

# **RĪGAS TEHNISKĀ UNIVERSITĀTE**

Datorzinātnes un informācijas tehnoloģijas fakultāte  
Lietišķo datorsistēmu institūts

**Ēvalds Urtāns**

Doktora studiju programmas “Datorsistēmas” doktorants

## **FUNKCIJU FORMĒŠANA DZIĻAJĀ MAŠĪNAPMĀCĪBĀ**

**Promocijas darba kopsavilkums**

Zinātniskais vadītājs  
profesors Dr. habil. sc. ing.  
Agris Nikitenko

RTU Izdevniecība  
Rīga 2021

## ANOTĀCIJA

Kļūdas funkciju konstruēšana dziļās mašīnāpmācības uzdevumiem iegūst aizvien lielāku interesi, jo empīriski iegūtās kļūdas funkcijas ieguvušas labākus rezultātus nekā standarta kļūdas funkcijas, kuras tika izvestas no klasiskās matemātikas teorijas. Šis darbs apraksta kļūdas funkciju un saistītu metožu ietekmi uz dziļo stimulēto mašīnāpmācību un dziļo metriku mašīnāpmācību. Jaunizveidotā MDQN kļūdas funkcija pārspēja DDQN kļūdas funkciju PLE datorspēļu vidēs, un jaunizveidotā Eksponenciālās Trijotnes kļūdas funkcija pārspēja standarta Trijotnes kļūdas funkciju ar biežāk lietotajiem papildinājumiem sejas re-identifikācijas uzdevumā, izmantojot VGGFace2 datu kopu. Tā sasniedz 85.7% precizitāti, izmantojot nulles-šāviena metodi. Šajā darbā iekļauts arī jaunizveidots UNet-RNN-Skip modelis, kurš uzlabo veiktspēju vērtību funkcijai ceļu plānošanas uzdevumos. Tas iegūst tādu pašu rezultātu, kā Vēriņu Iterāciju algoritms 99.8% gadījumu, un var tikt apmācīts uz 32x32 izmēra kartēm, bet pielietots uz lielākām kartēm, kā, piemēram 256x256. Jaunizveidotās kļūdas funkcijas un metodes ir veiksmīgi pielietotas vairākos komerciālos produktos sejas un balsis re-identifikācijā, audio signāla attīrīšanā un hromatogrāfijā.

# Saturs

<b>SAĪSINĀJUMI</b>	<b>4</b>
<b>1. VISPĀRĒJAIS DARBA APRAKSTS</b>	<b>6</b>
1.1. Tēmas aktualitāte . . . . .	6
1.2. Darba mērķis un uzdevumi . . . . .	8
1.3. Aizstāvamās tēzes . . . . .	9
1.4. Pētījuma metodika . . . . .	9
1.5. Zinātniskā novitāte un autora ieguldījums . . . . .	13
1.6. Darba struktūra un apjoms . . . . .	14
<b>2. FUNKCIJU VEIKTSPĒJAS UZLABOŠANA, IZMANTOJOT DZIĻO MAŠĪNAPMĀCĪBU</b>	<b>17</b>
2.1. Vērtību Iterāciju algoritms . . . . .	17
2.2. ConvNet un UNet modeļi . . . . .	20
2.3. RNN modeļi . . . . .	24
2.4. UNet-RNN-Skip modelis . . . . .	27
<b>3. FUNKCIJU FORMĒŠANA DZIĻAJAI STIMULĒTAJAI MAŠĪNAPMĀCĪBAI</b>	<b>31</b>
3.1. Q-Vērtību funkcija un politikas gradientu funkcijas . . . . .	31
3.2. DQN kļūdas funkcija . . . . .	32
3.3. MDQN kļūdas funkcija . . . . .	35
<b>4. FUNKCIJU FORMĒŠANA DZIĻAJAI METRIKU APMĀCĪBAI</b>	<b>38</b>
4.1. Nulles-šāvienu mašīnāpmācība un re-identifikācijas uzdevums . . . . .	38
4.2. Trijotnes Kļūdas funkcija . . . . .	41
4.3. Eksponenciālā kļūdas funkcija . . . . .	42
<b>5. EKSPERIMENTĀLIE REZULTĀTI UN PRAKTISKIE PIELIETOJUMI</b>	<b>45</b>
5.1. UNet-RNN-Skip rezultāti . . . . .	45
5.2. MDQN rezultāti . . . . .	49
5.3. Eksponenciālās Trijotnes kļūdas funkcijas rezultāti . . . . .	55
5.4. Darba praktiskais lietojums . . . . .	58
<b>6. Tālākie pētījumi</b>	<b>61</b>

7. Secinājumi	62
LITERATŪRA	64

## SAĪSINĀJUMI

A3C – asinhronie aktiera–kritiķa aģenti (Asynchronous Actor–Critic Agents)  
ACER – aktiera–kritiķa aģenti ar pieredzes atmiņu (Actor–Critic with Experience Replay)  
AI – mākslīgais intelekts (Artificial Intelligence)  
ASR – automātiskā runas atpazīšana (Automatic Speech Recognition)  
ConvNet, CNN – konvolūciju neironu tīkli (Convolutional Neural Network)  
CPU – centrālais procesors (Central Processing Unit)  
DQN – dziļais Q-tīkls (Deep Q–Network)  
DDPG – dziļais deterministiskais politikas gradients (Deep Deterministic Policy Gradient)  
DDQN – dubultais dziļais Q-tīkls (Double Deep Q–Network)  
DNN – dziļais neironu tīkls (Deep Neural Network)  
DML – dziļā metriķu apmācība (Deep Metric Learning)  
GPU – grafiskais procesors (Graphical Processing Unit)  
GRU – periodiskā šūna ar vārtiem (Gated Recurrent Unit)  
HPC – augstas veiktspējas skaitļošanas klasteris (High Performance Cluster)  
HPLC – augstas veiktspējas šķidrā hromatogrāfija (High Performance Liquid Chromatography)  
IMAPLA – svērtā svarīguma aktiera-skolnieķa arhitektūra (Importance Weighted Actor–Learner Architecture)  
LSTM – ilgtermiņa–īstermiņa atmiņas šūna (Long Short–Term Memory)  
MDQN – vairāku dziļo Q-modeļu kļūdas funkcija (Multi Deep Q–Network)  
MERLIN – atmiņas, stimulētas mašīnāpmācības un secinājumu tīkls (Memory, RL, and Inference Network)  
ML – mašīnāpmācība (Machine Learning )  
PLE – PyGame apmācību vides (PyGame Learning Environment )  
PPO – tuvinātā politikas optimizācija (Proximal Policy Optimization )  
RL – stimulētā apmācība (Reinforcement Learning)  
RNN – periodiskie neironu tīkli (Recurrent Neural Network)  
ResNet – pārpalikuma konvolūciju neironu tīkli (Residual Convolutional Neural Network )  
SLAM – vienlaicīga lokalizācija un kartēšana (Simultaneous Localization and Mapping)  
SLR – sistemātisks literatūras apskats (Systematic Literature Review)  
TD – atšķirība laikā (Temporal Difference)  
UNet – U-formas konvolūciju neironu tīkli (U–Shaped Convolutional Neural

Network)

VI – vērtību iterāciju algoritms (Value Iteration Algorithm)

VIN – vērtību iterāciju tīkls (Value Iteration Network)

# 1. VISPĀRĒJAIS DARBA APRAKSTS

## 1.1. Tēmas aktualitāte

Pēdējā desmitgadē dziļā mašīnāpmācība ir pārspējusi klasiskās mašīnāpmācības metodes, lai iegūtu tuvinātus risinājumus sarežģītām matemātiskām funkcijām daudzdimensiju datu kopām [115] [48] [118] [126]. Dziļās mašīnāpmācības modeļi nereti tiek izmantoti ar nepārraudzīto vai daļēji pārraudzīto apmācību, lai iegūtu matemātisku funkciju tuvinājumus, kuri apraksta sakarības datu kopā [69] [83] [19] [22]. Šādi modeļi tiek apmācīti, izmantojot kļūdas funkcijas, lai iegūtu latentas reprezentācijas ievades datiem, kuras raksturo to īpašības, sakarības un loģiku. Senāk datu īpašību inženierija un ekspertu sistēmas sakarību atpazīšanai bija : svarīgas daļas no pētījumiem, bet šodien kļūdas funkcijas un modeļu arhitektūras ir kļuvušas par aktuāliem pētāmiem objektiem. [57].

Ar dziļajām mašīnāpmācības metodēm ir iespējams iegūt augstāko precizitāti attēlu klasifikācijas uzdevumos [107] [45], dabīgās valodas modelēšanas uzdevumos [10] [95], runas atpazīšanas uzdevumos [78] [85], laika sēriju uzdevumos [9] [72] un citos uzdevumos, kur ievades dati satur daudz dimensiju.

Dziļās mašīnāpmācības metodes sasniedz arī augstākos rezultātus stimulētās mašīnāpmācības uzdevumos datorspēļu vidēs un robotikā, kur vadības signālus senāk spēja dot tikai cilvēks [35] [103] [21] [76]. Modernās dziļās mašīnāpmācības metodes piedāvā arī līdzīgu caurspīdību kā klasiskās statistiskās metodes lēmumu analīzei [63] [93].

Disertācijas mērķis ir izstrādāt jaunas kļūdas funkcijas un modeļus, lai tuvinātu jau zināmas funkcijas, balstoties uz empīriskiem pētījumiem, lai formētu funkciju īpašības un modeļus augstākai veiktspējai katrai datu kopai. Vēsturiski lielākā daļa kļūdu funkciju ir iegūtas no statistikas, informācijas vai varbūtību teorijām, bet salīdzinoši nesen empīriski atrastas kļūdas funkcijas var aizstāt esošās un iegūt labākus rezultātus [61] [89].

Lai sasniegtu mērķi, darbā tika apskatītas sekojošas, aktuālas tēmas:

1. Kļūdu funkciju formēšana dziļajā stimulētajā mašīnāpmācībā un dziļajā metriku mašīnāpmācībā. Kļūdas funkcijas dziļajā stimulētajā mašīnāpmācībā tika pārbaudītas datorspēļu vidēs, lai sasniegtu ātrāk augstākus rezultātus, kuri tiek novērtēti ar skaitlisku punktu vērtību. Savukārt kļūdas funkcijas dziļajā metriku mašīnāpmācībā tika pārbaudītas sejas

re-identifikācija un balss re-identifikācijas uzdevumos.

2. Modeļa arhitektūras un apmācību procesa izstrāde, lai tuvināti atrisinātu vērtību funkciju un paātrinātu Vērtību Iterāciju Algoritma darbību.
3. Eksperimenāla analīze jaunajām kļūdas funkcijām un metodēm, izmantojot akadēmiskās, sintētiskās un privātās datu kopas un vides, lai pārbaudītu to pielietojumu praktiskos risinājumos.

Jaunās kļūdas funkcijas, kuras tika izstrādātas šī darba ietvaros dziļajai metriku mašīnāpmācībai sasniedz 85.7% precizitāti seju re-identifikācijas uzdevumos, savukārt, standarta kļūdas funkcijas šajā pašā uzdevumā sasniedz 78.6% precizitāti. Šī precizitāte tika sasniegta izmantojot nulles-šāviena mašīnāpmācību [23] [116], kur sejas re-identifikācijas uzdevuma gadījumā, sejas, kuras tika iekļautas apmācību datu kopā, netika iekļautas testēšanas datu kopā, bet tā vietā precizitāte tiek rēķināta atrodot vislīdzīgākās sejas starp reģistrācijas seju paraugiem un neklasificētiem testa seju paraugiem. Šīs pašas kļūdas funkcijas tika pielietotas arī balss re-identifikācijas uzdevumam, kur tika sasniegta 88.4% precizitāte uz privātas datu, kopas, kura tiek izmantota komerciālā produktā. Līdzīgi, jaunās kļūdas funkcijas, kuras tika izstrādātas dziļajai stimulētajai mašīnāpmācībai sasniedza augstākus rezultātus PyGame Learning Environment vidē. Šīs pašas kļūdas funkcijas ir veiksmīgi tikušas pielietotas arī komerciālā produktā analītiskajā organiskajā ķīmijā, lai veiktu šķīdinātāja koncentrācijas gradienta optimizāciju, kura nepieciešama, lai atdalītu savienojumus paraugos. Visbeidzot, tika izstrādāti arī jauni dziļās mašīnāpmācības modeļi, lai uzlabotu veikspēju jau zināmajam Vērtību Funkcijai, kuru pielieto ceļu plānošanas uzdevumos mobilajiem robotiem. Jaunais modelis ir spējīgs iegūt tuvinātu rezultātu Vērtību Funkcijai, izmantojot paralēlo skaitļošanu. Modelis ir var iegūt vērtību karti vairākas reizes ātrāk par standarta metodi, izmantojot Vērtību Iterāciju Algoritmu.



## 1.2. Darba mērķis un uzdevumi

Disertācijas mērķis ir uzlabot dziļās mašīnāpmācības veiktspēju un apmācību rezultātus dažādiem pielietojumiem, izmantojot jaunizveidotas kļūdas funkcijas un modeļu arhitektūras priekš dziļo metriku apmācības un stimulētās mašīnāpmācības.

Jaunizveidotajām kļūdas funkcijām, kuras aprakstītas šajā darbā, jāspēj ātrāk konverģēt apmācības laikā, jāiegūst labākus rezultātus praktiskiem pielietojumiem un jābūt pielietojamām dažādos uzdevumos, sākot ar klasifikāciju sejas re-identifikācijai, analītiskajai ķīmijai un arī aģentu vadībai dažādās sarežģītās vidēs kā, piemēram, 3D datorspēlēs.

Konverģence dziļajā mašīnāpmācībā tiek sasniegta, kad izpildās  $|\frac{\mathcal{L}(f_{\theta}(x),y)_i}{\mathcal{L}(f_{\theta}(x),y)_{i-1}}| < \delta$ , kur relatīvā starpība starp vidējo vērtību kļūdas funkcijai  $\mathcal{L}$  ņemot vērā ievades datus  $x$  un patiesos datus  $y$  pašreizējā epochā un iepriekšējā epochā ir mazāka par  $\delta$ .

Atkarībā no datu kopas tīrības  $\delta$  var būt robežās no 0.1-0.001. Kļūdas funkcijas vērtībām nepieciešams samazināties, balstoties uz datu kopu vai apmācību vidi. Parasti kļūdas funkcijām ir globālais minimums ar vērtību nulle, izņemot stimulētajā mašīnāpmācībā, kur nereti nav iespējams noteikt patiesos stāvokļus, kuri iegūst augstāko balvas vērtību vidē.

Uzdevumi, kuri tika veikti disertācijas ietvaros:

1. Veikt literatūras analīzi jau eksistējošām kļūdas funkcijām, kuras ir līdzīgas DQN [68] un Tripleta Kļūdas funkcijām [23].
2. Izstrādāt jaunas kļūdas funkcijas, kuras ir līdzīgas DQN un Triplet Kļūdai, izmantojot uz nulles šāviena apmācību.
3. Izveidot jaunu modeli, lai iemācītos tuvināt Vērtības Funkciju, kura tiek izmantota Vērtību Iterāciju algoritmā [88].
4. Analizēt rezultātus jaunizveidotajai DQN funkcijai dažādās datorspēļu vidēs un analītiskās ķīmijas uzdevumos.
5. Analizēt rezultātus jaunizveidotās Tripleta Kļūdas funkcijas sejas re-identifikācijas uzdevumu veikšanā.
6. Analizēt rezultātus jaunizveidotajam modelim, lai iegūtu Vērtību Funkcijas tuvinājumu un salīdzināt tos ar pilnā Vērtību Iterāciju algoritma rezultātiem.

7. Publicēt pētījumu rezultātus zinātniskās publikācijās un pievienot tās disertācijai

### 1.3. Aizstāvamās tēzes

Disertācijā tiek aizstāvētas sekojošas tēzes:

1. Jaunizveidotā MDQN kļūdas funkcija Q-vērtību stimulētās māšīnāpmācības uzdevumos pārspēj DQN kļūdas funkcijas.
2. Jaunizveidotā Eksponenciālā Tripletu kļūdas funkcija dziļo metriku apmācības uzdevumos pārspēj klasisko Tripletu kļūdas funkciju.
3. Jaunizveidotais UNet-RNN-Skip modelis uzlabo veikspēju Vērtību Funkcijai, pielietojot to, lai atrisinātu Vērtību Iterācijas algoritma uzdevumu.
4. Jaunizveidotās MDQN un Tripletu Kļūdas funkcijas var tikt pielietotas praktiskos risinājumos seju re-identifikācijai, runātāju re-identifikācijai, audio trokšņu noņemšanai un hromatogrāfijas uzdevumiem analītiskajā ķīmijā.

### 1.4. Pētījuma metodika

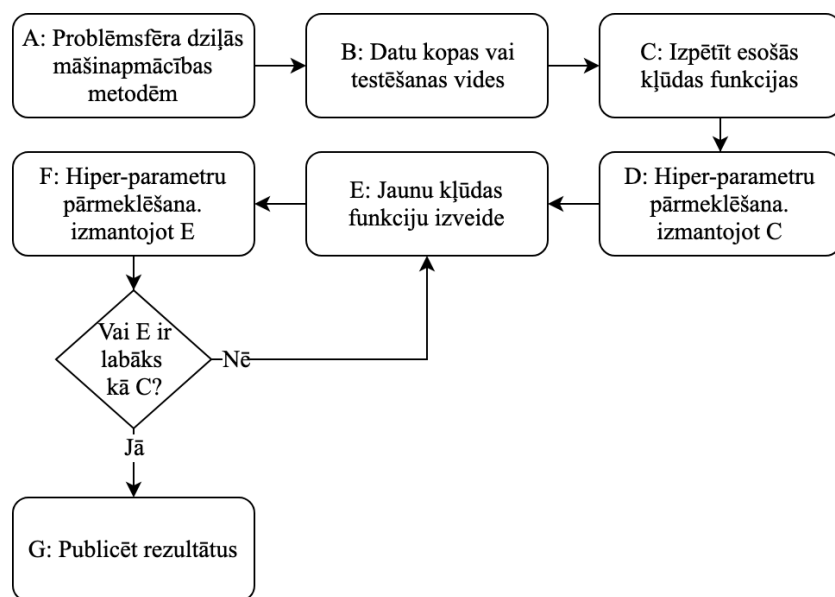
Disertācijas mērķi tika sasniegti, izmantojot analītiskos un eksperimentālos pētījumus, kuri tika publicēti zinātniskās publikācijās kā parādīts 1.6. nod.

Disertācijā aprakstītajos pētījumos tika izmantotas eksperimentālās un analītiskās pētījumu metodes.

Kvantitatīvās un kvalitatīvās pētījumu metodes tika izmantotas, lai analizētu zinātnisko literatūru, jau eksistējošas un arī jaunas metodes.

Pētījumos primāri tika izveidotas jaunas kļūdas funkcijas un metodes, lai uzlabotu DML (Deep Metric Learning) un RL (Reinforcement Learning) modeļu veikspēju un rezultātus.

Jaunizveidoto kļūdas funkciju izstrādes process shematiski attēlots 1. att.



1. att. Vienkāršota darbību shēma jaunu kļūdas funkciju izveidei dziļajai māšīnāpmācībai.

Funkciju izveides process sākas ar problēmsfēras (A) izvēli, kura būtu atbilstoša dziļajai māšīnāpmācībai. Šajā darbā problēmsfēras ietver DML balstītus latentos vektorus sejas un balss re-identifikācijas uzdevumiem; RL, izmantojot Q-Vērtību funkcijas aģentu politikas apmācībai datorspeļu vidēs; hromatogrāfijas šķīdinātāju gradientu atrašana, lai labāk atdalītu parauga vielas pa sastāvdaļām analītiskajā ķīmijā, izmantojot RL.

Pēc tam nepieciešams atrast piemērotas datu kopas vai testēšanas vides (B). Šajā pētījumā tika izmantotas seju re-identifikācijas un attēlu klasifikācijas datu kopas un arī datorspeļu vides RL aģentu testēšanai.

Tad jau eksistējošas kļūdas funkcijas (C) tika analizētas, izmantojot zinātnisko literatūru un jaunākās publikācijas. Tālāk šīs funkcijas tika realizētas dziļās māšīnāpmācības satvarā PyTorch [79] un tika pētītas to īpašības, ņemot vērā to parametrus.

Jau esošās kļūdas funkcijas tika testētas uz izvēlētajām datu kopām vai vidēm, pārmeklējot labākās hiper-parameteru kombinācijas (D). Hiper-parametri šajā gadījumā ir modeļa parametri, kurus modelis pats nemācās apmācības laikā. Katrs apmācību mēģinājums tiek izpildīts līdz tas sasniedz kļūdas konverģenci pret apmācību un testēšanas kļūdas funkciju skaitliskajām vērtībām. Konverģence dziļajā māšīnāpmācībā tiek sasniegta, kad kļūdas fun-

kcijas skaitliskā vērtība starp epohām vairs ievērojami nemainās. Gan apmācību, gan testēšanas kļūdas funkcijas rezultāta izmaiņa, piemēram, paliek zem 0.1% robežām, kā arī testēšanas kļūdas funkcijas rezultāts nedrīkst sākt pieaugt. Ja testēšanas kļūdas funkcijas rezultāts sāk pieaugt, tad tas var nozīmēt, ka ir notikusi pār-apmācīšanās. Epoha dziļajā mašīnāpmācībā ir viena iterācijā, kuras laikā tiek apmācīti modeļa parametri ar kļūdas funkciju, izmantojot visus apmācību un testēšanas datu paraugus. Katrai kļūdas funkcijai ir nepieciešams veikt pilnās pārlases pārmeklēšanu hiper-parametriem, jo vieni un tie paši hiper-parametri dažādu formu kļūdas funkcijām neiegūs labākos rezultātus vienlīdzīgi. Bez hiper-parametru pārmeklēšanas salīdzināšanas process nebūtu objektīvs.

