DATU KOPAS IZVĒLE NEIRONTĪKLU APMĀCĪŠANAI

OPTIMAL DATASET SELECTION FOR TRANSFER LEARNING

Autori: **Sandis DEKSNIS, Rolands PITERĀNS**
Zinātniskā darba vadītājs: **docents, Dr.sc.ing. Sergejs KODORS**
Rēzeknes Tehnoloģiju Akadēmija, Atbrīvošanas aleja 115, Rēzekne, Latvija

Abstract*. The proposed article describes transfer learning and Earth Mover's Distance (EMD) methodology application in machine learning. The goal was to find out the shortest distance among three datasets in order to identifyt dataset, which is more suited for neural network pretraining.*

The experiment was completed using Python programming language and Jupyter Notebook. Neural network pretrained on ImageNet dataset was applied as feature extractor. The extracted feature vectors of datasets were applied to calculate the minimal distance using EMD algorithm.

Keywords*: datasets, Earth Mover's Distance (EMD), ImageNet, neural networks, transfer learning*.

Ievads

Neirontīkli ir matemātiskais modelis, kas ļauj datoriem patstāvīgi atrast aproksimācijas funkciju, lai īstenotu datu klasifikāciju. Šo tīklu apmācīšana sākas ar ievadīto datu analīzi. Lai paātrinātu neirontīklu apmācīšanu tika izgudrota metode *"transfer learning"*, kura balstās uz apmācīta neirontīkla atkārtotas izmantošanas. *"Transfer learning"* metode ļauj atkārtoti izmantot gan apmācīta neirontīkla daļu, gan visu modeli jaunas problēmas risināšanai.

2018. gadā pētnieku grupa *Cui* u.c. mērija atšķirību starp datu kopām. Rezultātā autori izgudroja attāluma mērīšanas metodoloģiju, pielietojot *Earth Mover's Distance* (*EMD*), lai izvēlētu atbilstošu datu kopu *"transfer learning"* metodei. [2]

Pētījuma mērķis: izmantojot *EMD* metodoloģiju izmērīt distanci starp šādām datu kopām: *PlantPath2020*, *Animals* un *Flowers*.

Hipotēze: *EMD* attālums starp izvēlētām datu kopām būs dažāds.

Pētniecības metode: aprakstošā metode, eksperimentālā metode, *EMD* metodoloģija.

Materiāli un metodes

Aprakstošajā metodē mēs detalizēti izpētīsim un, apkopojot informāciju un pamatojoties uz daudzveidīgās literatūras aprakstiem, raksturosim tuvāku datukopu meklēšanu balstoties uz *"Earth Mover's Distance"* metodes.

Eksperimentālajā metodē mēs, izmantojot *Python* programmēšanas valodu, caur *Jupyter Notebook* (tīmekļa lietojumprogrammu) apstrādāsim datu kopas, pielietojot *"ImageNet"* priekšapmācītu neirontīklu raksturīgu pazīmju meklēšanai, un veiksim īsākā ceļa meklēšanu, tādējādi analizējot un veicot secinājumus par iegūtiem rezultātiem.

Īstenojot eksperimentu, tika pielietotās šādas tehnoloģijas un risinājumi:

- *Python* programmēšanas valoda;
- *Jupyter Notebook;*
- *Tensorflow;*
- *ImageNet* priekšapmācītājs neirontīkls ar *MobileNetV2* arhitektūru;
- Datu kopas: *PlantPath2020* [5], *Animals* [6] un *Flowers* [7];
- Simpleks metode;
- Dators ar *Intel Core i5-8250U* procesoru.

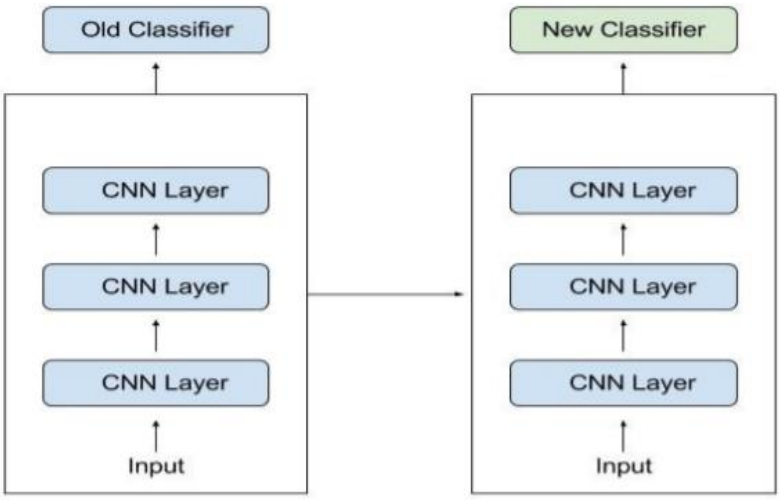
Transfer learning

Lielas datukopas apstrāde un anotēšana ir laikietilpīgs un relatīvi dārga, tāpēc dziļu mācību modeļu izmantošana nav tik vienkārša. Lai pārvarētu šo problēmu, pētnieki ir izpētījuši,

