Deep Learning – Projekt work

Image classification "in-the-wild"

Aloe Vera – Chinese Money Plant – Elephant Ear

Team: Manguszta

Czuth Csaba	B256KX
Jankura Renáta	ZA3A63

Bevezető

A kiválasztott képfeldolgozásos, előrejelzős feladat lényege, hogy növényekről szeretnénk megmondani, hogy a lehetséges fajok közül melyikbe tartozik (melyikhez áll közelebb).

A megvalósítás folyamán nagy segítségünkre volt a hasonló témájú, kép-klasszifikációs órai feladat: https://github.com/BME-SmartLab-VITMMA19/vision-assignment

Illetve a kapcsolódó Python, Pytorch objektumokhoz tartozó dokumentációk, valamint a Docker integrálásánál a következő requirement.txt magyarázó oldal: https://www.freecodecamp.org/news/python-requirementstxt-explained/

A képeket innen szereztük be: https://www.kaggle.com/datasets/kacpergregorowicz/house-plant-species?resource=download

Előfeldolgozás

A random választásunk az Aloe Vera, Chienese Money Plant, és az Elephant Ear hármasra esett, ezeket kellően szemmel is megkülönböztetőnek találtuk, mégis valahol hasonlóság is felfedezhető a típusok és képek között.

A folyamatot a képek átnevezésével, egy mappába helyezésével kezdtük. Szerettük volna, ha a fájlnév tartalmazza ugyanis a megoldást, amit általában nem így szokás, ám az egyszerűbb tesztelhetőség végett így tűnt praktikusnak.

A jpg, jpeg, png, JPG és webp kiterjesztéseket tartottuk meg, szerepeltek ugyanis másfélék is, amik vagy eltört, hibás megnevezések, vagy szimplán egy ritkább előfordulású képkiterjesztést. Ezeket ignoráltuk.

Továbbá meggeneráltuk a labels.csv-t, amely tartalmazza a „fájlnév” és „az osztály, amibe tartoznak” párost soronként.

Analízis

Kirajzoltuk a képarányok előfordulási gyakoriságát, ezalapján választva ki azt, hogy milyen méretre lenne ideális a többit is alakítani. 0.75-ből és 1.00-ból volt a legtöbb, így ezeket mindenképp bele akartuk venni, és adtunk még egy kis torzulásra is lehetőséget, hogy minél több képet meg tudjuk tartani.

Azokat, amelyeknél már jelentősebb vágást kellett volna alkalmazni, vagy nagy mértékű torzulás jönne létre, eldobtuk és nem használtuk a további feldolgozás során.

Csináltunk még egy kimutatást arról is, melyik fajról hány képünk maradt így, és az eredmény 250-350 darab/típus, így úgy ítéltük, hogy ez nem olyan jelentős eltérés, és nem egyenlítettük a darabszámokat egymáshoz, meghagytuk ezeket a különböző számokat, hogy megőrizzünk megint csak minél több képet.

Tanítás

A tanítás bemeneteként szolgáló képekhez adat augmentációs technikaként RandomResizedCrop és RandomHorizontalFlip transzformációs lépéseket végeztünk, ami segíthetett leküzdeni a kis datasetből származó kihívásokat.

Baseline modell konfiguráció

A tanítás során a képek klasszifikációjához konvolúciós neurális hálózatot használtunk. Ennek alapötletét a kapott órai feladat szolgáltatta. Baseline-nak is egy hasonló konfigurációt használtunk, kibővítve BatchNorm rétegekkel, amelyekkel a gyorsabb és megbízhatóbb konvergenciát igyekeztük elősegíteni. Egy Dropout lépést is hozzáadtunk, hogy csökkentsük a túltanulást.

Inkrementális modell fejlesztés

Az inkrementális modell fejlesztés során néhány más modellt próbáltunk ki a baseline modellhez képest, melyek kialakításakor LLM segítségével merítettük az ötleteket. Egy második modellt hoztunk létre LeakyRelu aktivációval. Majd egy harmadikat is, amelybe pedig több réteg került.

A modellek létrehozását követően egy ciklusban tanítottuk és teszteltük őket. Minden modellhez készült egy checkpoint is, amelybe elmentésre kerül az a modell, aminek a legjobb az „avg_per_class accuracy”-ja.

Kiértékelés

Metrikák, mire optimalizálunk:

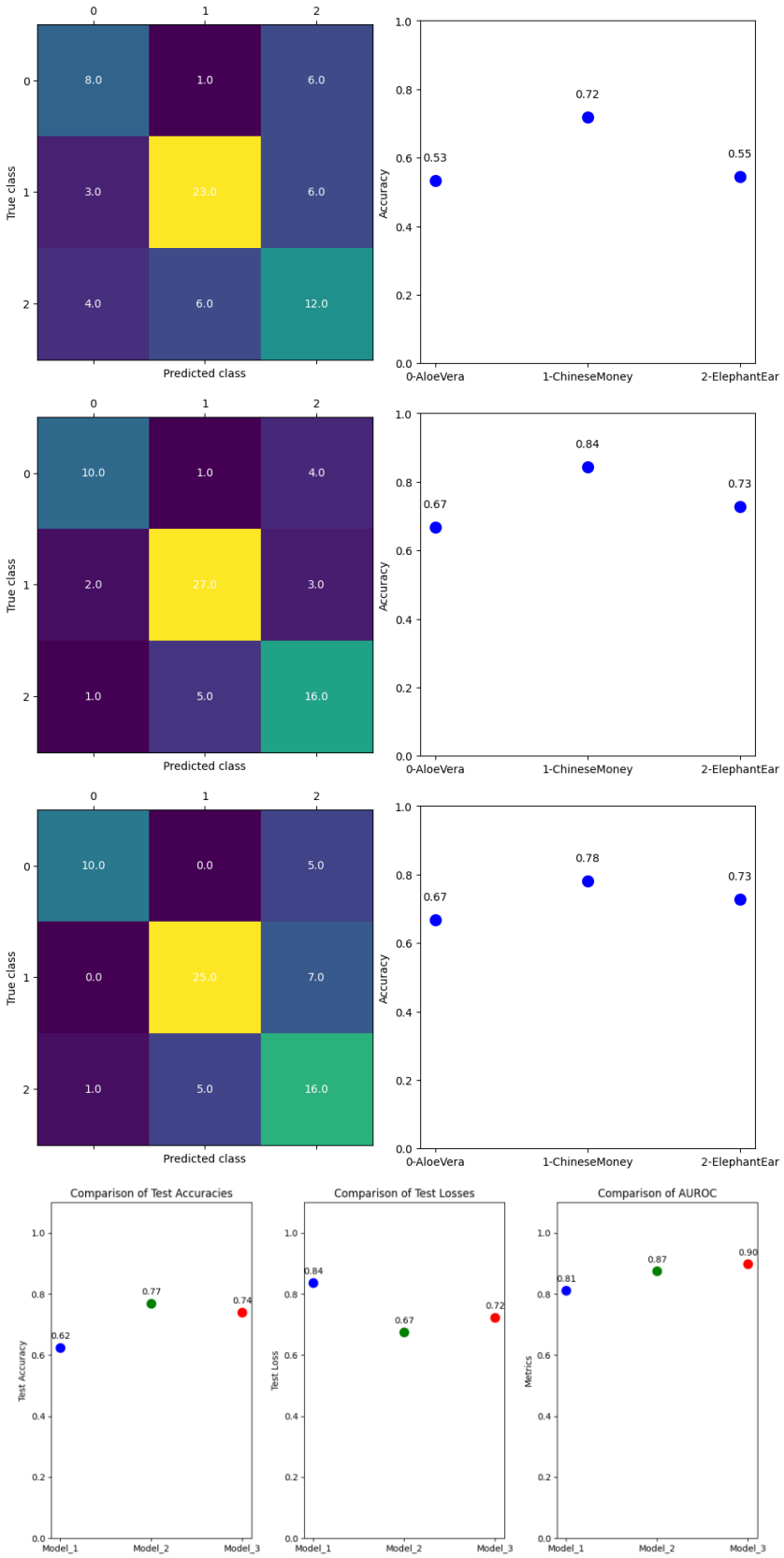
Tanítás közben monitoroztuk a loss-t, az accuracy-t, a per_class_accuracy-t, illetve számoltunk avg_per_class accuracy-t is, mivel úgy döntöttünk ez lesz a mérőszáma annak, hogy „milyen jó egy model”.

Hogy miért ezt választottuk, arra az a magyarázat, hogy nem tartottuk annyira fontosnak a másik alapvető metrikát, a loss-t, ugyanis nekünk most kevésbé fontos, hogy semmiképp se csússzon hiba a gépezetbe. Inkább szeretnénk arra törekedni, hogy minél több találat érkezzen. Azonban, mivel a képek száma se egyenlő, illetve csalóka lehet egy 0.66-os accuracy, ami igazából csak 2 osztályt talál el és a harmadikba teljesen bután működik, ezért hoztuk be a képbe a per_class_accuracy-t, majd, hogy ebből a 3 számból egyet tudjunk kovácsolni, így lett az átlag.

Mentjük még a confusion-matrix alakulását, illetve kiszámoljuk a AUROC értéket is.

Diagramok, kirajzolt eredmények:

Az inkrementális fejlesztés során 3 modellt hasonlítottunk össze egymással.



1. ábra: Kiértékelési metrikák eredményei a három modell esetében

Az 1. ábra első 3 sora soronként 1-1 modell adatait tartalmazza. A baloldalt a confusion-matrix, a jobbon pedig a per_class_accuracy látszik minden esetben.

Ugyanezen az ábrán a 4. sor azt szolgálja, hogy össze tudjuk hasonlítani a 3 modellt egy ábrán. A sorban az első ábra az általános accuracy-t hasonlítja össze, a soron következő a loss-okat, majd a AUROC-ot.

A tanított modellekből tehát az említett módon választottuk ki a legjobbat, és adtuk ezt tovább a Gradio-nak, hogy ezzel dolgozzon.

Gradio

A Machine Learning modell service-ként való deploy-olását a Gradio eszköz segítségével tettük meg. Ezt egyszerű volt konfigurálni a notebookban. Létre kellett hozni egy Gradio interface-t egy cellában, melynek legfontosabb bemenete az a logika, egy function-ben megvalósítva, amely kap egy képet, és az eredményt adja vissza. Ez lényegében annyit csinál, hogy előfeldolgozza a képet a modellnek megfelelő bemenetre, majd a modell jósol egy osztályt, amit a függvény visszaad. Ezután ezt az interfészt csak el kell indítani és a cella kimeneteként már meg is jelenik a grafikus UI, ahol feltölthető egy kép és kipróbálható a modell.

Konklúzió

A modellünkkel tudtunk javulást elérni, hiszen az alap 0.3 találati esély (random tippelés) helyett 0.7-0.8 körüli értékeket tudtunk elérni, egy viszonylag egyszerűbb megvalósítással is.

Kellemes meglepetés, hogy a Gradio-ba beadva a képeket milyen jó arányban találja el a valóságot. Természetesen, még lenne hova fejleszteni a teljesítményt, de jelen feladat keretében erre a megvalósításra jutottunk.