Tālāk notiek radošs process, izstrādājot jaunu kļūdas funkciju, kura laikā tiek ņemta vērā matemātikas teorija un citu funkciju īpašības. Tiek analizētas funkciju formas un vēlamās īpašības, balstoties uz ievades parametriem un kļūdas gradientiem. Šis process var arī tikt automatizēts, izmantojot funkciju konstruēšanas algoritmus un optimizācijas metodes kā, piemēram, Beijesa optimizāciju, Monte-Carlo optimizāciju vai ģenētisko algoritmu optimizāciju. Praksē kļūdas funkciju atrašana, izmantojot optimizācijas metodes ir nepraktiska, jo hiper-parametru pilnās pārlases pārmeklēšana katrai kļūdas funkcijai prasa ļoti daudz laika, bet šādā gadījumā būtu nepieciešams arī pārmeklēt hiper-parametrus arī pašai funkciju konstruēšanas metodei. Hiper-parametru pārmeklēšana tika veikta uz RTU HPC (High-Performance Cluster), kur tā prasīja vairākas nedēļas un pat mēnešus katrai datu kopai, kļūdas funkcijai, apmācību metodei un modeļa arhitektūrai.

Tālāk, kā jau minēts, tika veikta hiper-parametru pārmeklēšana (F) jaunajai kļūdas funkcijai tādā pašā veidā, kā jau eksistējošām kļūdas funkcijām. Ja jaunā kļūdas funkcija ieguva labākus rezultātus nekā eksistējošās kļūdas funkcijas, tad rezultāti tika publicēti zinātniskajā literatūrā (G), bet ja jaunā kļūdas funkcija nebija labāka, tad bija nepieciešams atgriezties kļūdas funkcijas izstrādes fāzē (E).

Katrai jaunajai kļūdas funkcijas izpētei tika pielietota sekojoša metodoloģija:

1. Līdzīgu kļūdu funkciju, modeļu un apmācību procedūru analīze.
2. Implementācija un testēšana jaunām kandidātu kļūdas funkcijām, modeļiem vai apmācību procedūrām.
3. Jauno un jau esošo kļūdas funkciju, modeļu vai apmācību procedūru

testēšana uz atskaites datu kopām vai vidēm RL gadījumā. Katrai datu kopai vai videi tika veikta pilnās pārleses hiper-parametru pārmeklēšana.

4. Jauno kļūdas funkciju, modeļu vai apmācību procedūru daļu ietekmes analīze.

RL kļūdas funkcijām tika veikts apskats ar visām mūsdienīgajām kļūdas funkcijām [109]. DQN (Deep Q-Network) tipa kļūdas funkcijas tika testētas vismaz 4 dažādās PLE (PyGame Learning Environment) datorspēļu vidēs [104], kā arī tika testētas hromatogrāfijas uzdevumam analītiskajā ķīmijā [24].

Vēlāk arī jauna eksponenciālā trijotnes kļūdas funkcija [110] tika izstrādāta un testēta nulles-šāviena re-identifikācijas uzdevumiem, izmantojot datu kopas kā, piemēram, VGGFace2 [13], EMNIST [15] un CIFAR10 [47]. Datu kopas tika pārkārtotas tā, lai klases, kuras ir iekļautas testa sadalījumā netiktu iekļautas arī apmācību sadalījumā, lai panāktu, ka šie modeļi strādā ar nulles-šāviena metodoloģiju [71].

Visbeidzot VI (Vērtību Iterācijas) funkcijas modelis tika izveidots, izmantojot UNet-RNN-Skip [111] modeli. Tas tika salīdzināts ar dažādiem citiem modeļiem kā, piemēram, UNet [87] modeļu varianti, lai uzlabotu VI algoritma veiktspēju. Kā arī tika radīta jauna sintētiska datu kopa, lai pārbaudītu VI darbību ceļu atrašanai kartē mobilajiem robotiem, un tā var tikt izmantota arī citiem RL tipa modeļu uzdevumiem.

Jaunradītās implementācijas ir publiski pieejamas tīmeklī atvērtā tipa programmatūras formātā. To saites ir iekļautas publikācijās. Tās tika implementētas, izmantojot PyTorch matemātiskās progamatūras satvaru [79]. PyTorch tika izvēlēts, jo tas spēj veidot dinamisku funkciju grafu, kuru var mainīt apmācību laikā, kā arī ir iespējams ērti atklūdot un mainīt funkciju gradientus. Katrai no metodēm un kļūdas funkcijām tika veikta hiper-parametru pārmeklēšana, izmantojot augstas veiktspējas datorus ar NvidiaV100 video kartēm iekš RTU HPC (High Performance Cluster).

## 1.5. Zinātniskā novitāte un autora ieguldījums

Disertācijā iekļautas sekojošas jaunas kļūdas funkcijas: MDQN kļūdas funkcija, kuru izmanto RL un kura aprakstīta detalizēti 3.3. nod.; Eksponenciālā Trijotnes Kļūdas funkcija, kuru izmanto DML un tā detalizēti aprakstīta 4.3. nod. Vēl tika radītas jaunas latento vektoru normalizācijas funkcijas Unit-Range un Unit-Bounce, kuras arī aprakstītas 4.3. nod. Līdzīgi tika radīts jauns UNet-RNN-Skip modelis, lai uzlabotu veiktspēju Vērtību funkcijai, lai izvēlētos aģenta politiku 2D vidē mobilā robota ceļu plānošanai un tas tika aprakstīts 2.4. nod. Visbeidzot tika radīts jauns sintētiskas datu kopas ģenerators OccupancyMapGenerator un iepriekš sagatavota datu kopa, lai testētu UNet-RNN-Skip modeli, un tas tika aprakstīts 2.4. nod.

Disertācijas autors ir norādīts, kā pirmais autors visās publikācijās, kuras pievienotas disertācijas pielikumā, izņemot publikācijā analītiskajā ķīmijā [24], kur disertācijas autors bija galvenais autors saistībā ar RL metodoloģiju uzdevuma risināšanai, izmantojot dziļo mašīnāpmācību, bet pārējie autori veica pētījumu saistībā ar analītisko ķīmiju.

Zinātniskie darbi, kuri pievienoti disertācijai tika publicēti 3 zinātniskajās konferencēs un 1 zinātniskajā monogrāfijā. Darbi ir ieguvuši vairākus apbalvojumus:

1. Labākais pētījums konferencē ICCDA 2020 (ASV)
2. Treša vieta doktorantu pētījumu konkursā ResearchSlam 2018 (Latvija)

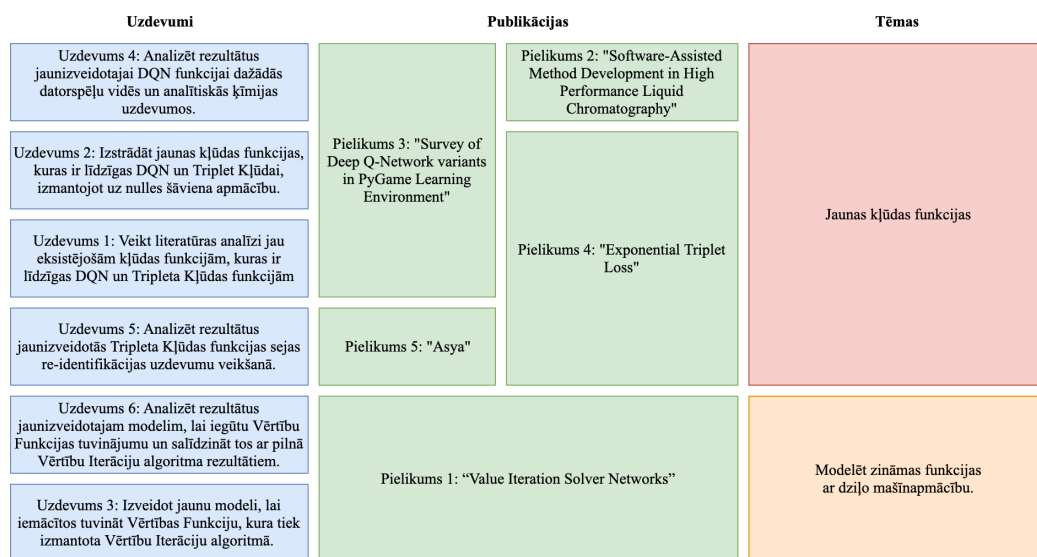
## 1.6. Darba struktūra un apjoms

Disertācijā ir iekļautas 75. lapas. Tā ir sadalīta 7 galvenajās nodaļās, un tā ir izveidota, kā publikāciju kopa ar paplašinātiem skaidrojumiem, jo lielākā daļa pētījumu ir publicēti zinātniskajās konferencēs, kur ir ierobežots pieļaujamo lapu skaits vienam darbam.

Disertācijas kopsavilkuma struktūra aprakstīta zemāk:

1. nod. Ievads, pētījumu pamatojums, pētījumu aktualitāte un pētījumu mērķi.
2. nod. Apraksta VI algoritma problemātiku, dod teorētisku ievadu esošajās dziļās mašīnāpmācības modeļu pieejām kā, piemēram, ConvNet, UNet, RNN un arī apraksta jaunizveidoto UNet-RNN-Skip modeli. UNet-RNN-Skip tika apmācīts, lai atdarinātu vērtības funkciju un samazinātu iterāciju skaitu, lai iegūtu optimālu politiku. Jaunizveidotais modelis tika izmantots, lai iemācītos jau eksistējošu neparalelizējamu funkciju tā, lai tā strādātu ātrāk un būtu paralelizējama.
3. nod. Apraksta Q-Vērtības funkcijā balstītu RL kļūdas funkcijas un metodes, testa vides un pieejas rezultātu analīzei, kā arī jaunizveidoto MDQL kļūdas funkciju, kura tika testēta un analizēta PLE vidēs un hromatogrāfijā analītiskās ķīmijas uzdevumā.
4. nod. Skaidro Dziļo Metriku Apmācību, izmantojot nulles-šāviena modeļus, kuri spēj iegūt latentu vektorus, kuri satur semantisko informāciju par ievades datiem. Šie modeļi tika apmācīti ar trijotnes kļūdas funkciju un jaunizveidoto eksponenciālo trijotnes kļūdas funkciju. Tās tika testētas uz vairākām datu kopām re-identifikācijas uzdevuma veikšanai.
5. nod. Apraksta eksperimentālos rezultātus jaunizveidotajām kļūdas funkcijām un modeļiem, kuri aprakstīti iepriekšējās nodaļās, kā arī apraksta praktiskos pielietojumus šīm kļūdas funkcijām un modeļiem.
6. nod. Apraksta tālāko pētījumu tēmas, kuras tika atrastas pētot jaunās kļūdas funkcijas un modeļus.
7. nod. Apraksta galvenos secinājumus un zinātnisko pienesumu pētījumiem, kuri aprakstīti šajā disertācijā.

Pilnajā disertācijas darbā ir papildus DQN apskatam iekļauts arī zinātniskās literatūras apskats, kurš analizē plašāk izmantotākās DML kļūdas funkcijas. Pilnajā darbā ir iekļauti pielikumi ar zinātniskajām publikācijām, uz kurām ir balstīts šis darbs. Sasaiste starp disertācijas mērķiem, tēmām un publikācijām ir attēlota 2. att.



2. att. Interconnection of published papers, thesis objectives, and research topics.



Publikācijas, kuras tika iekļautas disertācijas darbā un to galvenie jaunievedumi ir aprakstīti zemāk:

- Appendix A **“Value Iteration Solver Networks”, International Conference on Intelligent Autonomous Systems, 2020, IEEE, Evalds Urtans, Valters Vecins.** Tika izveidots jauns UNet-RNN-Skip modelis, lai uzlabotu VI algoritma veiktspēju, kā arī tika radīts OccupancyMapGenerator sintētiskas datu kopas ģenerators, lai testētu karšu plānotāju modeļus.
- Appendix B **Software-Assisted Method Development in High Performance Liquid Chromatography”; ISBN: 978-1-78634-545-5, Sep. 2018, Sergey V. Galushko, Irina Shishkina, Evalds Urtans, Oksana Rotkaja.** Tika izveidots jauns RL balstīts modelis, lai secīgi atrastu labākos šķīdinātāju gradientus priekš HPLC.
- Appendix C **“Survey of Deep Q-Network variants in PyGame Learning Environment”, International Conference on Deep Learning technologies, 2018, ACM, Evalds Urtans, Agris Nikitenko** Tika izveidota jauna RL apmācīšanas metode un jauna MDQN kļūdas funkcija.
- Appendix D **“Exponential Triplet Loss”, International Conference on Compute and Data Analysis, 2020, IEEE/ACM, Evalds Urtans, Valters Vecins, Agris Nikitenko** Tika izveidota jauna dziļo metriku apmācības kļūdas funkcija ar nosaukumu Eksponenciālā Trijotnes kļūdas funkcija.
- Appendix E **”asya: Mindful verbal communication using deep learning”, Cornell University, Computing Research Repository, 2020, Evalds Urtans, Austris Tabaks** Tika izveidota jauna sistēma, izmantojot Eksponenciālo Trijotnes kļūdas funkciju, lai veiktu balss reidentifikācijas uzdevumu.

## 2. FUNKCIJU VEIKTSPĒJAS UZLABOŠANA, IZMANTOJOT DZIĻO MAŠĪNAPMĀCĪBU

Šī nodaļa iepazīstina ar jaunu dziļās mašīnāpmācības paņēmienu, lai optimizētu veiktspēju klasiskai funkcijai, kura jau ir daudz pētīta un plaši pielietota, bet ļoti lēna. Tika pētīta vērtību funkcija, kuru izmanto Vērtību Iterāciju (VI) algoritms. Tā paredzēta, lai atrastu īsāko ceļu aizņemtības režģa kartēs no jebkuras pozīcijas tajā līdz mērķa pozīcijai.

Tās veiktspēja sarūk eksponenciāli, palielinot kartes izmēru, kā arī tā nav izpildāma paralēli. Savukārt, jaunais iteratīvais modelis, kurš tiek piedāvāts šajā darbā var iegūt līdzvērtīgus rezultātus kā VI algoritms daudz ātrāk, jo to var izpildīt paralēli. Ievērojami labākus rezultātus var iegūt, izmantojot lielāka izmēra kartes.

Vērtību funkcijas un Vērtību Iterācijas algoritma problēmsfēra ir aprakstīta 2.1. nod. Pēc tam dziļās mašīnāpmācības metodes, kuras var tik pielietotas, lai modelētu Vērtību funkciju, ir aprakstītas 2.2. nod. un 2.3. nod. Visbeidzot, jaunizveidotā metode ir aprakstīta 2.4. nod., un tās rezultāti ir parādīti 5.1. nod.

### 2.1. Vērtību Iterāciju algoritms

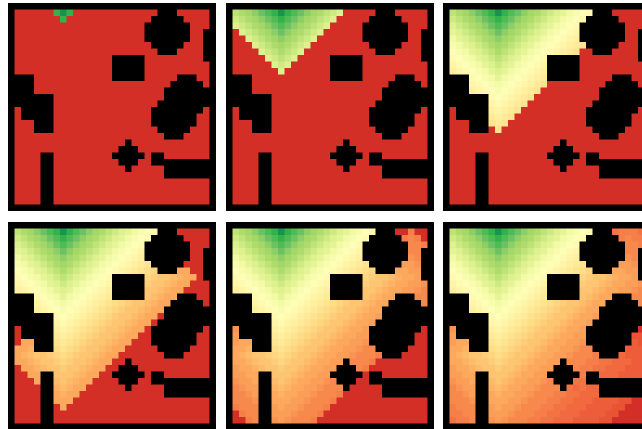
Vērtību Iterāciju Algoritms (VI) tiek lietots klasiskās stimulētās mašīnāpmācības uzdevumos, lai atrastu optimālu politiku vidē, kuras stāvokļi ir pilnībā novērojami. Tas var ņemt vērā arī stāvokļu pārejas modeli, kur šīs pārejas var būt nedroši nosakāmas [88]. VI nereti tiek izmantots, lai atrastu optimālu ceļu kartēs ar diskrētiem stāvokļiem. Ceļu meklēšanas uzdevumā tiek formalizēta un diskretizēta dabiska vide un tās šķēršļi. Nereti šos datus iegūst no sensoriem, kuri ir uzstādīti uz mobilas robota platformas. Sensori parasti var būt LIDAR sensori attāluma noteikšanai, izmantojot gaismu, ultraskaņas attāluma sensori, IMU akselometra un žiroskopa sensori pārvietojuma noteikšanai, magnetometrs, GPS u.c. sensori. Diskretizācija kartēm parasti tiek veikta, izmantojot aizņemtības režģa reprezentāciju. VI ir iteratīvs algoritms, kurš atkārtoti pielieto Vērtību funkciju 1. vienād. visām pozīcijām kartē, lai atrastu kumulatīvo pozīcijas vērtību, kā parādīts 3. att. Tālāk, vērtību gradients starp šīm pozīcijām norāda uz optimālu ceļu un stāvokļu pāreju politiku. Optimāla ceļa politika nodrošina, ka aģents var atrast ceļu

no jebkura stāvokļa diskretizētā kartē līdz pozitīvam terminālajam stāvoklim. Katram stāvoklis  $s$  tiek aprakstīts ar režģa pozīcijām. Katrā stāvoklī ir iespējams veikt darbības  $a$  kā, piemēram, pārvietoties augšup, lejup, pa kreisi, pa labi vai palikt nekustīgam iepriekšējā stāvoklī. Atkarībā no vides darbības var būt vairāk vai citādākas.

Darbība  $a$  ir jāizvēlas tā, lai maksimizētu summēto balvas vērtību šai darbībai  $R_a$ . Pēc tam tā tiek reizināta ar pārvietošanas modeļa varbūtību  $P_a$  un tiek pieskaitītas blakus esošo stāvokļu vērtības  $V(s')$ , kuras tiek reizinātas ar atlaides koeficientu  $\gamma$ , kā parādīts 1. vienād.

Vērtību funkcija tiek izsaukta iteratīvi visām kartes pozīcijām līdz vērtības konverģē starp iterācijām. Iterāciju skaits pieaug eksponenciāli, palielinoties kartes izmēram. Konverģence VI notiek, kad izpildās  $|\frac{V(s)_i}{V(s)_{i-1}}| < \delta$ , kur  $\delta$  ir vidējā skaitliskā starpība starp iterācijām, kurai var būt robežas no 0.1-0.001 atkarībā no vides, stāvokļu novērojumu tīrības un pieļaujamās kļūdas.

$$V_{i+1}(s) := \max_a \left\{ \sum_{s'} P_a(s, s') (R_a(s, s') + \gamma V_i(s')) \right\} \quad (1)$$



3. att. Secīgu VI algoritma iterāciju vizualizācija. Iterācijas ir attēlotas secīgi, sākot no augšējā kreisā stūra līdz augšējam labajam stūrim, tad no apakšējā kreisā stūra līdz apakšējam labajam stūrim. Kad ir sasniegta konverģence, tiek iegūta optimāla politika tā, lai no katra stāvokļa kartē var iegūt augstāko kumulatīvo balvas vērtību, nonākot pozitīvā terminālajā stāvoklī. Zaļā krāsā ir augstākās pozīciju vērtības (tuvāk pozitīvajam terminālajam stāvoklim), bet sarkanā krāsā ir attēlotas zemāko vērtību pozīcijas.

Pamatojums tam, lai modelētu Vērtību funkciju, izmantojot dziļo mašīnāpmācību ir tajā, ka šādi ir iespējams paātrināt šo algoritmu, to paralēlizējot un samazinot tā secīgo operāciju skaitu, kā parādīts 1. tab. Eksistē arī citi populāri algoritmi, kuri lieto heuristikas, lai samazinātu secīgo operāciju skaitu kā, piemēram, Dijkstra vai A\* algoritmi [88], bet, tos lietojot, samazinās arī rezultāta precizitāte.

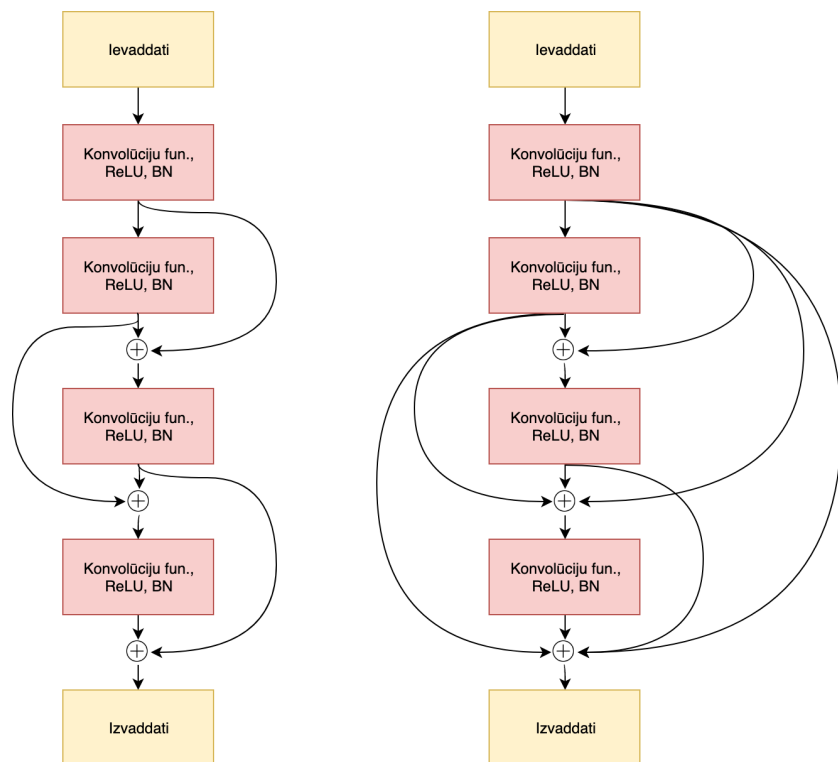
1. tabula.

Salīdzinājums jaunajam VI funkcijas modelim VSIN (Value Iteration Solver Network) ar klasiskajām funkcijām.

