Google Street View képek országokra vonatkozó klasszifikálása gépi tanulás segítségével

Feladat

A feladatot a GeoGuessr nevű játék ihlette, amelyben Google utcaképek alapján kell kitalálni, hol készült a kép. A tipp játékmódtól függően lehet ország (klasszifikáció), vagy egy pont a térképen (regresszió, amit általában klasszifikációra vezetnek vissza AI modelleknél). Az egyszerűség kedvéért országok klasszifikációját tűztem ki célul.

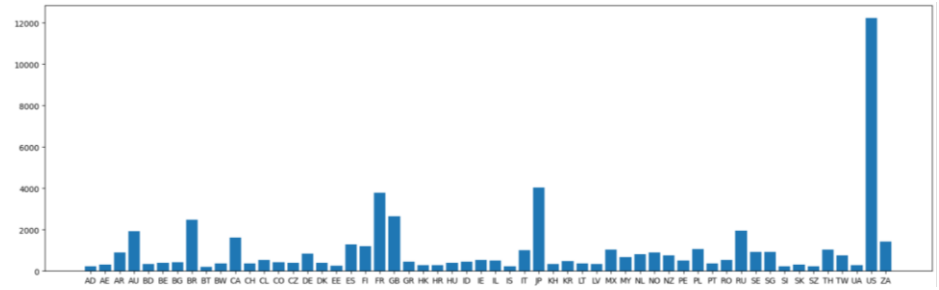
Adatbázis

A kezdeti adatbázis forrása:
https://huggingface.co/datasets/stochastic/random_streetview_images_pano_v0.0.2

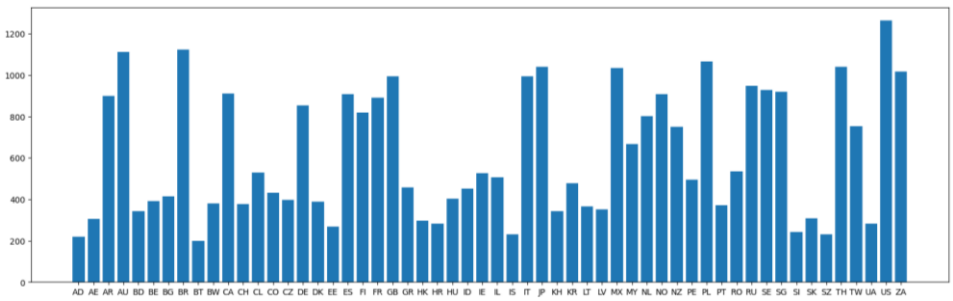
Az adatbázis kb. 11 000 képet tartalmaz 56 országból, amely országonként átlagosan nagyjából 200 mintát jelent. Szerencsére az adatok eloszlása közel egyenletes, viszont az első próbálkozásaim alapján arra a következtetésre jutottam, hogy a feladat túl nehéz standard technikákat (pl. transfer learning) alkalmazva, nagyjából 20% körüli pontosságot tudtam vele elérni. Ezért kibővítettem a minták halmazát egy másik adatbázissal: https://www.kaggle.com/datasets/ubitquitin/geolocation-geoguessr-images-50k

Ez már kb. 50 000 képet tartalmaz több mint 100 országból. Az országok számának növelése nehezíti a feladatot, ezért új ország nem került be az egyesített mintahalmazba.

Sajnos a második adatbázisban a minták eloszlása közel sem egyenletes, pl. az USA-ban készült képek száma kb. 12 000. Az egyesített adatbázisban a minta országok szerinti eloszlása:



Egy mély neuronháló könnyen rá tud tanulni a nagy gyakoriságú minták becslésére, ezért kézzel ritkítottam a sok mintával rendelkező országok képeit. A ritkítás nagyrészt véletlenszerűen történt, azonban próbáltam a kis információtartalmú (pl. sötétebb) képeket az eltávolításnál előnyben részesíteni. Mivel voltak olyan országok, amelyek az első adatbázishoz képest nem kaptak több mintát, ezért kénytelen voltam a kezdeti, szinte egyenletes és az új, torz eloszlás között kompromisszumot találni:



Az új eloszlás megválasztásánál figyelembe vettem a transfer learninggel elért eredményeket és az eredményekhez tartozó konfúziós mátrixokat is. Így kb. 34.000 mintakép maradt.

Háló betanítása

Az adatok feldolgozása

Az adatok előkészítése, a háló tanítása és a vizualizáció Pythonban, PyTorch és Matplotlib segítségével történik.

A képeket a `torch.utils.data.DataLoader` osztály példánya adagolja. A példányosításhoz szükség van két különböző, `torch.utils.data.Dataset` osztályból leszármazó adatbázisra, mivel az első adathalmaz .parquet, a második pedig sima .jpg képek formájában érhető el. A `ParquetImageDataset` és a `JpgImageDataset` osztályokat ebből a célból definiáltam.

