

TDK-dolgozat

Hartyányi Gábor

Hartyányi Gábor
Mérnök informatikus MSc

GYOMNÖVÉNYEK SZÁNTÓFÖLDI DETEKCIÓJA

Témavezető: Dr. Czúni László

Külső konzulens: Dr. Mesterházi Péter Ákos

Műszaki Informatikai Kar
Villamosmérnöki és Információs Rendszerek Tanszék

Pannon Egyetem
Lezárva: 2022.11.14



Köszönetnyilvánítás

Dolgozatom elkészültével szeretnék köszönetet mondani témavezetőmnek, Dr. Czúni Lászlónak az egyetemi Számítógépes látás tárgy keretében tanult ismeretek megszerzéséért és annak gyakorlatban történő hasznosításáért. Köszönetemet szeretném kifejezni továbbá a hónapokon át tartó segítségért, tanácsokért és a szigorú szakmai elvárásaiért, amivel a dolgozat elkészültét lehetővé tette.

Köszönöm az ötleteit, támogatását és a szakmai segítségét Dr. Mesterházi Péter Ákosnak, aki munkatársként rengeteg segítséget adott a téma kitalálásához és óriási támogatást nyújtott a mezőgazdasági kérdések vonatkozásában.

Köszönjük az OTKA K 135729 Geometriai tulajdonságok kinyerése lényegi vizuális jellemzőkből és az NVIDIA GPU Grant program támogatását.

Kivonat

A nemrégiben megkötött úgynevezett uniós „Green Deal” megállapodás értelmében 2030-ig mintegy 50 %-al kell csökkenteni a mezőgazdaságban kijuttatott kemikáliák mennyiségét [8]. Mivel a termőföld és a növénykultúrák változatlanok, csak egyetlen módja van a cél elérésének: a precíziósan kijuttatott input anyagok. Alapelve, hogy csak oda juttassunk ki vegyszert, ahová szükséges és csak a szükséges mennyiségben. A jelen kutatás a gyomfelismerés képi módszerével igyekszik a permetezőgép vezérlő rendszerek segítségére lenni. A cél egy olyan gépi tanulással támogatott képfeldolgozási rendszer kidolgozása, ami a kukorica kultúrában (későbbiekben egyéb más növénykultúrákban is) képes a kultúrnövényt megkülönböztetni a gyomoktól. Ennek megfelelően képes döntést hozni a permetező bekapcsolására és elzárására annak érdekében, hogy csak a szükséges helyre történjen kijuttatás. A dolgozatban – a fentiek bemutatása és a szakirodalom elemzése mellett - a gyomnövények képi detekciójára és lokalizációjára mutatok be módszereket, amelyeket saját készítésű adathalmazokon tanítok be és tesztelek.

Tartalomjegyzék

1. Bevezetés	6
1.1. A Green Deal megállapodás	6
1.2. Szántóföldi permetezési módszerek áttekintése	7
2. Irodalomkutatás	14
3. Képek készítése	17
3.1. Felvételkedészítés	17
3.1.1. Mezőgazdasági környezet	17
3.1.2. Felvételkedészítés eszköze	18
3.1.3. Felvételkedészítés módja és a felvételek listája	20
4. Képi adathalmaz létrehozása	26
4.1. Képfeldolgozás	26
4.1.1. Videó felvételek képekre bontása	26
4.1.2. Képek szelektálása	27
5. Gyomdetekciós modellek	30
5.0.1. Tesztek célja	30
5.0.2. A dolgozatban használt szoftvertechnológiák	31
5.1. Tensorflow-Keras CNN model	32
5.1.1. Tensorflow tesztkörnyezet	32
5.1.2. CNN alapmodell	33
5.1.3. Teszt adathalmaz	34
5.1.4. Tanítási tesztek 320x240 felbontású képekkel	35
5.2. EfficientNet	36
5.2.1. EfficientNetB0 csökkentett adathalmaz saját súlyokkal	37

5.2.2.	EfficientNetB0 transfer learning csökkentett adathalmazzal	40
5.2.3.	EfficientNetB0 teljes adathalmaz saját súlyokkal	41
5.3.	EfficientNetB0 teljes adathalmaz - Transfer Learning	42
6.	Permetezési terület pontos lehatárolása	46
6.1.	Terület lehatárolásának fontossága	46
6.1.1.	Kukoricaszárak detekciója	47
6.1.2.	Képek annotálása	49
6.1.3.	COCO JSON formátum	50
6.1.4.	Mask R-CNN környezet kialakítása	52
6.1.5.	Mask R-CNN tanítási folyamat és eredményei	53
6.1.6.	Kukorica tövének detekciója	54
6.1.7.	Egyenes illesztése a tőpontokra lineáris regresszió segítségével	56
6.1.8.	Sáveltolás térbeli leképezése	57
6.1.9.	Homográfia	59
6.1.10.	Aruco Marker ismertetése és használata	61
6.1.11.	ROI meghatározása homográfia segítségével	62
6.1.12.	Kép maszkolása	66
7.	Összefoglalás és lehetséges továbbfejlesztések	67
7.1.	További teendők	68
	Irodalomjegyzék	76

1. fejezet

Bevezetés

Az utóbbi évszázadban, de különösen az utóbbi néhány évtizedben a Föld lakossága egyre nagyobb ütemben növekszik. Az ezzel járó élelmiszer ellátás ennek megfelelően egyre nagyobb gondot okoz a világ számára. Ugyanez a tendencia figyelhető meg Európa és szűkebb hazánk tekintetében is. A lakosság megnövekedett számával arányosan egyre több élelmiszer megtermelésére van szükség, viszont a rendelkezésre álló földterületek nem növekednek, sőt az iparosodás miatt inkább fogyatkozik a művelhető földterületek száma. Ezek a tények arra kényszerítik a gazdálkodókat, hogy ugyanazon a földterületen a lehető leghatékonyabban termeljék meg ugyanazt - vagy ha lehet, akkor több-mezőgazdasági terméket, mint korábban. Ennek egyik sarokköve a precíziós gazdálkodás, amelynek egy speciális területére igyekszek fókuszálni a dolgozatomban.

1.1. A Green Deal megállapodás

A nemrégiben megkötött úniós "Green Deal" [8] megállapodás értelmében 2030-ig 50%-al kell csökkenteni a mezőgazdaságban kijuttatott vegyszerek mennyiségét. Ezek a vegyi anyagok lehetnek műtrágyák és különféle peszticidek (per-metszerek). A dolgozat keretében a peszticidek közül a gyomirtószer (her-bicidek) szántóföldi kijuttatására koncentrálok.

1.2. Szántóföldi permetezési módszerek áttekintése

Hagyományos permetezési módszerek

Hagyományos módon a különféle gyomirtószerek kijuttatását vontatott vagy önjáró permetezőgép segítségével végzik a szántóföldi kultúrában.



1.1. ábra. Önjáró permetezőgép (kép forrása: horschaustralia.com)



1.2. ábra. Vontatott permetezőgép (kép forrása: horschaustralia.com)

A kijuttatás körülményeit tekintve megkülönböztethetünk tarló és állomány permetezést. Első esetben kikelt kultúrnövény nélküli területre történik a ki-

juttatás, a második esetben viszont a kikelt haszonnövény állományban történik a vegyszerek kijuttatása. A permetezés folyamán a teljes területet egy előre meghatározott átlagos kijuttatási mennyiséggel permetezik le. A gyomírtószerek egyfomán jutnak a haszonnövényre, a gyom nélküli területre és a gyomokra egyaránt. A vegyszerek hatással vannak nemcsak a gyomokra, hanem a szántóföldi kultúrára is, azonban kisebb mértékben. Ez az alapja a gyomírtószerek hatásmechanizmusának.



1.3. ábra. A legelterjedtebb módszer szerint permetezéskor mindenhol jut vegyszer (kép forrása: www.istockphoto.com)

Precíziós kijuttatási módszerek

A modernebb eszközök képesek a permetező keret szakaszolására és a kijuttatott mennyiség szabályozására előre megadott kijuttatási terv szerint (prescription). Ez azonban a gyomirtás precíziósan történő végrehajtására nem igazán alkalmas.



1.4. ábra. Szakaszevezérelt permetezés (kép forrása: hardi.com)

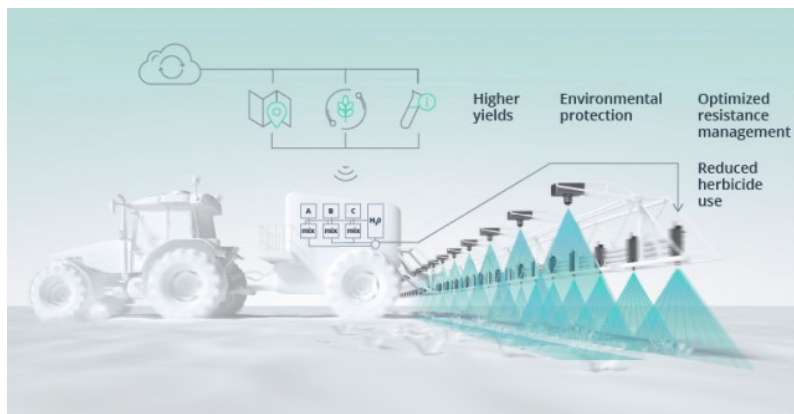
Alternatívaként megemlíthető a drónos úgynevezett foltpermetezés, ahol egy monitoring drónnal feltérképezett állományt speciális permetező drónnal kezelnek. Ez a módszer azonban a szabályozatlanság és egyéb hatékonysági kérdések miatt egyelőre nem járható út.



1.5. ábra. Drónnal történő permetezés (kép forrása: droneag.farm)

Pillanatnyilag a legkorszerűbb technológiának a számítógépes látáson alapuló megoldások számítanak. Ez a technológia a gyomnövényeket és a hasznos növényeket mesterséges intelligencia segítségével igyekszik felismerni és megkülönböztetni, majd ezen információk alapján a permetezőkeret fúvókáit oly

módon vezérli, hogy csak a szükséges területre jusson ki gyomirtószer.



1.6. ábra. Számítógépes látáson alapuló permetezés (kép forrása: Bosch)

Dolgozat fő céljai

A dolgozat fő témája ennek a gyomfelismerési feladatnak új megközelítésben történő vizsgálata.

A cél egy olyan módszer kidolgozása, ami lehetővé teszi a kukorica ültetvényen a gyomos területek felimerését a növény sorában. Az elsődleges célkitűzés a kultúrnövény (jelen esetben kukorica) fejlődési korai szakaszában előforduló gyomok felismerése és elkülönítése. A piacon fellelhető technológiák legtöbbször igyekeznek felismerni a gyomfajtákat. Ebben a dolgozatban nem cél az egyes gyomnövényszínek felismerése. Annak eldöntése a cél, hogy egy (járműre szerelt kamera által) megfigyelt terület tartalmaz-e gyomot vagy sem. Ennek megfelelően ezután tudjunk vezérlést adni egy permetezést kapcsoló szelepnek a zárására vagy megnyitására. Ez a fajta egyszerűsítése a felismerési eljárásnak lehetőséget ad a kivitelezés egyszerűsítésére (pl. gépi tanulási fázis rövidebb és egyszerűbb lehet). A gyakorlat azt mutatja, hogy ennél több információra nincs szükség ahhoz, hogy a vegyszermennyiséget jelentősen csökkenthessük. A megtakarítás akkor jelentkezhet, amikor a gyommentes területet felismerve megszakítjuk a kijuttatást.

A vizsgált módszer a kamera elhelyezésében különbözik a jelenleg használatos megoldásoktól. A különféle gyártók (pl. Amazone-BASF-Bosch [1]) megoldásaiban a növénykultúrát felülről, függőlegesen vizsgálják. Ennek a módszernek a hátránya lehet, hogy az idővel egyre nagyobb lomb kitakarja a gyomokat.

A javasolt, kidolgozandó megoldás a tövek közé eresztett kameraállásból igyekszik elkerülni a kitakarásokat a későbbi növényfejllettségi szakasz idején.



1.7. ábra. Amazone-BASF-Bosch számítógépes látáson alapuló permetezési megoldása (kép forrása: Amazone)

Amennyiben a dolgozat tárgyát képező vizsgálat sikerrel jár, akkor egy teljesen új eszköz megalkotására lesz lehetőség. Egy mechanikus talajművelő eszközre (kultivátorra) szerelve és permetezővel kiegészítve olyan eszközt kapunk, ami a sorközt vegyszermentesen mechanikus úton kezeli, miközben a sort (tövek közti rész) differenciáltan, precíziós eszközökkel permetezzük. Ezzel jelentős kijuttatási megtakarítást lehet elérni, hiszen minden más módszerhez képest töredéknyi területet kell gyomirtóval kezelni.



1.8. ábra. Sorközművelő kultivátor kijuttató adapterrel, amit AI módszerekkel lehetne vezérelni (kép forrása: Farmnet)

Azonban a földfelszínhez közel lévő, viszonylag nagyfelbontású és gyorsan mozgó kamera képalkotása okozhat problémákat a képminőségben, ezért valós alkalmazási körülmények között készült felvételeken szükséges a módszert validálni. A képi szenzor lehetséges porosodása, szennyeződésre külön megfontolásokat igényel.

A dolgozat során ennek megközelítésnek az elméleti és néhány gyakorlati, de kulcsfontosságú lépését vizsgálom meg és dolgozok ki megoldási javaslatot.

A dolgozat során először a felvételkedészítés és a képi adathalmaz összeállításával foglalkoztam. A munkálatokat júliusban kellett elkezdni, ugyanis akkor még lehetőségem volt a kukorica másodvetésekben felvételeket készítenem. A következőkben a rendelkezésre álló felvételeket különféle konvolúciós hálózatokon teszteltem. Arra voltam kíváncsi, hogy a közismert alaphálózatok segítségével milyen eredményeket lehet elérni a kukorica növénykultúrában készített képekkel. Ezeket a teszteket szűkített adatkészlettel végeztem el a teszt hatékonyságát szem előtt tartva. Ezekből a tesztekben és tapasztalatokból kiindulva összeraktam egy összetettebb neurális hálózatot, amelynek vizsgálatakor a teljes adatkészletet felhasználtam. Ebből a vizsgálatból arra szerettem volna választ kapni, hogy az általam használt perspektívában készült fotók alapján

lehetséges-e a gyomos és a gyom nélküli felvételek hatékony megkülönböztetése. A következő lépésben azt vizsgáltam, hogy a felvételek tartalmának leszűkítését a permetezni kívánt területre hogyan tudom megoldani. Ehhez felhasználtam a képi szegmentáció (pontosabban az egyedi objektumfelismerés és szegmentáció) és a homográfia eszközeit.