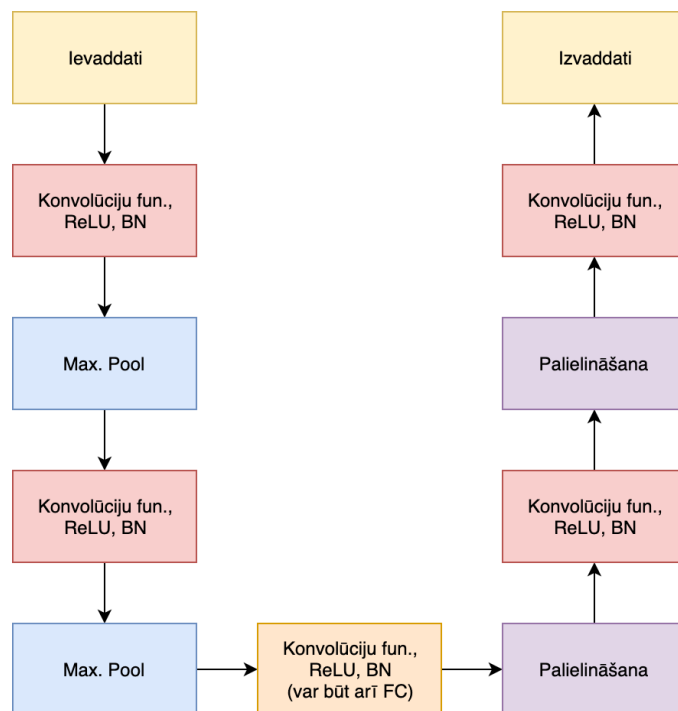
Metode	Sarežģītība	Kvalitāte
VI (Vērtību funkcija)	$O(m \cdot n^2)$	Optimāla
Dijkstra	$O(n^2)$	Laba
A*	$O(h \cdot n)$	Vidēja
VISN (Jaunievestā metode)	$O(h \cdot n)$	Laba

## 2.2. ConvNet un UNet modeļi

Lai palielinātu VI algoritma veiktspēju, šī funkcija var tikt modelēta, izmantojot ConvNet (Konvolūciju tīkla) modeļi [56] [48] [97] [102] [31] [38], kas tiek apmācīts, lai prognozētu skaitliskās vērtības  $V(s)$  funkcijai. Pats modelis var tik apmācīts Vērtību funkcijas rezultāta tuvinājuma aprēķināšanai. ConvNet modelis strādā, kā filtrs visai aizņemtības režģa kartei un prognozē skaitliskās vērtības VI algoritma rezultātiem bez vajadzības pēc iteratīva procesa. ConvNet modeļi var tikt paralelizēti uz mūsdienīgiem GPU, lietojot dziļās mašīnāpmācības satvarus. Šajā pētījumā tika izmantoti ResNet [31] un DenseNet [38] tipa ConvNet modeļi, kā iezīmju kodētāji vērtību funkcijai. ResNet un DenseNet modeļi, kuri tika izmantoti kā iekodētāji, shematiski prādīti 4. att.. Modeļa iezīmju kodētāks kondensē augstu dimensiju ievades datus uz zemu dimensiju latentajiem vektoriem, kurus tālāk var izmantot dziļākās modeļa daļās. Šajā pētījumā kā dekodera modelis tika izmantotas transponētas konvolūciju funkcijas [74]. Dekodera daļa no modeļa daļa rekonstruē zemu dimensiju latentos vektorus par augstas dimensijas rezultātiem, kuri tiek uzklāti, kā filtrs ievades datiem. Uzklāšana var notikt, izmantojot vienkāršu aritmētisku saskaitīšanu, reizināšanu vai aizvietošanu. Auto-Kodētāja modelis, kurš satur dekodēru parātis 5. att..



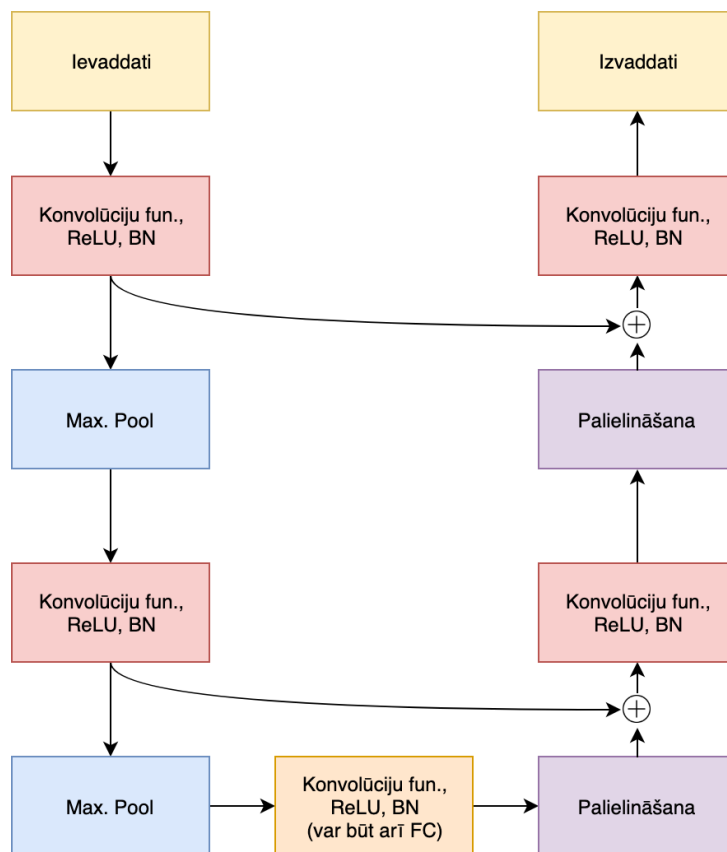
4. att. Salīdzinājums starp iekodētāju modeļiem, kur dots ResNet kreisajā pusē un DenseNet labajā pusē. Plusa zīme norāda uz aritmētisko saskaitīšanas operāciju.



5. att. Auto-Kodētāja modeļa piemērs. Funkciju grafa vidū, neobligāti, var tikt ievietota lineārās transformācijas funkcija FC (Fully Connected), ja modelim ir pieļaujami fiksēta izmēra ievades un izvades dati.

VI algoritma darbība ir līdzīga daudzkārtējam filtrēšanas uzdevumam, jo pati ievades datu struktūra netiek izmainīta, bet katrā iterācijā tiek pamainītas tikai nelielas detaļas.

Filtrēšanas un stila pārneses, un pat segmentēšanas uzdevumiem nereti tiek izmantots UNet tipa modelis [87], jo tas satur lēcienus savienojumus starp funkcijām, kuri uztur detalizējumu oriģinālajiem datiem dažādas iekodētāja dimensijās atkarībā no modeļa dziļuma. Šajā pētījumā UNet modeļi tika trenēti, lai iegūtu VI algoritma rezultātus ar vairākām iterācijām, kā arī vienā piegājienā.



6. att. Piemērs UNet modelim, un kā tajā tiek izmantoti lēciena savienojumi, kur datu īpašības tiek pieskatītas no iekodētāja puses uz dekodētāja pusi funkciju grafā.

Orģinālais UNet modelis, kas ir attēlots 6. att. satur funkciju grafu, kuru vienādojumi doti zemāk 2. vienād. Vienādojumos ievades dati  $x$  tiek padoti caur vairākām iekodētāja funkcijām, kur  $Conv$  funkcija ir lineāra 2D konvolūciju funkcija ar kodola izmēru  $3 \times 3$ , soli 1 un aplodu 1, un kuras rezultāts būs tāda paša izmēra izvades dati kā ievades dati. Šī funkcija nemaina aizņemtības režģa izmērus. Savukārt  $DeConv$  ir transponēta 2D konvolūciju funkcija kodola izmēru  $4 \times 4$ , soli 2 un aplodu 1, kas dos divas reizes lielāku izvades karti kā ievades datu karti. Līdzīgi arī  $MaxPool$  funkcijas rezultāts ar šiem pašiem parametriem dos 2 reizes mazāku izvades karti kā ievades karti, tikai katrā no kodola apgabaliem tiks izvēlēta maksimālā vērtība no vienas no režģa šūnām. Lēciena savienojumi, kuri ir parādīti 8. vienād. un



11. vienād. izmanto apvienošanas operāciju, bet segmentācijas uzdevumiem bieži tiek lietota arī saskaitīšanas operācija, kā tas ir ResNet modeļos [102]. Gala rezultāta skaitliskās vērtības  $y$  tiek ierobežots ar sigmoīda funkciju  $\sigma$ , kur katra vērtība tiek mērogota pret minimālo maksimālo kumulatīvo balvas vērtību kartē.

$$o_1 = ReLU(Conv(x)) \quad (2)$$

$$o_2 = MaxPool(o_1) \quad (3)$$

$$o_3 = ReLU(Conv(o_2)) \quad (4)$$

$$o_4 = MaxPool(o_3) \quad (5)$$

$$o_5 = ReLU(Conv(o_4)) \quad (6)$$

$$o_6 = DeConv(o_5) \quad (7)$$

$$o_7 = (o_6, o_3) \quad (8)$$

$$o_8 = Dropout(ReLU(Conv(o_7))) \quad (9)$$

$$o_9 = DeConv(o_8) \quad (10)$$

$$o_{10} = (o_9, o_1) \quad (11)$$

$$o_{11} = ReLU(Conv(o_{10})) \quad (12)$$

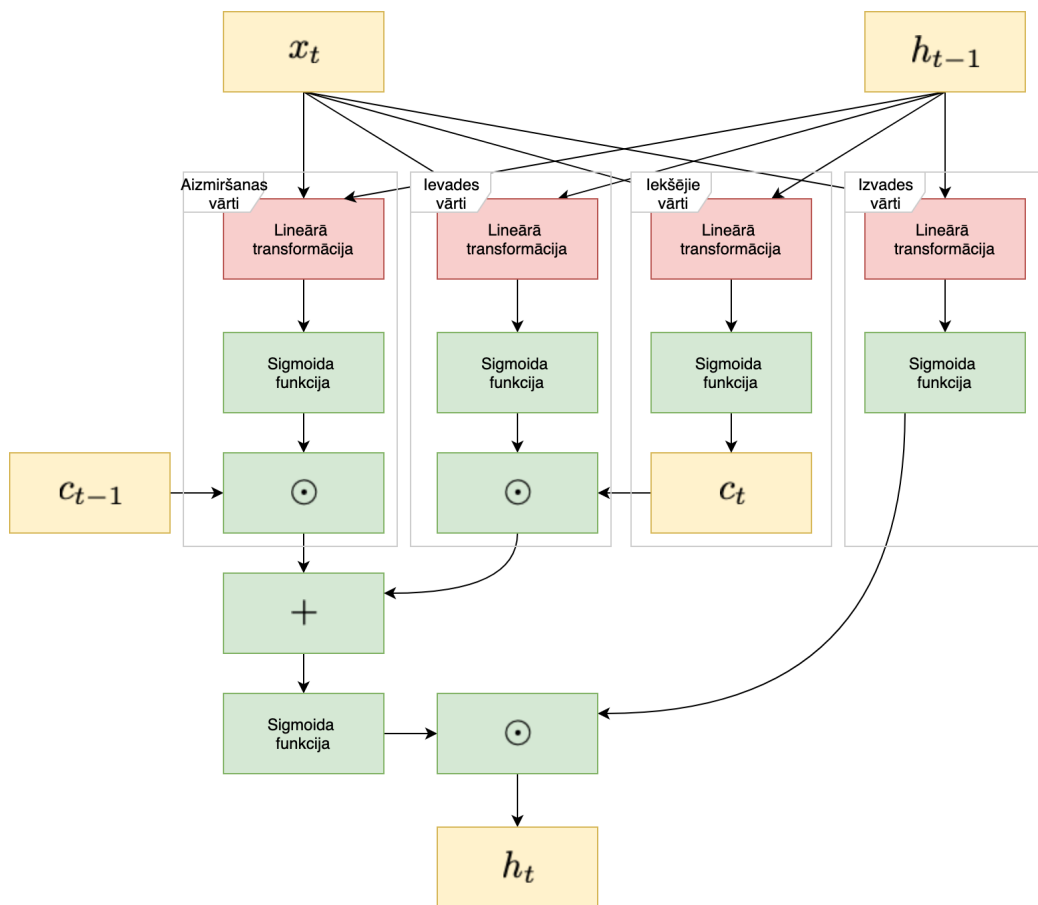
$$y = \sigma(o_{11}) \quad (13)$$

### 2.3. RNN modeļi

Lai modelētu iteratīvus procesus un laika sēriju tipa datus, nereti tiek izmantoti RNN (Rekurento Neironu Tīklu) modeļi. VI algoritma gadījumā šie modeļi tiek lietoti, lai samazinātu filtra uzdevuma sarežģītību, kad ir nepieciešams ar vienu ConvNet iterāciju prognozēt  $V(s)$  vērtības, neko nezinot par iepriekšējām vērtībām. Ar RNN tipa modeli vērtības tiek prognozētas secīgi ar iteratīviem soļiem līdzīgi kā to dara VI algoritms, bet RNN modelis nodrošina konvergenci ar daudz mazāku iterāciju skaitu.

Šajā pētījumā tika pielietoti populāri RNN modeļi, kā LSTM (Ilgtermiņa-Īstermiņa atmiņas šūna) [36] un GRU (Rekurentā šūna ar vārtiem) [28].

Lai uzlabotu veiktspēju un konverģences ātrumu, tika pielietotas parametru inicializācijas stratēģijas. Piemēram, nobīdes parametri aizmiršanas vārtiem LSTM modelī tika inicializēti ar skaitlisko vērtību 1, lai mazāk informācijas tiktu aizmirstas apmācību sākumā. Līdzīgi arī parametru nobīdes vektori GRU modelim tika inicializēti ar skaitlisko vērtību -1, lai panāktu šo pašu efektu [41] [54] [25] [106].



7. att. Function graph of LSTM model.

LSTM modeļa grafa vienādojumi ir doti 14. vienād. un shematiski attēloti 7. att. Apmācāmie svāri un nobīdes ir apzīmēti ar  $W$ ,  $U$  un  $b$ , sigmoīda funkcijā  $\sigma$  balstītie vārti ir  $f_t$  aizmiršanas vārti,  $i_t$  ievades vārti un  $o_t$  datu izvades vārti. Iekšējā stāvokļa vektors, kurš nav apmācāms ir  $c_t$  un neapmācāms slēptā stāvokļa vektors  $h_t$ , kuri apmācību sākumā tiek uzstādīti ar nulles skaitliskām vērtībām, bet ar katru sekvenču laika soli jeb iterāciju tiek izmainīti ar informāciju no iepriekšējiem laika soļiem. Katrā laika solī LSTM modeļa rezultāti sakrīt ar pašreizējo  $h_t$  vērtību. Pēc jaunākajiem pētījumiem  $b_f$  jāinicializē ar vērtību  $b_f = 1$  [106], tāpat jāpievieno RNN caurkrišanas (dropout) vai ZoneOut regularizācija added [49] [94],  $h_0$  un  $c_0$  var tikt apmācīti priekš pirmās iterācijas [75], Slāņa Normalizācijas funkcija jāpielieto pēc katras lineārās transformācijas LSTM modelī [3], un visbeidzot

jāpievieno lēciena savienojumi starp LSTM modeļu slāņiem, tos summējot  $h_t =_{k=i}^{layers} h_t^k$  līdzīgā veidā kā to dara ResNet modeli, lai vienmērīgi izplatītos kļūdas gradients dziļos mākslīgos neironu tīklos [102] [38] [106].

$$f_t = \sigma_g(W_f x_t + U_f h_{t-1} + b_f) \quad (14)$$

$$i_t = \sigma_g(W_i x_t + U_i h_{t-1} + b_i) \quad (15)$$

$$o_t = \sigma_g(W_o x_t + U_o h_{t-1} + b_o) \quad (16)$$

$$\tilde{c}_t = \sigma_h(W_c x_t + U_c h_{t-1} + b_c) \quad (17)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \quad (18)$$

$$h_t = o_t \odot \sigma_h(c_t) \quad (19)$$

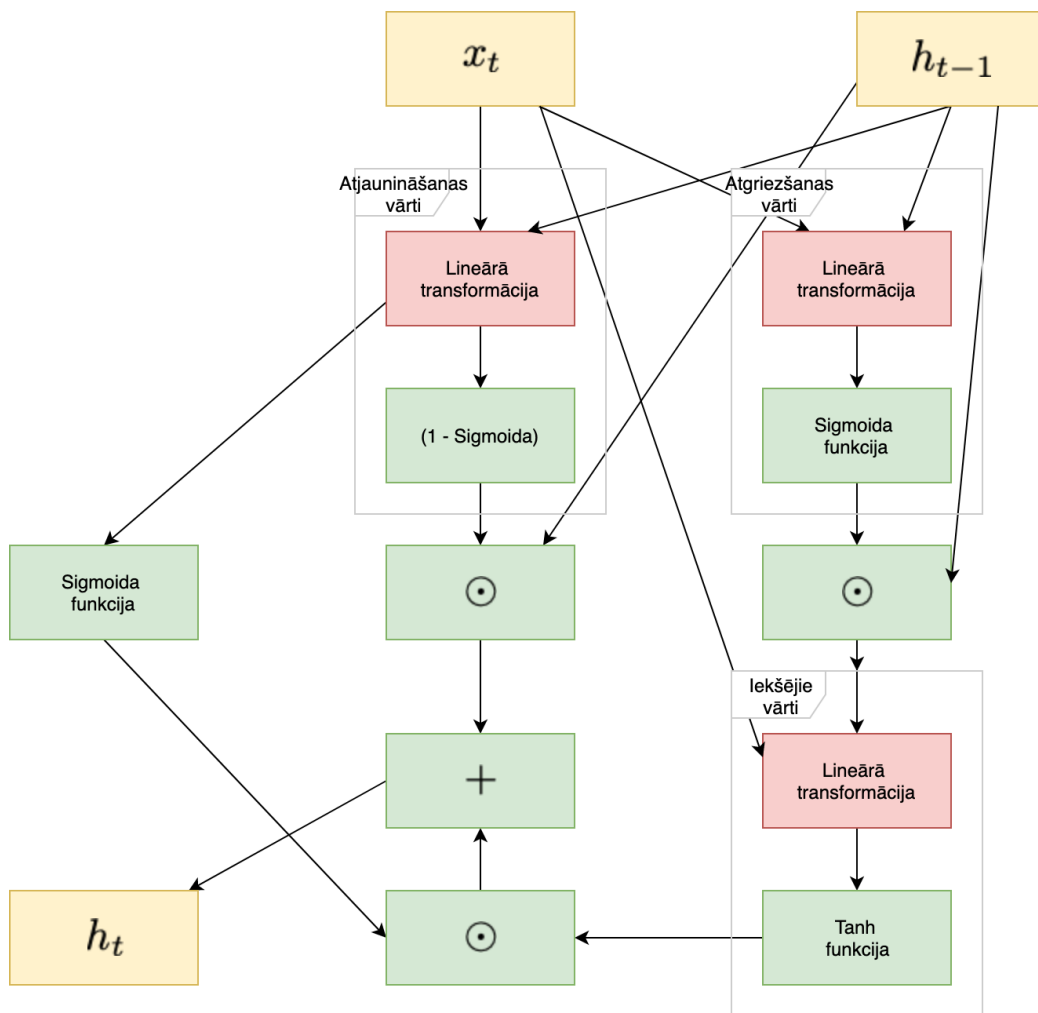
GRU modelis, kā parādīts zemāk 20. vienād. un 8. att. ir vienkāršota versija LSTM modelim ar mazāk apmācāmiem parametriem. Tas satur tikai  $z_t$  atjaunināšanas vērtus un  $r_t$  atgriešanas vērtus. Tāpat tas lieto ne tikai sigmoīda funkcijas, bet arī tanh funkcijas  $\phi$  līdzīgi kā tradicionālie vienkāršie RNN. Lai iegūtu augstāko veiktspēju ar šiem modeļiem, nepieciešams pielietot visas tās pašas izmaiņas, kā aprakstītas pie LSTM modeļiem augstāk, izņemot  $r_t$  vērtu nobīdi jāinicializē ar skaitlisko vērtību  $b_r = -1$  [106].

$$z_t = \sigma_g(W_z x_t + U_z h_{t-1} + b_z) \quad (20)$$

$$r_t = \sigma_g(W_r x_t + U_r h_{t-1} + b_r) \quad (21)$$

$$\hat{h}_t = \phi_h(W_h x_t + U_h (r_t \odot h_{t-1}) + b_h) \quad (22)$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \hat{h}_t \quad (23)$$



8. att. Function graph of GRU.

## 2.4. UNet-RNN-Skip modelis

Kā viena daļa no disertācijas tika jaunizveidots modelis, kurš apvieno UNet un RNN modeļus vienā. Jaunais modelis UNet-RNN-Skip satur vairākas īpašības no modeļiem, kurus izmanto citām vajadzībām, lai nodrošinātu jaunā modeļa spēju sasniegt ātrāku konvergenci VI algoritma rezultāta iegūšanai. Tas satur UNet, LSTM un lēciena savienojumus līdzīgi kā ResNet, bet tie tiek izmantoti laika sēriju un segmentācijas uzdevumiem.

