Képfeldolgozás a gyakorlatban

Objektumfelismerés OpenCV & TensorFlow segítségével

A projektem célja, hogy rövid és egyszerű kódon keresztül végezzek objektumfelismerést. Miért? Az internetnek hála boldog-boldogtalan talál ehhez eszközöket, minimális kódolási tudással is tud alkotni *valamit*. Amellett, hogy saját felvételeimre alkalmazok objektumfelismerést, szeretnék rávilágítani néhány hátrányosságára a rövidnek és egyszerűbbnek tűnő kódban. Ugyanis működőképes, de mégis, nem a leghatékonyabb megoldás.

Mi történt az eredeti tervvel, amiről kérdeztem a Tanárnőt? Annyi sok más őrültséget csinálnak a kötelező tárgyak tanárai, hogy nem tudok időt szakítani arra a projektre, bármennyire is szeretném megcsinálni. A virágos projektet túl komplex szinten képzeltem el, és ha beadnám az alapjait, úgy érezném, csak valami puritán, nem törődöm kód kerülne a Tanárnő elé, attól meg rosszul érezném magam, így inkább ez lett.

Előkészületek

Mielőtt bármihez is hozzáfogtam, körbenéztem az interneten. Ez tűnt a legjobb kezdő lépésnek, látni szerettem volna, milyen eszközöket kínál egy egyszerű google keresés, mi lesz az első dolog, amivel szembesül az, aki random rákeres a témára. A Wikipédiás oldaltól lejjebb görgetve kismillió tutorialt, kódrészletet, videókat találunk, mind különböző hozzáértési-szintet követel. Az első és legnagyobb következtetés, amit levonhattam, az a tény, hogy mennyire nem szeretik az emberek a C++-t használni. Minden megoldás IS Pythonra vezet, keveset lelni más nyelvben, max úgy, hogy a C++ verzió mellett Python is feltűnik. Természetesen nem lep meg, én sem preferálom a C++-t, egyáltalán nem áll a kezemre, és már szinte traumatikus élmény benne dolgozni, annyira erőltették középsuliban. Ez az én személyes problémám, de lényeg a lényeg: a kódunkat Pythonban fogjuk írni.

**Miért?**

* Python ilyen: jó
* Kezdő barát, legalábbis szerintem (nagy előny, hogy nincs ; buli)
* Imádok Jupyter Notebookban dolgozni (ez volt a fő-fő ok)

TensorFlow

Avagy nem fogok nekiállni betanítani egy új modellt. Mert miért tenné, aki a legrövidebb megoldást keresi? Nyilván, ezáltal nincs saját képeken betanult modellünk, ez befolyásolhatja a felismerés pontosságát. Ez az első hátrány:

Nem a mi képeinken tanult be a modell, nem biztos, hogy minden esetben 100%-os pontossággal ismer fel objektumokat.

Szóval, előre betanult a modell, de min? Ha a labels.csv fájlt megnyitjuk, láthatjuk, hogy a felhasznált modellünk 91 féle objektumot tud felismerni. Ez ahhoz képest elég sok, de nem feltétlenül jelenti azt, hogy a mi képeinkhez biztosan jó. Persze, ha tudom, mi van biztosan a listában, tudok pont olyan képeket csinálni. No és ha a kiskutyámat szeretném felismertetni? Itt van a baj. Meg is találtuk a második hátrányt:

Limitált a felismerhető alakok száma.

Javíthat, ha nagyobb modellt választunk, biztos akadna, ami felismerné az én Mollymat, viszont most maradunk a rövidebb modellnél.

TensorFlow Lite

TensorFlow Lite is a set of tools that enables on-device machine learning by helping developers run their models on mobile, embedded, and edge devices. – mondja a Google, ha beütjük ezt.

Bővebben megfogalmazva, olyan modellel fogunk dolgozni, ami kisebb. Egyrészt, mert 5 objektum az elvárás, másrészt, mert a prioritás továbbra is az egyszerűség (harmadrészt, betelt a laptopom tárhelye és nincs pénz új SSD-re). A kiválasztott modellünk a notebookban került linkelésre.