2. fejezet

Irodalomkutatás

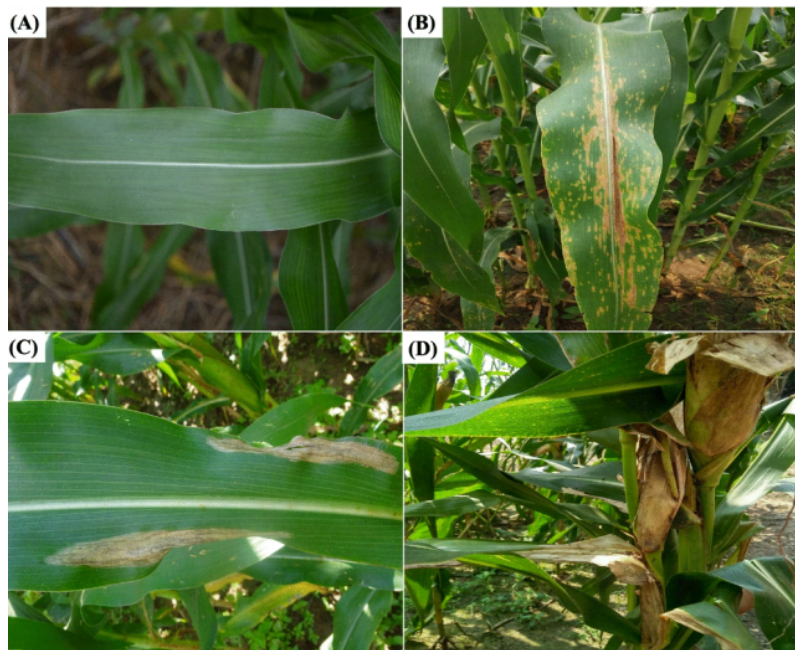
A dolgozat elkészítését azzal kezdtem, hogy felkutattam a témában néhány elérhető cikket és különféle online publikációt. Kíváncsi voltam, hogy mi a tudomány mai állása a gyomdetektálás kapcsán. Mivel manapság egyre fontosabb szerepet tölt be a precíziós gazdálkodás, ezért bőségesen áll rendelkezésre a témában különféle forrásanyag.

A Computer Science Review 2021-ben megjelent "Smart Farming in Europe" [18] című cikkében átfogó képet ad a jelenlegi technológiákról, amiket a modern precíziós gazdálkodásban használatosak, beleértve a különféle mélytanulási módszereket is. Emerson, Nuno és António Pereira által 2020-ban „A Systematic Review of IoT Solutions for Smart Farming” [19] címmel közzétett publikációjában részletesen kitér a különféle IOT és adatátviteli technológiákra, valamint ezen túlmenően vizsgálja a különböző mezőgazdasági feladatok (monitoring, betegség megelőzés, öntözés menedzsment...stb) ellátásához szükséges érzékelőket. Összegyűjtötték a számítógépes látás és a mesterséges intelligencia mezőgazdasági használatáról szóló publikációkat.

Először arra gondoltam, hogy a gyomok felismerését színérzékeléssel kellene megoldani. A háttértől való megkülönböztetés miatt ez nyilvánvalónak tűnt számomra. A növényfajták színeltérései szintén azt erősítették bennem, hogy jó eredményeket lehet elérni ezzel a módszerrel. Ezzel kapcsolatban Rebecca Stone részletes cikkben [22], egy példán keresztül mutatja be a „színtereket” (color spaces) és a szín alapú szegmentációnak a használatát a Python nyelv és az OpenCV segítségével. Az írásban a szerző ismerteti az RGB és CMYK szín terek mellett a HSV és a HSL (hue, saturation, and brightness/luminance) színterek lényegét is. Hasonló a témában a TechVidvan oldalán elérhető "Detect Objects of Similar Color using OpenCV in Python" [25] című írás is. Itt

már kimondottan növényi levélzet szegmentálását mutatják be konkrét példán keresztül. Mindezek mellett kíváncsi voltam a mélytanulós módszerek használatára is. Ezzel kapcsolatban szintén találtam írásokat. A „CNN feature based graph convolutional network for weed and crop recognition in smart farming” [15] című cikkben a szerzők gyomok és haszonnövények felismerését tesztelték különféle CNN hálózatok segítségével. Egy másik cikkben [6] a szerzők a növények tápanyaghiány betegségeit igyekeztek detektálni a fejlett EfficientNet hálózatokkal. Egy további érdekes tanulmány [12] sárgarépa földön egy mezőgazdasági robot által készített képekből, szegmentációs eljárással igyekezett a gyomokat és a sárgarépát felismerni és megkülönböztetni. A szerzők a 60 képből álló képadatbázist, a maszkokat és az annotációs készletet is rendelkezésre bocsátotta a GITHUB-on.

A következőkben annak néztem utána, hogy vajon milyen kukoricát ábrázoló adatbázisok érhetők el az interneten keresztül. Az általam talált adatbázisok mindegyike olyan képeket tartalmazott, amik laboratóriumban készültek a növény fejlődését követve [27], vagy a kukorica betegségeit ábrázolta [28].



2.1. ábra. Kukorica betegségeket ábrázoló fotók (Kép forrása: [28])

Mivel a céljaimhoz meglehetősen sajátos felvételekre volt szükség, ezért

úgy döntöttem, hogy saját képi adatbázist hozok létre.

Áttekintve a felkutatott irodalmat arra a következtetésre jutottam, hogy sokat fejlődtek a neurális hálók az elmúlt néhány évben, ezért érdemes lehet ezekre építeni. Így elkezdtem egy saját módszer kidolgozását. Kíváncsi voltam, hogy a modernebbeknek tartott általános, képfeldolgozásra tervezett hálózatok, mint a Mask R-CNN és az EfficientNet mire képesek a gyomdetektálásban.

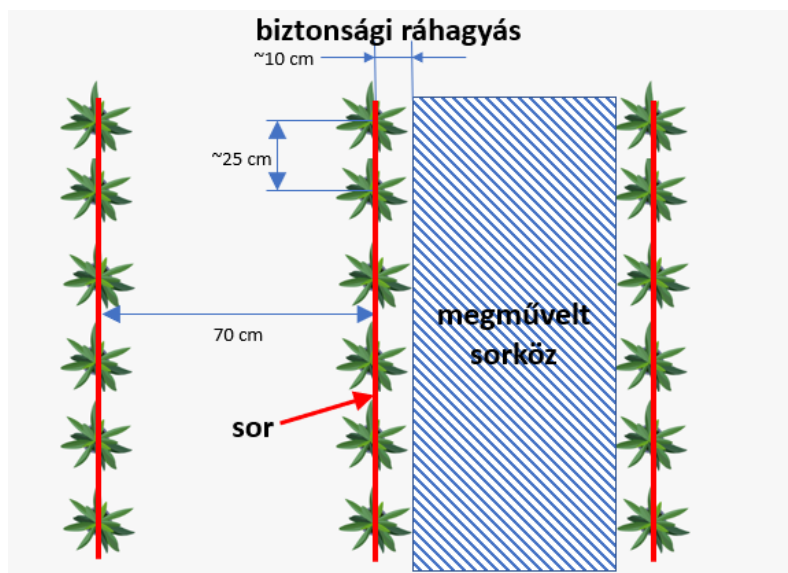
3. fejezet

Képek készítése

3.1. Felvételkészítés

3.1.1. Mezőgazdasági környezet

A felvételkészítés másodvetésű kukorica vetésállományban történt idegen termőterületen. A vetési idő május végén-június elején volt. (Pontos vetési idő és kukorica fajta nem ismert.) A tipikus sortávolság a kukorica esetén általában 70 cm. A tőtávolság erősen eltérő lehet a kultúrnövény fajtájától és a termelési céltól függően. A mi esetünkben ez átlagosan 25 cm volt. A növények magassága a terület tápanyag és vízellátottságától függően 20-50 cm között változott.



3.1. ábra. Kukorica sorok szerkezete

3.1.2. Felvételkészítés eszköze

A felvételek készítésére egy Gopro 7-es kamera [3] került felhasználásra.

A felvételkészítéskor a kamera elhelyezésének fontos szerepe volt. A kamerát olyan helyre kellett szerelni, ahonnan a lehető legjobban megfigyelhető a talaj, valamint a növény szerkezet. Továbbá fontos szempont volt, hogy a levélzet minél kevesebbet takarjon ki a képből. Ezek a szempontok miatt a kamerát a lehető legalacsonyabbra kellett elhelyezni.

Az első felvételek készítésekor a kamerát egy mezőgazdasági munkagépre szereltem fel (talajlazító). A felrögzített kamera 25 cm-re a földtől került rögzítésre 45 fokos szöget bezárva. Felbontás 2704x1520 px 60fps mellett. A vízszintes síkon mért távolság a kukorica sortól hozzávetőleg szintén 25 cm volt. Ez a távolság a traktor nyomvonalának eltérései és a veteményezett sor egyenetlensége miatt nem állandó.



(a) Munkagépre szerelt kamera elhelyezése (b) Munkagépre szerelt kamera rögzítése

3.2. ábra. Kamera elhelyezése és rögzítése a munkagépen

A talajlazítás művelet végzésekor tipikusan 10-15 km/óra sebességgel haladnak a traktorok. A felvételeket két haladási sebességgel készítettem. Először 10 km/óra sebességet állítottunk be a vontatón. Sajnos a felvételek elmosódottak

lettek, emiatt a következő felvételt 8 km/óra mellett ugyanolyan beállításokkal végeztem. A felvételek továbbra is elmosódottak maradtak. Ennek az lehet a megoldása, hogy nagyobb fényerejű kamerát kell használnom, ahol rövidebb az expozíciós idő vagy mesterséges megvilágítással kell megvilágítani a felvétel helyszínét, aminek hatására szintén csökkenthető a zársebesség.



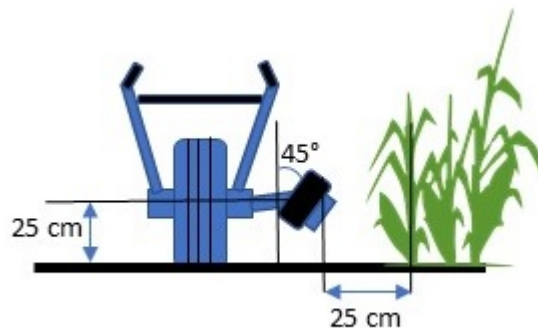
3.3. ábra. Traktorral 10 km/h sebességgel készített elmosódott felvétel



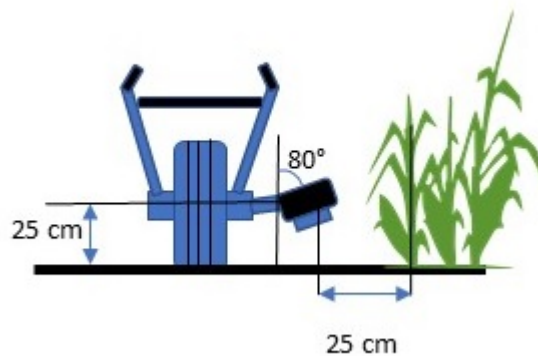
3.4. ábra. Traktorral 8 km/h sebességgel készített elmosódott felvétel

A fentiekből látható, hogy a kép veszített az élességéből és nem tudtam, hogy mekkora befolyással lesz a feldolgozás eredményére. Ebben a dolgozatban úgy döntöttem, hogy lassabban készítem el a képeket annak érdekében, hogy elkerüljem az életlen képek problémáját. Ezt úgy oldottam meg, hogy készítettem egy kerékkel rendelkező kézi eszközt, amire felrögzítettem a kamerát. A kamera egy csuklós rögzítő segítségével a kerék tengelyével egyvonalba került elhelyezésre. Ez biztosította, hogy a szerkezet függőleges hosszirányú dőlésétől függetlenül egy magasságban maradjon a talajhoz viszonyítva. Min-

den egyéb beállítás és minden geometriai paraméter ugyanaz maradt, mint a traktorral készült felvételek esetén, csak a felvétel készítésekor alkalmazott haladási sebesség volt kisebb. A sebességet igyekeztem 4 km/h -val tartani. Ennek ellenőrzésére egy mobiltelefont rögzítettem az eszközre és egy GPS alapú sebességmérő szoftverrel mértem a sebességet. [2] Ezzel az eszközzel készítettem a felvételeket, amiből azután a képi adatkészlet összeállt.



3.5. ábra. Kézikocsira szerelt kamera sematikus rajza 45 fokos kameraszöggel



3.6. ábra. Kézikocsira szerelt kamera sematikus rajza 80 fokos kameraszöggel

3.1.3. Felvét elkészítés módja és a felvételek listája

A mintafelvételek legtöbbje 45 fokos kameraszög állásban készült a földhöz képest. A gyomosodás szempontjából készítettem gyomtól teljesen mentes, részben gyomos és csak gyomokat tartalmazó felvételeket is.

Készültek olyan képek, amik a függőlegeshez képest 80 fokos szöget bezáró kamerával lettek rögzítve. Ennek az lehet az előnye, hogy jobban a sorban a tövek közti területet figyeli és kevésbé lehet zavaró a sorközökben megjelenő gyom, mint "zaj".



3.7. ábra. Gyommentes kukorica felvétel 45 fokos kameraszög állással



3.8. ábra. Gyommentes kukorica felvétel 80 fokos kameraszög állással



3.9. ábra. Csak gyomot tartalmazó felvétel 45 fokos kameraszög állással



3.10. ábra. Csak gyomot tartalmazó felvétel 80 fokos kameraszög állással



3.11. ábra. Kukoricát és gyomot vegyesen tartalmazó kép 45 fokos kameraszög állással



3.12. ábra. Kukoricát és gyomot vegyesen tartalmazó kép 80 fokos kameraszög állással

A felvételeket 60 fps sebességgel 2704x1520 px felbontással és H264-MPEG-4 AVC (part10) (avc1) kódolással készültek. Igyekeztem több, rövidebb felvételt készíteni annak érdekében, hogy az utófeldolgozás során ne kelljen nagy állományokkal dolgoznom. A sorokat a napirány miatt szintén igyekeztem vál-

Képek készítése

togatni. A legnagyobb gondot a gyomos felvételek megkeresése okozta, ugyanis az idei szárazság a gyomfejlődést is akadályozta. A rendelkezésre álló kukorica ültetvény jól előkészített és gondozott talajjal rendelkezett, ami tovább csökkentette a gyomok terjedését.

A felvételek teljes összesítő táblázata az alábbi táblázatban tekinthető meg.