A képeken különböző transzformációkat alkalmazunk: a random vízszintes tükrözés és [-1...1) intervallumba történő normálás a tapasztalataim szerint javított az eredményeken. A két adatbázisban a képek mérete különbözik, és szeretnénk az uniójukból véletlenszerű batcheket generálni, ezért azonos méretre kell őket szabni. Az első adatbázisban a képek arányaiban szélesebbek (illetve a függőleges tengelyen ugyanannyi információt tartalmaznak, vagyis ugyanakkor a látószög), ezért azokból vágunk ki a második adatbázis képaránya alapján darabokat.

A képek mérete így 662x1536 pixel. Tapasztalataim azt mutatták, hogy minél kevésbé csökkentem a felbontást, annál többet tud tanulni a modell. Másképp: a kép kis részlete (pl. távoli oszlop) is tartalmazhat releváns információt. Azonban mélyebb hálók tanítása nulláról ezzel a képaránnyal rendkívül időigényes, ezért több megközelítést is meg kell fontolni.

Különböző megközelítések implementációja

A feladat megoldásához többféle megközelítés létezik, azonban a legegyszerűbb az előredefiniált konvolúciós neuronhálók betanítása, finomhangolása. Próbálkoztam többfajta architektúrával, azonban elsősorban a ResNet50 hálón kísérleteztem.

ResNet50

Az architektúra méretét tekintve kompromisszumos, hiszen egy közép- /felsőkategóriás GPU-val akár nulláról is be lehet tanítani nagyobb felbontású képeken, így sok megközelítést ki lehet próbálni.

	FC	Weights	α	epochs	Im size	Freeze	Test acc
1	(1000, num_classes)	ImNetV2	1e-3 ⟶ 1e-6	3	512	front ⟶ none	*0.49*
2	(1000, 300) ⟶ (300, num_classes)	ImNetV2	1e-4 ⟶ 1e-6	5	512	front	*0.56*
3	(1024, 512) ⟶ (512, num_classes)	ImNetV2	1e-4 ⟶ 1e-5	4	512	front	*0.50*
4	(1024, 512) ⟶ (512, num_classes)	ImNetV2	1e-4 ⟶ 1e-5	3	512	front	*0.57*
5	(1024, 512) ⟶ (512, 256) ⟶ (256, num_classes)	ImNetV2	1e-4 ⟶ 1e-5	2.5	512	front	*0.54*
6	(1000, 300) ⟶ (300, num_classes)	random	1e-4 ⟶ 1e-6	3	256	none	*0.50*
7	(1000, 300) ⟶ (300, num_classes)	random	1e-4 ⟶ 1e-6	2	392	none	*0.51*

Jelölések: **FC** – fully connected layers; **Im size** – a képek rövidebb oldalának hossza; **Freeze** - a hálónak mely súlyait fagyasztottuk be, front – layer4 és fc kivételével az összes réteg súlya, none – a háló egyik paramétere sem

A táblázatból leolvasható, hogy az ImageNet adatbázison előretanított súlyok (amik az alacsonyabb szintű mintázatfelismerésért felelősek) megfelelőek a feladathoz, mivel hasonló pontosságot lehet velük elérni, emellett a betanítás sokkal gyorsabb (512-es mérettel nekem legalább egy napig tartott volna a betanítás random súlyokkal, ezért nincs feltüntetve). Emellett látszik, hogy a fully connected rétegek mérete sem befolyásolja jelentős mértékben a háló teljesítőképességét (persze normális kereteken belül maradva).

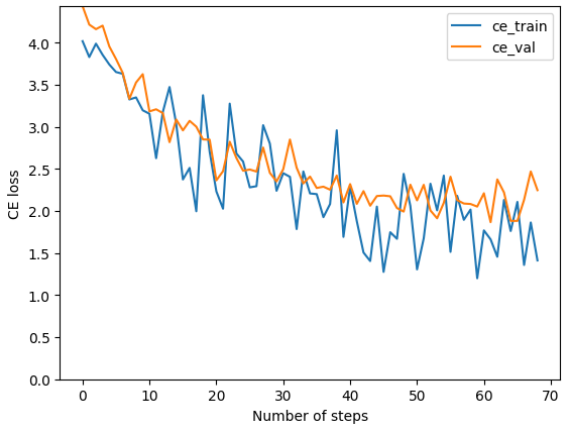
ResNet101, ResNet152, stb.

Kipróbáltam nagyobb méretű hálók betanítását is (természetesen transfer learninget alkalmazva), azonban nem tudtam elérni jelentősen jobb eredményeket, 55-60% körüli pontosság fölött egyik sem tudott teljesíteni.

