InceptionV3 – Augmentáció

Feladat:

Egy pretrained InceptionV3 nevű modellel valósítottam meg egy classification feladatot, ahova cica kutya képeket használtam fel az internetről.

InceptionV3:

Az InceptionV3-ról annyit érdemes megjegyezni, hogy több millió különböző képpel van tanítva rengeteg különböző témakörben, gyakorlatilag lefedi az összes általános témakört, esetünkben természetesen nem alkalmazható, mert ezeket a vasúti alkatrészek túlságosan specifikusak, de sok esetben hasznos lehet. Egy mátrixot ad vissza. A ResNet50 egy hasonló pretrained modell, ezzel is leteszteltem és hasonlóan nagyon jó eredményeket adott ki, de különösebben nem foglalkoztam vele, említés szintjén viszont érdekes lehet.

InceptionV3 kipróbálása magában:

A feladathoz tartozó colab doksi elején lehet tesztelni önmagában az InceptionV3-at csak egy .jpg képet kell neki adni és ő megcsinálja a predikciót.

Pl(1).: Rottweiler:

https://www.zooplus.hu/magazin/wp-content/uploads/2019/01/Rottweiler.jpg

Válasz:

[[('n02106550', 'Rottweiler', 0.9355747), ('n02108422', 'bull_mastiff', 0.0017547387), ('n02088466', 'bloodhound', 0.0015604508),……]]

A modell 93.556% eséllyel mondja azt, hogy egy Rottweilerről van szó,kevesebb mint 0.2%, hogy bull mastiff és innentől kezdve a maradékon osztozkodik az összes modellben lévő kutyafajta.

Pl(2): Golden retriever:

https://static.posters.cz/image/1300/fototapetak/golden-retriever-dog-416x254-cm-premium-non-woven-130gsm-i81322.jpg

Válasz:
[[('n02099601', 'golden_retriever', 0.61719364), ('n11879895', 'rapeseed', 0.26706433), ('n02099712', 'Labrador_retriever', 0.05226206),…..]]

A modell 61,7%-ot jósol arra, hogy a kutya egy golden retriever és 26,7%-ot, hogy rapeseed a fajtája. Látszik mivel a golden retriever egy sokkal kevésbé unique felépítésű így nehezebben tudja eltalálni a modell. Ha jól tudom erről is beszéltünk, hogy érdemes lenne nem csak a tippelt osztályt kiíratni hanem az esélyeket is.

A modell további tanítása:

Lépése:

1. Képek letöltése, kicsomagolása
2. A modell betöltése, képek átméretezése (299x299 az esetünkben)
3. Alapmodell felépítése amivel még tanítani fogjuk
4. Adat augmentáció
5. Tanítás

Az InceptionV3-nál meg kell adni, hogy a legfelső rétegeket includeolja-e én ezt falsera állítottam. Aztán GlobalAveragePooling2D()-vel 1Ds adatokat állítottam elő, mert a Dense réteg cask 1ds adatot kaphat, ezen relut hívtam, kipróbáltam tanh-van sigmoidal is. Az a tapasztalat, hogy szinte minden esetben a relu a legjobb ha nem is feltétlen javít a pontosságon gyorsabban lefut (van ahol rosszabb).

Kezdetben tanítható paraméterekkel, augmentáció nélkül:

- -Lassabb
- -Kifogy a képekből hamar a kicsi a pool és magas az epoch szám
- -Val_loss: 8.3588 val_accuracy: 0.6900 (magas val_loss alacsony accuracy)
- -Nincs augmentáció, de mindenképp érdemes normalizálni amit meg is lehet tenni az ImageDataGeneratorrel, csak rescalelni kell a képet 1/255 értékkel.

-Nem tanítható paraméterekkel, augmentáció nélkül:

- -Val_loss: 0.0030 - val_accuracy: 1.0000
- -Itt jegyeztem meg, hogy érdemesebb let volna nem egy pretrained modellel vizsgálni az augmentáció előnyeit, lehetőségeit. Persze átlagban még így is jó volt az augmentáció, mert nem egy tesztesetből kell kiindulni. Sok futtatás után azért tudott javítani.

ImageDataGenerator tanulságok:

- -Ne zoomoljunk szinte 100%-osan a képekbe, mert sokszor nehezebben tudja felismerni.

- -Kevés kép esetén nagyon megéri augmentálni, nem fogyunk ki képekből javítja a losst és az accuracy.

- -Érdemes horizontal flipet alkalmazni, zoom_ranget, brightness_ranget, rotation_ranget és gyakorlatilag az összes paraméterén jól működött a dolog jobb eredményeket adott mint augmentáció nélkül.

- -Igazából minden ilyen augmentáció hasznos lehet, hogy kicsit dúsítsuk a képeink számát, sajnos abból a szempontból nem volt a legjobb a modell, hogy alapjáraton túlságosan jókat tippelt, de itt

is tudtuk javítani az accuracyn 3%-ot ami a valós életben kifejezetten fontos lehet továbbá egy nagyon jó modellt szinte tökéletessé tud tenni.

Aktiváció – Optimalizáció:

Sok érdekességet szűrtem le, általánosságban a relu nagyon jó és gyors tud lenni, legtöbbször édemes ezt alkalmazni mint aktivációs függvény viszont sok helyen óvatosnak kell lenni.

```
predictions = Dense(1, activation='sigmoid')(x)
```

Amikor a predictionst állítjuk elő a dense réteggel, ott sigmoidot használva nagyon jó eredményeket érhetünk el viszont ha ezt lecseréljük relura, az eddig 95-100% predikció leromlik a random tipp tehát kb 50%-ra. Ugyanez tanh-val úgy 60-65% fele állandósul.

```
model.compile(optimizer='Adam', metrics=['accuracy'],loss='binary_crossentropy')
```

Adammal nagyon jól működött a rendszer de kipróbáltam még pár optimizerrel amikről hallottam, a legtöbbet nem ismerem cask lekerestem érdekesség képp, természetesen binary_crossentropy kell , mert kettő kimeneti osztályunk van, amúgy categorical_crossentropy.

AdaDeltát használva Adam helyett:

val_loss: 0.6118 - val_accuracy: 0.6700

Az eredmény sokat romlott.

Adagradot használva Adam helyett:

val_loss: 0.2814 - val_accuracy: 0.9700

Az eredmény szinte változatlan maradt ellenben a val_loss sokszorosára ugrott.

SGD-t használva Adam helyett:

```
val_loss: 0.1262 - val_accuracy: 0.9850
```

Az eredmény szinte változatlan ez is nagyon jól működött, bár a val_loss itt is emelkedett, jobban működik az SGD-vel mint az Adagraddal vagy az AdaDeltával.

Utolsó megjegyzés:

Rengeteg helyen folyamat cserélgettem az aktivációs rétegeket, főleg a relut, tanh, sigmoidot, mert ezeket ismerem valamennyire. Szinte mindig ahogy azt már említettem a relu lett a legjobb de azért voltak olyan esetek amikor nem, ellenben gyorsabb mint a sigmoid és tanh tehát körülbelül azonos

val_loss val_accuracynél érdemes relut használni. Fontos hiba, ami tanulságos lehet, hogy vigyázni kell az batch-size számának helyes megválasztására, többször is elrontottam, hogy túl magasan hagytam így kifogytam a képekből, néha pedig már nem javult a modell. Sok technika van, hogy lehet védekezni ezek ellen, valamelyik projektben használtam EarlyStoppingot, Dropout is hasznos lehet. Nem érdemes nagyon magas epochszámot sem választani ha mégis érdemes EarlyStoppingot alkalmazni.