3.1. táblázat. Felvétel összesítő táblázat

fájlnev	méret (MB)	felvétel kelte	hossz	kam.szög	fényviszony	terület	sebesség (km/h)
GH010398.MP4	629	2022.07.02 15:05	01:27	45	napcsütés	főleg gyom	4
GH010437.MP4	598	2022.07.19 18:33	01:21	45	árnyék	gyom	4
GH010415.MP4	272	2022.07.11 18:39	00:37	45	árnyék	gyom	4
GH010436.MP4	596	2022.07.19 18:31	01:23	45	árnyék+bevilanó szebefény	gyom	4
GH010425.MP4	463	2022.07.19 18:17	01:04	45	ellenfény+árnyék	gyom	4
GH010409.MP4	615	2022.07.02 15:39	01:25	45	nap+árnyék	gyom	4
		Összesen:	07:17				
GH010434.MP4	436	2022.07.19 18:28	01:00	80	nap+árnyék	gyom	4
		Összesen:	01:00				
GH010435.MP4	375	2022.07.19 18:29	00:52	45	nap+árnyék	gyom+kukorica	4
GH010424.MP4	241	2022.07.19 18:15	00:33	45	nap+árnyék	gyom+kukorica	4
GH010410.MP4	329	2022.07.11 18:32	00:45	45	nap+árnyék	gyom+kukorica	4
GH010404.MP4	319	2022.07.02 15:28	00:44	45	nap+árnyék	gyom+kukorica	4
GH010403.MP4	333	2022.07.02 15:27	00:46	45	nap+árnyék	gyom+kukorica	4
GH010402.MP4	802	2022.07.02 15:27	01:51	45	nap+árnyék	gyom+kukorica	4
GH010401.MP4	859	2022.07.02 15:24	01:59	45	nap+árnyék	gyom+kukorica	4
GH010400.MP4	504	2022.07.02 15:22	01:10	45	nap+árnyék	gyom+kukorica	4
GH010399.MP4	2100	2022.07.02 15:21	04:59	45	nap+árnyék	gyom+kukorica	4
GH010408.MP4	425	2022.07.02 15:34	00:59	45	napcsütés	gyom+kukorica	4
GH010407.MP4	390	2022.07.02 15:33	00:54	45	napcsütés	gyom+kukorica	4
GH010406.MP4	420	2022.07.02 15:32	00:58	45	napcsütés	gyom+kukorica erősen gyomos	4
GH010405.MP4	457	2022.07.02 15:30	01:07	45	napcsütés	gyom+kukorica erősen gyomos	4
GH010397.MP4	717	2022.07.02 15:01	01:39	45	napcsütés	gyom+kukorica	4
GH010396.MP4	384	2022.07.02 14:59	00:53	45	napcsütés	gyom+kukorica	4
GH010395.MP4	363	2022.07.02 14:58	00:50	45	napcsütés	gyom+kukorica	4
GH010394.MP4	323	2022.07.02 14:56	00:44	45	napcsütés	gyom+kukorica	4
GH010393.MP4	234	2022.07.02 14:54	00:32	45	napcsütés	gyom+kukorica	4
GH010392.MP4	412	2022.07.02 14:51	00:57	45	napcsütés	gyom+kukorica erősen gyomos	4
GH010391.MP4	368	2022.07.02 14:48	00:51	45	napcsütés	gyom+kukorica	4
		Összesen:	24:03				
GH010433.MP4	85,9	2022.07.19 18:27	00:11	80	nap+árnyék	gyom+kukorica	4
GH010432.MP4	249	2022.07.19 18:27	00:34	80	nap+árnyék	gyom+kukorica	4
GH010431.MP4	392	2022.07.19 18:26	00:54	80	nap+árnyék	gyom+kukorica	4
GH010430.MP4	441	2022.07.19 18:25	01:01	80	nap+árnyék	gyom+kukorica	4
GH010429.MP4	304	2022.07.19 18:24	00:42	80	nap+árnyék	gyom+kukorica	4
		Összesen:	03:22				
GH010438.MP4	509	2022.07.19 18:35	01:10	45	ellenfény	kukorica	4
GH010422.MP4	243	2022.07.19 18:14	00:33	45	ellenfény	kukorica	4
GH010421.MP4	306	2022.07.19 18:13	00:42	45	ellenfény	kukorica	4
GH010420.MP4	328	2022.07.19 18:12	00:45	45	ellenfény	kukorica	4
GH010439.MP4	479	2022.07.19 18:36	01:06	45	nap+árnyék	kukorica	4
GH010423.MP4	333	2022.07.19 18:15	00:46	45	nap+árnyék	kukorica	4
GH010419.MP4	490	2022.07.19 18:11	01:08	45	nap+árnyék	kukorica	4
GH010418.MP4	269	2022.07.19 17:40	00:37	45	nap+árnyék	kukorica fiatal állomány	4
GH010413.MP4	299	2022.07.11 18:37	00:41	45	nap+árnyék	kukorica	4
GH010411.MP4	654	2022.07.11 18:34	01:31	45	nap+árnyék	kukorica	4
GH010412.MP4	612	2022.07.11 18:36	01:25	45	nap+árnyék+bevilanó ellenfény	kukorica	4
GH010416.MP4	243	2022.07.19 17:36	00:33	45	napcsütés	kukorica fiatal állomány/elforgatott kép	4
GX010387_1.mp4	1200	2022.06.24 11:08	02:52	45	napcsütés	kukorica	8
GX010387_2.mp4	1160	2022.06.24 11:08	02:46	45	napcsütés	kukorica	10
		Összesen:	16:35				
GH010428.MP4	293	2022.07.19 18:23	00:40	80	nap+árnyék	kukorica	4
GH010427.MP4	507	2022.07.19 18:20	01:10	80	nap+árnyék	kukorica	4
GH010426.MP4	308	2022.07.19 18:18	00:42	80	nap+árnyék	kukorica	4
GH010414.MP4	366	2022.07.11 18:38	00:50	80	nap+árnyék	kukorica	4
		Összesen:	03:22				

Képek készítése

3.2. táblázat. Felvételek összesítése terület fajtánként

kamera szög	terület fajtája	képkockák száma	30. képkockák száma
45	gyomos	29 340	978
80	gyomos	3 600	80
45	gyom+kukorica	83 460	2 782
80	gyom+kukorica	12 120	404
45	kukorica	59 700	1 990
80	kukorica	12 120	404
80	80 fokos összesen	27 840	888
45	45 fokos összesen	172 500	5 750

4. fejezet

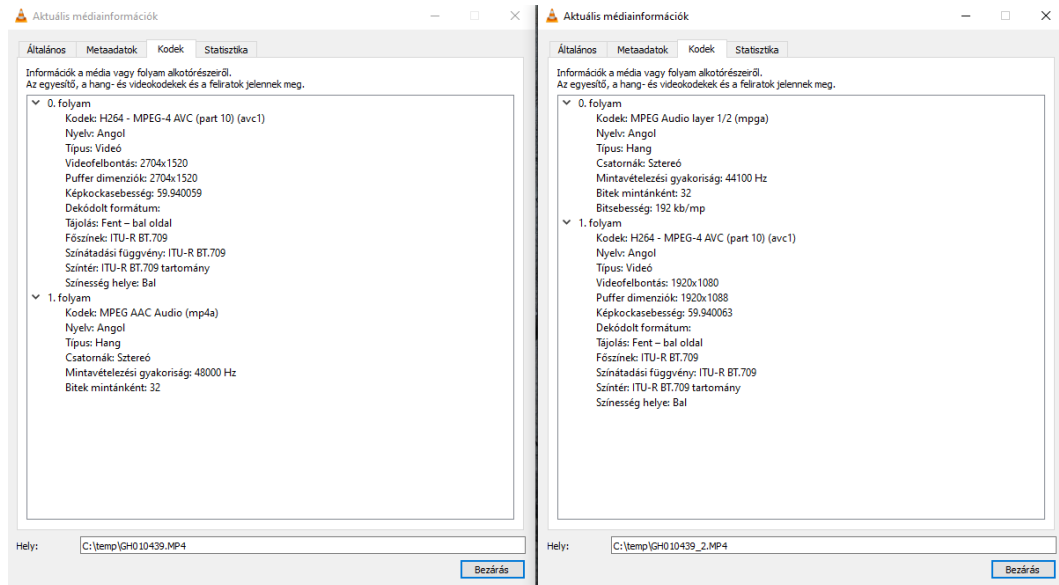
Képi adathalmaz létrehozása

4.1. Képfeldolgozás

4.1.1. Videó felvételek képekre bontása

A videófelvevételek elkészülte után a videóból ki kellett nyerni a képeket. A képkockák egy python program segítségével az opencv felhasználásával kerültek kinyerésre.

A képkockák beolvasása minden felvétel esetén az 54. frame-nél megállt. Megoldást az adta, hogy a videót VLC player segítségével átkonvertáltam a program GUI által nyújtott "VIDEO for MPEG4" formátumba. Ezek után gond nélkül fel tudtam dolgozni az összes felvételt. Ezzel a képkockák mérete 2704x1520 px helyett 1920x1080 px -re változott, de az én szempontomból ez lényegtelen, hiszem maximum 640x480px méretű képkockákon végeztem a vizsgálatokat. A videó átalakítása előtti és utáni média információs ablak így néz ki:



4.1. ábra. Videófelvétel átalakítása után a média információ összehasonlítása

A videófelvételeken általánosságban 52-54 képkocka alatt halad el egy objektum a kép egyik szélétől a másik széléig. Megvizsgálva a felvételeket azt tapasztaltam, hogy 30 képkockánként lementve a frame-ket nem túl nagy ($\sim 10 - 10\%$) átfedéssel információvesztés nélkül lehet a képeket letárolni. Ennek megfelelően a felvételek képkockákra bontása során csak minden 30. felvételt tartottam meg. A kimentett felvételek egyenkénti ellenőrzése és kiválogatása rendkívül munkaigényes feladatnak bizonyult. Öt napot vett igénybe az összes kép átválogatása.

4.1.2. Képek szelektálása

Az átkonvertált videófelvételekből kimentett és átválogatott képekből egy újabb python program segítségével szétválogattam a képeket az alábbiak szerint. Az összes felvétel közül minden 4. képet elkülönítettem. Ezzel a felvételeket 25 %-át félretettem további célra és a 75 % megmaradt tanítási célra. A 25% félretett kép egy ötödét teszt célra válogattam ki. A maradék pedig a validációs képeket alkották. Ezzel az összes képből 75% ment tanító képnek, 20% ment validálásra és 5% ment teszt célra. A szétválogatást végző programkód a függelékben megtalálható.

A felvételeket, ahogyan a videófelvételekben láthattuk külön kategóriák-

ba szedtem szét. Külön könyvtárakba kerültek a csak gyomot tartalmazó ("gyom"), a gyomot nem tartalmazó kukorica felvételek ("gyn") és a gyomot és kukoricát vegyesen tartalmazó felvételek ("gyomos"). A szétválogatás eredményeként létrejöttek az alábbi adathalmazok.

4.1. táblázat. Tiszta kukoricát tartalmazó, gyommentes felvételekből származó képkockák száma és eloszlása

Video ID	Tanító	Validáló	Tesztelő
GH010439	100	26	7
GH010411	129	35	10
GH010412	117	34	9
GH010419	100	28	6
GH010420	69	18	5
GH010421	64	17	4
GH010422	51	13	4
GH010423	70	18	4
GH010424	45	12	3
GH010438	106	29	7
GH010413	0	0	82
összesen	851	230	141

4.2. táblázat. Csak gyomot tartalmazó felvételekből származó képkockák száma és eloszlása

Video ID	Tanító	Validáló	Tesztelő
GH010398	93	25	6
GH010436	119	32	8
GH010409	127	34	9
GH010425	97	26	26
GH010437	121	32	9
GH010415	0	0	69
összesen	557	149	127

4.3. táblázat. Gyomot és kukoricát vegyesen tartalmazó felvételekből származó képkockák száma és eloszlása

Video ID	Tanító	Validáló	Tesztelő
GH010399	201	54	14
GH010400	69	18	5
GH010401	110	32	7
GH010408	84	22	6
GH010407	71	19	5
GH010406	0	0	116
GH010391	48	12	4
GH010392	65	17	5
GH010393	21	6	2
GH010394	41	11	3
GH010395	60	16	4
GH010396	21	22	2
GH010397	67	18	5
GH010405	94	25	7
GH010402	76	20	6
GH010403	53	14	4
GH010404	47	12	4
GH010410	9	2	1
összesen	1137	320	200

5. fejezet

Gyomdetekciós modellek

5.0.1. Tesztek célja

Az előzőekben bemutatott képi adathalmaz elkészítése után azt vizsgáltam, hogy a felvételeket milyen konvolúciós eljárással lehet sikeresen feldolgozni. A feladat elvégzésére több út is állt előttem. Az interneten számtalan tanulmány jelent meg a különféle módszerekről. A legtöbb cikk azt a módszert taglalja, amikor különféle neurális hálózatokkal a gyomot vagy a kultúrnövényt igyekeznek mélytanulós módszerrel fajtánként felismerni és megkülönböztetni [15] [24].

Én egy olyan utat választottam, amely során nem szükséges a gyomnövényeket fajtánként felismerni. Véleményem szerint ez túl sok tanítási folyamatot jelentene, ami a gyomnövények sokféleségét tekintve rendkívül szerteágazó lehet. Ezek miatt a gyakorlat szempontjából nem tartom gazdaságilag kivitelezhetőnek. E helyett azt az utat követtem, amiben elég csak a főbb szóba jöhető szántóföldi növényeket felismerni. (Ami nem az, az mind gyomnak tekinthető.) Magyarországon a legelterjedtebb szántóföldi "kapás" növényünk közül nagyságrendileg kevesebb, mint 10 fajta található meg, míg a gyom populáció lényegesen szélesebb fajtaválasztékot mutat. A dolgozat keretében csak a kukorica felismerését tűztem ki célul. A fentiek alapján tehát elmondható, hogy bináris a feladat (el kell dönten, hogy kukorica vagy gyom van a képen). Az emberek szabad szemmel jól meg tudják különböztetni a kukoricát, ezért kíváncsi voltam arra, hogy ezt egy egyszerű neurális hálózat megoldja-e.

A következő feladat a számítógép különböző szoftver komponenseinek a felkonfigurálása volt.

5.0.2. A dolgozatban használt szoftvertechnológiák

Programozási nyelvnek a Pythont választottam a gyors fejlesztési folyamat érdekében. Igaz ugyan, hogy a programkód végrehajtása rendkívül lassú, de tesztelési feladatokra a kényelmes használata és a bőséges támogatás miatt nagyon megfelel. A különböző feladatok elvégzésére használok Matplotlib-et, Numpy-t, SciPy-t, Scikit-image-t, Pandas-t, Keras-t, Tensorflow-t és OpenCV-t. A következőkben röviden bemutatok néhányat ezek közül.

Fejlesztési környezetként Jupyter Notebook-ot, Anaconda-t és a Google Colab-ot használtam. A Python kód írásához vegyesen használtam Visual Studio Code-ot és Spyder-t.

Numpy

A NumPy nyílt forráskódú kiegészítő csomag a Python programozási nyelvhez, mely a nagy, többdimenziós tömbök és mátrixok használatát támogatja egy nagy magas szintű matematikai függvénykönyvtárral.

SciPy

A SciPy csomag a Scientific Python rövidítése, és ahogyan a neve is utal rá, a természettudományok területét igyekszik lefedni. Ugyanaz a szerzője, mint a NumPy csomagé, számos esetben épít is rá.

Scikit-image

A scikit-image egy nyílt forráskódú képfeldolgozó könyvtár a Python programozási nyelvhez. Algoritmusokat tartalmaz pl. képszegmentáláshoz, geometriai transzformációkhoz, színtér-manipulációhoz, elemzéshez, szűréshez, morfológiához, képi jellemzők detektálásához. Úgy tervezték, hogy jól használható legyen a Python NumPy és SciPy numerikus és tudományos könyvtárakkal.