Indul a Mandula – azaz telepíteni kell az atya úr istent is

A notebookban kikommentelve ugyan, de ott hagytam, mennyi dolgot kellett telepíteni. (Itt most felejtsük el a tényt, hogy Anaconda-ban environment-tel is kínlódtam egy sort, úgymond irreleváns most.)

Tensorflow és tensorflow\_hub magától értetődőek mostmár, hogy miért kerültek oda, opencv pedig úgyszint evidens. Lássuk a többit:

**Numpy** – amolyan matematika szeretetcsomag a Pythonhoz, többdimenziós tömbök, mátrixok kezeléséhez jó

**Pandas** – programkönyvtár, ami adatok feldolgozásához és elemzéséhez kell

**Matplotlib**.**pyplot** – matlab-izáljuk a témaköröket, módosítgatni tudunk vele ilyen-olyan ábrákat

A képen szöveg, képernyőkép, számítógép látható

Automatikusan generált leírás

A képen szöveg, képernyőkép, számítógép látható

Automatikusan generált leírás

Mostmár tényleg indul a Mandula

1. A modell: Első lépésben betöltjük a kiválasztott modellünket. Letölthető, viszont tárhelyet spórol, ha a weboldalról töltjük be. Ha már van erre lehetőség, élünk is vele.
2. A kép: A képet úgy módosítjuk, hogy az a modell által feldolgozhatóvá váljon. Itt hátrány lehet, hogy minden képből is négyzetet fogunk csinálni:

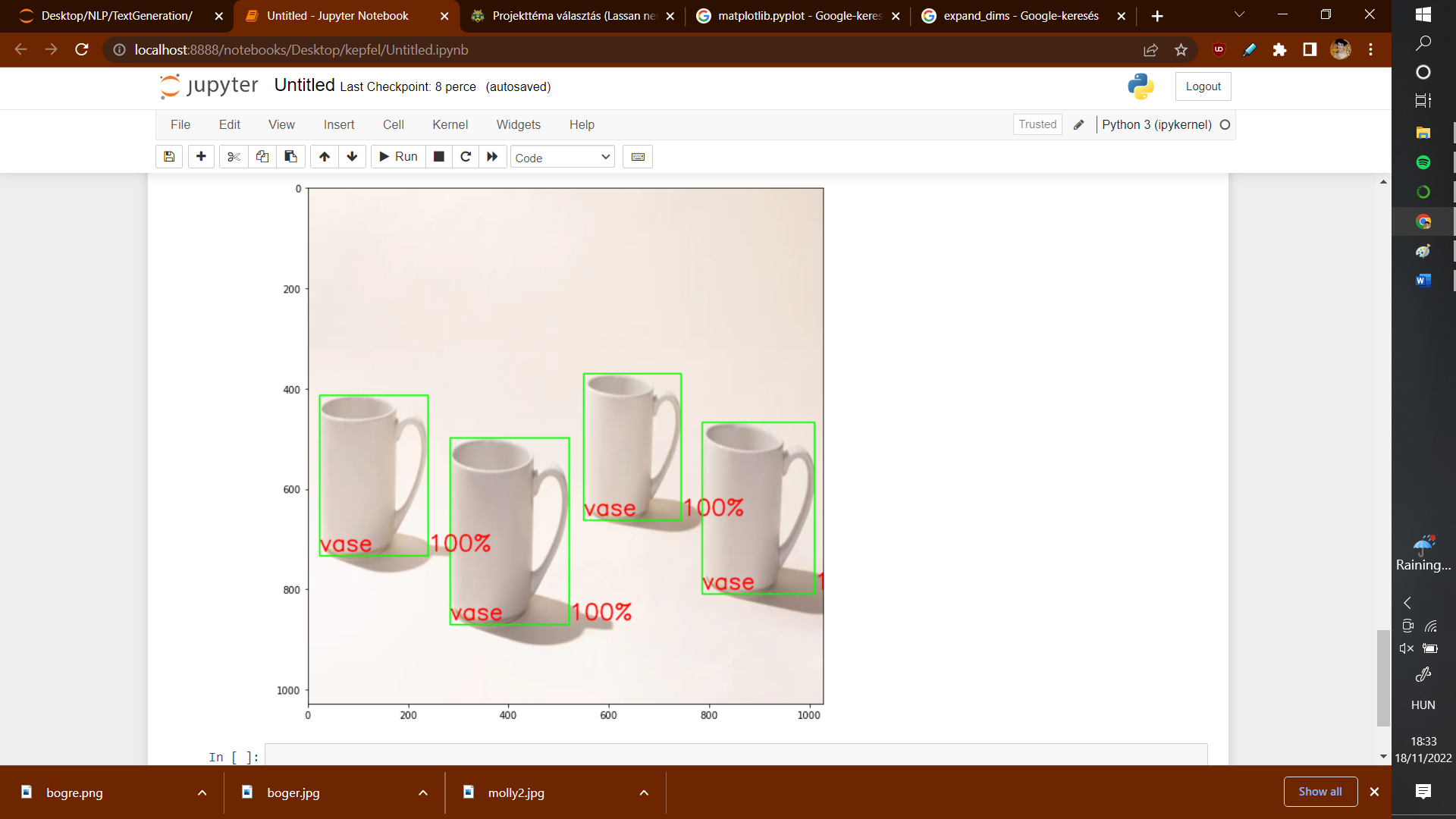
Az objektum torzul, így nem biztos, hogy pontosan azt kapjuk eredménynek, ami valójában szerepel a képen.