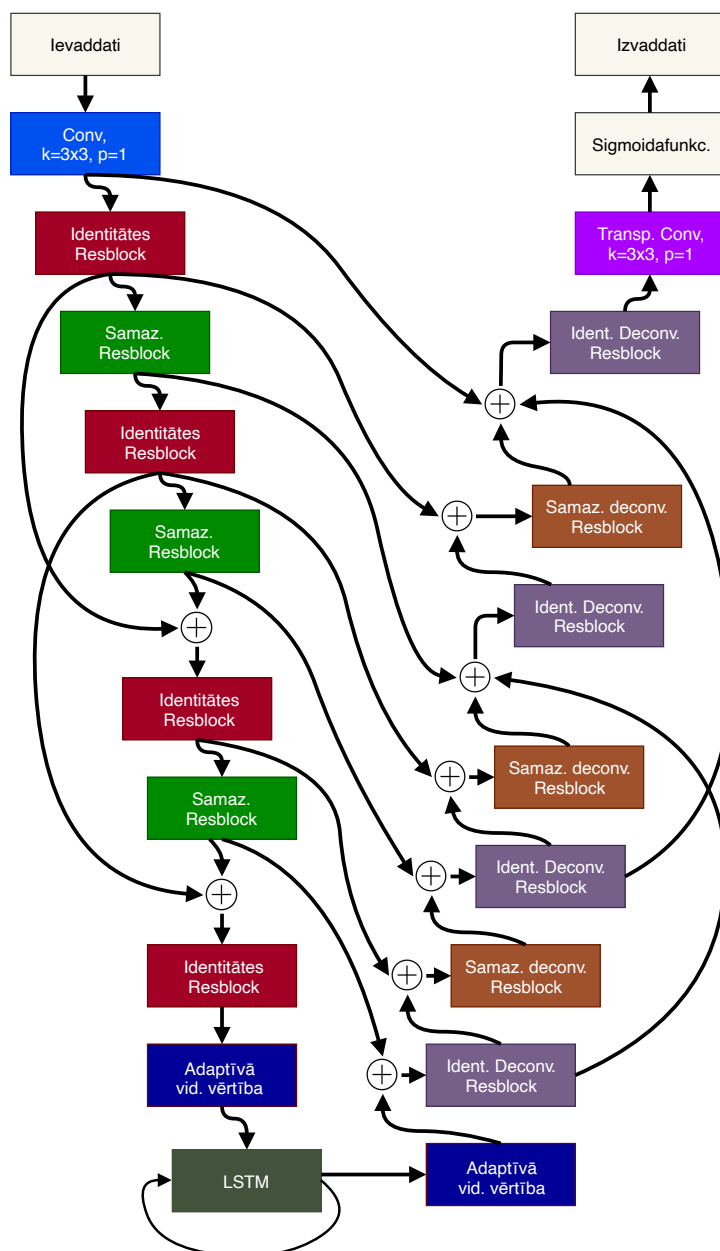
UNet-RNN-Skip modelis tika radīts, lai iegūtu tuvinātas skaitliskās vērtī-

bas funkcijai, kuru izmanto VI algoritmā, bet tā, lai to varētu izpildīt paralēli pilnīgi visām aizņemtības režģa šūnām. Savukārt, VI algoritmam nepieciešams šīs vērtības izrēķināt secīgos soļos, apskatot katru šūnu aizņemtības režģī. Tāpat jaunais modelis tika radīts, lai aprēķinātu ar vienu iterāciju tuvinātu skaitlisko vērtību, kuru VI algoritms aprēķina ar vairākām iterācijām.

Vēl viens ieguvums UNet-RNN-Skip modelim ir tas, ka to var apmācīt uz daudz mazāka izmēra kartēm kā, piemēram, 32x32, bet pēcāk to var produkcijā izmantot uz lielām kartēm kā, piemēram, 256x256, bez nepieciešamības to papildus apmācīt, jo modelis ir iemācījies Vērtību Funkciju nevis sakarības pašā kartē.

Visas kartes tika ģenerētas, izmantojot jaunu OccupancyMapGenerator algoritmu, kas arī bija daļa no šī pētījuma un ir aprakstīts 5.1. nod. Eksperimentālie rezultāti un modeļa īpašības ir aprakstītas 5.1. nod.

UNet-RNN-Skip modelī svarīga detaļa ir LSTM novietojums “U formas” pašā apakšā funkciju grafā, bet tajā pašā laikā modelis arī izmanto UNet lēciena savienojumu funkcijas no kodētāja uz dekodētāja daļām līdzīgi kā ResNet, UNet++ [136] un UNet 3+ [39] modeļos. Pretēji parastam UNet, kurš izmanto īpašību apvienošanu, paplašinot matricu izmērus, kā parādīts 6. att., jaunais modelis izmanto aritmētisko saskaitīšanu lēciena funkcijās. ResNet [31] funkciju bloki tika izveidoti divos dažādos veidos, kā parādīts 9. att.



9. att. UNet-RNN-Skip modelis. Ar krāsām tiek iezīmēti dažādi funkciju bloki, kuri tiek izmantoti šajā modelī.

Vispirms tika izveidots identitātes ResBlock funkcija, kā parādīts 24. vien-

ād.

$$y_{c \times w \times h} = \text{ReLU}(\text{Conv3x3}(x_{c \times w \times h})) \quad (24)$$

$$y'_{c \times w \times h} = \text{BatchNorm}(y_{c \times w \times h}) \quad (25)$$

$$z_{c \times w \times h} = \text{ReLU}(\text{Conv3x3}(y'_{c \times w \times h})) \quad (26)$$

$$z'_{c \times w \times h} = z_{c \times w \times h} + x_{c \times w \times h} \quad (27)$$

$$u_{c \times w \times h} = \text{ReLU}(z'_{c \times w \times h}) \quad (28)$$

$$u'_{c \times w \times h} = \text{BatchNorm}(u_{c \times w \times h}) \quad (29)$$

Pēc tam tika izveidota BottleNeck funkcija, kurš samazina iezīmju karšu izmērus ar konvolūciju funkcijas palīdzību, bet tajā pašā laikā arī palielina iezīmju skaitu karšu kanālu dimensijā, kā parādīts 30. vienād. ResBlock funkcija ir līdzīga pirms-aktivizācijas ResBlock funkcijai [32], kur partiju normalizācija tiek izpildīta pirms lineārajām transformācijām. Tādas pašas funkcijas spoguļskatā tika izmantotas arī transponēto konvolūciju funkciju daļā, kuras palielina karšu izmērus, bet samazina kanālu iezīmju skaitu.

$$y_{n \cdot c \times \frac{w}{n} \times \frac{h}{n}} = \text{ReLU}(\text{Conv3x3}(x_{c \times w \times h})) \quad (30)$$

$$y'_{n \cdot c \times \frac{w}{n} \times \frac{h}{n}} = \text{BatchNorm}(y_{n \cdot c \times \frac{w}{n} \times \frac{h}{n}}) \quad (31)$$

$$z_{n \cdot c \times \frac{w}{n} \times \frac{h}{n}} = \text{ReLU}(\text{Conv3x3}(y'_{n \cdot c \times \frac{w}{n} \times \frac{h}{n}})) \quad (32)$$

$$x'_{n \cdot c \times \frac{w}{n} \times \frac{h}{n}} = \text{Conv1x1}(x_{c \times w \times h}) \quad (33)$$

$$z'_{n \cdot c \times \frac{w}{n} \times \frac{h}{n}} = z_{n \cdot c \times \frac{w}{n} \times \frac{h}{n}} + x'_{n \cdot c \times \frac{w}{n} \times \frac{h}{n}} \quad (34)$$

$$u_{n \cdot c \times \frac{w}{n} \times \frac{h}{n}} = \text{ReLU}(z'_{n \cdot c \times \frac{w}{n} \times \frac{h}{n}}) \quad (35)$$

$$u'_{n \cdot c \times \frac{w}{n} \times \frac{h}{n}} = \text{BatchNorm}(u_{n \cdot c \times \frac{w}{n} \times \frac{h}{n}}) \quad (36)$$

### 3. FUNKCIJU FORMĒŠANA DZIĻAJAI STIMULĒTAJAI MAŠĪNAPMĀCĪBAI

Šī nodaļa iepazīstina ar jaunu kļūdas funkciju un apmācību procedūru dziļajai Q-Vērtību funkcijas apmācībai. Jaunizveidotā MDQN kļūdas funkcija izmanto dinamisku funkcijas grafu, kurš maina kļūdas funkcijas formu apmācības procesa laikā. Tika veikta arī citu līdzīgu funkciju analīze. Jaunizveidotās procedūras arī iekļauj metodes, kura ļauj vizualizēt stāvokļu vērtības un politiku, lai “melnās kastes modeļus” padarītu par interpretējamiem, “baltās kastes modeļiem”.

Dziļās stimulētās mašīnāpmācības problēmsfēra aprakstīta 4.1. nod. Jau pazīstamas DQN kļūdas funkcijas un to paveidi aprakstīti 3.2. nod. Visbeidzot, jaunizveidotā kļūdas funkcija aprakstīta 3.3. nod., un tās rezultāti doti 5.2. nod.

#### 3.1. Q-Vērtību funkcija un politikas gradientu funkcijas

Stimulētās mašīnāpmācības algoritmi jau bija labi izpētīti pat pirms dziļās mašīnāpmācības metodēm, taču līdz ar jaunajām metodēm to kapacitāte ir ievērojami palielinājusies [68]. Līdzīgi, kā iepriekšējās nodaļās aprakstītais VI algoritms, arī šie modeļi aprēķina labāko politiku  $\pi$  balstoties uz stāvokļu kumulatīvajām vērtībām  $r_t$  un reizēm izmantojot tikai pašu stāvokli  $s_t$ .

Eksistē trīs galvenās metožu kopas stimulētajā mašīnāpmācībā:

1. Q-Vērtību funkcijā balstītās metodes;
2. Politikas gradienta metodes;
3. Aktiera-kritiķa metodes (Q-Vērtību funkcijas un politikas gradienta metožu apvienojums).

Šīs metodes, galvenokārt, savā starpā atšķiras ar kļūdas funkcijas formu, kas ir viena no svarīgākajām sastāvdaļām, lai šīs metodes spētu strādāt.

Politikas gradienta metodes strādā uz politikas gradienta teorēmas pamata 37. vienād., kur kļūdas gradients politikas funkcijai  $\pi$  ir novērtējuma vērtība iepriekšējo stāvokļu trajektorijai  $\tau$ , pašreizējam stāvoklim  $s_t$  un katrai iespējamai darbībai šajā stāvoklī  $a_t$ , reizinot ar kumulatīvo balvas vērtību  $R$  38. vienād., trajektorijai  $\tau$ . Šī gradienta funkcija ir nestabila un grūti



konverģējama, bet nesenie pētījumi to ir padarījuši daudz stabilāku un praktiski izmantojamu, kā piemēram kļūdas funkcijas TRPO [92], PPO [91], MERLIN [125], IMPALA [21], u.c. kļūdas funkcijas.

$$\nabla \pi_{\theta}(a_t|s_t) = \nabla \log \pi_{\theta}(a_t|s_t) R(\tau) \quad (37)$$

Kumulatīvā balvas vērtība  $R$  38. vienād. ir summēta balvas vērtība stāvokļu trajektoriju balvu vērtības, kur tās katrā laika solī  $t$  tiek reizinātas ar atlaides vērtību  $\gamma$ . Atlaides vērtība  $\gamma$  nosaka, cik ļoti nozīmīgas ir ātri iegūstamas balvas pret balvām, kuras var iegūt epizodes beigās. Jo mazāka ir šī skaitliskā vērtība, jo vairāk modelis liek uzsvāru uz ātri iegūstamām balvām. Dažās vidēs, kur aģents var viegli nokļūt negatīvā terminālā stāvoklī, šī vērtība būtu jāuzstāda zemāka. Parasti šo vērtību uzstāda kā hiper-parametru vērtību robežās 0.9-0.99, jo nozīmīgākie notikumi datorspēļu vidēs parasti notiek pēc ilgāka laika un pie tiem jānonāk ar mērķtiecīgu plānošanu.

$$R = \sum_{t=0}^n \gamma^t r_t \quad (38)$$

Otrā metožu kopa sakņojas Bellmana vienādojumā 39. vienād., kas tuvinā  $R$  vērtību katrā stāvoklī  $s_t$  pie darbības  $a_t$ , to modelējot ar Q-Vērtību funkcijas modeli. Secīgi apskatot darbības  $a_t$  un izvēloties to, kura sniedz augstāko Q-Vērtību, ir iespējams noteikt politiku  $\pi$ .

$$Q_{\pi}(s_t, a_t) = r_t + \max_{a'} Q_{\pi}(s_{t+1}, a') \quad (39)$$

Trešā metožu kopa ir Aktiera-Kritiķa metodes, kuras sakņojas abās iepriekšējās metodēs. Tās, galvenokārt, apvieno abas iepriekšējās kļūdas funkcijas, izmantojot jauktu kļūdas funkciju ar vairākām daļām starp kurām ir koeficienti. Daudzi veiksmīgi šādu metožu varianti ir tikuši izstrādāti kā, piemēram, DDPG [60], A3C [67], GPU A3C [4], ACER (Actor-Critic with Experience Replay) [124], u.c. metodes.

Lai arī liktos, ka labākos rezultātus vienmēr iegūtu kombinētās kļūdas funkcijas, tomēr labākos rezultātus nereti izdodas iegūt tieši ar katru kļūdas funkciju atsevišķi, bet tas ir atkarīgs arī no vides īpatnībām.

### 3.2. DQN kļūdas funkcija

Viens no veiksmīgākajiem dziļās stimulētās mašīnāpmācības modeļiem ir DQN (Deep Q-Network). Pēc sākotnējiem un veiksmīgajiem DQN [68] rezultātiem tika radīta virkne citu Q-Vērtības funkcijā sakņotu modeļu uzlabojumi

[2]. Sākotnējais DQN tika testēts uz Atari spēlēm, kurām ir daudzdimensiju stāvokļu reprezentācijas - operatīvās atmiņas apgabala matrica vai spēles attēla pikseļu matrica katram spēles stāvoklim, kā parādīts 10. att. Vēl cita bieži izmantota stimulētās mašīnāpmācības vide ir OpenAI Gym [6], kurā tiek izmantotas maza skaita dimensijas stāvokļu reprezentācijai, piemēram, MoonLander vidē tiek dota informācija par kosmosa kuģa atrašanās vietu, ātrumu un leņķi attiecībā pret mērķa pozīciju uz Mēness, kur tam ir jānolaižas. Bet šajā pētījumā tika izmantotas PLE (PyGame Learning Environment) vides, kuras satur gan daudzdimensiju vides, gan vides ar mazāku skaitu dimensiju un ir iespējams pat izmainīt vides uzvedību, kamēr modelis uz tās tiek apmācīts, lai būtu iespējams dziļāk izpētīt modeļa īpašības.

Pēdējos gados DQN modelim ir veiktas dažas mazas, bet nozīmīgas izmaiņas, kuras ir devušas ievērojamus uzlabojumus tā veikspējā. Viens no svarīgākajiem uzlabojumiem ir prioritārā pieredzes atmiņa [90], kura nodrošina, ka apmācībai tiek izvēlētas stāvokļu pārejas ar lielāko temporālo kļūdu. Vēl viens nozīmīgs uzlabojums ir Duelā DQN [123], kurš ar modeļa arhitektūras palīdzību tam liek iemācīties atpazīt ātri iegūstamās balvas un ilgtermiņa balvas, kuras nereti ir daudz nozīmīgākas. Nākamais uzlabojums ir DDQN (Dubultais DQN) [30] [29], kurš nodrošina stabilāku DQN kļūdas funkcijas 40. vienād. konverģenci, izmantojot jaunu kļūdas DDQN kļūdas funkciju 41. vienād. DDQN kļūdas funkcijas gadījumā apmācības laikā  $Q_{target}$  modeļa svāri tiek kopēti un sasaldēti no  $Q_{\Theta}$  ik pēc katra, iepriekš definēta skaita laika soļu.

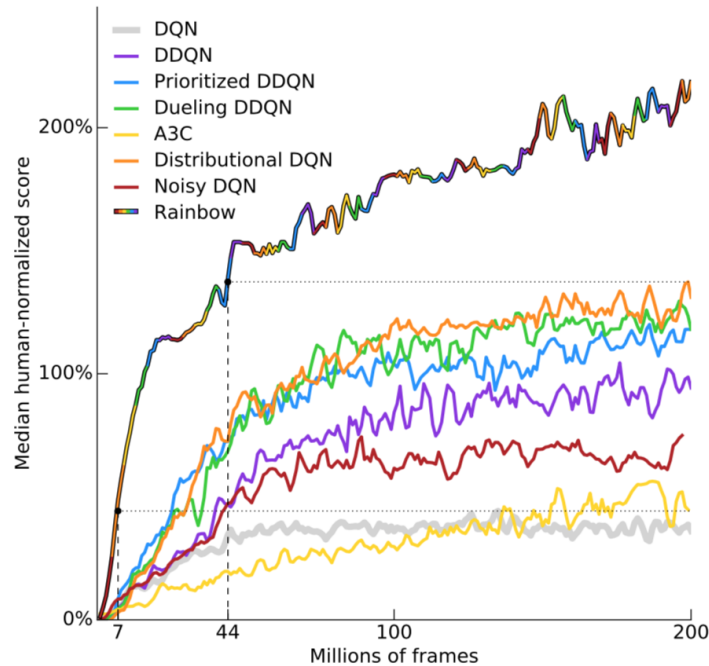
$$\mathcal{L}_{dqn} = \begin{cases} (r_t + \gamma \max_{a'} Q_{\Theta}(s_{t+1}, a') - Q_{\Theta}(s_t, a_t))^2 & \text{if } t < t_{last} \\ (r_t - Q_{\Theta}(s_t, a_t))^2 & \text{if } t = t_{last} \end{cases} \quad (40)$$

$$\mathcal{L}_{ddqn} = \begin{cases} (r_t + \gamma \max_{a'} Q_{target}(s_{t+1}, a') - Q_{\Theta}(s_t, a_t))^2 & \text{if } t < t_{last} \\ (r_t - Q_{\Theta}(s_t, a_t))^2 & \text{if } t = t_{last} \end{cases} \quad (41)$$



10. att. Atari spēles, kuras tika izmantotas, lai pārbaudītu DQN modeli [68].

Visbeidzot Varavīksnes-DQN modelis parāda, ka apvienojot visas nelielās izmaiņas kopā vienā modelī, tas pārspēj katru no izmaiņām atsevišķi [35], kā parādīts 11. att. Attēlā dots salīdzinājums normalizētiem Atari spēļu rezultātiem, izmantojot dažādas kļūdas funkcijas un metodes atkarībā no miljoniem kadru, kuri bija nepieciešami, lai apmācītu aģentu. Varavīksnes-DQN modelis izmanto DDQN kļūdas funkciju kā arī iepriekš minētās un vēl citas izmaiņas.



11. att. Varavīksnes-DQN modeļa un tā daļu salīdzinājums, izmantojot normalizētu Atari balvas vērtību [35].

### 3.3. MDQN kļūdas funkcija

Šajā disertācijā ir aprakstīta jauna MDQN (Multi Deep Q-Network) kļūdas funkcija, kura ir dinamiska un maina savas īpašības apmācības laikā. Līdzīgi kā DDQN tā satur mērķa modeļus, kuri tiek atjaunināti ik pēc noteikta skaita laika soļu, bet atšķirībā no DDQN kļūdas funkcijā tiek izmantota vairāk kā viena kopija DQN svaru. Divas, trīs vai vairāk kopijas svaru var tikt izmantotas, lai samazinātu neseno un nejaušu notikumu ietekmi uz apmācības procesu. Šīs svaru kopijas tiek mainītas savā starpā balstoties uz dažādiem soļu skaitiem, tādējādi liekot modelim atcerēties dažādus notikumus katrā no DQN modeļu kopijām, lai veidotu pilnvērtīgi galveno DQN modeli. MDQN kļūdas funkcija pseidokodā ir aprakstīta Algoritmā 1. alg.

Atkarībā no MDQN varianta vairāki DQN modeļu svāri  $Q_1, Q_2, \dots, Q_n$  tiek inicializēti apmācības sākumā. Piemēram, MDQN-3 satur 3 paralēlus Q

modeļus, pēc tam sākotnējie  $Q_a$  un  $Q_b$  tiek nejauši izvēlēti no DQN modeļu kopas. Kadru skaitītāji  $c_a, c_b, \dots, c_n$  tiek uzstādīti ar skaitlisko vērtību nulle. Hiper-parametri  $threshold_a, threshold_b, \dots, threshold_n$ , kuri nosaka mērķa modeļa maiņu, tiek atrasti, izmantojot parametru režģa pārmeklēšanu, taču tos var arī uzstādīt kā apmācāmus parametrus. MDQN arī izmanto stāvokļu pāreju kopas  $a_t, s_t, s_{t+1}, R_t$ , kuras ir līdzīgas tām, kas ir redzētas atmiņā, lai apmācītu Q-Vērtību funkciju, izmantojot patiesās kumulatīvās balvas vērtības. Šīs pārejas var viegli viegli salīdzināt atmiņā, kad tiek izmantotas vides ar mazu skaitu dimensiju, kā tas ir PLE (PyGame Learning Environment) [104], kā arī tās var tik atrastas arī daudzdimensiju (pikseļu matricu) vidēs, izmantojot dziļo metriku apmācību, kura aprakstīta nākamajā nodaļā.

1. algorithms. MDQN loss function

```

1: procedure TRAIN
2:    $Q_1, Q_2, \dots, Q_n$ 
3:    $threshold_a, threshold_b, \dots, threshold_n$ 
4:    $c_a, c_b, \dots, c_n = 0$ 
5:    $Q_a = Sample(Q_1, Q_2, \dots, Q_n)$ 
6:    $Q_b = Sample(Q_1, Q_2, \dots, Q_n)$ 
7:   while  $Training = True$  do
8:     for do  $\{a_t, s_t, s_{t+1}\}$  sample from ReplayBuffer
9:        $\forall n, c_n = c_n + 1$ 
10:      if  $\forall n, c_n > threshold_n$  then
11:         $Q_b = Q_a$ 
12:         $Q_a = Q_n$ 
13:         $c_n = 0$ 
14:      if  $\{a_t, s_t, s_{t+1}\}$  similar exist in ReplayBuffer then
15:         $Q_a(a_t, s_t) \leftarrow \sum_{t=0}^{t+1} \gamma^t R_t$ 
16:      else
17:        if  $s_t \neq$  terminal state then
18:           $Q_a(a_t, s_t) \leftarrow R_t + \gamma \max_a Q_b(a, s_{t+1})$ 
19:        else
20:           $Q_a(a_t, s_t) \leftarrow R_t$ 
21:      while  $s_t \neq$  terminal state do
22:         $a_t \leftarrow \max_a average(\{Q_a(a, s_t), Q_b(a, s_t)\})$ 
23:        ...
24:      store  $\{a_t, s_t, s_{t+1}, r_t\}$  in ReplayBuffer

```

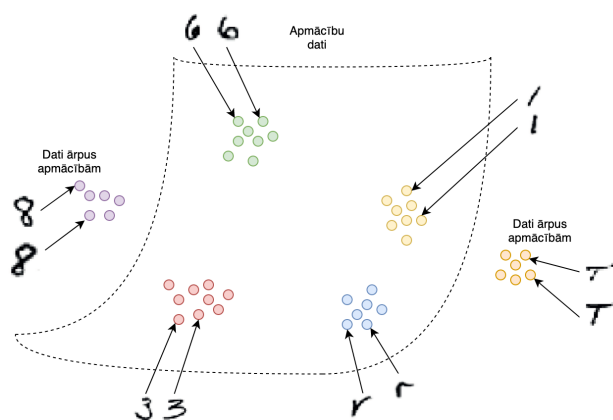
## 4. FUNKCIJU FORMĒŠANA DZIĻAJAI METRIKU APMĀCĪBAI

Šī nodaļa iepazīstina ar jaunizveidoto Eksponenciālo Trijotnes kļūdas funkciju un ar jaunām apmācības procedūrām dziļo metriku apmācības uzdevumiem. Tajā arī tiek iepazīstināts ar jaunām normalizācijas funkcijām latentajiem vektoriem, kuras nodrošina leņķisko attālumu īpašības, izmantojot Eiklīda attālumus.