https://doi.org/10.17770/het2021.25.6777

ka zināšanas par iepriekšējiem objektiem palīdz atpazīt jaunus objektus, izmantojot to līdzību un saikni ar jaunajiem objektiem. Pamatojoties uz šo ideju, daži pētījumi liecina, ka klasifikācijai var izmantot dziļas mācīšanās modeļus (*deep learning models*), kas apmācīti citiem klasifikācijas uzdevumiem [3]. Tādējādi neirontīkla modeļus, kas apmācīti konkrētai datu kopai vai uzdevumam, var pielāgot jaunam uzdevumam pat no cita domēna. Šī pieeja ir pazīstama kā *transfer learning*.

Datoru redzējumā neironu tīkli parasti mēģina noteikt malas to agrākajos slāņos, formas vidējā slānī un dažas uzdevumam raksturīgas iezīmes vēlākos slāņos. Ar pārejas mācīšanos (*transfer learning*) tiek izmantoti agrākie un vidējie slāņi, un tikai pēdējie tiek apmācīti no jauna (skat. 1.att.). Tas palīdz izmantot marķētos datus par uzdevumu, uz kuru tā sākotnēji tika apmācīta. Piemēram, modelim, kas apmācīts atpazīt mugursomu uz attēla, tiek pārapmācīts atpazīt saulesbrilles. Iepriekšējos slāņos modelis ir iemācījies atpazīt objektu pamatīpašības, ka līnijas, krāsas, tekstūras, utt., bet pēdējos slāņos, tas kombinē šīs īpašības sarežģītākās raksturīpašības, piemēram, ģeometriskas figūras.



1. att. *Transfer learning* konceptuāls modelis [3]

Lai paātrinātu mašīnapmācīšanu (*machine learning*) un neirontīklu klasifikācijas precizitāti, pētnieki pēc iespējas izmanto gatavus modeļus no iepriekšējiem uzdevumiem, izvelējoties veiksmīgākus risinājumus jaunu uzdevumu izpildīšanai.

Raksturīgu iezīmju meklēšana

Klasifikācijas "funkcijas" atrašana ir galvenais mākslīgā intelekta uzdevums. Tas iegūst visatbilstošākās attēla iezīmes un piešķir tiem semantiskās nozīmes. Attēlu klasifikācijā izšķirošs solis ir analizēt attēla īpašības un sakārtot skaitliskās pazīmes klasēs. Klasifikācijas funkcijas iegūšana balstās uz šādu īpašību analīzes:

- Krāsu iezīmes (*color features*). Attēlu klasifikācijā un attēlu iegūšanā krāsa ir vissvarīgākā iezīme. Krāsu histogramma ir visizplatītākā krāsu iezīmju iegūšanas metode. To uzskata par krāsas sadalījumu attēlā. Krāsu funkcijas efektivitāte ir saistīta ar faktu, ka tie ir neatkarīgi un nejutīgi pret attēla izmēru, pagriešanu un tālummaiņu;
- Tekstūras īpatnības (*texture features*). Tekstūras iezīmju iegūšana ir ļoti izturīga metode lielam attēlam, kurā ir atkārtots reģions. Tekstūra ir pikseļu grupa, kurai ir noteikts raksturojums. Tekstūras pazīmju metodes iedala divās kategorijās: telpiskās faktūras pazīmju iegūšana un spektrālās faktūras pazīmju izvilkšana;

- Formas īpašības (*shape features*). Formas pazīmes tiek ļoti izmantotas objektu atpazīšanā un formas aprakstā. Formas pazīmju iegūšanas paņēmieni tiek klasificēti kā: reģiona un kontūras bāzes. Kontūrmetodes aprēķina pazīmi no robežas un neņem vērā tās iekšpusi, savukārt reģiona metodes aprēķina pazīmi no visa reģiona.