Pandas

A pandas a Python programozási nyelv egyik programkönyvtára, amely adatok feldolgozására és elemzésére szolgál. Különösen adattáblák és idősorok feldolgozásához szolgáltat megfelelő adatszerkezeteket. Ez a szabad szoftver

a három-záradékos BSD licenc alatt jelent meg. A név a "panel data" angol kifejezésből származik, amely az ökonometria területén azt jelenti, hogy egy egyedről több időperiódusból rendelkezünk adatokkal.

OpenCV

Az OpenCV (Open Source Computer Vision Library) egy ingyenes, nyílt forráskódú függvénykönyvtár, amely a gépi látás megvalósításához szükséges programozási funkciókat tartalmaz. Több platformon és több programozási nyelvvel használható, leggyakrabban Python vagy C++-ban írt programokban alkalmazzák.

Keras

A Keras az egyik vezető magas szintű neurális hálózat API. Pythonban íródott és több háttér neurális hálózati számítási motort támogat.

Tensorflow

A TensorFlow egy ingyenes és nyílt forráskódú szoftverkönyvtár gépi tanulás-hoz és mesterséges intelligenciához. Számos feladatban használható, de különös hangsúlyt fektet a képzésre és a mély neurális hálózatok következtetésére.

5.1. Tensorflow-Keras CNN model

Legelőször Tensorflow környezetben Keras API függvények segítségével végeztem tesztek alap konvolúciós modelleken (CNN). Azért választottam ezt az utat, mert ezzel kapcsolatban bőséges irodalom és leírás található az interneten. [7] [16] [26]

5.1.1. Tensorflow tesztkörnyezet

A képek feldolgozására Tensorflow-t használtam Keras függvények segítségével. Az első tesztek egy laptopra telepített Anaconda környezetben futó Jupyter Notebook segítségével történtek. A szükséges összetevők telepítése után (python, Anaconda, Ipython, Jupyter, pandas) a böngészőben localhost-on elérhetővé válik a jupyter notebook.

5.1.2. CNN alapmodell

Elsőként egy Tensorflow-ban megvalósított CNN alapmodell került kiválasztásra.

A CNN modell két konvolúciós réteget tartalmaz a hozzájuk tartozó max-pooling réteggel. A konvolúció 3x3-as kernellel történik, amiből 64 db van. A kimeneti függvény ReLu. Ezt követi egy Flatten transzformáció, majd két darab dense transzformáció, ami 64 illetve 2 neuront hoz létre. Ez utóbbi a kimenetünk. 640x480-as képméret mellett a gépet annyira leterhelte, hogy időnként lefagyott vagy újraindult. Emiatt a képek méretét felére (320x240) csökkentettem. A végeredmény így nézett ki:

```
model = tf.keras.Sequential([
    tf.keras.layers.Conv2D(64, 3, strides=(1, 1),
                           activation='relu', padding='same',
                           input_shape=[320,240, 3]),
    tf.keras.layers.MaxPooling2D(pool_size=2),
    tf.keras.layers.Conv2D(128, 3, strides=(1, 1),
                           activation='relu', padding='same',
                           ),
    tf.keras.layers.MaxPooling2D(pool_size=2),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dense(1, activation='sigmoid')])
```

5.1. ábra. Keras CNN konvolúciós modell kódja

```
model.summary()
```

```
Model: "sequential_12"
```

Layer (type)	Output Shape	Param #
conv2d_24 (Conv2D)	(None, 320, 240, 64)	1792
max_pooling2d_24 (MaxPooling)	(None, 160, 120, 64)	0
conv2d_25 (Conv2D)	(None, 160, 120, 128)	73856
max_pooling2d_25 (MaxPooling)	(None, 80, 60, 128)	0
flatten_12 (Flatten)	(None, 614400)	0
dense_24 (Dense)	(None, 64)	39321664
dense_25 (Dense)	(None, 2)	130
Total params: 39,397,442		
Trainable params: 39,397,442		
Non-trainable params: 0		

5.2. ábra. Keras CNN konvolúciós modell összegzése

5.1.3. Teszt adathalmaz

A teszt során elsősorban csökkentett mennyiségű képet használtam fel mind-össze két kategóriából. A vizsgált két kategória:

- csak gyomokat tartalmazó fotók,
- tiszta, gyom nélküli kukorica fotók.

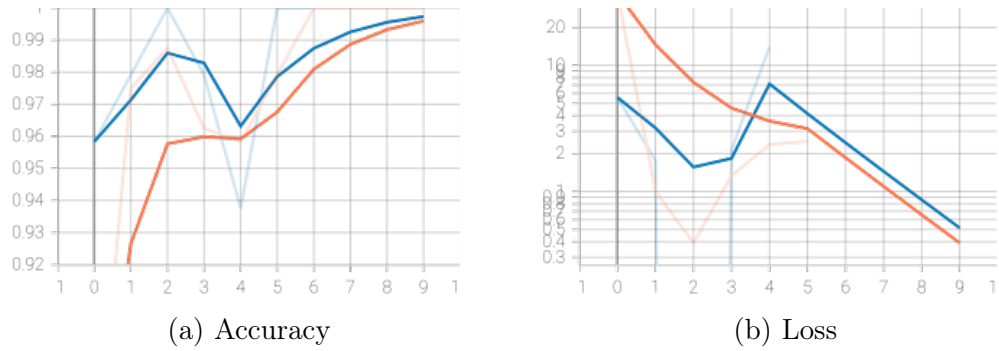
A becsléshez (teszteléshez) használt tesztképek részben azokból a videófelvevételekből kerültek elkülönítésre, amikből a tanításhoz és validáláshoz használtunk fel képeket ("ismert videó"), illetve további 10-10 olyan kép is felhasználásra került, ami teljesen más videófelvevételekből származnak ("új videó"). Máshogyan fogalmazva arra törekedem, hogy olyan videófelvevételből is kerüljön ki tesztkép, amiből korábban nem származott tanító kép. A képek a következő megoszlásban történtek felhasználásra:

5.1. táblázat. Teszteléshez használt csökkentett adathalmaz összetétele

	Training	Validation	Test
Gyom nélküli kukorica kép (Video ID:GH010439,GH010411)	120	30	10
Csak gyomot tartalmazó kép (Video ID:GH010398,GH01436)	120	30	10
Gyom nélküli kukorica kép (Video ID:GH010406)	0	0	10
Csak gyomot tartalmazó kép (Video ID:GH010415)	0	0	10

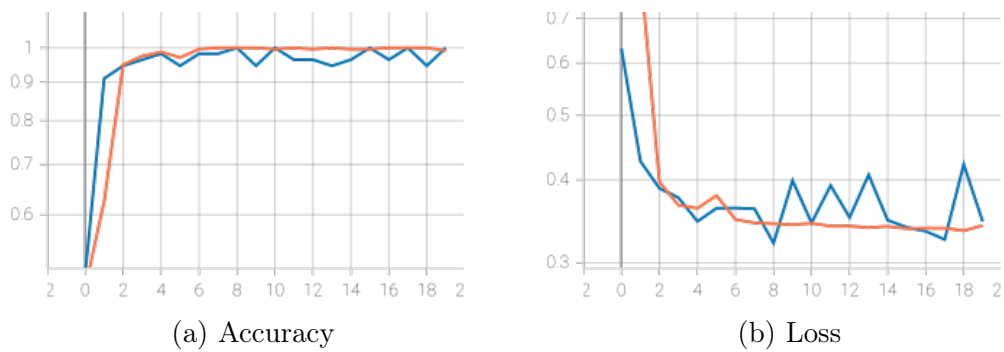
5.1.4. Tanítási tesztek 320x240 felbontású képekkel

10 epoch



5.3. ábra. CNN model 10 epoch 320x240 felbontású 300 tanító képpel
(Training: Orange - Validation: Blue)

20 epoch



5.4. ábra. CNN model 20 epoch 320x240 felbontású 300 tanító képpel
(Training: Orange - Validation: Blue)

5.2. táblázat. Tesztelési halmazon mért pontosság CNN hálózattal

epoch	10	20
gyom nélkül (ismert videóból)	100%	90%
gyom nélkül (új videóból)	100%	100%
gyom (ismert videóból)	90%	100%
gyom (új videóból)	50%	100%
összesen:	85%	98%

5.2. EfficientNet

A tesztjeim közt a következő "off-the-shelf" megoldás az EfficientNet volt.

Az EfficientNet [23] egy konvolúciós neurális hálózati architektúra egyedi skálázási módszerrel, amely egyszerre skálázza a mélység/szélesség/felbontás összes dimenzióját. A hagyományos gyakorlattól eltérően, amely ezeket a tényezőket tetszőlegesen skálázza, az EfficientNet skálázási módszere egységesen, optimálisan próbálja skálázni a hálózat szélességét, mélységét és felbontását rögzített skálázási együtthatókkal.

Az EfficientNetnek alapvetően nyolc változatát lehet elérni "off-the-shelf" módon. Mindegyik változat saját bemenő képfelbontással dolgozik. A hálózat számozása 0-tól 7-ig különbözteti meg egymástól a változatokat. A számozás növekedése együtt jár a hálózat összetettségének a növekedésével is.

5.3. táblázat. EfficientNet különböző változataihoz tartozó képfelbontások

Base model	resolution
EfficientNetB0	224
EfficientNetB1	240
EfficientNetB2	260
EfficientNetB3	300
EfficientNetB4	380
EfficientNetB5	456
EfficientNetB6	528
EfficientNetB7	600

A teszt során az előzőekben ismertetett tanító, validáló és tesztalmazzal készítettem vizsgálatokat.

5.2.1. EfficientNetB0 csökkentett adathalmaz saját súlyokkal

Legelőször saját súlyokkal lett elvégezve a vizsgálat, illetve a modell legfelső rétege is megtartásra került. Ezt a következő sorral lehet megtenni:

```
outputs=EfficientNetB0(include_top=True,weights=None,classes=NUM_CLASSES)
```

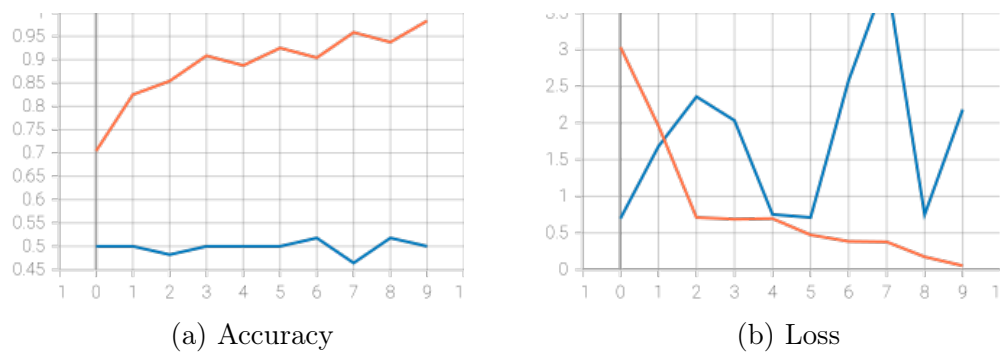
Ez azt jelenti, hogy a modellünk teljes mértékben tanítható és külső, előre beállított súlyokat nem használ a tanulási folyamat során. A Keras függvények használatával azt is meg lehet adni, hogy kimenetként hány neuront használjon, azaz hányféle osztályba sorolja a tanító képeket. Ebben a tesztben a tanító képeken nem alkalmaztam augmentációt. Azaz nem használtam semmiféle elforgatást vagy egyéb módosítást a túltanulás megelőzésére.

Model: "model_1"

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 224, 224, 3)]	0
efficientnetb0 (Functional)	(None, 2)	4052133
Total params: 4,052,133		
Trainable params: 4,010,110		
Non-trainable params: 42,023		

5.5. ábra. EfficientNetB0 transfer learning model

10 epoch képi augmentáció nélkül



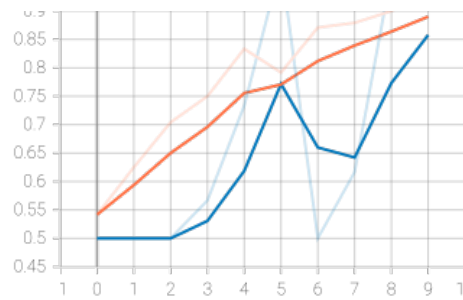
5.6. ábra. EfficientNetB0 10 epoch 224x224 felbontású 300 tanító képpel augmentáció nélkül (Training: Orange - Validation: Blue)

10 epoch képi augmentációval

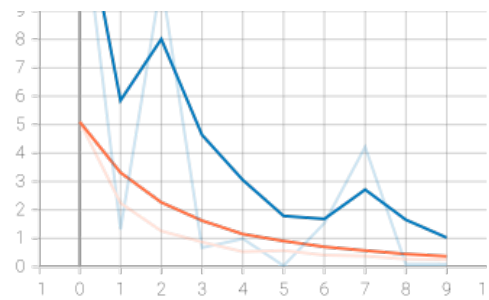
A következő tesztben kipróbáltam a képi augmentáció hatását a tanítóképeken. Az alábbi sorok szerint használtam a különféle Keras által nyújtott "on the fly" augmentációs eljárásokat.

```
train_datagen = ImageDataGenerator( rescale=1.0 / 255,
    rotation_range=40,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
```

```
horizontal_flip=True,
fill_mode="nearest",
)
```



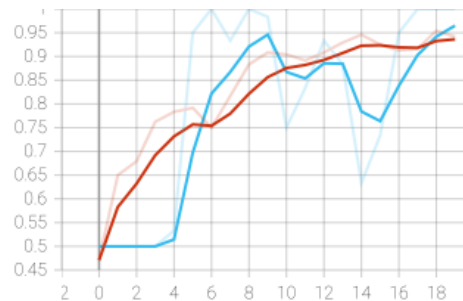
(a) Accuracy



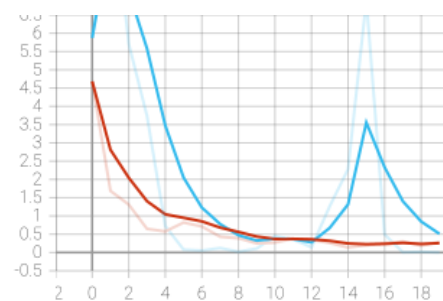
(b) Loss

5.7. ábra. EfficientNetB0 10 epoch 224x224 felbontású 300 tanító képpel augmentáció használatával
(Training: Orange - Validation: Blue)

20 epoch képi augmentációval



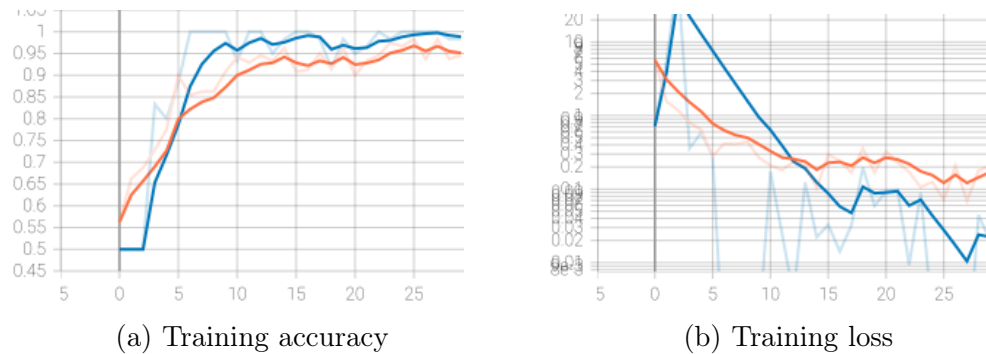
(a) Accuracy



(b) Loss

5.8. ábra. EfficientNetB0 teszt saját súllyal 20 epoch-al 300 tanító képpel augmentáció használatával
(Training: Red - Validation: Blue)

30 epoch képi augmentációval



5.9. ábra. EfficientNetB0 teszt saját súllyal 30 epoch-al 300 tanító képpel augmentáció használatával
(Training: Orange - Validation: Blue)

5.2.2. EfficientNetB0 transfer learning csökkentett adathalmazzal

A következőkben az előre tanított "imagenet" súlyokkal végeztem a tanítást. Továbbá a hálózatot kiegészítettem további rétegekkel. Ahhoz, hogy ezt meg tudjam tenni a legfelső dense réteget le kell tiltani. Ezt a következő sorral lehet megtenni:

```
outputs = EfficientNetB0(include_top=False, weights="imagenet")
```

A teljes kód a függelékben található meg.