Problēmsfēra re-identifikācijas uzdevumam aprakstīta 4.1. nod. Jau zināmas dziļās mašīnāpmācības kļūdas funkcijas, lai risinātu re-identifikācijas uzdevumu aprakstītas 4.2. nod. Visbeidzot, jaunā kļūdas funkcija un normalizācijas funkcijas aprakstītas 4.3. nod. To rezultāti doti 5.3. nod.

### 4.1. Nulles-šāviena mašīnāpmācība un re-identifikācijas uzdevums

Nulles-šāviena mašīnāpmācība ir metode [66] [23] [44] [62], kura balstās uz modeli, kurš tiek pielietots kategorizācijas uzdevumam jaunām, apmācības laikā neredzētām klasēm, kā parādīts 12. att. Parasti datu kopas apmācībai un testēšanai ir no vienas un tās pašas sfēras, piemēram, seju fotoattēlu, balss ierakstu, auto fotoattēlu, u.c. datu kopas. Ieguvums šādiem modeļiem ir tajā, ka tos nav nepieciešams vēlreiz apmācīt, kad uzdevumā parādās jaunu klašu piemēri, un šo modeļu uzvedību var pielāgot arī pēc apmācības fāzes. Pielāgošanu veic, mainot parametrus klasterizācijas un latentu vektoru telpas normalizācijas parametriem pēc apmācības. Vēl viens ieguvums ir, ka jaunajām klasēm nepieciešami tikai pāris datu piemēri, nevis tūkstošiem datu piemēru kā parasti tas būtu nepieciešams klasifikācijas modeļiem, kuri neizmanto nulles-šāviena mašīnāpmācības metodes [116]. Nulles-šāviena apmācība ir efektīva pie maza paraugu skaita, pat ar 2 datu paraugiem var sasniegt augstu klasifikācijas precizitāti [116].



12. att. Nulles-šāviena mašīnāpmācības klasteru vizualizācija, izmantojot EMNIST datu kopu, kur tās sadalītas tā, lai apmācības un testa datu kopās nebūtu iekļautas vienas un tās pašas klases.

Dažreiz zinātniskā literatūra nulles-šāviena apmācību sauc arī par viena-šāviena apmācību, lai gan tehniski viena-šāviena apmācība nozīmē, ka viens paraugs no mērķa klasēm ir ticis iekļauts apmācības procedūrā. Tāpat eksistē arī pāris-šāvienu mašīnāpmācība, kur pāris paraugu no mērķa klasēm tiek iekļauti apmācību kopā. Šādus modeļus parasti papildus apmāca vēlāk, izmantojot pārcelšanas apmācības metodes (transfer learning), lietojot šīs pāris jaunās klases. Nulles-šāviena mašīnāpmācības gadījumā modelis netiek nemaz apmācīts ar jauno klašu paraugiem.

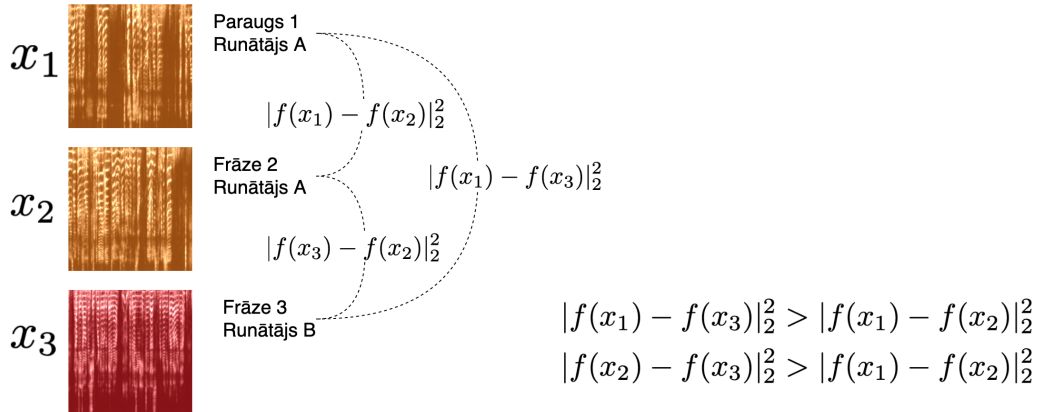
Modelis ir kļuvis par kodētāju, kurš spēj daudzus dimensiju datus kā, piemēram, fotoattēlus, iekodēt pāris dimensiju latentajos vektoros, starp kuriem attālumi jeb distances nodrošina to klasterizāciju pēc semantiskās informācijas. Šādus nulles-šāviena modeļus arī nereti sauc par dziļo metriku apmācību (DML) vai distanču metriku apmācību.

Šie modeļi bieži tiek lietoti seju re-identifikācijas [23] un citu biometrisku datu verifikācijas sistēmām kā parādīts 13. att.

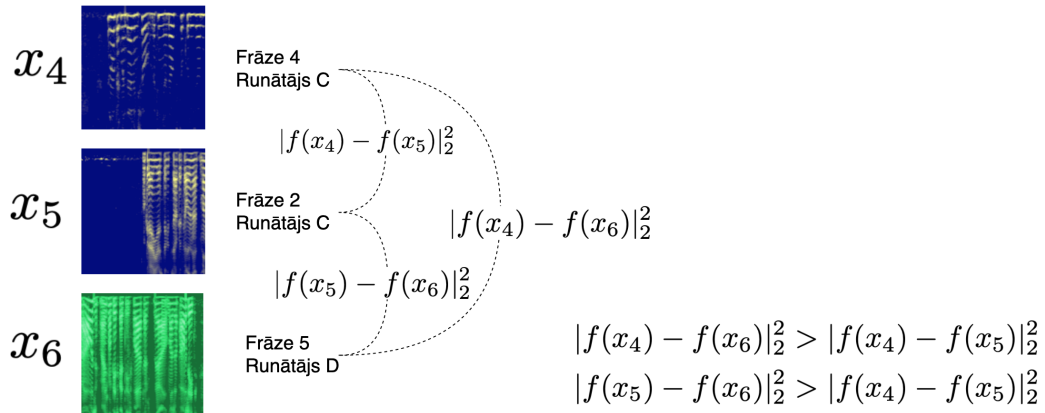
Šādās sistēmās katra persona tiek uzņemta sistēmā, izmantojot vienu vai vairākus paraugus ar sejas fotoattēlu, kuri pieder jaunai datu klasei, kuru pēcāk modelis spēj atpazīt jebkurā fotoattēlā, kurš netika iekļauts uzņemšanas procedūrā.



**Runātāja verifikācijas uzdevums  
(paraugi, kuri iekļauti apmācībā)**



**Runātāja re-identifikācijas uzdevums  
(paraugi, kuri nav iekļauti apmācībā)**



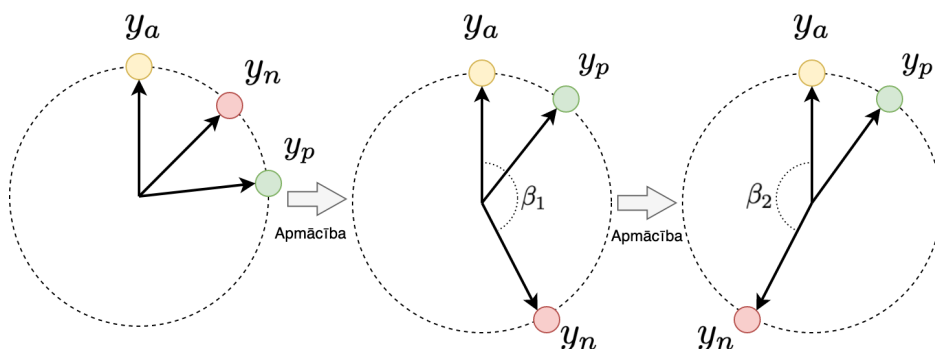
13. att. Atšķirība starp runātāja verifikācijas un runātāja re-identifikācijas uzdevumu (implementēts praktiskā pielietojumā - komerciālā lietotnē "asya") Re-identifikācijas uzdevums pieder pie nulles-šāviena apmācības, jo modelis  $f$  nav redzējis audio paraugus  $x_4, x_5, x_6$  apmācības laikā.

## 4.2. Trijotnes Kļūdas funkcija

Lai modelis no ievades datiem iemācītos dziļas reprezentācijas latentajā vektoru telpā, izmantojot nulles-šāviena apmācību, parasti tiek izmantota Kontrastīvā kļūdas funkcija [8] vai Trijotnes kļūdas funkcija [23].

Trijotnes kļūdas funkcijas 42. vienād. mērķis ir samazināt attālumu latentajiem vektoriem starp vienas klases paraugiem  $\|y_a - y_p\|_2^2$  un palielināt attālumu starp dažādu klašu paraugiem  $\|y_a - y_n\|_2^2$  kā parādīts 14. att.

$$\mathcal{L}_{std} = \left| \|y_a - y_p\|_2^2 - \|y_a - y_n\|_2^2 + \alpha \right|_+ \quad (42)$$

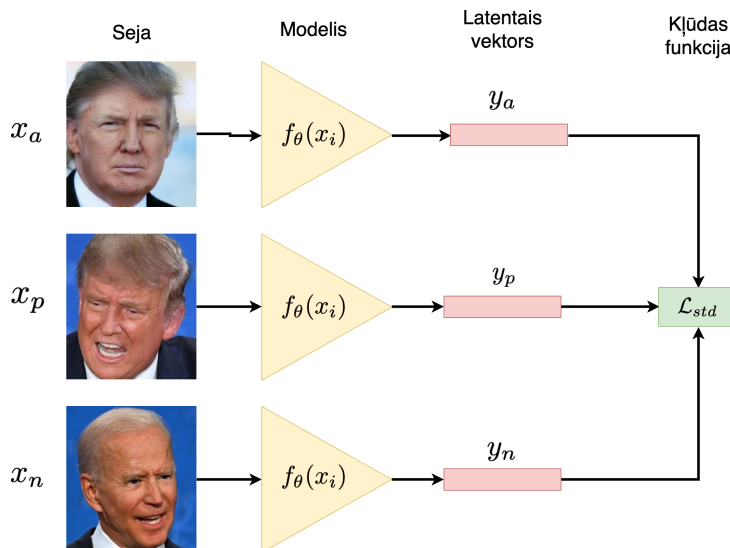


14. att. Apmācības process, izmantojot trijotnes kļūdas funkciju 42. vienād. ar kosinusa distanci starp enkura  $y_a$  vektoru, un tās pašas klases pozitīvo vektoru  $y_p$ , un citas klases negatīvo vektoru  $y_n$ . Kad ir sasniegts maksimālais attālums starp vektoru pāriem, tad šis attālums atkal samazinās.

Tajā pašā laikā funkcija ir izveidota tā, lai tā nereducētu visus vienas klases vektorus vienā punktā telpā, bet, lai starp tiem būtu robežas attālums  $\alpha$ . Nereti kā attālumu metriku izmanto kosinusa vai Eiklīda distances. Vēlams izmantot kosinusa distanci, jo tai ir cikliska daba, kur paraugu attālums sasniedzot robežvērtību 2 atkal samazinās atpakaļ līdz robežvērtībai 0.

Trijotnes kļūdas funkcijām nepieciešams paraugu meklēšanas algoritms un ierobežojumi kādi paraugi drīkst tikt lietoti ar šo kļūdas funkciju. Piemēram, parasti modelis tiek apmācīts ar pozitīvajiem vienas klases paraugu pāriem, starp kuriem ir lieli attālumi un starp negatīvajiem paraugu pāriem, kuriem ir mazi attālumi. Šādi paraugi, pabeidzot apmācību, nav vēlamī, tāpēc tie dod lielāko ietekmi uz kļūdas funkciju, taču šo paraugus ir īpaši jāmeklē, un tie parasti netiek atlasīti nejaušā veidā.

Modelis  $f_\theta(x)$  ir kodētājs, kurš reducē dimensiju skaitu no daudz-dimensiju ievades datiem  $x$  uz mazāka skaita dimensiju latentu vektoru  $y$ . Latentam vektoram jābūt vismaz 32 dimensijām vai vairāk, kā paskaidrots pielikumā [110]. Parametri  $\theta$  ir vienādi visiem datu paraugiem  $x$ , kā parādīts 15. att.



15. att. Piemērs trijotnes kļūdas funkcijai seju re-identifikācijas uzdevumam. Kodētāja modelim  $f_\theta(x_i)$  ir kopēji svāri. Dimensiju skaits  $y_i$  vektoram ir ievērojami mazāks, kā  $x_i$  ievades datiem.

### 4.3. Eksponenciālā kļūdas funkcija

Disertācijā līdz ar citiem jaunievedumiem, ir aprakstīta jaunizveidotā Eksponenciālā Trijotnes kļūdas funkcija  $\mathcal{L}_{exp}$ , kuras skaitlisko vērtību telpā kļūdas telpa parādīta 16. att., kurā apmācība konverģē ātrāk par Trijotnes kļūdas funkciju  $\mathcal{L}_{std}$ , kura aprakstīta iepriekšējā nodaļā.

Eksponenciālajai Trijotnes kļūdas funkcijai  $\mathcal{L}_{exp}$  ir forma, kura nepieļauj negatīvo pāru attālumus, kuri būtu mazāki par pusi no maksimālā attāluma sfēriskajā telpā  $\max(f_{emb}(x))$ , kurš bez mērogošanas parasti ir  $\max(f_{emb}(x)) = 2.0$ .

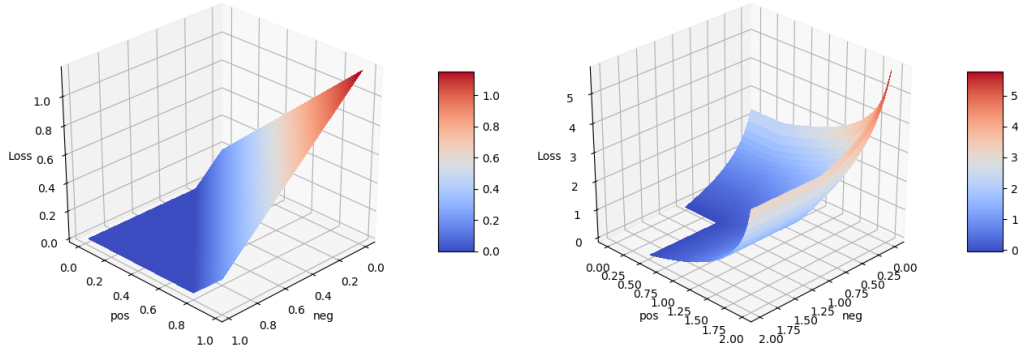
Savukārt  $c_n$  ir minimālā klasteru distance līdzīgi kā robežvērtība  $\alpha$ , kuru lieto Trijotnes kļūdas funkcijā  $\mathcal{L}_{std}$ . To izrēķina, dalot maksimālo latentu vektoru

telpas distanci  $\max(f_{emb}(x))$  ar klašu skaitu apmācību apmācību kopā  $K$  43. vienād.

$$c_n = \frac{\max(f_{emb}(x))}{K} \quad (43)$$

$$emb_p = \frac{\|y_a - y_p\|_2^2}{\max(f_{emb}(x))} \quad emb_n = \frac{\|y_a - y_n\|_2^2}{\max(f_{emb}(x))} \quad (44)$$

$$\mathcal{L}_{exp} = -\log(1.0 - \frac{|emb_p - c_n|_+}{1 - c_n} + \epsilon) - \log(1.0 - \frac{|0.5 - emb_n|_+}{0.5} + \epsilon) \quad (45)$$



16. att. Salīdzinājums starp  $\mathcal{L}_{std}$  un  $\mathcal{L}_{exp}$  kļūdas funkcijām. Pozitīvie vienas klases pāru attālumi attēloti ar  $\|y_a - y_p\|_2^2$  (pos) un negatīvie dažādu klašu attālumi attēloti ar  $\|y_a - y_n\|_2^2$  (neg).

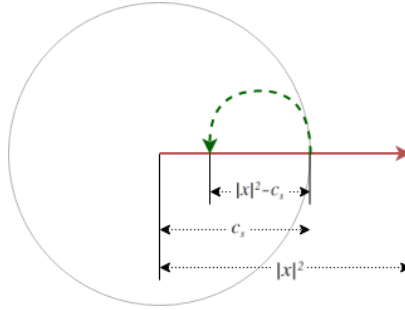
Lai vēl vairāk uzlabotu  $L_{exp}$  veiktspēju, vairākas citas kļūdas funkcijas no neseniem pētījumiem tika pievienotas kļūdas funkcijai, lai izveidotu kompozīta kļūdas funkciju  $L_{comp}$  49. vienād. Tika pievienotas L2-ierobežotā krusteniskā-entropijas kļūdas funkcija  $\mathcal{L}_{class}$  [84] un Centra kļūdas funkcija  $\mathcal{L}_{center}$  [128] 47. vienād. 48. vienād. L2-ierobežotās krusteniskās-entropijas kļūdas funkcijas  $\mathcal{L}_{class}$  gadījumā pirms Softmax funkcijas latentais vektors  $f(x)$  tiek L2 normalizēts un mērogots ar  $s$ . Savukārt  $\mathcal{L}_{center}$  gadījumā, apmācību laikā klašu latentie vektori tiek akumulēti un tiek izrēķināti  $c_{y_i}$  klašu masas centru vektori, kuri tiek izmantoti šajā kļūdas funkcijā.

$$\mathcal{L}_{class} = - \sum_{i=1}^M y_i \log \frac{e^{W_i^T s |f(x_i)|_2^2 + b_i}}{\sum_{j=1}^C e^{W_j^T s |f(x_i)|_2^2 + b_j}} \quad (46)$$

$$\mathcal{L}_{center'} = \sum_{i=1}^M \|x_i - c_{y_i}\|_2^2 \quad (47)$$

$$\mathcal{L}_{center} = \sum_{i=1}^M \left| \|x_i - c_{y_i}\|_2^2 - \frac{c_n}{2} \right|_+ \quad (48)$$

$$\mathcal{L}_{comp} = \mathcal{L}_{exp} + C_{center} \mathcal{L}_{center} + C_{class} \mathcal{L}_{class} \quad (49)$$



17. att. Ilustrācija Unit-Bounce normalizācijas funkcijai priekš latentajiem vektoriem Eksponeciālajā Trijotnes kļūdas funkcijā.

Vēl viens jaunievedums šajā darbā ir latentu vektoru normalizācijas funkcija Unit-Bounce 50. vienād. un 51. vienād. Tai ir līdzīgas īpašības kā kosinusa distancei, bet tā tiek aprēķināta ar Eiklīda distancēm. Kad latentais vektors sasniedz sfēras rādiusu  $c_s$ , tas virzās atpakaļ latentās telpas centra virzienā kā parādīts 17. att., bet kad tas sasniedz otru malu sfērai tas atkal virzās uz centru pretējā virzienā. Tādējādi tiek panākts, ka visa 3D latentā telpa tiek efektīvi izmantota, nevis kā L2 gadījumā, kur vektori atrodas tikai uz sfēras virsmas.

$$f'_{emb}(x) = \begin{cases} f_{bounce}(x), & \text{if } |x|^2 \geq 1 \\ x, & \text{citos gadījumos} \end{cases} \quad (50)$$

$$f_{bounce}(x) = \begin{cases} |x|^2 - \left\lfloor \frac{|x|^2}{c_s} \right\rfloor - c_s \frac{x}{|x|^2}, & \text{if } \left\lfloor \frac{|x|}{c_s} \right\rfloor \bmod 2 = 0 \\ c_s \frac{x}{|x|^2} - |x|^2 - \left\lfloor \frac{|x|^2}{c_s} \right\rfloor, & \text{citos gadījumos} \end{cases} \quad (51)$$

## 5. EKSPERIMENTĀLIE REZULTĀTI UN PRAKTISKIE PIELIETOJUMI

Šajā nodaļā doti galvenie eksperimentu rezultāti jaunizveidotajām kļūdas funkcijām un metodēm, kuras aprakstītas 2. nod. , 3. nod. un 4. nod. Jaunizveidotās metodes tika pārbaudītas dažādās problēmsfērās, sākot ar VI algoritma optimizācijas, dziļās mašīnāpmācības dažādās datorspēļu vidēs, dziļās metriku apmācības sejas re-identifikācijas uzdevumam, un beidzot ar praktiskiem pielietojumiem analītiskajā ķīmijā un cilvēka balss analizē. Pilni eksperimentālie rezultāti atrodami publikācijās, kuras tika pievienotas pilnajai disertācijai kā pielikumi [111] [109] [24] [112] [110].