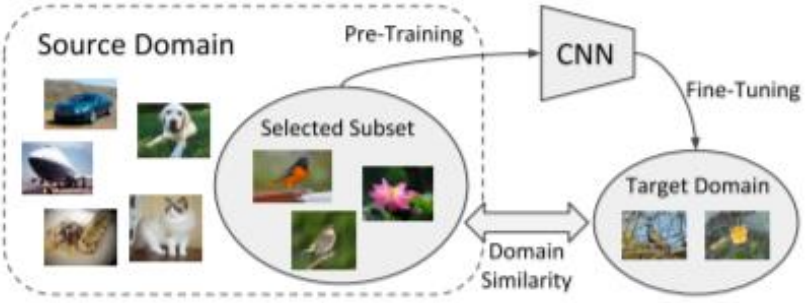
Attēlu segmentēšana starp attēlu atpazīšanas metodēm

Attēlu segmentēšana ir objekta telpiska reģiona konstatēšana analīzes attēlā, lai iegūtais gabals būtu precīzs objekta attēlojums, kas ir daudz mērķtiecīgāks un vieglāk analizējams. Eksistē divi objekta segmentēšanas veidi, pamatojoties uz kameras mobilitātes[3]:

- segmentācija ar statisko kameru;
- segmentācija ar kustīgo kameru.

Statiskās kameras gadījumā, tā atrodas noteiktā vietā ar fiksētu skata leņķi, kad objekts un fons ir stingri nemainīgi. Kustīgas kameras gadījumā, attēls visu laiku mainās, - šāda segmentācija ir sarežģītāka, jo redzamā scēna visu laiku ir dažāda.

***"Transfer learning"* apmācības shēma**



2. att. Pārneses apmācības shēmas pārskats [4]

Ņemot vērā interesējošo izvēlēto domēnu, no sākuma neirontīkls tiek apmācīts atlasītajā apakškopā no avota domēna, pamatojoties uz piedāvāto līdzības mērījumu, un pēc tam precīzi noregulējam mērķa domēnu.

"Transfer learning" apmācību ir atpazīšanas īpašību pārvietošana no avota domēna (*source domain*) uz mērķa domēnu (*target domain*) (skat. 2.att). Darbu, kādu nepieciešams pielietot, lai pārmācītu neirontīklu klasificēt attēlus no cita domēna, sauc par datukopas attālumu. Domēnu attālumu var aprēķināt, pielietojot *"Earth Mover's Distance"* (*EMD*) [4].

Earth Mover's Distance

Attāluma mērīšana starp entītijām ir pamatfunkcija, kas tiek pielietota daudzos analīzes uzdevumos. Piem., ceļa meklēšana, datu ieguve, datu meklēšana, kopu veidošana un klasifikācija ir atkarīga no attāluma mērījumiem starp objektiem. *"Earth Mover's Distance"*, kas pazīstams arī kā diskrētais Vaseršteina attālums, ir izstrādāts, lai izmērītu attālumu starp diviem varbūtības sadalījumiem. Vaseršteina attālumam, papildus lietojumiem attēlu iegūšanā, ir svarīgi pielietojumi mašīnmācīšanās (*machine learning*) jomā. [1]

Katra kopa ir vidēji liela (200 – 500Mb). Kopu analīzei tika izmantota *EMD* metodoloģija, kur ar ģeneratīvo modeli izveido paraugu kvalitāti, mērot attālumu starp sintētiskajiem paraugiem un sākotnējiem datiem. *EMD* vēl nav atradusi plašu piemērošanu. Tas ir saistīts ar tā lielo skaitļošanas sarežģītību – kubiskā sarežģītība no datu kopas lieluma, kas neatļauj to pielietot ļoti lielām, miljonu vai vairāk objektu, kolekcijām.

Rezultāti un to analīze

Eksperimenta veikšanai tika sagatavotas trīs datu kopas ar fotoattēliem ‘JPEG’ formātā: *PlantPath2020* [5], *Flowers* [7], *Animals* [6]; un kods, kas tika izpildīts *Jupyter Notebook* vidē.