Az alábbi ábrán látható a kiegészített rétegekkel készült hálózat:

Model: "sequential_7"

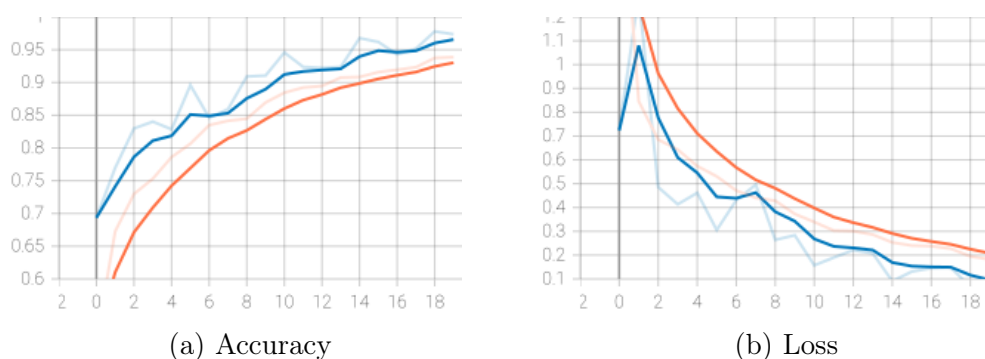
Layer (type)	Output Shape	Param #
efficientnetb0 (Functional)	(None, None, None, 1280)	4049571
gap (GlobalMaxPooling2D)	(None, 1280)	0
dropout_out (Dropout)	(None, 1280)	0
fc_out (Dense)	(None, 2)	2562
input_5 (InputLayer)	multiple	0
Total params: 4,052,133		
Trainable params: 2,562		
Non-trainable params: 4,049,571		

5.10. ábra. EfficientNetB0 model kiegészített rétegekkel

A képek felbontása a B0 változat miatt 224x224-re lett beállítva és továbbra is RGB képek lettek használva. Teszteredmények a következők:

20 epoch

5.2.3. EfficientNetB0 teljes adathalmaz saját súlyokkal



5.11. ábra. EfficientNetB0 teszt saját súllyal 20 epoch-al 300 tanító képpel augmentáció használatával
(Training: Orange - Validation: Blue)

5.3. EfficientNetB0 teljes adathalmaz - Transfer Learning

Az előző lépésekben sikeres teszteket futtattam csökkentett adatkészleten. Bebizonyosodott, hogy a kevés tanító epoch ellenére is képes a hálózat a gyomdetekcióra. A csökkentett adatmennyiség és a kevés epoch jó eredményt hozott az alap CNN hálózaton, de nagy valószínűséggel lényegesen nagyobb adatkészlettel és változatosabb tartalmú képekkel a jóval fejlettebb EfficientNet hálózat jobban teljesítene. Emiatt úgy döntöttem, hogy az előző teszt során használt EfficientNet hálózatot módosítva, teljes adatkészleten futtatom a tanítási folyamatot.

A teszthez az EfficientNetB0 változatát egészítettem ki két fully-connected réteggel a következők szerint:

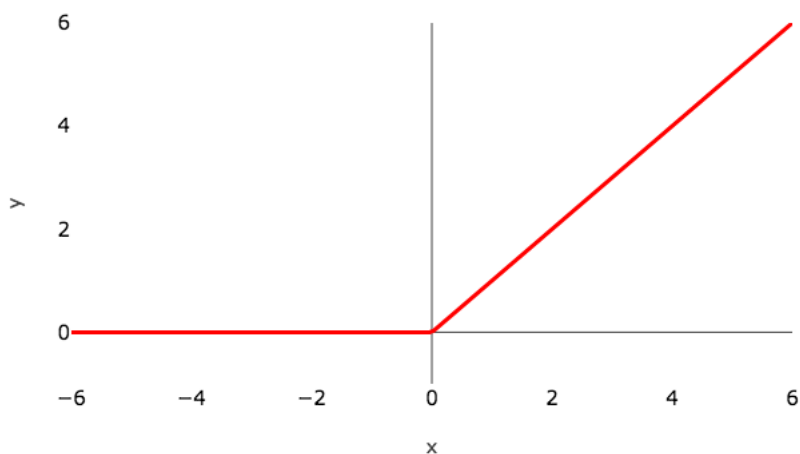
Elsőként az EfficientNetB0 legfelső fully-connected (dense) rétegét elhagytam (include_top=False). Ezután egy BatchNormalization és egy Dropout réteg után két saját dense réteggel egészítettem ki, amihez Swish aktivációt használtam. Legvégül a modell output rétegeként ismét egy fully-connected réteget használtam softmax aktivációval. A leírtak a következőképpen néztek ki:

5.4. táblázat. EfficientNetB0 model kiegészített rétegekkel.

Layer	Type	Shape	Param
input_1 (EfficientNetB0)	(InputLayer)	(None, 224, 224, 3)	0
EfficientNetB0 other layers...	...	...	...
top_activation (EfficientNetB0)	(Activation)	(None,7,7,1280)	0
avg_pool(EfficientNetB0)	(GlobalAveragePooling2D)	(None,1280)	0
—Added layers—	—	—	—
batch_normalization	(BatchNormalization)	(None,1280)	5120
dropout	(Dropout)	(None,1280)	0
dense	(Dense)	(None,512)	655872
batch_normalization_1	(BatchNormalization)	(None,512)	2048
activation	(Activation - swish)	(None,512)	0
dropout_1	(Dropout)	(None,512)	0
dense_1	(Dense)	(None,128)	65664
batch_normalization_2	(BatchNormalization)	(None,128)	512
activation_1	(Activation - swish)	(None,128)	0
dense_2(output)	(Dense,activation="softmax")	(None,3)	387

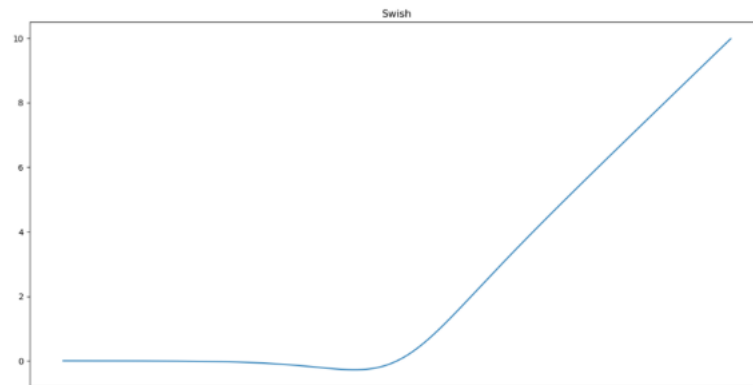
Swish aktivációs függvény

A Swish (A Self-Gated) függvény [20] rövid bemutatásához szükség van először a ReLu (Rectified Linear Unit) függvény ismeretére. A ReLu egy széles körben használt aktiválási függvény, különösen a konvolúciós neurális hálózatoknál. Előnye, hogy könnyen kiszámítható és nem okoz „Vanishing Gradient” problémát [14]. Mindössze az a gond vele, hogy nem nulla központú. (Az aktiváló függvény kimenetének nullánál szimmetrikusnak kell lennie, hogy a gradiensek ne tolódjanak el egy adott irányba.) A “dying ReLU” sajnos előfordulhat, amikor is a kimenet minden negatív bemenetnél nulla. Emiatt egyes csomópontok teljesen elhalnak és nem tanulnak semmit. Egy másik probléma lehet a ReLU-val az aktiválások felrobbanása, mivel a felső határ végtelen. Ez néha használhatatlan csomópontokhoz vezet.



5.12. ábra. ReLu aktiváló függvénygörbe

A Swish ($f(x) = x \cdot \text{sigmoid}(x)$) függvény valamivel jobban teljesít a ReLU-hoz képest. A grafikonja meglehetősen hasonlít a ReLU grafikonjához. Mivel azonban nem változik hirtelen egy ponton (mint a ReLU $x = 0$ esetén), ez megkönnyíti a konvergálást tanítás közben. A Swish hátránya, hogy meglehetősen számításigényes.



$$\begin{aligned} f(x) &= x * \text{sigmoid}(x) \\ &= x * (1 + e^{-x})^{-1} \end{aligned}$$

5.13. ábra. Swish aktiváló függvénygörbe

Teszteredmények az EfficientNetB0 model kiegészített rétegekkel történő vizsgálatával

5.5. táblázat. Hyperparameters

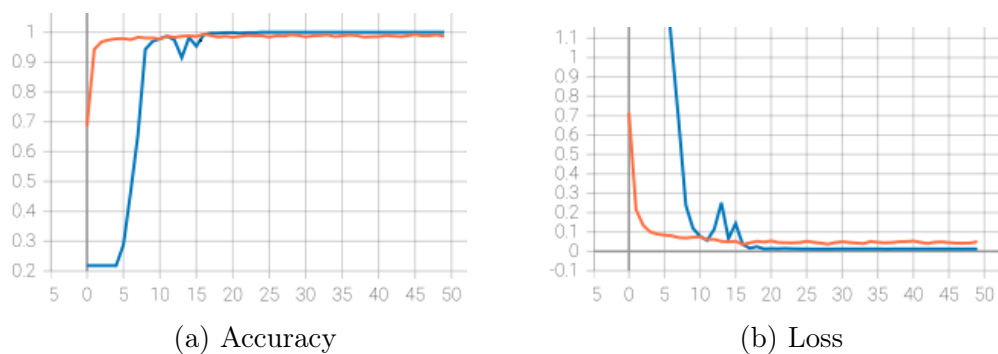
Dropout	0.7 ; 0.5
Network Weight Initialization	Imagenet
Activation function	swish+softmax
Learning Rate	from 0.0001 to 3.8146971692576415e-10
Number of epochs	50
Batch size	32
Training data	2545
Validation data	683
Test data	297

A Keras rendelkezik egy előre gyártott osztálydefinícióval, aminek a segítségével tanítás közben lehet a learning rate nagyságát változtatni, amennyiben a loss értékek nem változik a megadott (patience) epoch után. Ennek a paramétereit láthatjuk az alábbi táblázatban.

5.6. táblázat. ReduceLROnPlateau paraméterek

Kezdeti learning rate (lr)	0.0001
changing factor (lr változtatás mértéke - $\text{new_lr} = \text{lr} * \text{factor}$)	0.5
patience (epoch száma, ami után megváltoztatja az lr-t)	2

A tanítási folyamat után a teszteredmény a következő lett:



5.14. ábra. EfficientNetB0 50 epoch-al 2545 tanító képpel augmentáció használatával
(Training: Orange - Validation: Blue)

Az eredményekből szépen látszik, hogy sikerült egy olyan hálózatot összerakni, aminek a segítségével a gyomdetekció elvégezhető. A kitűzött feladat ezen része megoldottnak vehető (természetesen egy termék kifejlesztéséhez tovább érdemes növelni a tanító és tesztképek halmazát és változatosságát). A következő célként megfogalmazott feladat a képek felesleges részeinek az eltávolítása volt fejlett sordetekcióval, hiszen a kukorica tövek közvetlen közelében van csak szükség permetezésre, a többi helyen a mechanikai tisztítás megoldja a problémát.

6. fejezet

Permetezési terület pontos lehatárolása

6.1. Terület lehatárolásának fontossága

Az előző pontokban ismertetetteknek megfelelően az EfficientNet kiegészített rétegekkel alkalmassá vált, hogy a videófolyamból kinyert képek alapján eldöntse, hogy egy felvétel tartalmaz-e gyomot vagy sem. Mivel a feladat célkitűzésében az szerepel, hogy a kultúrnövények (jelen esetben kukorica) sorában kell detektálnunk a gyomokat és az ezen kívül álló terület (pl. sorközökben) számunkra lényegtelen, ezért fontos további lépés a felvételen szereplő terület pontos lehatárolása. Ennek szemléltetésére alábbiakban láthatunk egy képet.



6.1. ábra. Számunkra a zöld sávval jelölt terület a lényeges

A fenti képen látható, hogy a számunkra releváns területen kívüli rész felesleges „zaj”-ként szerepel a képen. Emiatt szükséges ezen területek kitakarása maszkolási eljárással. Ez azonban további problémákat vet fel. Első lépésben meg kell határozni, hogy a kép mely területe a fontos számunkra. Fel kell ismernünk és meg kell találnunk a kukorica növények sorát. Ha ez megvan, akkor homográfia segítségével meg kell mondanunk, hogy a képen megtalált sorhoz

képest 10 cm-el eltolt párhuzamosok hol találhatók meg a vizsgált képen. A következőkben ezeken a lépéseken megyek végig.

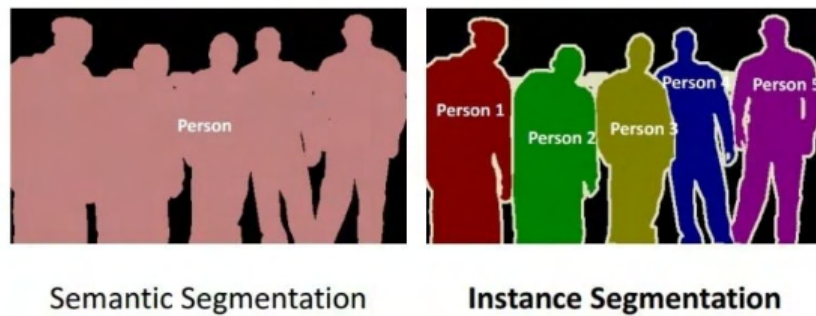
6.1.1. Kukoricaszárak detekciója

Ahhoz, hogy meg tudjuk állapítani a kukoricasorok pontos pozícióját elengedhetetlen, hogy szegmentáljuk a kukorica növényeket. Miután sikeresen szegmentáltuk a növényt, meg kell keresnünk a növények tövét (ahol a földdel érintkeznek). Ezek a pontok fogják adni a sorokat. A szegmentáláshoz Mask R-CNN-t [13] használtam TensorFlow környezetben Keras segítségével. Alábbiakban ezt fogom bemutatni.

