. Clean accuracies — Cora: GCN 0.821, GAT 0.835, GIN 0.770; CiteSeer: GCN 0.713, GAT 0.705, GIN 0.639.

		Cora	CiteSeer
FGA	GCN	0.548	0.588
	GAT	0.455	0.389
	GIN	0.558	0.403
Nettack	GCN	0.592	0.565
	GAT	0.507	0.403
	GIN	0.555	0.464
RND	GCN	0.580	0.597
	GAT	0.536	0.388
	GIN	0.591	0.464

Score distributions. Figure 1 shows the distribution of Portrait Divergence scores between attacked graphs and the original graph, overlaid with the same comparison but using our link-prediction metric to generate null graphs with a similar magnitude of change.

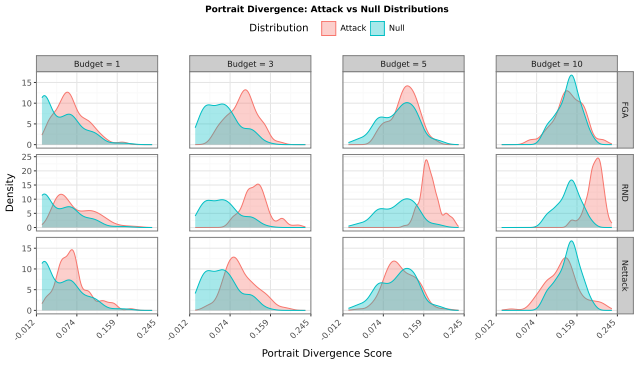


Figure 1: Portrait Divergence Distribution across attacks and budgets for Cora.

Attack scores exhibit a slightly higher mean than the null distribution, though the degree of separation varies by attack: FGA and Nettack are positioned considerably closer to the null than RND. A progressive rightward shift is observed in both distributions as the budget increases, reflecting the growing structural distance from the original graph as more edges are modified. That this shift is shared by both null and attack distributions is expected, though it confirms that the null is well-calibrated to the scale of perturbation at each budget.

Figure 2 shows the opposite pattern: attacked graphs are found to be *more* similar to the original than null graphs, as measured by Graphlet Disagreement. This is interpreted as evidence that gradient-based attacks, while trained to preserve degree structure, incidentally conserve local subgraph motifs — whereas Adamic-Adar null perturbations alter subgraph structure more aggressively. Despite this reversal, separation between attack and null distributions is found to be sharper under Graphlet Disagreement than under Portrait Divergence, making it a more powerful discriminator in practice.

This reversal necessitates a change in test direction: rather than flagging candidate graphs whose disagreement score exceeds the null, graphs whose score falls *below* the null are treated as anomalous. An important consequence follows — a graph identical to the original would be flagged as perturbed under this criterion. Graphlet Disagreement should therefore be applied *conditional* on a substantial number of changes having already been observed; it identifies whether those changes are structurally stealthy, not whether changes have occurred at all.

Null distribution. The null distributions are generated relying on classical link prediction scores as proxies for plausible graph change: they estimate which non-edges are likely to form and which existing edges are structurally weak. Adamic-Adar, Jaccard, and Preferential Attachment are compared on held-out links. Adamic-Adar and Jaccard performed similarly, while Preferential Attachment was less competitive. Adamic-Adar is used for the current implementation as a standard and interpretable method, while noting that stronger link-prediction models could be explored in future work.

Detection power. Detection power ($1 - \beta$, where β represents the type II error rate) is assessed in Figure 3. Across all measures and

 DOI: https://doi.org/10.26493/scores26.10

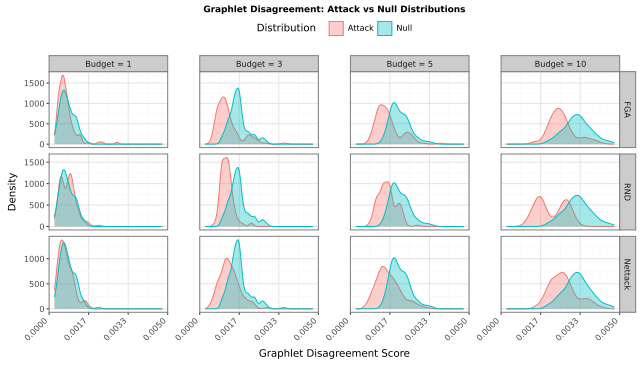


Figure 2: Graphlet Disagreement Distribution across attacks and budgets for Cora. Type I error is the critical p-value that is allowed and used as a threshold. Power refers to the share of malicious attacks that the test catches at a given p-value threshold.

attacks, the proposed framework substantially exceeds the degree distribution baseline implicit in Nettack's stealth constraints, and power is found to increase monotonically with budget.