A betanítás és a betanított hálók döntéseinek vizualizációja

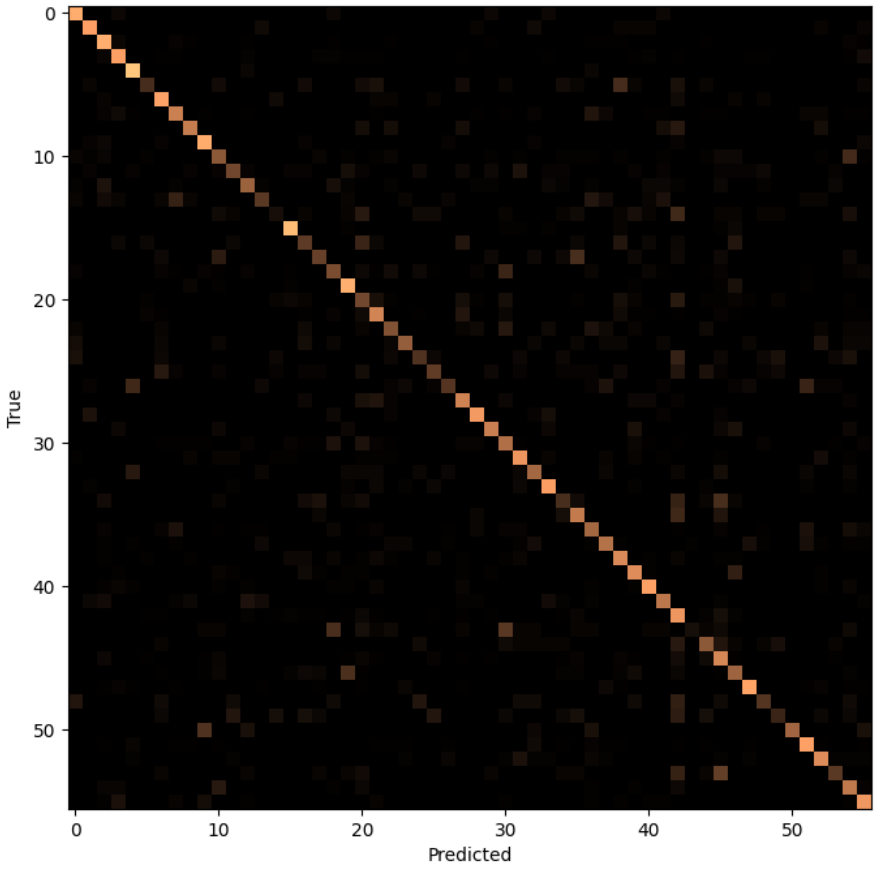
Tanítási grafikon

Egy példa:



Konfúziós mátrix

Egy konfúziós mátrix segíthet vizualizálni például azt, hogy melyik országot melyikkel kever gyakran a háló, vagy hogy információ hiányában mit tippel gyakran. Egy példa konfúziós mátrixra 55-60%-os pontosság esetén:



Class activation map vizualizáció

Az országok felismerésénél különösen érdekes lehet, mi alapján dönt egy mélyháló. Ha értelmezhető eredményeket kapunk, akkor érdekes/hasznos információkat kaphatunk arról, milyen különlegesség figyelhető meg egy-egy ország utcaképén. Ez akár a GeoGuessr játékosoknak is tippeket adhat, de általánosságban is izgalmas, mi alapján érezheti az ember (akár magyarázat nélkül is), hogy pl. nem Szlovákiában, hanem Magyarországon van.

A Grad-CAM technikát alkalmazom, amely az feed forward után a PyTorch `register_full_backward_hook()` függvénye segítségével (amit az utolsó rétegre hívunk meg) lejegyzi az utolsó réteg kimenetéhez (ami a legmagasabb szintű mintázatokat tartalmazza) tartozó gradiens értékeket, amiket csatornánként átlagolunk. Vagyis megkapjuk, hogy melyik mintázat (amit egy csatornával azonosítunk) jelenléte/hiánya segítené a hálót tovább tippjének biztosításában. Az átlagolás a transzláció invariancia miatt indokolt.

A `register_forward_hook()` metódus segítségével kinyerjük az utolsó réteg aktivációit, majd ezeket súlyozzuk csatornánként az előbb kapott (átlag)gradiens értékekkel. Így minden csatornához kaptunk egy vele megegyező méretű mátrixot, amely pixelértékei akkor lesznek magasabbak, ha a filter által felismert mintázat minél inkább jelen van a kép adott részén, és ennek az erősödése minél inkább javítaná a tipp magabiztosságát (magas a gradiens). Végül ezeket a mátrixokat átlagoljuk, normáljuk és a méretarányosan az eredeti képre illesztjük valamilyen áttetszési értékkel.