Mask R-CNN

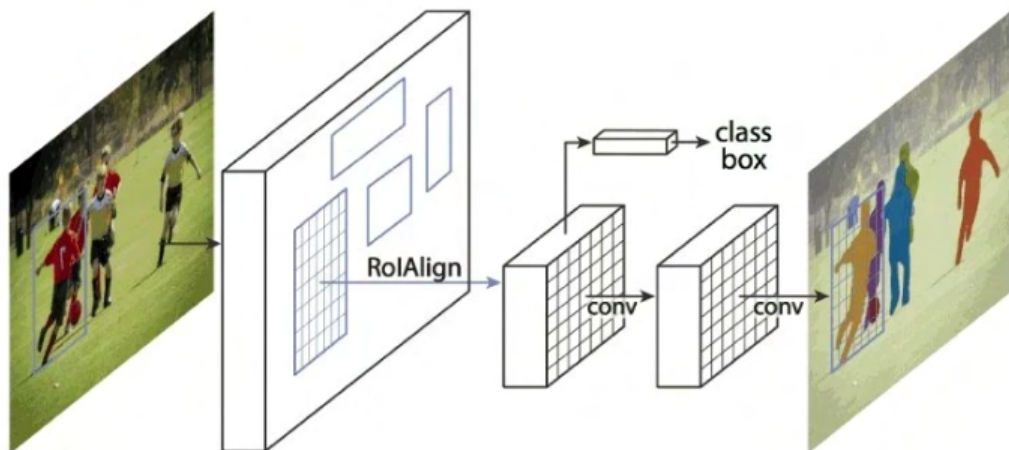
Mi az a Mask R-CNN? ahmedA Mask R-CNN [13] vagy Mask RCNN egy konvolúciós neurális hálózat (CNN), ami jelenleg az egyik legkorszerűbb objektum szintű képszegmentálás (instance segmentation) terén. A Mask R-CNN-t tulajdonképpen a Faster R-CNN nevű [21] Region-Based Convolutional Neural Network-re épülő továbbfejlesztésnek tekinthetjük. A számítógépes látásban az image segmentation nem más, mint az a folyamat, aminek során a pixeleket különböző címkével látjuk el, amelyek megfelelnek objektum típusoknak. Ezt a szegmentációt használhatjuk arra, hogy objektumokat és azok körvonalait megtaláljuk egy képen. A Mask R-CNN alapvetően kétféle szegmentációval foglalkozik. Ezek a Semantic Segmentation és az Instance Segmentation.

- Semantic Segmentation esetén minden egyes pixelt a képen osztályozunk egy fix meghatározott osztálynak megfelelően anélkül, hogy az objektumokat megkülönböztetnénk egymástól. Lényegében az összes egy osztályba sorolható objektumot egyként kezel, nem lehet különbséget tenni a példányok között. Az egyes osztályok szemantikus kategóriáknak felelnek meg.
- Instance Segmentation (Instance Recognition) esetén lehetőség van minden egyes önálló objektum megkülönböztetésére. Ebben a formában ez detekció, objektum lokalizáció és objektum klasszifikáció is egyben.



6.2. ábra. A Semantic és az Instance Segmentation közti különbség (kép forrása: towardsdatascience.com)

A Mask R-CNN a Faster R-CNN-re [21] épül. A Faster R-CNN két kimenettel rendelkezik minden egyes objektum kijelölésére. Egy osztály címke és egy úgynevezett bounding-box, amivel bekeretezi az objektumot. Ezzel szemben a Mask R-CNN-nek van egy harmadik kimenete is az objektum maszk részére. A Mask R-CNN-t lehet a Faster R-CNN kiterjesztésének is tekinteni abban a tekintetben, hogy az osztálycímke és a bounding-box mellé egy sokkal kifinomultabb maszkot is biztosít a felismert objektumok részére.

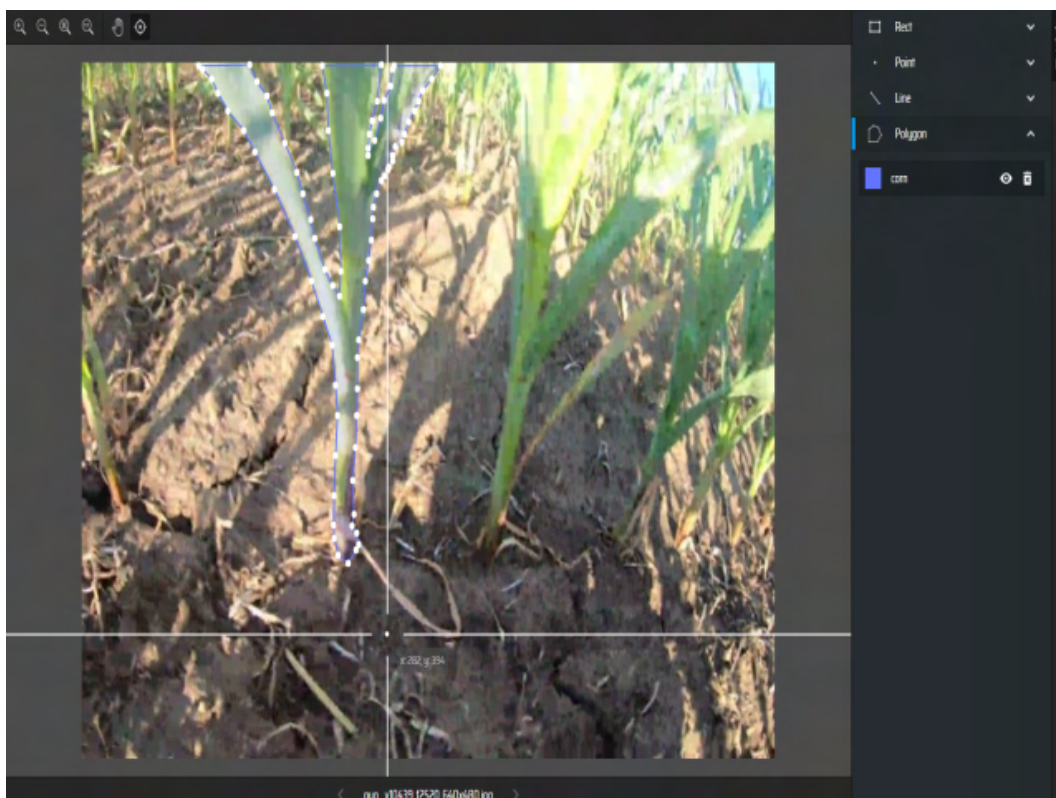


6.3. ábra. Mask R-CNN Instance szegmentációjának durva vázlatos elve (kép forrása: [13])

6.1.2. Képek annotálása

A szegmentáció elvégzésének első lépése ebben az esetben is tesztadat készlet létrehozása volt. A szegmentáció esetében egy célszoftver segítségével ki kell jelölni a tesztképeken azokat az objektumokat, amiket meg szeretnénk tanítani a hálózatnak, valamint az objektumokat címkékkel kell ellátni kategóriák szerint. Erre a célra én a Make Sense nevű ingyenes szoftvert használtam [4]. A szoftver előnye, hogy online használható és nem tárol el a feldolgozott adatokból vagy képekből saját részre semmit sem.

A szoftver segítségével lehetőség van többek közt négyszög és poligon kijelölésre. A kijelölt területhez az említett címkéket kell hozzárendelni. Poligon kijelölés esetén az online tool képes VGG JSON és COCO JSON formátumban menteni az adatokat. Mindkét esetben egyetlen fájl készül az összes kép annotációs adatával. Én a COCO JSON formátumot választottam az annotációhoz.



6.4. ábra. Annotáció a Make Sense online tool-al

6.1.3. COCO JSON formátum

A JSON formátum az alábbi szekciókból áll:

- Info: Általános információk a dataset-ről.
- Lincenses: Esetleges licenszekkel kapcsolatos információk.
- Images: A képek teljes listáját tartalmazza. Nem tartalmazhat Bonding-box-ot, címkét, szegmentációs adatokat. Csak és kizárólag a képekről tartalmaz információkat. Tartalmazza a fájlnevet, a szélesség és magasság adatokat, valamint egy egyedi azonosítót, ami alapján összerendelésre kerül az annotációs adatokkal.

```
"images": [{
    "file_name": "gyn__v10439_f2100_640x480.jpg",
    "height": 480,
    "id": 1,
    "width": 640
}, {
    "file_name": "gyn__v10439_f210_640x480.jpg",
    "height": 480,
    "id": 2,
    "width": 640
}, {
    "file_name": "gyn__v10439_f2160_640x480.jpg",
    "height": 480,
    "id": 3,
    "width": 640
}, {
```

6.5. ábra. COCO JSON formátum "images" szekció

- Categories: Kategóriák felsorolása.
 - id: egyedi kategória azonosító
 - name: kategória megnevezés

```
"categories": [{  
  "id": 1,  
  "name": "corn"  
},  
],
```

6.6. ábra. COCO JSON formátum "categories" szekció

- Annotations: Az annotációról tartalmaz információkat.
 - segmentation : koordináta pontok listája, amelyek az objektum formáját reprezentálják.
 - area : pixelszám (pl. 10px X 20px téglalap alakú terület = 200)
 - iscrowd : 0 esetén egyszerű objektumot reprezentál, 1 esetén több objektumot képvisel.
 - image_id: A képek adatainál megadott azonosító.
 - bbox : bounding box, adatai [top left x position, top left y position, width, height]
 - category_id: Kategória azonosító
 - id: egyedi annotációs azonosító.

```

"annotations": [{
  "area": 21820.8305806477,
  "bbox": [14.322250639386146, 0.6138107416879797,
  "category_id": 1,
  "id": 0,
  "image_id": 1,
  "iscrowd": 0,
  "segmentation": [[130.3324808184143, 323.47826086
65.88235294117644, 120.92071611253199, 29.0537084
, 65.67774936061383, 137.69820971867006, 109.2583
  ], {
    "area": 10923.319444535296,
    "bbox": [292.3785166240409, 0, 92.68542199488496,
    "category_id": 1,
    "id": 1,
    "image_id": 1,
    "iscrowd": 0,
    "segmentation": [[308.9514066496164, 340.05115089
320.613810741688, 0.6138107416879797, 385.0639386
311.4066496163683, 300.76726342711004]]
  }, {

```

6.7. ábra. COCO JSON formátum "annotations" szekció

6.1.4. Mask R-CNN környezet kialakítása

Mask R-CNN tesztelésére és használatára kézenfekvő választás a Matterport féle GIT repository [17]. Ez egy meglehetősen jól dokumentált és sokak által tesztelt összeállítás. A csomag népszerűségét az is jelzi, hogy több mint 11 ezer „forkolás” történt erről a helyről. Legnagyobb hátránya, hogy sajnos csak TensorFlow 1.0 -t támogat. Emiatt az Ahmed Gad által módosított, de nagyon jól karbantartott GIT repository-ját választottam [10], ami egyébiránt elég jól naprakészen támogatja a Tensorflow 2-t. A telepítés során azonban kiderült, hogy mégsem teljesen naprakész minden egyes rész. Szükségessé vált a telepítendő csomagok verzióinak a módosítása, amit a következőkben részletezek.

A Mask R-CNN használatára az Anaconda környezetet választottam. Az Anacondán belül létrehoztam egy önálló virtuális környezetet python 3.7.11-es verzióval. A használatához szükséges volt a pip, numpy és a scipy csomagok importálására. A fent említett repository-t letöltve (akár klónozással) a fájlok

közt található egy requirements.txt fájl, ami felsorolja a szükséges csomagokat a telepítéshez. A fent említett okok miatt az ebben lévő függőségek verziószámait módosítani kellett. Ezek a következők:

- numpy==1.20.3
- scipy==1.4.1
- Pillow==8.4.0
- cython==0.29.24
- matplotlib
- scikit-image==0.16.2
- tensorflow==2.2.0
- keras==2.3.1
- opencv-python==4.5.4.60
- h5py==2.10.0
- imgaug==0.4.0
- IPython[all]

A módosításokhoz a segítséget a hivatkozásban szereplő linken találtam meg [5]. Mindezt azért írtam le, hogy illusztráljam, hogy általában milyen nehézségek jelentkeznek egy-egy modell telepítése során, és ha mindezek nincsenek valahol dokumentálva, akkor több munkanap is elmehet az egyes megfelelő csomagok azonosításával, telepítésével, mire a rendszer működőképes lesz.

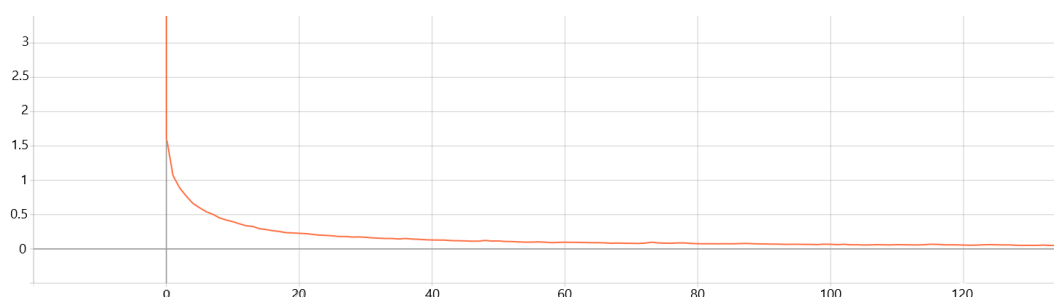
A Mask R-CNN használatához a környezet ezzel teljesen használhatóvá vált.

6.1.5. Mask R-CNN tanítási folyamat és eredményei

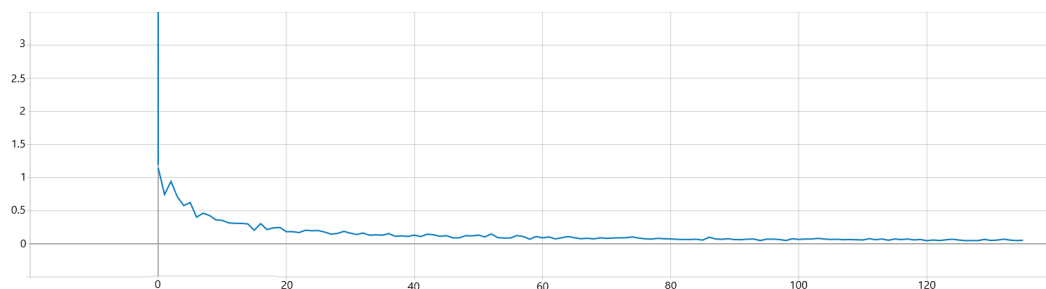
Ebben a fázisban a fő cél az volt, hogy a videófelvevételekből kinyert képeken szegmentálva felismerjem a kukorica növényeket további feldolgozásra.