A képen szöveg látható

Automatikusan generált leírás

1. Címkék és keretezés: Betöltjük a címkéinket, illetve megnézzük, mennyire tudja felismerni, mi van a képen. Ezután az opencv eszközeit felhasználva fogjuk bekeretezni és feliratozni az objektumainkat. Az is kiírásra kerül, mennyire biztos benne, hogy eltalálta, mi van a képen.

Első próbaként random google-ről szedett képre futtattam le, és lám: felsült. Az ok nagy eséllyel a torzulás miatt van, így megnyújtva valóban vázának néz ki, szóval nem tévedett nagyot. Legalább 100% biztos benne, fő a magabiztosság. Arra jó volt, hogy megmutassa, több egyforma tárgyat is fel tud ismerni.



Első tesztek – Google-csuprok, csak, hogy lássam, egyáltalán működik-e, mielőtt nekiállnék fotózgatni

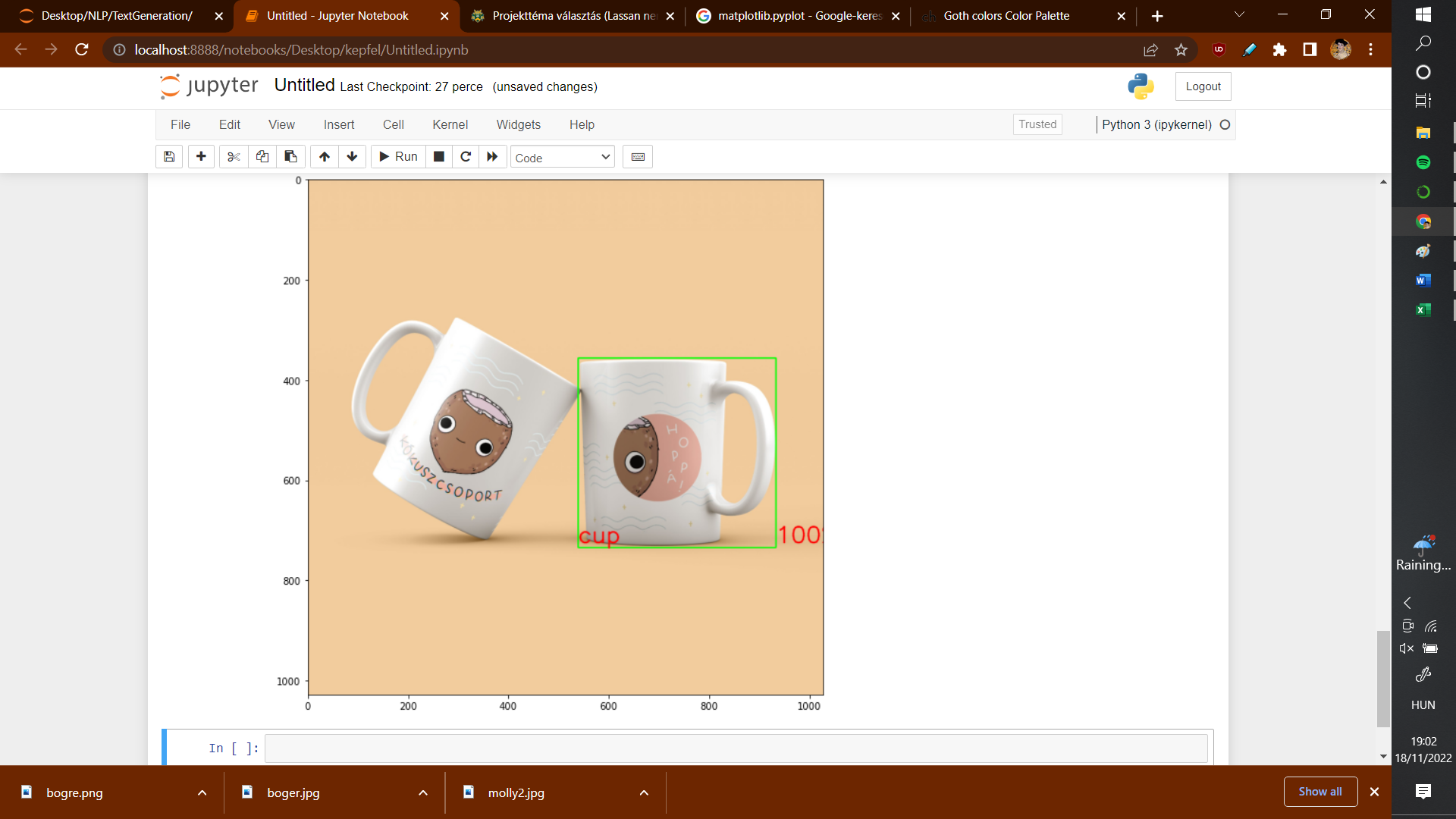
Fentebb szerepelt a bögre-vázánk, ezúttal levágtam a képből, hogy kevésbé torzuljon. Mostmár bögrék, és alá is támasztottuk az egyik hátrányt, a torzulást.

A képen szöveg, képernyőkép, beltéri, konyhai felszerelés látható

Automatikusan generált leírás

Újabb Google-kép: Két bögre, vészesen közel egymáshoz

Elkülönülő objektumokat felismert, ám ennek az eredménye felteszi a kérdést: Milyen távol kell legyenek egymástól az egyes tárgyak, hogy mind felismerésre kerüljenek?