### 5.1. UNet-RNN-Skip rezultāti

UNet-RNN-Skip modelis tika eksperimentāli testēts VI algoritma problēmas risināšanai, lai paātrinātu vērtību funkcijas konvergenci. VIN VIN (Value Iteration Network) modelis ir balstīts uz jaunizveidoto UNet-RNN-Skip, kuru var pielietot arī citu problēmu risināšanai. VIN modelis ir pielietojams dažādu izmēru aizņemtības režģu kartēm, kā parādīts 18. att. un 3. tab., un to nav nepieciešams papildus apmācīt noteiktam kartes izmēram. VIN modeļa konverģences ātrums samazinās lineāri starp dažādu izmēru kartēm, bet VI algoritma konverģences ātrums samazinās eksponenciāli, palielinoties kartes izmēram. Metrika "mērķa sasniegšanas proporcija" (success rate) tika izmantota, lai novērtētu modeļa stabilitāti uz testa datu kopas. Šī metrika norāda uz procentuālu skaitu režģa šūnu, no kurām ar VIN palīdzību var nokļūt līdz pozitīvam terminālam stāvoklim kartē. VI algoritmam šīs metriks vērtība vienmēr būs 1.0, jo pēc konverģences VI algoritms ir spējīgs iegūt optimālu ceļa plānojumu no jebkuras režģa šūnas kartē. Pētījumā arī tika noskaidrots, ka VI uzdevumam UNet modeļi pārspēja precizitātē kovolūciju AE modeļus, kā parādīts 2. tab.

2. tabula.

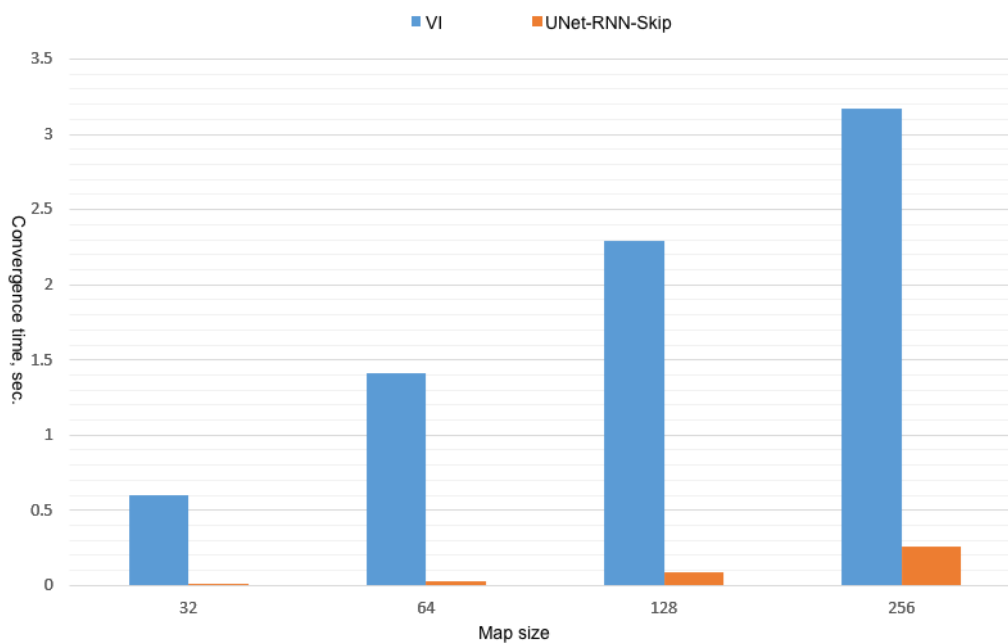
Salīdzinājums ConvNet un UNet modeļu rezultātiem, lai atrisinātu VI algoritmu.

Modelis	Kļūda	Mērķa sasnieg. pr.	Epočas laiks (min)
Conv-AE-RNN	8.58E-06	0.598	10.862
UNet-RNN-Skip	3.04E-06	0.998	15.833

3. tabula.

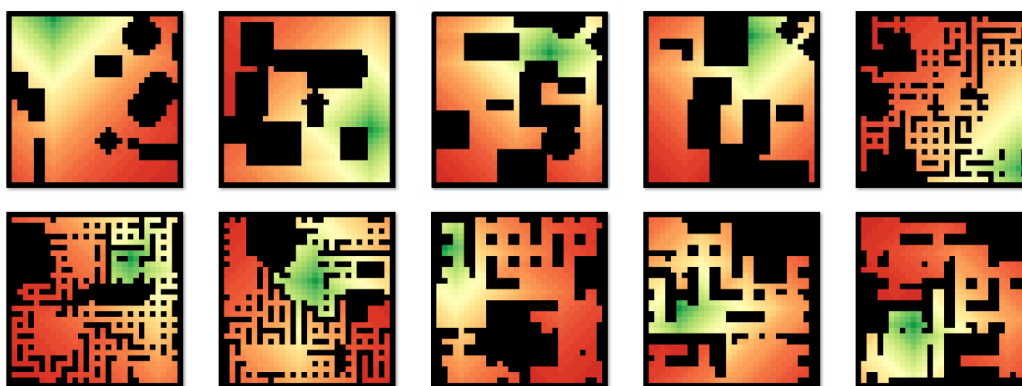
Salīdzinājums konverģences ātrumam (sek.) starp VIN un VI metodēm.

Modelis / Kartes izm.	32	64	128	256
VI	2.95	24.873	195.902	1473.108
VIN	0.031	0.071	0.236	0.833



18. att. Salīdzinājums konverģences ātrumam (sek.) starp VIN un VI metodēm.

Līdz ar jaunizveidoto UNet-RNN-Skip modeli šī pētījuma ietvaros tika izveidots arī sintētiskas datu kopas ģenerators OccupancyMapGenerator, kurš spēj izveidot aizņemtības režģu kartes ar šķēršļiem, kā parādīts 19. att. Ģenerators ir spējīgs izveidot jebkura izmēra 2D aizņemtības režģa karti, izmantojot komandrindas argumentus. Tas izveido kartes PNG attēlu formātā un arī izpilda VI algoritmu uz šīm kartēm, un šo rezultātu arī saglabā kā PNG attēlu. Ģeneratoram var norādīt šķēršļu daudzumu kartē. Šķēršļi sastāv no nejauši izveidota labirinta, taisnstūriem un apliem. Dijkstra ceļu meklēšanas algoritms [18] tika pielietots, lai pārbaudītu, vai no visām režģa šūnām var nonākt pozitīvā terminālā stāvoklī. Labirinta izveidošanai tika lietots rekursīvs atpakaļ-iešanas algoritms. Pilns karšu ģenerēšanas algoritms dots pseudo kodā zemāk 2. alg.



19. att. Piemēri sintētiskajām kartēm, kuras ģenerētas ar OccupancyMapGenerator un šūnu vērtībām, lai sasniegtu pozitīvu terminālu stāvokli (Zaļš – augstākā balvas vērtība, Sarkans - zemākā balvas vērtība).



2. algorithms. OccupancyMapGenerator map generation algorithm

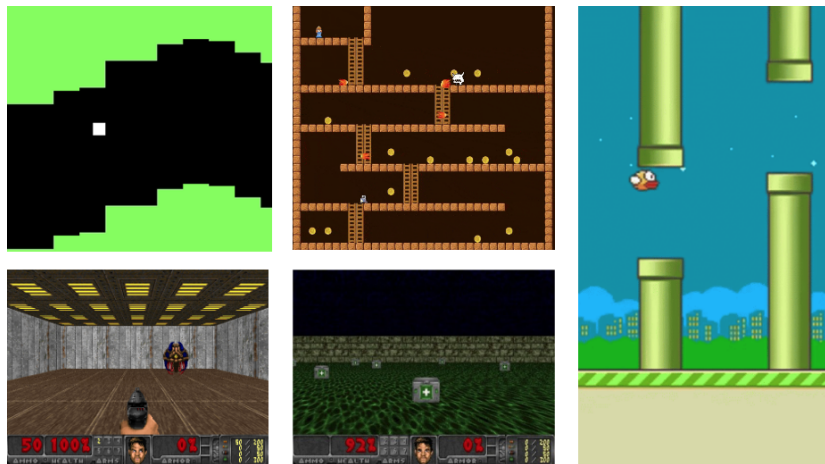
```

1: procedure GENERATEMAP
2:   size
3:   types_obstacles = {maze, circles, rectangles}
4:   max_coverage
5:   iterations_obstacles
6:    $\epsilon_{VI}$ 
7:    $M_{size \times size} \leftarrow generateZeros(size)$ 
8:   if maze  $\in$  types_obstacles then
9:      $M \leftarrow generateMaze(M)$ 
10:  if circles  $\in$  types_obstacles or rectangles  $\in$  types_obstacles then
11:    for iterations_obstacles do
12:       $coverage = \frac{walkable}{size^2}$ 
13:      if  $coverage < max\_coverage$  then
14:         $M \leftarrow generateObstacles(M, types\_obstacles)$ 
15:      else
16:        break
17:       $goal_{x,y} \leftarrow RandomWalkablePosition(M)$ 
18:      for  $pos_{x,y}$  in  $M$  do
19:        if  $pos_{x,y} \in walkable$  then
20:           $reachable \leftarrow dijkstra(M, goal_{x,y}, pos_{x,y})$ 
21:          if  $\neg reachable$  then
22:             $M \leftarrow fillHole(M, pos_{x,y})$ 
23:       $M_{vi} \leftarrow valueIteration(M, \epsilon_{VI})$ 
24:       $store(M_{vi}, M)$ 

```

## 5.2. MDQN rezultāti

MDQN kļūdas funkcija [109] tika pārbaudīta dažādās datorspēļu vidēs, izmantojot PLE (PyGame Learning Environment) [104]. Tā tika notestēta spēlēs kā, piemēram, “Flappy Bird”, Pong, “3D Raycast Maze”, Doom, kā parādīts 20. att. Šīs spēles satur daudz-dimensiju spēles stāvokļus jeb zīmētā attēla pikseļu matricu katram spēles kadram, bet dažas spēles kā, piemēram, Pong piedāvā arī maz-dimensiju spēles stāvokļus, kuri ir noderīgi, lai ātri pārbaudītu jaunus modeļus vai kļūdas funkcijas pirms tās tiek apmācītas uz sarežģītākiem daudz-dimensiju spēļu stāvokļiem. Kā redzams 4. tab., MDQN kļūdas funkcija sasniedza augstākus rezultātus kā DDQN kļūdas funkcija, kura publikācijas izstrādes laikā bija labākā metode Q-funkcijā balstītai stimulētajai mašīnāpmācībai [30]. Visās vidēs un kļūdas funkciju kombinācijās rezultāti tika iegūti ar hiper-parametru režģa pārmeklēšanu, lai nodrošinātu jēgpilnu salīdzinājumu starp metodēm, tāpat eksperimenti tika atkārtoti 20 reizes.



20. att. PLE spēļu piemēri. Augšējā rindā no kreisās puses “Pixel Helicopter”, “Monster Kong”, bet no labās puses “Flappy Bird”. Apakšējā rindā 2 varianti Doom spēlēm, kuri arī tika izmantoti šajā pētījumā (Video ar aģentu, kurš apmācīts ar MDQN kļūdas funkciju pieejams <https://www.youtube.com/watch?v=oqN6rtnv1EI>).

MDQN kļūdas funkcija ieguva labākus rezultātus nekā DDQN kļūdas funkcija un to rezultāti bija stabilāki, atkārtojot eksperimentus vairākas reizes, kā parādīts 4. tab.

4. tabula.

Vidēja balvas vērtība, atkārtojot eksperimentus 20 reizes, un, izmantojot dažādas kļūdas funkcijas, PLE vidēs.

Kļūdas funk.	Vide	Vid. balva
MDQN	Flappy Bird	17.2
DDQN	Flappy Bird	16.9
MDQN	Pong	1.7
DDQN	Pong	1.3
MDQN	3D Raycast Maze	3.9
DDQN	3D Raycast Maze	3.7

Līdz ar MDQN kļūdas funkciju, šajā darbā tika izveidota arī jauna metode Q-funkcijas vērtību vizuālai attēlošanai, katram spēles stāvoklim pikseļu matricā. Jaunās Q-Vērtību kartes tiek iegūtas, manipulējot ar objektiem pašā datorspēles vidē, tādējādi iegūstot sapratni par aģenta politiku katrā iespējamā spēles stāvoklī. Tā kā PLE vides ir veidotas ar atvērtu kodu, tad ir iespējams manipulēt ar spēlētāja atrašanās vietu spēlē, kas ļauj izrēķināt Q-vērtību katrā pikseļa pozīcijā viena stāvokļa ietvaros, kā tas parādīts 21. att.



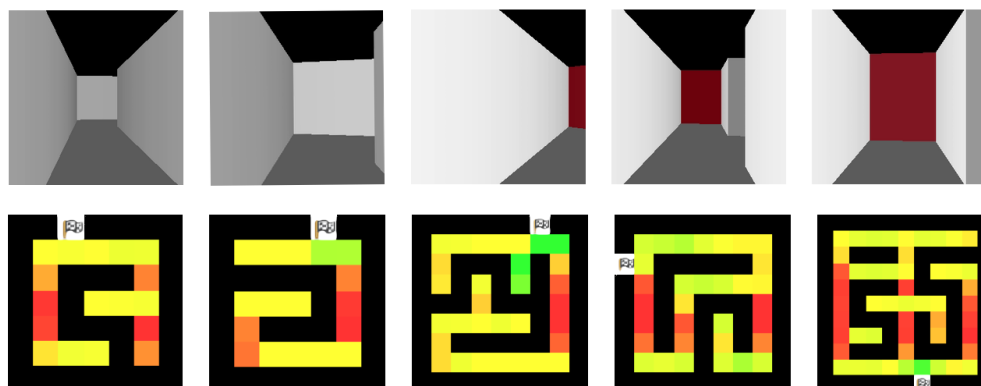
21. att. Q-Vērtību karte FlappyBird vidē, kur katra pikseļa vērtība nosaka kāda būtu stāvokļa Q-Vērtība, ja spēlētājs (putns) atrastos šajā pozīcijā. Sarkana krāsa apzīmē zemu Q-Vērtību, bet zaļa krāsa apzīmē augstu Q-Vērtību. Pa kreisi ir Q-Vērtību kartes pirms apmācības, pa vidu apmācības laikā, pa labi pēc apmācības.

Pat 3D vidēm, kā “3D raycast maze” (3D labirints no pirmās personas skatupunkta) ir iespējams noteikt katra stāvokļa Q-Vērtību, pagriežot kameru ap Z asi un izrēķinot vidējo Q-Vērtību no katra kadra, kas tika iegūts 360 grādu pozīcijās. Pēc tam vidējās vērtības tiek saglabātas virs-skatā, kā parādīts 22. att. Kā parādīts 5. tab. MDQN kļūdas funkcija ieguva augstāko vidējo balvas vērtību 20 apmācību atkārtojumos un zemāko novirzi, salīdzinot ar DDQN un DQN kļūdas funkcijām. Ar MDQN-2 un MDQN-3 tiek apzīmēts modeļu kopiju skaits 2 un 3, kuri tiek izmantoti MDQN kļūdas funkcijā.

5. tabula.

Dažādu kļūdas funkciju rezultāti 20 apmācību atkārtojumos, izmantojot “3D raycast maze” vidi.

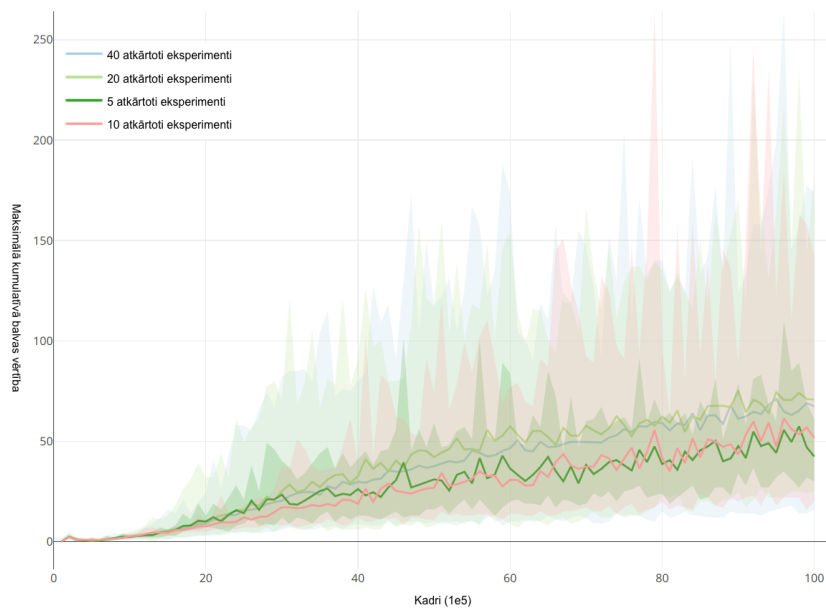
Kļūdas func.	Apmāc. Ātr.	Vid. balva	Std. balvai
MDQN-2	1.00E-05	3.904359232	0.728045918
DQN	1.00E-05	3.88654262	2.124993494
MDQN-2	1.00E-06	3.7166532	0.154117942
DDQN	1.00E-06	3.713829593	1.524318234
DDQN	1.00E-05	3.638360789	1.662039807
DDQN	0.0001	3.267777864	2.889255991
MDQN-3	1.00E-05	3.056116361	2.339890159
DQN	1.00E-06	3.026868771	2.028895348
MDQN-3	1.00E-06	2.770128326	0.714132328
MDQN-2	0.0001	2.545370799	4.120312752
DQN	0.0001	2.24425396	2.153779645
MDQN-3	0.0001	2.174641347	3.541216037



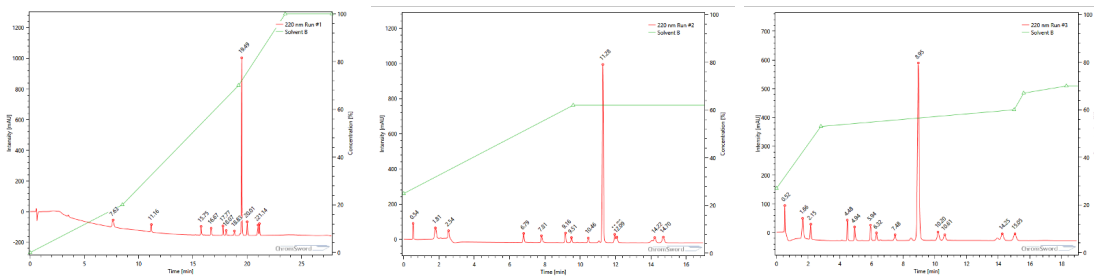
22. att. Augšējā rindā kadri no “3D raycast maze” vides. Apakšējā rindā Q-Vērtību kartes šai videi no virs-skata. Kreisā pusē pirms apmācības, pa vidu apmācības laikā un labajā pusē pēc apmācības. Apmācību laikā tika pakāpeniski palielināta labirinta sarežģītība. Sarkana krāsa apzīmē zemu Q-Vērtību, bet zaļa krāsa apzīmē augstu Q-Vērtību.

Visbeidzot, līdz ar MDQN kļūdas funkcijas izveidi, tika arī veikta citu Dziļo Q-Funkciju modeļu analīze un salīdzinājums. Tika secināts, ka šīs metodes ir ļoti jūtīgas uz nejaušo skaitļu iestādījumu spēles sākumā kā arī pašu spēļu nenoteiktību. Kā parādīts 23. att., nepieciešams izmantot pietiekamu skaitu atkārtotu eksperimentu jeb apmācību atkārtojumus ar tiem pašiem hiper-parametriem, lai noskaidrotu augstāko rezultātu katrā vidē. Tika noskaidrots, ka nepieciešams 20 apmācību atkārtojumi, lai noteiktu precīzu vidējo vērtību apmācību rezultātiem.

MDQN kļūdas funkcija un citas dziļās māšīnāpmācības metodes arī tika pielietotas komerciālā produktā SIA ChromSword. Kā parādīts 24. att. ar 3 secīgiem eksperimentiem, izmantojot šos modeļus, ir iespējams atrast HPLC šķīdinātāja gradientu, lai panāktu labāko piķu sadalījumu nezināmām vielām. Šos modeļus pielieto analītiskajā ķīmijā un zāļu izstrādē.



23. att. Novirze rezultātu vērtībās, atkārtojot apmācību dažādu skaitu reižu, izmantojot DQN kļūdas funkciju “Flappy Bird” vidē.



24. att. Secīgi HPLC eksperimenti, kurus izveido stimulētās māšīnāpmācības aģents, lai atrastu labāko šķīdinātāja gradientu, lai atdalītu pīkus nezināmām vielām.

### 5.3. Eksponeciālās Trijotnes kļūdas funkcijas rezultāti

Klašu re-identifikācijas uzdevuma rezultāti, izmantojot Eksponeciālo Trijotnes kļūdas funkciju  $L_{exp}$ , doti 6. tab., kur Standarta Trijotnes Kļūdas funkcija  $L_{std}$ , Eksponeciālā Trijotnes kļūdas funkcija  $L_{exp}$ , Centra kļūdas funkcija  $L_{cen}$  un klasifikācijas kļūdas funkcija  $L_{cls}$ . Visi rezultāti tika iegūti, veicot režģa pārmeklēšanu hiper-parametriem, katrai metodei atsevišķi. Tas ir viens no iemesliem, kāpēc arī standarta trijotnes kļūdas funkcija  $L_{std}$  ieguva augstus rezultātus daudzās no dotajām datu kopām. Visi klasifikācijas rezultāti tika iegūti, izmantojot nulles šāvienu modeļus un re-identifikācijas uzdevumu, kur pareizā klase tika noteikta pēc latentā vektora attāluma līdz klases masas centram jeb vidējam latentā vektoram no visiem vienas klases paraugiem.

Klases un paraugi, kuri tika iekļauti apmācību kopā netika iekļauti testēšanas kopā. Abām  $L_{std}$  un  $L_{exp}$  tika izmantota vairāku daļu kļūdas funkcija, kuras ietvēra arī Centra kļūdas funkciju  $L_{cen}$  un L2-Softmax klasifikācijas kļūdas funkciju  $L_{cls}$ . Klasifikācijas kļūdas funkcija  $L_{cls}$  aprēķināta tikai apmācības kopai, jo modelim nebija zināms testa kopā iekļauto klašu skaits. Pētījumā seju re-identifikācijas uzdevuma veikšanai tika izvēlēta datu kopa VGGFace2 [13] ar 9000 klasēm bet modeļi tika arī testēti uz biežāk izmantotām attēlu klasifikācijas datu kopām kā, piemēram, MNIST [55], Fashion-MNIST [130], EMNIST (Extended-MNIST) [15] un CIFAR10 [47]. Arī šim datu kopām apmācības un testa kopu sadalījums tika pārdalīts tā, lai to klases nepārklātos. Visās datu kopās tika atliktas 20% klases ar to paraugiem testēšanai, bet 80% palika apmācībai.