6.1. táblázat. Hyperparameter-ek a Mask R-CNN szegmentáció tanításhoz

Network Weight Initialization	<i>COCO</i>
Number of epochs	<i>130</i>
Steps per Epoch	<i>100</i>
Training data	<i>50</i>
Validation data	<i>7</i>
Classes (background+classes)	<i>1+1</i>
Learning rate	<i>0.001</i>



6.8. ábra. Mask R-CNN loss értékek 135 epoch tanítás alatt



6.9. ábra. Mask R-CNN loss értékek 135 epoch validáció alatt

6.1.6. Kukorica tövének detekciója

Elkészült az annotált tanítóképek segítségével feltanított Mask R-CNN modellünk. Ezt a tanítás végén elmentettük a következő szegmentálási feladatokhoz. A detekció kódjának alapját a korábban használt forrásból merítettem. Az algoritmus a megadott könyvtárból egyenként kiolvassa a képeket és a model.detect() osztály metódussal elvégzi a szegmentálást. A szegmentálás ered-

ménye egy "dictionary" típusú változóba kerül. A változó különféle dimenziójú tömböket tartalmaz.

6.2. táblázat. Mask R-CNN detect() metódus visszatérési adatszerkezete.

Key	Type	Size
class_ids	Array of int32	(6,)
masks	Array of bool	(480, 640, 6)
rois	Array of int32	(6, 4)
scores	Array of float32	(6,)

class_ids: Ez a tömb tartalmazza a szegmentált objektumok klasszifikációját vektor forájában.

masks: Számunkra ez a legfontosabb adat. A tömb a harmadik dimenzió tengelye (axis) mentén minden egyes felismert objektum részére tartalmaz a kép felbontásával megegyező maszk tömböt. Minden egyes elem egy képpontnak felel meg. Ahol a felismert objektum érvényes pontja szerepel, ott TRUE értéket vesz fel a tömb, egyébként FALSE értékeket tartalmaznak az elemek. Ezzel minden egyes objektumhoz egy bináris maszkot tudunk rendelni.

rois: A Region of Interest, azaz az objektum köré rajzol négyszög koordinátáit adja meg.

scores: A felismert objektumok kategóriájának valószínűségét tartalmazza.



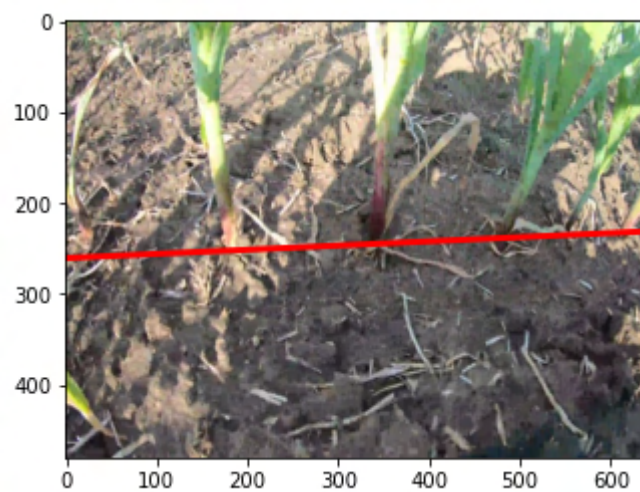
6.10. ábra. Mask R-CNN által szegmentált kukorica növények

A kukorica növények töveinek megtalálásához feltételeztem, hogy az objektum legalsó pixele elegendő közelségben van a növény tövéhez. Ennek érdekében minden egyes „mask” mátrixban meg kellett keresnem a legalsó olyan pixelt, ami "TRUE" értékű volt, ehhez elkészítettem a megfelelő python programot.

6.1.7. Egyenes illesztése a tőpontokra lineáris regresszió segítségével

A következő lépés az egyenes illesztése a megtalált tőpontokhoz (kukorica tövek találkozási pontja a talajjal). Mivel a felismerés és a szegmentálás pontatlanságával számolni kell, valamint a növények sem mindig egyvonalban kelnek ki, ezért nem esnek teljesen egyvonalba a pontok. Ez miatt nem lehet egyszerűen egy egyenessel összekötni a megtalált tőpontokat. A sorok megállapításához egy olyan módszert kellett alkalmazni, ami a legjobban megközelíti a valódi sor egyenesét.

A pontokra legjobban illeszkedő egyenes megtalálásához a lineáris regressziót használtam. A programkódon belül a `scipy` csomag `curve_fit()` függvényét használtam fel. Felmerült, hogy ha előfordulnak hibás detekciók egy-egy képen, célszerűbb lenne a hibatűrőbb RANSAC [9] algoritmus használata. Mivel valós alkalmazásban a kamera mozgása nagyjából folytonos, az egyenes számítását időben átlagolni is lehet, így ezen témakör alaposabb vizsgálata videóelemzést igényelne, ami bár nem túl bonyolult, az időhatárok miatt túlmutat ezen dolgozat keretein.



6.11. ábra. Egyenes illesztése a növények töveihez

6.1.8. Sáveltolás térbeli leképezése

Ezzel addig jutottam el, hogy az objektum felismerés és szegmentálás után sikerült egy egyenest illesztenem azokra a pontokra, amelyek a növények tövét jelentik a képen. A következő lépésben a gyomfelismerés és permetezés szempontjából lényeges terület behatárolása volt a cél (ROI).



6.12. ábra. Kukoricasor és permetezés szempontjából érdekes (ROI) terület

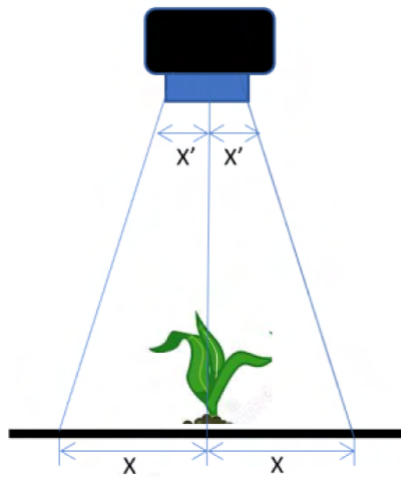
Visszaulva az eredeti koncepcióra, miszerint a célként megfogalmazott technológiában a kultúrnövény vetemény sorközét mechanikai módszerrel kellene gyomtalanítani és csak a növények sorát - azaz a tövek közti területet - kell permetezni, számunkra csak ez a tövek közti terület a lényeges. Ehhez a területhez hozzá kell még venni a mechanikai gyomkezelés során tartott biztonsági zónát a növény sorától.



6.13. ábra. Kukoricasor és a hozzá tartozó eltolt egyenesek

6.1.9. Homográfia

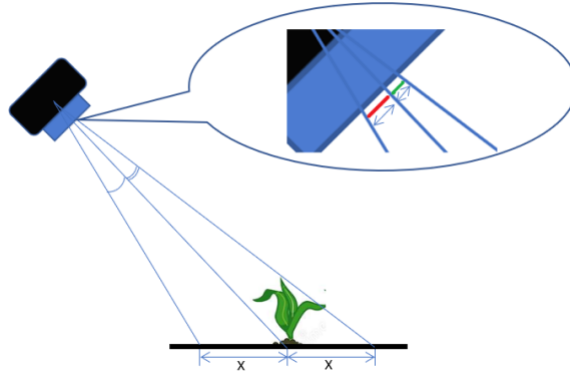
Abban az esetben, ha a kamera érzékelősíkja párhuzamosan lenne elhelyezve a talaj felszínével, akkor egyszerűen az eddig megtalált egyenesre kellene párhuzamost helyezni a ROI meghatározására. Azonban ez nem így van, így a kamera képsíkjának és a felszín egymással bezárt szögének, a kamera optikai leképezésének függvényében (perspektíva hatás) geometriai transzformációkra van szükség.



6.14. ábra. Földfelszínnel párhuzamos kamera képelkötése

A feladat megoldása során azzal a feltételezéssel kellett élnem, hogy a földfelszín állandó szöget zár be a kamera képsíkjával. Ez a gyakorlatban kissé változhat (a talaj helyi egyenetlenségei miatt), de a permetezésben használt technológia (szórófejek szórásképének pontatlansága) miatt az ezzel keletkezett hiba elhanyagolható.

A kamera az általam használt szcenárióban 45 fokos szöget zárt be a földfelszín síkjával. Ez olyan torzítást eredményezett, amiben a kamera érzékelőjéhez közelebb eső ROI pontok pixelenkénti terület lefedettsége kisebb volt, mint az érzékelőtől távolabb eső területről alkotott kép esetében.



6.15. ábra. Földfelszínnel 45 fokos szöget bezáró kamera képalkotása

Egyszerűbben megfogalmazva a kamerához közelebb eső terület egységnyi részéről alkotott kép pixelenként kisebb területet fed le, mint a távolabb eső területhez tartozó pixelek esetén.

Megoldást a homográfia segítségével találtam. Az eljárás lényege, hogy a perspektívikus torzítást egy homográfia mátrix segítségével tudom korrigálni. Az eljárásban a "világ" ismert pontjaihoz hozzá tudok rendelni a képalkotó érzékelőn egy-egy pontot (képi koordinátát). A kettő közti kapcsolatot egy úgynevezett homográfia mátrix segítségével tudom létrehozni, mégpedig egy egyszerű mátrix szorzás segítségével. Az ismert pontok segítségével kiszámolt mátrix-szal azután bármely ugyanazon a síkon elhelyezkedő pont koordinátáját meg tudom határozni.

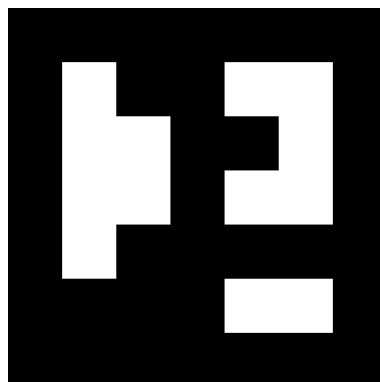
$$s \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = H \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

6.16. ábra. Homográfia mátrix

Azaz egy homográfia kalibrációt kellett elvégeznem. Ehhez, a világi ismert pontok meghatározásához az Aruco kódokat [11] használtam.

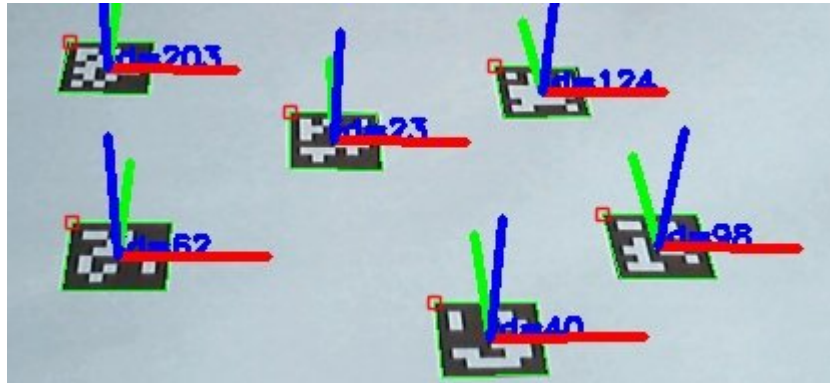
6.1.10. Aruco Marker ismertetése és használata

Az Aruco marker [11] olyan optikai jelölő, amelyet a gépi látásra alkalmas programok használnak, mint például a virtuális valóság, kiterjesztett valóság vagy robot navigáció. A két legfőbb előnye a kiváló képi detektálhatóság és a közvetlen 3D pózbecslés (mivel a kód több részlete is jól behatárolható a képeken). A jelölőkből készleteket lehet regenerálni egy szótár segítségével. A szótár egy olyan lista, melynek elemeit a markerek bináris kódolásával kapjuk. Az Aruco marker négyzet alakú, amely a könnyebb felismerhetőség miatt egy vastag fekete kerettel rendelkezik. Ezen belül egy bináris mátrix helyezkedik el. A bináris mátrix kódolva tartalmazza a marker azonosítóját, ami a kódszótáron belüli sorszámából áll. A marker mérete egyben a belső mátrix mérete is. Az Aruco markerekkel kapcsolatos számításokat az OpenCV függvénykönyvtárain keresztül végeztem el.



6.17. ábra. 5x5 cm-es Aruco marker

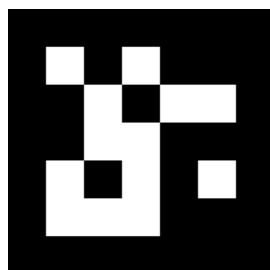
A detektálás során a beépített algoritmus képes a marker formáját és ezzel együtt a perspektivikus torzítását és megállapítani. A mintázat alapján a négyzet alakú marker négy sarkát képes megtalálni. Ezzel és a torzítás mértékével képes a kamera pose-át meghatározni.



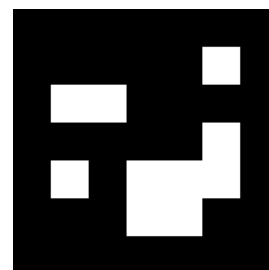
6.18. ábra. Aruco marker pose becslési alkalmazása

6.1.11. ROI meghatározása homográfia segítségével

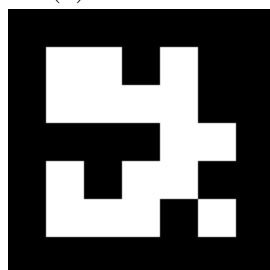
A fentiek ismeretében a ROI terület meghatározására a homográfia eszközeit használtam fel. Az eljárásban a kukorica felvételek készítésekor használt beállításokkal megegyező geometriai beállításokat használva (kamera pose) készítettem tesztfelvételeket Aruco kódokról. Első lépésben az opencv segítségével legyártottam 4 db Aruco markert. Ezek az opencv függvénykönyvtár 5cm X 5cm -es 1000 darabos szótárából kerültek ki 0-3 közti azonosítóval.