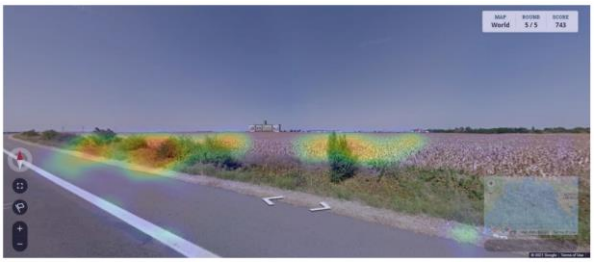
A hálók nagyrészt hasonló képrészleteket alapján döntenek, így nem kötöm modellhez a további példaképek activation mapjeit. A lenti példák közül mindegyik képet helyesen ismert fel a háló, továbbá azok kerültek nagyobb eséllyel kiválasztásra, amelyek valamilyen értelmezhető részletet emelnek ki.

Magyarország

A hazai mintaképek között dominálnak azok, amelyek vidéken készültek. Látható, hogy a háló a tetőszerkezeteket ítélte itthoni különlegességnek.

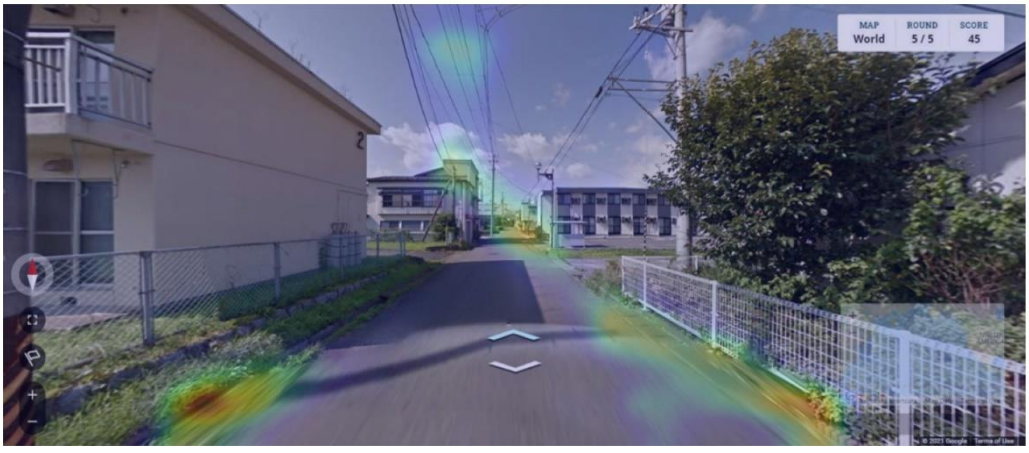
Románia

Habár Magyarországot, Románia, Szlovákiát, Bulgáriát (és esetleg Oroszországot) könnyen keveri a modell, érdekes látni, mi alapján képes néha mégis magabiztos tippet adni.



Japán

Vannak olyan országok, amelyekben Google Street View-hoz köthető sajátosságokat ismert fel a modell, például azt, hogy a Japánban készült képek alja nagyobb területen ki van homályosítva. Az ilyen felismerések a játékosoknak hasznos tipp lehet, azonban általános országfelismerő természetesen nem épülhet ilyen döntésekre.



Tajvan

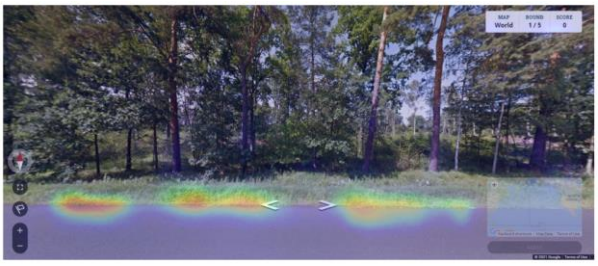
Andorra

Andorrában egy városi utcaképen jól látszik, hogy a háló a hegyeket találja különlegesnek.

Banglades

Lengyelország

Lengyelországban rendkívül sok értelmezhető, az országra jellemző attribútumot lehet felismerni a példákon (villanyoszlop lyukai, kis fehér oszlopok, stb).




UK


USA



Izland

Brazília

Források

- https://towardsdatascience.com/grad-cam-in-pytorch-use-of-forward-and-backward-hooks-7eba5e38d569
- https://medium.com/the-owl/using-forward-hooks-to-extract-intermediate-layer-outputs-from-a-pre-trained-model-in-pytorch-1ec17af78712
- https://medium.com/@tef1/geoguessr-guesser-98e01efb5235
- https://www.finndayton.com/CS229_Final_Report.pdf
- https://www.youtube.com/watch?v=ts5lPDV--cU
- https://www.youtube.com/watch?v=h1GU3EeMw_I&pp=ygUbZ2VvZ3Vlc3NyIGFpIG5ldXJhbCBuZXR3b3Jr
- https://www.youtube.com/@TraversedTV/videos