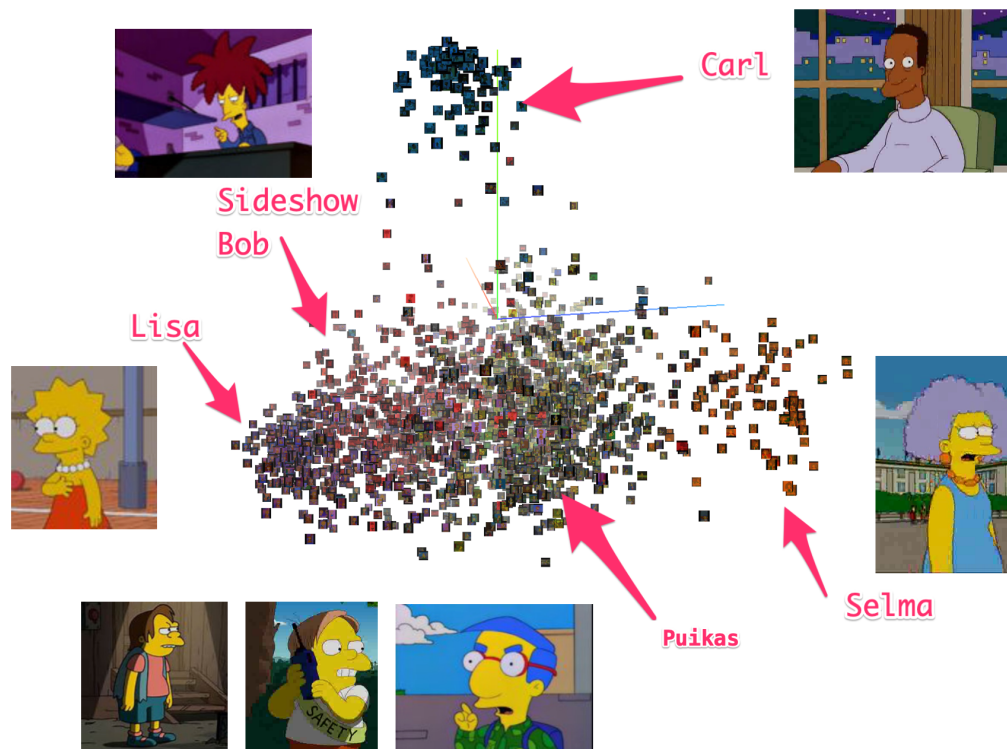


6. tabula.

Nulles-šāviena precizitātes salīdzinājums uz testa datu kopas dažādām kļūdas funkcijām.

Kļūdas func. / Prec.	MNIST	FMNIST	EMINST	CIFAR10	Simpsons	VGGFace2
$L_{std}$	<b>99.6</b>	91.4	82.0	56.2	91.0	77.4
$L_{std} + L_{cls}$	<b>99.6</b>	92.1	85.0	79.8	91.2	76.3
$L_{std} + L_{cen}$	97.5	71.5	61.7	52.1	90.9	76.4
$L_{std} + L_{cen} + L_{cls}$	97.7	82.0	70.9	62.8	91.2	78.6
$L_{exp}$	<b>99.6</b>	92.7	82.7	85.7	91.5	85.0
$L_{exp} + L_{cls}$	<b>99.6</b>	<b>93.1</b>	85.2	87.2	90.9	84.1
$L_{exp} + L_{cen}$	<b>99.6</b>	<b>93.1</b>	85.7	85.3	<b>92.1</b>	84.0
$L_{exp} + L_{cen} + L_{cls}$	<b>99.6</b>	<b>93.1</b>	<b>86.0</b>	<b>87.3</b>	91.7	<b>85.7</b>

Papildus šim pētījumam tika izmantota arī Simpsonu datu kopa, kuru var iegūt no Kaggle datu zinātnieku sacensību vietnes. Arī uz šīs datu kopas tika apmācīts re-identifikācijas uzdevums. Rezultāti, projicējot latentos vektorus no 32 dimensijām uz 3 dimensijām, izmantojot PCA redzami 25. att. Paraugā ir iespējams novērot, ka dažādas multfilmas varoņu vizuālās īpašības tiek veiksmīgi sagrupētas un to savstarpējais attālums atbilst semantiskajai nozīmei.



25. att. PCA vizualizācija Simsonu testa dartsu kopai, izmantojot modeli, kurš apmācīts ar Eksponenciālo Trijotnes kļūdas funkciju  $L_{exp}$ , un šos paraugus nav izmantojis apmācībai.

## 5.4. Darba praktiskais lietojums

Jaunās kļūdas funkcijas un modeļi, kuri aprakstīti šajā disertācijā var tikt pielietoti dažādās praktiskās implementācijās, dažas no kurām jau ir notestētas komerciālos produktos. Katrai jaunizveidotajai kļūdas funkcijai un modelim praktiskie pielietojumi uzskaitīti zemāk. Praktiskie pielietojumi UNet-RNN-Skip:

1. VI algoritma optimizācija, lai iegūtu ātrāku vērtību konvergenci un ātrāk veiktu ceļa plānošanas uzdevumu. Šis pielietojums tika notestēts pētījumu ietvaros. Šis modelis var iegūt rezultātus ar par kārtu mazāk iterācijām kā VI algoritms un ir mērogojamāks uz lielākām kartēm. Kaut arī modelis tika apmācīts uz mazākām kartēm tas, vienlīdz labi spēj strādāt uz daudz lielāka izmēra kartēm, saglabājot ātruma
2. VI algoritma optimizācija priekš stimulētās mašīnāpmācības uzdevumiem. Modeļi var pielietot arī, lai apmācītu politiku lēmumu pieņemšanas uzdevumos, kur, līdzīgi kā ceļa plānošanas uzdevumā, stāvokļi var saturēt informāciju, kura nepieciešama lēmuma pieņemšanai aģentam, bet vērtību gradients darbību grafā norāda uz lēmumu.
3. Stila pārneses uzdevums video datiem [40] [120]. Jauno modeļi var izmantot arī, piemēram, lai veiktu melnbaltu filmu krāsošanu, pievienot troksni, lai iegūtu vecmodīgu stilu vai pārveidotu filmas par multfilmām.
4. Stila pārneses uzdevums audio datiem [16] [1]. Jauno modeļi var pielietot, lai klonētu balsis vai ģenerētu mākslīgas balsis, izmantojot ierakstītu balss paraugu kā ievades datus. Modeļi var pielietot 1D audio signālam vai 2D spektrogrammām, kur to pēc tam var rekonstruēt ar Griffin-Lim algoritmu.
5. Trokšņu noņemšanas uzdevums video datiem [20] [105]. Jauno modeļi var pielietot, lai noņemtu troksni un vecu video ierakstu defektus. Tāpat modeļi var pielietot, lai uzlabotu video kompresiju, samazinot video detalizētību apgabalos, kuros skatītājs visticamāk nepamanītu atšķirību.
6. Trokšņu noņemšanas uzdevums audio datiem [58] [131]. Šis uzdevums veiksmīgi ir ticis implementēts SIA Asya komerciālajos risinājumos.

Modelis tiek izmantots, lai reālajā laikā noņemtu fona trokšņus dabīgās vidēs, atstājot tikai balss signālu. Pateicoties šim modelim SIA Asya mobilā lietotne ir spējīga sniegt sarunu analīzi trokšņainās vidēs kā, piemēram, kafejnīcās vai ofisos.

Praktiskie pielietojumi MDQN kļūdas funkcijai:

1. Q-Vērtību funkcijā balstītu modeļu apmācība dažādiem stimulētās māšīnāpmācības uzdevumiem kā, piemēram, vides simulācijai [42], pabraucošām mašīnām [134], virtuālajiem asistentiem [50] un robotu vadībai [77]. Disertācijas ietvaros jaunā MDQN kļūdas funkcija tika testēta PLE datorspēļu kontroles uzdevumos. MDQN kļūdas funkcija ir īpaši efektīva maz-dimensiju vidēs vai arī, izmantojot dziļo metriku apmācību, lai iekodētu daudz-dimensiju stāvokļus maz-dimensiju latentajos vektoros.
2. Šķīdinātāju gradientu optimizācija priekš HPLC, kuru izmanto analītiskajā ķīmijā [26]. Jaunā kļūdas funkcija ir izmēģināta komerciālā risinājuma SIA ChromSword. Modelis ir spējīgs atrast šķīdinātāja gradientu, lai veiktu vielu sastāvdaļu sadalīšanu ar secīgiem eksperimentiem divu stundu laikā, bet pirms tam šis uzdevums ar manuālu darbu prasīja vairākas dienas.

Praktiskie pielietojumi Eksponeciālo Trijotnes kļūdas funkcijai:

1. Iekodētāja modeļa pirms-apmācība, izmantojot dziļo metriku apmācību dažādiem uzdevumiem kā, piemēram, stimulētajai mašīnāpmācībai, laika sēriju apmācībai, klasifikācijai un regresijai [129]. Pirms-apmācība izmantojot Eksponeciālo Trijotnes kļūdas funkciju var uzlabot apmācības ātrumu pamat-uzdevumam. Nosacījums šādas metodes pielietojumam ir tajā, ka datu kopai jā satur klasificētus datus vai arī jā satur informācija par paraugu līdzību. Šo informāciju iespējams arī iegūt, izmantojot nepārraudzītās apmācības metodes, kā, piemēram, VAE (Variational Auto Encoders) [19].
2. Re-identifikācijas uzdevums attēliem vai biometriskajiem datiem [23] [5]. Pētījuma ietvaros jaunā kļūdas funkcija tika veiksmīgi pielietota sejas un produktu attēlu re-identifikācijas uzdevumā. Tāpat tā veiksmīgi tika pielietota SIA Asya komerciālajā risinājumā balss re-identifikācijas uzdevumam. SIA Asya veidotā mobilā lietotne reālā laikā izpilda balss

re-identifikācijas uzdevumu, izmantojot balsis biometriskos datus dabiskās sarunās. To arī ir iespējams izmantot dažādos e-komercijas risinājumos, kur nepieciešams atrast produktus pēc vizuālās līdzības.

## 6. Tālākie pētījumi

Par pamatu turpmākiem pētījumiem kļūdas funkciju īpašībām un formai var tik izmantoti rezultāti un secinājumi, kuri aprakstīti šajā darbā. Pētījumi Q-Vērtību funkcijas apmācībā un stimulētajā mašīnāpmācībā ir, galvenokārt, saistīti ar kļūdas funkciju, kura ir nemainīga neprognozējamā apmācību vidē. Q-Vērtību modeļu kļūdas funkcijai varētu pievienot ziņkārības komponenti, lai motivētu vides izpēti bez ārējas balvas vērtības. Balvas vērtību vajadzētu noteikt, izmantojot tikai stāvokļa atšķirību no prognozētā stāvokļa, izmantojot pasaules modeli. Šajā virzienā ir jau veikti pētījumi, taču problēma vēl nav pilnībā atrisināta [12] [11].

Vēl viens pētniecības virziens būtu pievienot stimulētās mašīnāpmācības modelim vairākas rakstīšanas un lasīšanas galvas, lai tas izmantotu papildus atmiņas tabulas, kurās varētu tikt glabāti redzēto stāvokļu latentie vektori, to secības un balvu vērtības. Šie modeļi tiktu apmācīti saglabāt un izmantot iekodētus faktus par vidi, lai maksimizētu kumulatīvo balvas vērtību pat ja tie nebūs redzējuši šos faktus apmācību procesa laikā, līdzīgi kā strādā nulles-šāviena mašīnāpmācība, kura tika aprakstīta iepriekšējās nodaļās. Modelis spētu iemācīties algoritmu ar kura palīdzību var atrisināt vidi nevis tikai sakarības pašā vidē. Lai šādu modeli apmācītu varētu pielietot arī papildus kļūdas funkcijas, kuras regulētu rakstīšanas un lasīšanas galvu darbību. Nesen šajā virzienā ir publicēti pētījumi, taču problēma nav atrisināta daudzdimensiju vidēm [125] [81] [51].

Kļūdas funkcijas, lai apmācītu lasīšanas un rakstīšanas galvas ārējās atmiņas modeļiem un ziņkārības modeļiem ir līdzīgas dziļo metriku apmācības kļūdas funkcijām kā trijotnes kļūdas funkcija un kontrastējoša kļūdas funkcija. Dziļo metriku apmācība priekš nulles šāviena, viena šāviena un k-šāviena apmācības arī nav vēl atrisināta problēma. Šo pētījumu varētu papildināt ar KL (Kullback–Leibler) diverģences funkciju vai kādu citu funkciju, kura novērtē varbūtības blīvumu attiecībā pret iepriekš definētiem varbūtīgajiem sadalījumiem. Nereti kombinācijā ar KL izmanto normālo sadalījumu ar apmācāmu vidēju un standartnovirzes vērtību. Vēl papildus tam būtu vēlams pievienot arī rekonstrukcijas kļūdas funkciju, kuru var iegūt no VAE tipa modeļiem vai arī diskriminatora kļūdas funkciju, izmantojot GAN tipa modeļus, kuri nereti strādā labāk kā vienkārši auto-kodētāju modeļi.

Vēl eksponenciālo trijotnes kļūdas funkciju varētu pielietot arī pirmsapmācības uzdevumiem un nepārraudzītai apmācībai, lai bez marķētāju palīdzības paplašinātu datu kopas vai iekodētu daudzdimensiju ievades datus

latentajos vektorus laika sēriju vai cita vieda modeļiem.

Nesen zinātniskajā literatūrā publicētas jaunas kļūdas funkcijas kā, piemēram, CapsNet robežu kļūdas funkcija (kapsulu tīkliem) [89] un Fokālā kļūdas funkcija [61]. Šīs funkcijas netika iegūtas no informācijas teorijas kā, piemēram, krusta-entropijas kļūdas funkcija, bet gan tās ir iegūtas eksperimentālā ceļā un tās pārspēj rezultātos klasiskās kļūdas funkcijas. Abām kļūdas funkcijām ir forma kļūdas telpā, kas palīdz tām, atkarībā no parametriem, ātrāk konverģēt. Šie pētījumi norāda uz to, ka gan klasifikācijas uzdevumos, gan citos dziļās mašīnāpmācības uzdevumos, visticamāk, vēl var atrast labākas kļūdas funkcijas, kur to forma varētu spēlēt lielu lomu.

## 7. Secinājumi

Disertācijā tika piedāvātas un pārbaudītas jaunas kļūdas funkcijas, modeļu arhitektūras un apmācību algoritmi. Tajā parādīta funkciju formas nozīme dziļajā mašīnāpmācībā.

Šī darba jaunievedumi iekļauj UNet-RNN-Skip modeli, OccupancyMapGenerator datu kopas ģeneratoru, MDQN kļūdas funkciju, Q-Vērtību kartes, Eksponeciālo Trijotnes kļūdas funkciju, Unit-Range latentās telpas normalizācijas funkciju, Unit-Bounce latentās telpas normalizācijas funkciju, un citas metodes, kuras ir publicētas zinātniskajā literatūrā un pievienotas pielikumos. Eksperimentālie rezultāti parāda, ka jaunās metodes iegūst labākus rezultātus kā populārākās dziļās mašīnāpmācības metodes katrā problēmsfērā, kur tās tika testētas.

Sejas re-identifikācijas uzdevumam un dziļo metriku apmācības uzdevumam, izmantojot VGGFace2 datu kopu, Eksponeciālās Trijotnes kļūdas funkcija ieguva labākos rezultātus, sasniedzot 85.7% precizitāti ar nulles-šāviena metodi. Eksponeciālā Trijotnes kļūdas funkcija konverģē ātrāk kā citas Trijotnes kļūdas funkcijas. Unit-Range normalizācijas funkcija un Unit-Bounce normalizācijas funkcija nodrošina labāku latentās telpas sadalījumu kā L2 normalizācijas funkcija, un jaunizveidotajām normalizācijas funkcijām ir līdzīgas īpašības kā kosinusa distancei, tikai Eiklīda telpā. Disertācijas ietvaros tika pievienota papildus nodaļa ar padziļinātu literatūras apskatu dziļo metriku apmācībai un to kļūdas funkcijām. Literatūras apskats parāda, ka jaunizveidotās kļūdas funkcijas šajā pielietojumā ir aktīva pētījumu tēma, kura vēl nav pilnībā atrisināta un turpina attīstīties vairāk nekā 30 gadu garumā.

Stimulētās mašīnāpmācības uzdevumam dažādās datorspēļu vidēs MDQN kļūdas funkcija iegūst lielākas kumulatīvās punktu summas par DDQN un DQN kļūdas funkcijām. Tā arī nodrošina funkcionalitāti, lai varētu izveidot Q-Vērtību kartes, kuras ļauj ieskatīties modeļa politikā un lēmumu pieņemšanas procesā, izmantojot vizuālu politikas reprezentāciju katram stāvoklim. Tika noskaidrots arī, ka ir nepieciešams veikt vismaz 20 atkārotus eksperimentus ar vieniem un tiem pašiem hiper-parametriem MDQN un citām kļūdas funkcijām, lai novērstu nejaušu svaru inicializācijas ietekmi vidēs kā PLE, kurās lielu lomu spēlē nejaušības. MDQN kļūdas funkcijas pētījums sevī ietver literatūras apskatu, salīdzinājumus un analīzi saistībā ar Q-Vērtību kļūdas funkcijām, kuras uzrādīja labākos rezultātus uz publicēšanas brīdi. Literatūras analīze Q-Vērtību kļūdas funkciju gadījumā parāda, ka šī vēl joprojām ir aktīva pētījumu tēma, kura nav vēl līdz galam atrisināta.

Vēl šīs disertācijas ietvaros tika radīts UNet-RNN-Skip modelis, kurš modelē vērtību funkciju un sniedz par kārtu ātrāku izpildes ātrumu, salīdzinot ar klasisko VI algoritmu. Tas iegūst tādu pašu politiku kā VI algoritms 99.8% gadījumu, un var tikt apmācīts uz 32x32 kartēm, bet pēc tam izmantots uz lielākām 256x256 kartēm. Arī jaunizstrādāto OccupancyMapGenerator karšu ģeneratoru var pielietot, lai iegūtu datu kopas aizņemtības režģa uzdevumiem. Šīs datu kopas var tikt pielietotas VI algoritma tipa uzdevumiem, kā arī ar to nesaistītiem uzdevumiem kā, piemēram, SLAM uzdevumiem.

Visas tēzes, kuras dotas 1.3. nod. tika pierādītas.

Jaunās MDQN un Eksponenciālās Trijotnes kļūdas funkcijas ir jau tikušas veiksmīgi pielietotas komerciālos produktos analītiskās ķīmijas uzdevumu veikšanā SIA ChromSword un biometrisko datu re-identifikācijas uzdevumam SIA Asya. Vēl jaunās kļūdas funkcijas var tikt pielietotas dažādiem citiem praktiskiem pielietojumiem, kuri aprakstīti 5.4. nod. Pielietojumi iekļauj nulles-šāviena uzdevumus ne tikai biometriskajiem datiem, bet arī, piemēram, līdzīgu produktu atrašanai, izmantojot foto-attēlu. Vēl tās var pielietot ceļu plānošanā, analītiskajā ķīmijā, runas modeļos, stimulētajā mašīnā =apmācībā un robotu vadības uzdevumos. Pētījums uzsver jaunu kļūdas funkciju efektivitāti dziļās mašīnāpmācības uzdevumos, un šīs funkcijas ir nevis izvestas no esošas matemātikas teorijas, bet atrastas, izmantojot empīriskas metodes.



## LITERATŪRA

- [1] Sercan Ö. Arik u. c. “Neural Voice Cloning with a Few Samples”. *ArXiv* abs/1802.06006 (2018).
- [2] Kai Arulkumaran u. c. “A Brief Survey of Deep Reinforcement Learning”. *ArXiv* abs/1708.05866 (2017).
- [3] Jimmy Ba, J. Kiros un Geoffrey E. Hinton. “Layer Normalization”. *ArXiv* abs/1607.06450 (2016).
- [4] Mohammad Babaeizadeh u. c. “GA3C: GPU-Based A3C for Deep Reinforcement Learning”. *CoRR* abs/1611.06256 (2016). URL: <http://arxiv.org/abs/1611.06256>.
- [5] H. Bredin. “TristouNet: Triplet Loss for Speaker Turn Embedding”. *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2017), 5430.—5434. lpp.
- [6] Greg Brockman u. c. “OpenAI Gym”. *ArXiv* abs/1606.01540 (2016).
- [7] J. Bromley u. c. “Signature Verification Using a ”Siamese” Time Delay Neural Network”. *Int. J. Pattern Recognit. Artif. Intell.* 1993.
- [8] Jane Bromley u. c. “Signature Verification Using A ”Siamese” Time Delay Neural Network”. *IJPRAI* 7.4 (1993), 669.—688. lpp. DOI: 10.1142/S0218001493000339. URL: <https://doi.org/10.1142/S0218001493000339>.
- [9] Edward De Brouwer u. c. “GRU-ODE-Bayes: Continuous Modeling of Sporadically-Observed Time Series”. *ArXiv* abs/1905.12374 (2019).
- [10] T. Brown u. c. “Language Models Are Few-Shot Learners”. *ArXiv* abs/2005.14165 (2020).
- [11] Yuri Burda u. c. “Exploration by Random Network Distillation”. *ArXiv* abs/1810.12894 (2019).
- [12] Yuri Burda u. c. “Large-Scale Study of Curiosity-Driven Learning”. *ArXiv* abs/1808.04355 (2019).
- [13] Qiong Cao u. c. “VGGFace2: A Dataset for Recognising Faces across Pose and Age”. *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)* (2017), 67.—74. lpp.