Vispirms tika lejupielādēts jau gatavs *ImageNet* priekšapmācīts neirontīkls, kas tika izmantots katras klases pazīmju iegūšanai (*feature extraction*), (skat. 3.att.).

```
model = tf.keras.applications.MobileNetV2(
    input_shape=(224, 224, 3),
    alpha=1.0,
    include_top=True,
    weights="imagenet",
    input_tensor=None,
    pooling=None,
    classes=1000
)

model = tf.keras.models.Model( inputs=model.inputs, outputs=model.layers[-2].output )
model.compile( optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'] )
model.summary()
```

```
block_16_depthwise_relu (ReLU)  (None, 7, 7, 960)    0           block_16_depthwise_BN[0][0]
__________________________________________________________________________________________________
block_16_project (Conv2D)       (None, 7, 7, 320)    307200      block_16_depthwise_relu[0][0]
__________________________________________________________________________________________________
block_16_project_BN (BatchNorma (None, 7, 7, 320)    1280        block_16_project[0][0]
__________________________________________________________________________________________________
Conv_1 (Conv2D)                 (None, 7, 7, 1280)   409600      block_16_project_BN[0][0]
__________________________________________________________________________________________________
Conv_1_bn (BatchNormalization)  (None, 7, 7, 1280)   5120        Conv_1[0][0]
__________________________________________________________________________________________________
out_relu (ReLU)                 (None, 7, 7, 1280)   0           Conv_1_bn[0][0]
__________________________________________________________________________________________________
global_average_pooling2d (Globa (None, 1280)         0           out_relu[0][0]
==================================================================================================
Total params: 2,257,984
Trainable params: 2,223,872
Non-trainable params: 34,112
```

3.att. Pazīmju iegūšanas neirontīkla sagatavošana

Katrai datu kopai ar “ImageNet” neirontīkla palīdzību tika izveidota pazīmju matrica jeb vārdnīca. Tas nozīmē, ka neirontīkls apstrādā katru kopas elementu un izvada vektoru ar vidējām skaitliskām vērtībām, kur katra vērtība ir attiecināma uz noteiktu pazīmi fotoattēlā (skat. 1. - 3. tabulu).

1. tabula

Datu kopas “PlantPath2020” pazīmju skaitlisko vērtību piemērs (15 pirmās)

Klases	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
healthy	0,04	0,25	0,12	0,14	0,32	0,23	0,17	0,09	0,04	0,02	0,07	0,58	0,8	0,51	0,3	0,25
rust	0,03	0,13	0,11	0,06	0,45	0,23	0,18	0,12	0,05	0,01	0,08	0,54	0,63	0,64	0,36	0,17
scab	0,02	0,19	0,27	0,08	0,36	0,1	0,17	0,08	0,02	0,01	0,07	0,37	0,68	0,38	0,2	0,13

2. tabula

Datu kopas “Flowers” pazīmju skaitlisko vērtību piemērs (15 pirmās)

Klases	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
daisy	0,03	0,71	0,01	0,04	0,06	0,18	0,03	0,08	0,15	0,02	0,06	0,06	0,09	0,12	0,3	0,12
dandelion	0,07	0,4	0,01	0,25	0,04	0,3	0,02	0,05	0,25	0,06	0,12	0,19	0,03	0,17	0,08	0,14
rose	0,09	0,61	0,04	0,07	0,3	0,29	0,3	0,1	0,16	0,11	0,09	0,15	0,11	0,3	0,16	0,11
sunflower	0,09	0,63	0,01	0,21	0,09	0,24	0,13	0,19	0,1	0,03	0,15	0,09	0,13	0,26	0,12	0,13
tulip	0,07	0,6	0,01	0,06	0,14	0,28	0,18	0,12	0,18	0,08	0,04	0,07	0,08	0,15	0,25	0,15

3. tabula

Datu kopas “Animals” pazīmju skaitlisko vērtību piemērs (15 pirmās)

Klases	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
butterfly	0,09	0,38	0,02	0,04	0,07	0,14	0,05	0,07	0,04	0,02	0,23	0,05	0,07	0,19	0,16	0,04
dogs	0,04	0,18	0,08	0,02	0,1	0,07	0,09	0,09	0,07	0,01	0,06	0,06	0,07	0,1	0,07	0,06
elephants	0,08	0,11	0,05	0,04	0,09	0,24	0,08	0,03	0,03	0,02	0,11	0,05	0,47	0,07	0,15	0,06
horses	0,22	0,12	0,08	0,01	0,1	0,05	0,08	0,03	0,08	0,03	0,04	0,02	0,16	0,12	0,23	0,03

Tālāk katrai datu kopai tika aprēķināta klašu varbūtība (skat. 4. – 6. tabulas).