Performance is, however, found to be dataset dependent. On CiteSeer, D-Measure achieves the highest power across all types of attacks, with Portrait divergence being competitive at low budgets and Graphlet Disagreement scaling better at higher budgets. On Cora, D-Measure's performance collapses, while Graphlet Disagreement emerges as the strongest overall performer. Portrait Divergence detects random attacks reliably on both datasets but offers limited power against more sophisticated attacks.

Taken together, Graphlet Disagreement is identified as the most consistent detector across datasets and attack types, though the caveats regarding test direction discussed above apply.

4 DISCUSSION

Results reveal a clear detectability hierarchy among the three attacks. RND, as the simplest attack, is also the most easily detectable, which lends credence to our results. Nettack is consistently the hardest to detect, irrespective of comparison method. This is likely because, out of all attack types, it is the only one that sets hard limits on the types of changes it makes. FGA and RND rely on the budget to constrain them, but that proves to be much too permissive.

The central insight provided by this endeavour is that adversarial attack methods modify the graph in ways that are structurally different from what one might expect from link prediction indices. Given the measures of link prediction as a plausible path that networks take through time, it becomes easier to identify and investigate attacks.

However, the technique rests on assumptions that may prove to be somewhat unrealistic. First, and most importantly, the null baseline, which is based on the Adamic-Adar index , may not be sufficiently realistic and may not translate to other types of graphs. If temporal data is available, then a GNN may be used to generate transition matrices for graphs, such that more realistic nulls are constructed, tailored to the network in question. Second, the tests

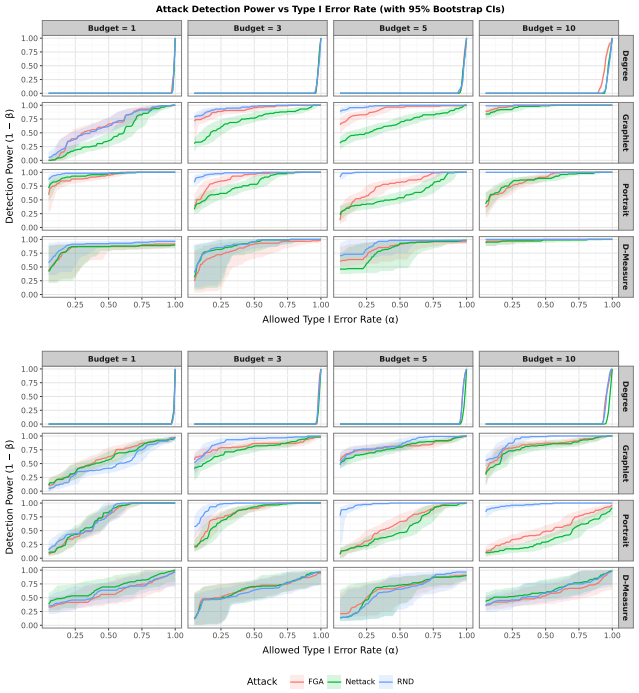


Figure 3: Power curves across datasets and budgets. Top: CiteSeer. Bottom: Cora.

can sometimes give degenerate results. For example, for graphlet disagreement, the attacks are usually below the null distribution, which makes us test whether the observed statistic is sufficiently below the null; but a graph that is identical to the original would surely be flagged as malicious under this test. Therefore, we stress that the performance of the test should be *conditional* on there having been changes to the graph. This is a structural limitation of the measure, not an artefact of tuning: because gradient-based attacks are optimised to minimise local disruption, any sufficiently unperturbed graph — attacked or not — will resemble the original more than a random structured null does. Graphlet Disagreement is therefore best understood as a *second-stage* discriminator, applied only once a change (via Degree Distribution, Portrait Divergence, or an external signal) has already been flagged, rather than as a standalone detector.