- [14] Sumit Chopra, Raia Hadsell un Yann LeCun. “Learning a Similarity Metric Discriminatively, with Application to Face Verification”. *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005), 20-26 June 2005, San Diego, CA, USA*. 2005, 539.—546. lpp. DOI: 10.1109/CVPR.2005.202. URL: <https://doi.org/10.1109/CVPR.2005.202>.
- [15] Gregory Cohen u. c. “EMNIST: An Extension of MNIST to Handwritten Letters”. *ArXiv* abs/1702.05373 (2017).
- [16] J. Cong u. c. “Data Efficient Voice Cloning from Noisy Samples with Domain Adversarial Training”. *ArXiv* abs/2008.04265 (2020).
- [17] Haowen Deng, Tolga Birdal un Slobodan Ilic. “PPFNet: Global Context Aware Local Features for Robust 3D Point Matching”. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2018), 195.—205. lpp.
- [18] E. Dijkstra. “A Note on Two Problems in Connexion with Graphs”. *Numerische Mathematik* 1 (1959), 269.—271. lpp.
- [19] E. Dupont. “Learning Disentangled Joint Continuous and Discrete Representations”. *NeurIPS*. 2018.
- [20] T. Ehret u. c. “Model-Blind Video Denoising via Frame-to-Frame Training”. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2019), 11361.—11370. lpp.
- [21] Lasse Espeholt u. c. “IMPALA: Scalable Distributed Deep-RL with Importance Weighted Actor-Learner Architectures”. *ArXiv* abs/1802.01561 (2018).
- [22] Chelsea Finn, S. Levine un P. Abbeel. “Guided Cost Learning: Deep Inverse Optimal Control via Policy Optimization”. *ArXiv* abs/1603.00448 (2016).
- [23] Florian Schroff, Dmitry Kalenichenko un James Philbin. “FaceNet: A Unified Embedding for Face Recognition and Clustering”. *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*. 2015, 815.—823. lpp. DOI: 10.1109/CVPR.2015.7298682. URL: <https://doi.org/10.1109/CVPR.2015.7298682>.
- [24] S. Galushko u. c. “ChromSword: Software for Method Development in Liquid Chromatography”. 2018.

- [25] Yuan Gao un Dorota Glowacka. “Deep Gate Recurrent Neural Network”. *ArXiv* abs/1604.02910 (2016).
- [26] Tarun Gogineni u. c. “TorsionNet: A Reinforcement Learning Approach to Sequential Conformer Search”. *ArXiv* abs/2006.07078 (2020).
- [27] Jacob Goldberger u. c. “Neighbourhood Components Analysis”. *Advances in Neural Information Processing Systems*. Izdevis L. Saul, Y. Weiss un L. Bottou. 17. sējums. MIT Press, 2005. URL: <https://proceedings.neurips.cc/paper/2004/file/42fe880812925e520249e808937738d2-Paper.pdf>.
- [28] Klaus Greff u. c. “LSTM: A Search Space Odyssey”. *IEEE Transactions on Neural Networks and Learning Systems* 28.10 (2017. g. okt.), 2222.—2232. lpp. ISSN: 2162-237X, 2162-2388. DOI: 10.1109/TNNLS.2016.2582924. arXiv: 1503.04069. URL: <http://arxiv.org/abs/1503.04069> (aplūkots 25.04.2019.).
- [29] Hado V. Hasselt. “Double Q-Learning”. *Advances in Neural Information Processing Systems 23*. Izdevis J. D. Lafferty u. c. Curran Associates, Inc., 2010, 2613.—2621. lpp. URL: <http://papers.nips.cc/paper/3964-double-q-learning.pdf>.
- [30] Hado van Hasselt, Arthur Guez un David Silver. “Deep Reinforcement Learning with Double Q-Learning”. *CoRR* abs/1509.06461 (2015). URL: <http://arxiv.org/abs/1509.06461>.
- [31] Kaiming He u. c. “Deep Residual Learning for Image Recognition”. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), 770.—778. lpp.
- [32] Kaiming He u. c. “Identity Mappings in Deep Residual Networks”. *ArXiv* abs/1603.05027 (2016).
- [33] Alexander Hermans, Lucas Beyer un Bastian Leibe. “In Defense of the Triplet Loss for Person Re-Identification”. *ArXiv* abs/1703.07737 (2017).
- [34] J. Hershey u. c. “Deep Clustering: Discriminative Embeddings for Segmentation and Separation”. *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2016), 31.—35. lpp.

- [35] Matteo Hessel u. c. “Rainbow: Combining Improvements in Deep Reinforcement Learning”. *ArXiv* abs/1710.02298 (2018).
- [36] S. Hochreiter un J. Schmidhuber. “Long Short-Term Memory”. *Neural Computation* 9 (1997), 1735.—1780. lpp.
- [37] C. Huang, Chen Change Loy un X. Tang. “Local Similarity-Aware Deep Feature Embedding”. *NIPS*. 2016.
- [38] Gao Huang, Zhuang Liu un Kilian Q. Weinberger. “Densely Connected Convolutional Networks”. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017), 2261.—2269. lpp.
- [39] Huimin Huang u. c. “UNet 3+: A Full-Scale Connected UNet for Medical Image Segmentation”. *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2020), 1055.—1059. lpp.
- [40] Satoshi Iizuka un Edgar Simo-Serra. “DeepRemaster: Temporal Source-Reference Attention Networks for Comprehensive Video Enhancement”. *ArXiv* abs/2009.08692 (2019).
- [41] Rafal Józefowicz, Wojciech Zaremba un Ilya Sutskever. “An Empirical Exploration of Recurrent Network Architectures”. *ICML*. 2015.
- [42] A. Kiani, Chris Wang un Angela Xu. “Sepsis World Model: A MIMIC-Based OpenAI Gym ”World Model” Simulator for Sepsis Treatment”. *ArXiv* abs/1912.07127 (2019).
- [43] B. Kitchenham u. c. “Systematic Literature Reviews in Software Engineering - A Systematic Literature Review”. *Inf. Softw. Technol.* 51 (2009), 7.—15. lpp.
- [44] Gregory R. Koch. “Siamese Neural Networks for One-Shot Image Recognition”. 2015.
- [45] A. Kolesnikov u. c. “Big Transfer (BiT): General Visual Representation Learning”. *arXiv: Computer Vision and Pattern Recognition* (2019).
- [46] Martin Köstinger u. c. “Large Scale Metric Learning from Equivalence Constraints”. *2012 IEEE Conference on Computer Vision and Pattern Recognition* (2012), 2288.—2295. lpp.

- [47] Alex Krizhevsky. “Learning Multiple Layers of Features from Tiny Images”. 2009.
- [48] Alex Krizhevsky, Ilya Sutskever un Geoffrey E. Hinton. “ImageNet Classification with Deep Convolutional Neural Networks”. *Communications of the ACM* 60.6 (2017. g. 24. maijs), 84.—90. lpp. ISSN: 00010782. DOI: 10 . 1145 / 3065386. URL: <http://dl.acm.org/citation.cfm?doid=3098997.3065386> (aplūkots 25.04.2019.).
- [49] David Krueger u. c. “Zoneout: Regularizing RNNs by Randomly Preserving Hidden Activations”. *ArXiv* abs/1606.01305 (2017).
- [50] Katya Kudashkina, P. Pilarski un R. Sutton. “Document-Editing Assistants and Model-Based Reinforcement Learning as a Path to Conversational AI”. *ArXiv* abs/2008.12095 (2020).
- [51] Guillaume Lample u. c. “Large Memory Layers with Product Keys”. *ArXiv* abs/1907.05242 (2019).
- [52] M. Law, N. Thome un M. Cord. “Quadruplet-Wise Image Similarity Learning”. *2013 IEEE International Conference on Computer Vision* (2013), 249.—256. lpp.
- [53] M. Law, R. Urtasun un R. Zemel. “Deep Spectral Clustering Learning”. *ICML*. 2017.
- [54] Quoc V. Le, Navdeep Jaitly un Geoffrey E. Hinton. “A Simple Way to Initialize Recurrent Networks of Rectified Linear Units”. *ArXiv* abs/1504.00941 (2015).
- [55] Yann LeCun un Corinna Cortes. “MNIST Handwritten Digit Database”. (2010). URL: <http://yann.lecun.com/exdb/mnist/> (aplūkots 14.01.2016.).
- [56] Yann LeCun u. c. “Comparison of Learning Algorithms for Handwritten Digit Recognition”. 1995.
- [57] H. Lee u. c. “Convolutional Deep Belief Networks for Scalable Unsupervised Learning of Hierarchical Representations”. *ICML '09*. 2009.
- [58] J. Lee u. c. “Dynamic Noise Embedding: Noise Aware Training and Adaptation for Speech Enhancement”. 2020.
- [59] Chao Li u. c. “Deep Speaker: An End-to-End Neural Speaker Embedding System”. *CoRR* abs/1705.02304 (2017). arXiv: 1705.02304. URL: <http://arxiv.org/abs/1705.02304>.

- [60] Timothy P. Lillicrap u. c. “Continuous Control with Deep Reinforcement Learning”. *CoRR* abs/1509.02971 (2015). URL: <http://arxiv.org/abs/1509.02971>.
- [61] Tsung-Yi Lin u. c. “Focal Loss for Dense Object Detection”. *2017 IEEE International Conference on Computer Vision (ICCV)* (2017), 2999.—3007. lpp.
- [62] Teng Long u. c. “Zero-Shot Learning via Discriminative Representation Extraction”. *Pattern Recognition Letters* 109 (2018), 27.—34. lpp.
- [63] Scott Lundberg un Su-In Lee. “A Unified Approach to Interpreting Model Predictions”. *NIPS*. 2017.
- [64] R. Manmatha u. c. “Sampling Matters in Deep Embedding Learning”. *2017 IEEE International Conference on Computer Vision (ICCV)* (2017), 2859.—2867. lpp.
- [65] Michael Phi. *Illustrated Guide to LSTM’s and GRU’s: A Step by Step Explanation*. 2020. g. 6. janv. URL: <https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21>.
- [66] Erik G. Miller, Nicholas E. Matsakis un Paul A. Viola. “Learning from One Example through Shared Densities on Transforms”. *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No.PR00662)* 1 (2000), 464—471 vol.1.
- [67] Volodymyr Mnih u. c. “Asynchronous Methods for Deep Reinforcement Learning”. *ICML* 48 (2016), 1928.—1937. lpp. URL: <http://arxiv.org/abs/1602.01783>.
- [68] Volodymyr Mnih u. c. “Playing Atari with Deep Reinforcement Learning”. *CoRR* abs/1312.5602 (2013). URL: <http://arxiv.org/abs/1312.5602>.
- [69] Igor Mordatch. “Concept Learning with Energy-Based Models”. *ICLR*. 2018.
- [70] Yair Movshovitz-Attias u. c. “No Fuss Distance Metric Learning Using Proxies”. *2017 IEEE International Conference on Computer Vision (ICCV)* (2017), 360.—368. lpp.
- [71] “N-Shot Learning: Learning More with Less Data”. (). URL: <https://blog.floydhub.com/n-shot-learning/>.

- [72] D. Neil, M. Pfeiffer un Shih-Chii Liu. “Phased LSTM: Accelerating Recurrent Network Training for Long or Event-Based Sequences”. *NIPS*. 2016.
- [73] Binh X. Nguyen u. c. “Deep Metric Learning Meets Deep Clustering: An Novel Unsupervised Approach for Feature Embedding”. *ArXiv* abs/2009.04091 (2020).
- [74] Hyeonwoo Noh, Seunghoon Hong un Bohyung Han. “Learning Deconvolution Network for Semantic Segmentation”. *2015 IEEE International Conference on Computer Vision (ICCV)* (2015), 1520.—1528. lpp.
- [75] “Non-Zero Initial States for Recurrent Neural Networks - R2RT”. (). URL: <https://r2rt.com/non-zero-initial-states-for-recurrent-neural-networks.html>.
- [76] OpenAI u. c. “Solving Rubik’s Cube with a Robot Hand”. *ArXiv* abs/1910.07113 (2019).
- [77] OpenAI u. c. “Solving Rubik’s Cube with a Robot Hand”. *ArXiv* abs/1910.07113 (2019).
- [78] D. Park u. c. “Improved Noisy Student Training for Automatic Speech Recognition”. *ArXiv* abs/2005.09629 (2020).
- [79] Adam Paszke u. c. “Automatic Differentiation in PyTorch”. (2017).
- [80] K. Petersen u. c. “Systematic Mapping Studies in Software Engineering”. *EASE*. 2008.
- [81] Alexander Pritzel u. c. “Neural Episodic Control”. *ArXiv* abs/1703.01988 (2017).
- [82] Qi Qi u. c. “A Simple and Effective Framework for Pairwise Deep Metric Learning”. *Computer Vision – ECCV 2020*. Izdevis Andrea Vedaldi u. c. Cham: Springer International Publishing, 2020, 375.—391. lpp. ISBN: 978-3-030-58583-9.
- [83] Hubert Ramsauer u. c. “Hopfield Networks Is All You Need”. *ArXiv* abs/2008.02217 (2020).
- [84] Rajeev Ranjan, Carlos D. Castillo un Rama Chellappa. “L2-Constrained Softmax Loss for Discriminative Face Verification”. *CoRR* abs/1703.09507 (2017). arXiv: 1703.09507. URL: <http://arxiv.org/abs/1703.09507>.

- [85] Mirco Ravanelli, Titouan Parcollet un Yoshua Bengio. “The Pytorch-Kaldi Speech Recognition Toolkit”. *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2019), 6465.—6469. lpp.
- [86] Oren Rippel u. c. “Metric Learning with Adaptive Density Discrimination”. *ICLR abs/1511.05939* (2016).
- [87] Olaf Ronneberger, Philipp Fischer un Thomas Brox. “U-Net: Convolutional Networks for Biomedical Image Segmentation”. *MICCAI*. 2015.
- [88] Stuart J. Russell un Peter Norvig. *Artificial Intelligence: A Modern Approach (2nd Edition)*. Prentice Hall, 2002. g. dec. ISBN: 0-13-790395-2. URL: <http://www.amazon.ca/exec/obidos/redirect?tag=citeulike09-20&path=ASIN/0137903952>.
- [89] Sara Sabour, Nicholas Frosst un Geoffrey E. Hinton. “Dynamic Routing between Capsules”. *ArXiv abs/1710.09829* (2017).
- [90] Tom Schaul u. c. “Prioritized Experience Replay”. *CoRR abs/1511.05952* (2015). URL: <http://arxiv.org/abs/1511.05952>.
- [91] John Schulman u. c. “Proximal Policy Optimization Algorithms”. *ArXiv abs/1707.06347* (2017).
- [92] John Schulman u. c. “Trust Region Policy Optimization”. *ICML*. 2015.
- [93] Ramprasaath R. Selvaraju u. c. “Grad-CAM: Why Did You Say That? Visual Explanations from Deep Networks via Gradient-Based Localization”. *CoRR abs/1610.02391* (2016). URL: <http://arxiv.org/abs/1610.02391>.
- [94] Stanislau Semeniuta, Aliaksei Severyn un Erhardt Barth. “Recurrent Dropout without Memory Loss”. *COLING*. 2016.
- [95] M. Shoeybi u. c. “Megatron-Lm: Training Multi-Billion Parameter Language Models Using Model Parallelism”. *ArXiv abs/1909.08053* (2019).
- [96] Ari Silburt u. c. “Lunar Crater Identification via Deep Learning”. *Icarus* 317 (2019), 27.—38. lpp.
- [97] Karen Simonyan un Andrew Zisserman. “Very Deep Convolutional Networks for Large-Scale Image Recognition”. *CoRR abs/1409.1556* (2015).



- [98] Kihyuk Sohn. “Improved Deep Metric Learning with Multi-Class n-Pair Loss Objective”. *NIPS*. 2016.
- [99] Hyun Oh Song u. c. “Deep Metric Learning via Facility Location”. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017), 2206.—2214. lpp.
- [100] Hyun Oh Song u. c. “Deep Metric Learning via Lifted Structured Feature Embedding”. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), 4004.—4012. lpp.
- [101] Hyun Oh Song u. c. “Learnable Structured Clustering Framework for Deep Metric Learning”. *ArXiv* abs/1612.01213 (2016).
- [102] Christian Szegedy, Sergey Ioffe un Vincent Vanhoucke. “Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning”. *AAAI*. 2016.
- [103] Haoran Tang u. c. “#Exploration: A Study of Count-Based Exploration for Deep Reinforcement Learning”. *ArXiv* abs/1611.04717 (2017).
- [104] Norman Tasfi. “PyGame Learning Environment”. *GitHub repository* (2016). URL: <https://github.com/ntasfi/PyGame-Learning-Environment>.
- [105] Matias Tassano, J. Delon un T. Veit. “FastDVDnet: Towards Real-Time Deep Video Denoising without Flow Estimation”. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2020), 1351.—1360. lpp.
- [106] “Tips for Training Recurrent Neural Networks”. (). URL: <https://danijar.com/tips-for-training-recurrent-neural-networks/>.
- [107] Hugo Touvron u. c. “Fixing the Train-Test Resolution Discrepancy”. *Advances in Neural Information Processing Systems (NeurIPS)*. 2019.
- [108] “Triplet Loss and Online Triplet Mining in TensorFlow | Olivier Moindrot Blog”. (). URL: <https://omoindrot.github.io/triplet-loss>.
- [109] E. Urtans un Agris Nikitenko. “Survey of Deep Q-Network Variants in PyGame Learning Environment”. *ICDLT '18*. 2018.
- [110] E. Urtans, Agris Nikitenko un Valters Vecins. “Exponential Triplet Loss”. *Proceedings of the 2020 the 4th International Conference on Compute and Data Analysis* (2020).

- [111] E. Urtans un Valters Vecins. “Value Iteration Solver Networks”. *2020 3rd International Conference on Intelligent Autonomous Systems (ICoIAS)* (2020), 8.—13. lpp.
- [112] Evalds Urtans un Ariel Tabaks. *Asya: Mindful Verbal Communication Using Deep Learning*. 2020. arXiv: 2008.08965 [eess.AS].
- [113] E. Ustinova un V. Lempitsky. “Learning Deep Embeddings with Histogram Loss”. *NIPS*. 2016.
- [114] R. Vaillant, C. Monrocq un Y. Le Cun. “Original Approach for the Localisation of Objects in Images”. *IEE Proceedings - Vision, Image and Signal Processing* 141.4 (1994), 245.—250. lpp.
- [115] Athanasios Voulodimos u. c. “Deep Learning for Computer Vision: A Brief Review”. *Computational Intelligence and Neuroscience* 2018 (2018).
- [116] Chanchin Wang, Xue Zhang un Xipeng Lan. “How to Train Triplet Networks with 100K Identities?”: *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)* (2017), 1907.—1915. lpp.
- [117] J. Wang u. c. “Deep Metric Learning with Angular Loss”. *2017 IEEE International Conference on Computer Vision (ICCV)* (2017), 2612.—2620. lpp.
- [118] Jinjiang Wang u. c. “Deep Learning for Smart Manufacturing: Methods and Applications”. *Journal of Manufacturing Systems* 48 (2018), 144.—156. lpp.
- [119] X. Wang u. c. “Multi-Similarity Loss with General Pair Weighting for Deep Metric Learning”. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2019), 5017.—5025. lpp.
- [120] Xinrui Wang un Jinze Yu. “Learning to Cartoonize Using White-Box Cartoon Representations”. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2020), 8087.—8096. lpp.
- [121] Xinshao Wang u. c. “Ranked List Loss for Deep Metric Learning”. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2019), 5202.—5211. lpp.

- [122] Ziyu Wang, Nando de Freitas un Marc Lanctot. “Dueling Network Architectures for Deep Reinforcement Learning”. *CoRR* abs/1511.06581 (2015). URL: <http://arxiv.org/abs/1511.06581>.
- [123] Ziyu Wang, Nando de Freitas un Marc Lanctot. “Dueling Network Architectures for Deep Reinforcement Learning”. *CoRR* abs/1511.06581 (2015). URL: <http://arxiv.org/abs/1511.06581>.
- [124] Ziyu Wang u. c. “Sample Efficient Actor-Critic with Experience Replay”. *ArXiv* abs/1611.01224 (2017).
- [125] Greg Wayne u. c. “Unsupervised Predictive Memory in a Goal-Directed Agent”. *ArXiv* abs/1803.10760 (2018).
- [126] “We Analyzed 16,625 Papers to Figure out Where AI Is Headed next | MIT Technology Review”. (). URL: <https://www.technologyreview.com/2019/01/25/1436/we-analyzed-16625-papers-to-figure-out-where-ai-is-headed-next/>.
- [127] Kilian Q. Weinberger un L. Saul. “Distance Metric Learning for Large Margin Nearest Neighbor Classification”. *NIPS*. 2005.
- [128] Yandong Wen u. c. “A Discriminative Feature Learning Approach for Deep Face Recognition”. *ECCV*. 2016.
- [129] Cameron R. Wolfe un Keld T. Lundgaard. “E-Stitchup: Data Augmentation for Pre-Trained Embeddings”. 2019.
- [130] Han Xiao, Kashif Rasul un Roland Vollgraf. “Fashion-Mnist: A Novel Image Dataset for Benchmarking Machine Learning Algorithms”. *ArXiv* abs/1708.07747 (2017).
- [131] Yong Xu u. c. “Dynamic Noise Aware Training for Speech Enhancement Based on Deep Neural Networks”. *INTERSPEECH*. 2014.
- [132] Dong Yi, Zhen Lei un S. Li. “Deep Metric Learning for Practical Person Re-Identification”. *ArXiv* abs/1407.4979 (2014).
- [133] Y. Yuan, Kuiyuan Yang un Chao Zhang. “Hard-Aware Deeply Cascaded Embedding”. *2017 IEEE International Conference on Computer Vision (ICCV)* (2017), 814.—823. lpp.
- [134] Q. Zhang un Tao Du. “Self-Driving Scale Car Trained by Deep Reinforcement Learning”. *ArXiv* abs/1909.03467 (2019).

- [135] W. Zheng, S. Gong un T. Xiang. “Reidentification by Relative Distance Comparison”. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35 (2013), 653.—668. lpp.
- [136] Zongwei Zhou u. c. “UNet++: A Nested u-Net Architecture for Medical Image Segmentation”. *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support : 4th International Workshop, DLMIA 2018, and 8th International Workshop, ML-CDS 2018, held in conjunction with MICCAI 2018, Granada, Spain, S...* 11045 (2018), 3.—11. lpp.