4. tabula

Datu kopas “Animals” klašu varbūtības

	Varbūtība
Butterfly	0,22
Dogs	0,38
Elephants	0,15
Horses	0,25

5. tabula

Datu kopas “Flowers” klašu varbūtības

	Varbūtības
Daisy	0,18
Dandelion	0,24
Rose	0,18
Sunflower	0.17
Tulip	0.23

6. tabula

Datu kopas “PlantPath2020” klašu varbūtības

	Varbūtības
Healthy	0,30
Rust	0,36
Scab	0,34

Tālāk notiek datu kopu salīdzināšana savā starpā jeb distances vektoru meklēšana starp divu datu kopu klasēm, pielietojot pazīmju vārdnīcas vektorus. Vispirms tiek pieslēgtas divas datu kopas vārdnīcas (skat. 10. att.), tad pielietojot Eiklīda distances formulu tiek aprēķināti attālumi starp divām klasēm (skat. 7.- 9. tabulu).

```
source = pd.read_csv('C:/JProjects/Results/1_PlantPath2020_Dataset.csv', sep=';', header=0, index_col=0)
target = pd.read_csv('C:/JProjects/Results/1_flowers_Dataset.csv', sep=';', header=0, index_col=0)
```

10. att. Datu kopu pazīmju vārdnīcu pieslēgšana

7. tabula

Distanču vektori starp datu kopu klasēm (*Flowers – Animals*)

	Butterfly	*Dogs*	*Elephants*	*Horses*
Daisy	5,23	6,09	7,24	6,16
Dandelion	4,28	4,70	6,42	4,87
Rose	5,65	6,80	7,12	6,43
Sunflower	6,01	7,04	7,59	6,82
Tulip	4,36	4,94	6,53	4,92

8. tabula

Distanču vektori starp datu kopu elementiem (*PlantPath2020 – Animals*)

	Butterfly	*Dogs*	*Elephants*	*Horses*
Healthy	11,45	13,09	12,05	12,47
Rust	9,45	11,00	10,37	10,50
Scab	9,43	10,91	10,35	10,39

9. tabula

Distanču vektori starp datu kopu elementiem (*PlantPath2020 – Flowers*)

	Daisy	*Dandelion*	*Rose*	*sunflower*	*tulip*
Healthy	11,33	11,31	10,09	10,92	11,18
Rust	9,73	9,47	8,64	9,39	9,42
Scab	9,68	9,44	8,59	9,48	9,41

Tad, izmantojot *"Simplex"* metodi, tiek aprēķināts minimalais attālums starp pašām datu kopām saskaņā ar *EMD* metodoloģiju, kur mazāks skaitlis nozīmē mazāku attālumu jeb līdzīgākas pazīmes starp divu datu kopu klasēm. Rezultāti tika apkopoti tabulā (skat. 10. tabulu.).

10. tabula

Distances starp datu kopām

		Mērķa datu kopa		
		Plants	*Flowers*	*Animals*
Avota datu kopa	*Plants*	0	9,77	10,95
	Flowers	9,77	0	5,59
	Animals	10,95	5,59	0

Secinājumi

Izmantojot *EMD* algoritmu tika aprēķināts attālums starp trīs datu kopām, kur mazākā iegūta vērtība pēc datu kopas apstrādāšanas, norādīja uz mazāko attālumu starp datu kopām. Mazākais attālums starp izvēlēto datu kopu un citu datu kopu nozīmē, ka algoritms ir atradis līdzīgas iezīmes datu kopu elementu starpā.

Mūsu experinments parādija, ka mazākais attālums ir starp datukopām *"Animals"* un *"Flowers"* un tā vērtība ir ~5,6.

Literatūra

1. Chi Zhang, Yujun Cai, Guosheng Lin, Chunhua Shen "DeepEMD: Differentiable Earth Mover's Distance for Few-Shot Learning" Sk. internetā. (2020.) - https://arxiv.org/pdf/2003.06777.pdf
2. Haris Pozidis, Kubilay Atasu "Linear-Complexity Earth Mover's Distance Approximations for Efficient Similarity Search" Sk. internet (07.16.2019.) - https://www.ibm.com/blogs/research/2019/07/earth-movers-distance/
3. Rohan Saha, Debaruna Saha "Transfer Learning – A Comparative Analysis" Sk. internetā. (2018.) - https://www.researchgate.net/publication/329786975_Transfer_Learning_-_A_Comparative_Analysis
4. Yin Cui, Yang Song, Chen Sun, Andrew Howard, Serge Belongie "Large Scale Fine-Grained Categorization and Domain-Specific Transfer Learning" (arXiv:1806.06193v1) Sk. internetā. (2018.) - https://openaccess.thecvf.com/content_cvpr_2018/papers/Cui_Large_Scale_Fine-Grained_CVPR_2018_paper.pdf
5. Dataset "Plants_Dataset[99 classes]" by Muhammad jawad - https://www.kaggle.com/muhammadjawad1998/plants-dataset99-classes
6. Dataset "Animals-10" by Corrado Alessio - https://www.kaggle.com/alessiocorrado99/animals10
7. Dataset "Flowers Recognition" by Alexander Mamaev - https://www.kaggle.com/alxmamaev/flowers-recognition.