The use of a structured null distribution represents a deliberate methodological choice. A purely random null flags any sufficiently large perturbation as anomalous regardless of structural plausibility; by contrast, null perturbations generated via the Adamic-Adar index are calibrated to reflect credible edge dynamics, such that rejection of the null constitutes stronger evidence of malicious intent. A further advantage is that no temporal data are required: given any static graph, a link prediction index yields a plausible set of candidate perturbations. Where greater realism is desired, temporal observations or generative models such as the Stochastic Block Model could be substituted, and their comparison against the present benchmark is identified as a natural direction for future work.

REFERENCES

[1] Lada A Adamic and Eytan Adar. 2003. Friends and Neighbors on the Web. *Social Networks* 25, 3 (July 2003), 211–230. https://doi.org/10.1016/S0378-8733(03)00009-1

[2] James P. Bagrow and Erik M. Bollt. 2019. An Information-Theoretic, All-Scales Approach to Comparing Networks. *Applied Network Science* 4, 1 (Dec. 2019), 45. https://doi.org/10.1007/s41109-019-0156-x arXiv:1804.03665 [cs.SI]

[3] Battista Biggio and Fabio Roli. 2018. Wild Patterns: Ten Years After the Rise of Adversarial Machine Learning. *Pattern Recognition* 84 (Dec. 2018), 317–331. https://doi.org/10.1016/j.patcog.2018.07.023 arXiv:1712.03141 [cs.CV]

[4] Jinyin Chen, Yangyang Wu, Xuanheng Xu, Yixian Chen, Haibin Zheng, and Qi Xuan. 2018. Fast Gradient Attack on Network Embedding. https://arxiv.org/abs/1809.02797v2.

[5] Hanjun Dai, Hui Li, Tian Tian, Xin Huang, Lin Wang, Jun Zhu, and Le Song. 2018. Adversarial Attack on Graph Structured Data. https://doi.org/10.48550/arXiv.1806.02371 arXiv:1806.02371 [cs.LG]

[6] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and Harnessing Adversarial Examples. https://doi.org/10.48550/arXiv.1412.6572 arXiv:1412.6572 [stat.ML]

[7] Tomaz Hocevar and Janez Demsar. 2013. A Combinatorial Approach to Graphlet Counting. *Bioinformatics (Oxford, England)* 30 (Dec. 2013). https://doi.org/10.1093/bioinformatics/btt717

[8] Tiago A. Schieber, Laura Carpi, Albert Díaz-Guilera, Panos M. Pardalos, Cristina Masoller, and Martín G. Ravetti. 2017. Quantification of Network Structural Dissimilarities. *Nature Communications* 8, 1 (Jan. 2017), 13928. https://doi.org/10.1038/ncomms13928

[9] Daniel Zügner, Amir Akbarnejad, and Stephan Günnemann. 2018. Adversarial Attacks on Neural Networks for Graph Data. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2847–2856. https://doi.org/10.1145/3219819.3220078 arXiv:1805.07984 [stat.ML]

 DOI: https://doi.org/10.26493/scores26.10

 DOI: https://doi.org/10.26493/scores26.10

Index of Authors

Antolinc, Luka, 9

Balant, Denis, 41
Baldouski, Daniil, 53
Berkovič, Klemen, 5, 13, 17
Bošković, Borko, 17
Boškovič, Borko, 9, 31
Bržan, Nik, 5
Bujić, David, 5

Catana, Diana Ioana, 75
Chonev, David, 63

Damiš, Nik, 13
Drozg, Nik, 13

Forštnarič, Oskar, 9, 31
Fras, Nejc, 27

Golob, Aljoša, 27

Hameršak, Anže, 9
Hristova, Sara, 17

Ignjatovic, Vuk, 67

Jereb, Janez Ignacij, 45
Jodl, Anja, 35

Krabonja, Enej, 9, 31
Krész, Miklós, 53

Lah, Tjaša, 31

Marolt, Maks, 31
Matič, Luka, 71
Medved, Pavel, 13
Mićović, Irina, 17

Nikov, Mitko, 35

Pliberšek, Miha, 9, 31
Polajžer, Igor, 13
Popič, Jan, 27
Požar, Matic, 49

Rajter, Matej, 27
Rogelj, Peter, 67, 71

Sestito, Pietro, 75
Simon, Caio, 75
Stevanović, Đorđe, 75

Toropenko, Alla, 23
Tupkalenko, Dmytro, 53
Tushych, Olha, 23

Vališer, Jan, 27
Vidovič, Blaž, 13

Zugan, Dani, 57

Šišić, Nedim, 67
Šoberl, Domen, 63

Organisers and sponsors:

