Tesztek élesben – nekiálltam fotózgatni

1. A saját bögrém – felismeri-e a flancos „pixeles” csempe előtt?

A képen szöveg, képernyőkép, számítógép, megjelenítés látható

Automatikusan generált leírás

1. A kanalam – nem ismeri fel a kanalat. Sejthető, hogy másfajta kanálon tanult be

A képen szöveg, képernyőkép, monitor, számítógép látható

Automatikusan generált leírásA képen szöveg, képernyőkép, számítógép látható

Automatikusan generált leírás

1. A saját bögrém és egy kanál – feismeri a bögrét, de a kanalat nem

A képen szöveg, képernyőkép, számítógép, megjelenítés látható

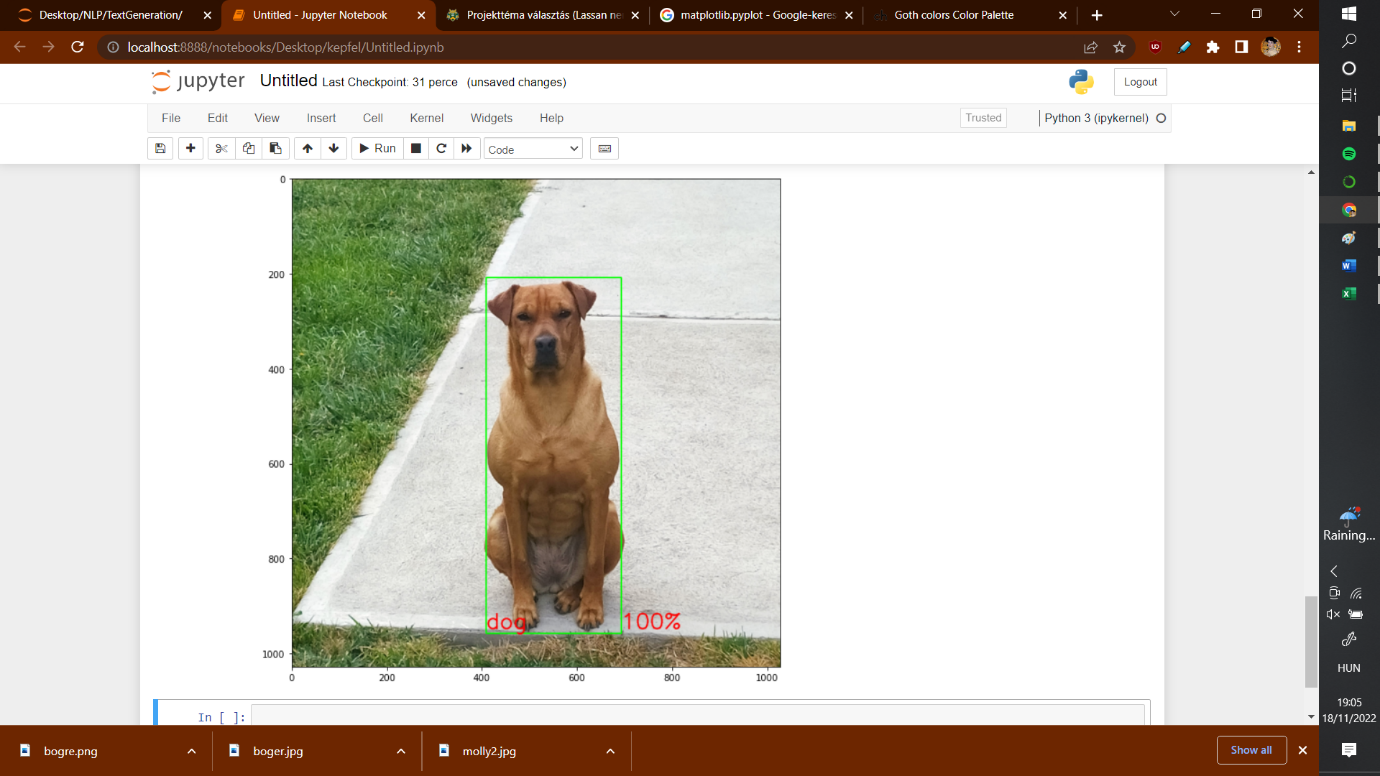
Automatikusan generált leírás

1. Anya tálja

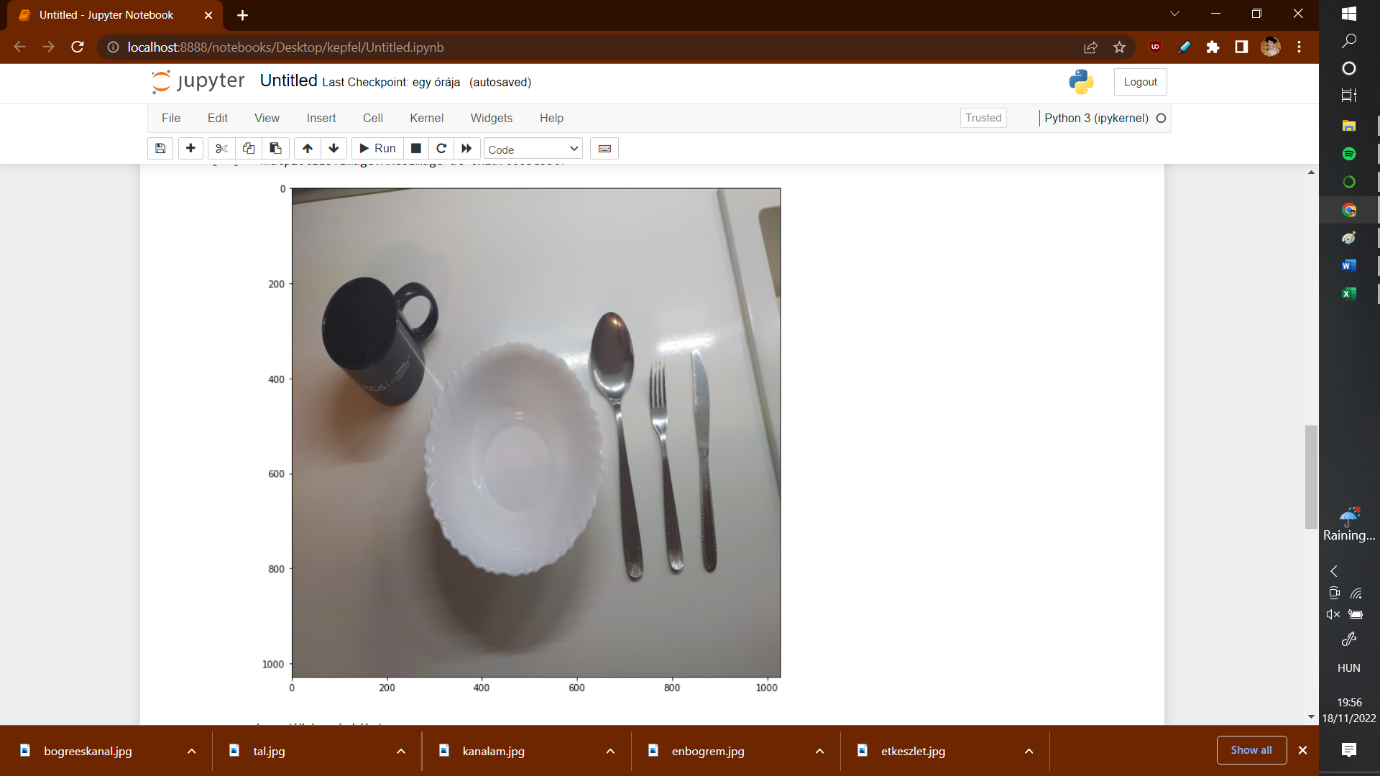
A képen szöveg, képernyőkép, monitor, megjelenítés látható

Automatikusan generált leírás

1. A kutyám – hoppá, van a labelek között kutyus! És lám, Molly valóban kutyus!



1. Egy konkrét étkészlet - vadulunk, de hiába, ugyanis olyan szöget sikerült elkapni, amiből nem ismer fel semmit sem, ugyanazt a hibaüzenetet kaptuk, mint a kanálnál



Konklúzió

A modell csak szórakozásra alkalmas, több olyan esetbe is belefutottunk, ahol nem tudja felismerni, mi van a képen, vagy téved valamilyen oknál fogva. A kód maga rövid, lényegretörő, ebből a szempontból hatékony. 1-1 kép feldolgozása viszont sok időt vesz igénybe. Ez természetesen rendszerfüggő is, de erősen von le a hatékonyságból. Biztos vagyok benne, hogy legalább fél óra csak a képek tesztelgetésével ment el, főleg, mert feszt elfelejtettem screenshotot készíteni – részletkérdés.

**A megoldás összes hátránya tehát:**

* Mivel nem saját képeken tanult be, nem biztos, hogy mindent ismer fel, hiába van a listában.
* Limitált a felismerhető alakok száma.
* Jól elkülönülő alakok kellenek.
* Primitív, rugalmatlan kód.
* Hosszú futási idő.

**Köszönöm a figyelmet!**

Gellén Rebeka, J97N2Y, 2022/11/18