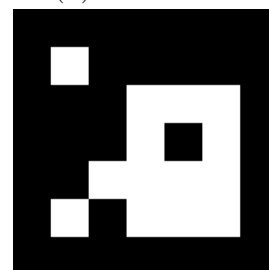
(a) 5x5-1000-0



(b) 5x5-1000-1



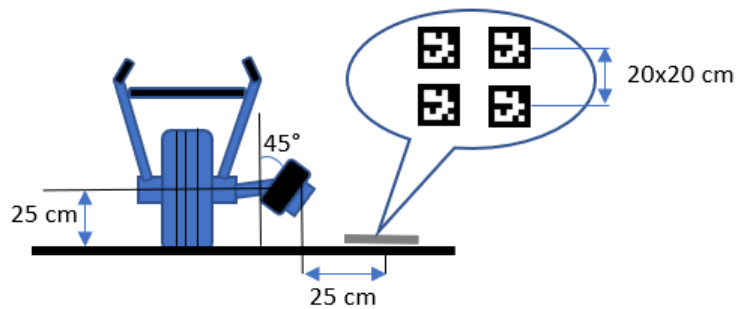
(c) 5x5-1000-2



(d) 5x5-1000-3

6.19. ábra. Pose becslésre használt Aruco marker készlet

Feltételezve a relatív kicsi markerekből adódó detekció pontatlanságát, több marker elhelyezésével, több távolabbi objektumpont kijelölését végeztem el. A markereket négyzet alakban egymástól 20 cm távolságra rögzítettem egy fehér táblán. A négyzetes elrendezésű markereket úgy helyeztem el, hogy a markerek által alkotott sarokpontok képzeletbeli átlóinak metszéspontja a kamerától olyan távolságra legyen, mint a kukoricasorok voltak az állományban (25 cm). (Ez egyébként az elvi számításokban nem fontos, de így az esetleges (radiális, tangenciális) lencsetorzításokkal sem kell számolni, mivel a homográfia meghatározása a kép közepén, a legkevésbé torzított helyen fog történni.)

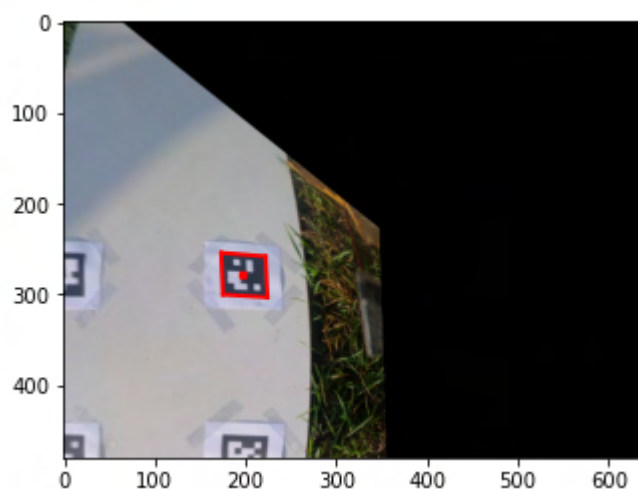


6.20. ábra. Aruco markerek elhelyezése a homográfia mátrix meghatározásához



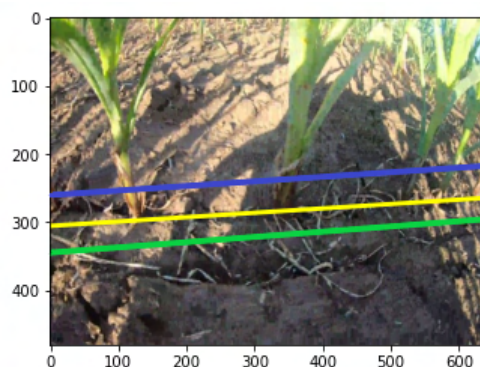
6.21. ábra. Aruco markerek ugyanabból a nézőpontból, mint a kukorica felvételek esetén

A felvételt ezután egy python kód és az OpenCv függvénykönyvtár segítségével dolgoztam fel. Első lépésben az Aruco kódok detektálása történt meg a `detectMarkers()` függvénnyel. Így megkaptam a négy felismert Aruco marker 4-4 sarokpontját. Ebből kiszámoltam a markerek középpontjait. A markerek középpontjainak ismeretében a `findHomography()` függvénnyel kiszámoltam a homográfiás mátrixunkat. A `warpPerspective()` függvénnyel a kép átalakítható olyan perspektívába, ami a kamera érzékelőjével párhuzamos síkban helyezkedik el.



6.22. ábra. Aruco markerek homográfiás átalakítása után a felvétel párhuzamos síkba került az érzékelővel

Az így alkotott kép perspektívikus torzításmentesnek tekinthető. A markerek közti távolság ismeretében könnyen leképezhető az általunk keresett 10 cm-es távolság a képen. A kukorica sor vonatkozásában ezt a homográfiás mátrixot tudtam felhasználni a felvételek leképezésére. Az előzőekben már meghatároztam a felismert kukorica növények töveire illesztett egyenest. Ennek pontjai ismertek számomra. Ezeket a pontokat a mátrix segítségével le tudom képezni a párhuzamos síkra. Ezután ezen a párhuzamos síkon kijelölöm a sorok vonalához képest a 10 cm-es párhuzamos eltolások pontjait és inverz mátrix művelettel kiszámolom a való világban a koordinátákat. Ezzel megkaptam a számomra érdekes terület határvonalait.

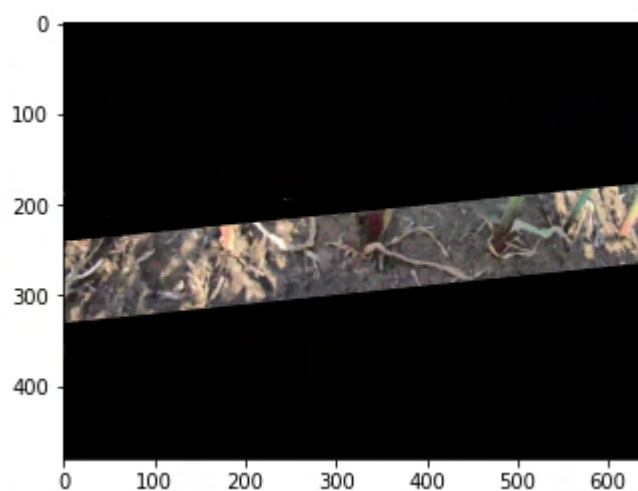


6.23. ábra. A sorokat jelölő középvonal és a ROI határvonalai

6.1.12. Kép maszkolása

Az eddigiek alapján meghatároztam, hogy a kukoricasortól 10-10 cm-re lévő területet kell figyelembe vennem a gyomdetektáláshoz. Ezt neveztük ROI-nak. Gyakorlatban a korábban már említett szórófej szórásképi pontatlanság és az ültetési egyenetlenségek miatt néhány centiméteres hiba nem okoz számottevő problémát. (A szórófej mindenképpen nagyobb területet fog lefedni.) Ezekre alapozva számoltam ki a határvonalak helyét.

Az előzőekben megkapott ROI határvonalak ismeretében az OpenCV fill-Poly() függvényének segítségével aránylag egyszerűen elvégezhető a képek maszkolása.



6.24. ábra. A határvonalak mentén maszkolt kukorica sor

7. fejezet

Összefoglalás és lehetséges továbbfejlesztések

A dolgozat során arra a kérdésre kerestem a választ, hogy egy teljesen új perspektívából megfigyelt kukoricasorban hogyan lehet a gyomdetekciót korszerű számítógépes látás eszközeivel elvégezni. A munkálatokat a vegetációs időszak figyelembevétele miatt meglehetősen hamar kellett kezdeni. Az irodalomkutatási feladatokat júniusban kezdtem el, az első próbafelvételeket pedig júliusban tudtam elvégezni. Emiatt a felvételnépszerűsítésre korlátozott lehetőségeim maradtak. A dolgozat készítése során számtalan python scriptet kellett írnom a különböző részfeladatokhoz (képkockák beolvasása, képek átméretezése, képek kiválogatása...stb.).

- A dolgozat során elkészítettem egy képi adatbázist, amelyet kategorizáltam a gyom jelenléte szerint.
- A képi adatbázissal teszteket végeztem arra vonatkozólag, hogy az egyszerű konvolúciós neurális hálózatokkal, illetve az EfficientNet alap változataival lehetséges-e hatékonyan megkülönböztetni a gyomot tartalmazó felvételeket a gyommentes kukorica felvételektől.
- Egy összetettebb, EfficientNet-re épülő hálózatot készítettem, amivel sikerült megfelelő pontossággal a gyomfelismerést elvégezni.
- A következő lépés annak a vizsgálata volt, hogy milyen módon tudom a számunkra érdekes területet elkülöníteni a képeken, hogy csak azokra fókuszálva történjen a gyomdetekció.
 - Ennek keretében szükségessé vált a kukorica növény szegmentálása.

- Szegmentálás után a növény tövének meghatározása útján a növény sorainak megkeresése.
- A sordetekció után a homográfia segítségével meghatároztam a sorhoz képest azt a területet, ami a gyomdetekció szempontjából érdekes lehet (ROI).
- Ezek után képi maszkolási művelettel sikerült kitakarni a gyomdetekció szempontjából felesleges területet.

7.1. További teendők

Mivel korlátozott lehetőségem volt a felvételek készítésére, ezért a gyomok felismeréséhez lényegesen több, változatos felvételt kell készíteni a biztosan stabil eredmények eléréséhez (pl. por, szennyeződések hatásai). A tapasztalatom szerint az alkalmazott számítási folyamatok meglehetősen számításigényesek. Főleg, ami a Mask R-CNN szegmentációs eljárását illeti. Ezt az esetleges gyakorlati használatához mindenképpen vizsgálni kell. A következő lépések közé tartozik a tesztelt eljárások egy rendszerré építése és annak gyakorlati tesztjei. Sok kérdést vet fel a gyakorlati kivitelezés szempontjából a permetező szelep vezérlése és a kamera elhelyezése. Gondolni kell például a porosodás és egyéb szennyeződés kivédésére. További megfontolást igényel a detektálási folyamat sebessége és a traktor haladási sebességének az összefüggése. Vajon a kamera előrehelyezésével a permetező szelephez képest mekkora időt tudunk nyerni a számítási feladatok elvégzésére a különböző sebességgel haladó traktor esetében?

Ábrák jegyzéke

1.1. Önjáró permetezőgép (kép forrása: horschaustralia.com)	7
1.2. Vontatott permetezőgép (kép forrása: horschaustralia.com) . . .	7
1.3. A legelterjedtebb módszer szerint permetezéskor mindenhova jut vegyszer (kép forrása: www.istockphoto.com)	8
1.4. Szakaszvezérelt permetezés (kép forrása: hardi.com)	9
1.5. Drónnal történő permetezés (kép forrása: droneag.farm)	9
1.6. Számítógépes látáson alapuló permetezés (kép forrása: Bosch) .	10
1.7. Amazone-BASF-Bosch számítógépes látáson alapuló permite- zési megoldása (kép forrása: Amazone)	11
1.8. Sorközművelő kultivátor kijuttató adapterrel, amit AI módsze- rekkel lehetne vezérelni (kép forrása: Farmnet)	12
2.1. Kukorica betegségeket ábrázoló fotók (Kép forrása: [28])	15
3.1. Kukorica sorok szerkezete	17
3.2. Kamera elhelyezése és rögzítése a munkagépen	18
3.3. Traktorra 10 km/h sebességgel készített elmosódott felvétel . .	19
3.4. Traktorra 8 km/h sebességgel készített elmosódott felvétel . . .	19
3.5. Kézikocsira szerelt kamera sematikus rajza 45 fokos kameraszöggel	20
3.6. Kézikocsira szerelt kamera sematikus rajza 80 fokos kameraszöggel	20
3.7. Gyommentes kukorica felvétel 45 fokos kameraszög állással	21
3.8. Gyommentes kukorica felvétel 80 fokos kameraszög állással . . .	21
3.9. Csak gyomot tartalmazó felvétel 45 fokos kameraszög állással . .	22
3.10. Csak gyomot tartalmazó felvétel 80 fokos kameraszög állással . .	22
3.11. Kukoricát és gyomot vegyesen tartalmazó kép 45 fokos kamera- szög állással	23

ÁBRÁK JEGYZÉKE

3.12. Kukoricát és gyomot vegyesen tartalmazó kép 80 fokos kamera- szög állással	23
4.1. Videófelvétel átalakítása után a média információ összehasonlítása	27
5.1. Keras CNN konvolúciós modell kódja	33
5.2. Keras CNN konvolúciós modell összegzése	34
5.3. CNN model 10 epoch 320x240 felbontású 300 tanító képpel (Training: Orange - Validation: Blue)	35
5.4. CNN model 20 epoch 320x240 felbontású 300 tanító képpel (Training: Orange - Validation: Blue)	35
5.5. EfficientNetB0 transfer learning model	38
5.6. EfficientNetB0 10 epoch 224x224 felbontású 300 tanító képpel augmentáció nélkül (Training: Orange - Validation: Blue) . . .	38
5.7. EfficientNetB0 10 epoch 224x224 felbontású 300 tanító képpel augmentáció használatával (Training: Orange - Validation: Blue)	39
5.8. EfficientNetB0 teszt saját súllyal 20 epoch-al 300 tanító képpel augmentáció használatával (Training: Red - Validation: Blue) .	39
5.9. EfficientNetB0 teszt saját súllyal 30 epoch-al 300 tanító képpel augmentáció használatával (Training: Orange - Validation: Blue)	40
5.10. EfficientNetB0 model kiegészített rétegekkel	41
5.11. EfficientNetB0 teszt saját súllyal 20 epoch-al 300 tanító képpel augmentáció használatával (Training: Orange - Validation: Blue)	41
5.12. ReLu aktiváló függvénygörbe	43
5.13. Swish aktiváló függvénygörbe	44
5.14. EfficientNetB0 50 epoch-al 2545 tanító képpel augmentáció hasz- nálatával (Training: Orange - Validation: Blue)	45
6.1. Számunkra a zöld sávval jelölt terület a lényeges	46
6.2. A Semantic és az Instance Segmentation közti különbség (kép forrása: towardsdatascience.com)	48
6.3. Mask R-CNN Instance szegmentációjának durva vázlatos elve (kép forrása: [13])	48
6.4. Annotáció a Make Sense online tool-al	49
6.5. COCO JSON formátum "images" szekció	50

ÁBRÁK JEGYZÉKE

6.6. COCO JSON formátum "categories" szekció	51
6.7. COCO JSON formátum "annotations" szekció	52
6.8. Mask R-CNN loss értékek 135 epoch tanítás alatt	54
6.9. Mask R-CNN loss értékek 135 epoch validáció alatt	54
6.10. Mask R-CNN által szegmentált kukorica növények	56
6.11. Egyenes illesztése a növények töveihez	57
6.12. Kukoricasor és permetezés szempontjából érdekes (ROI) terület	58
6.13. Kukoricasor és a hozzá tartozó eltolt egyenesek	58
6.14. Földfelszínnel párhuzamos kamera képelkötése	59
6.15. Földfelszínnel 45 fokos szöget bezáró kamera képalkotása	60
6.16. Homográfia mátrix	60
6.17. 5x5 cm-es Aruco marker	61
6.18. Aruco marker pose becslési alkalmazása	62
6.19. Pose becslésre használt Aruco marker készlet	62
6.20. Aruco markerek elhelyezése a homográfia mátrix meghatározá- sához	63
6.21. Aruco markerek ugyanabból a nézőpontból, mint a kukorica fel- vételek esetén	64
6.22. Aruco markerek homográfias átalakítása után a felvétel párhü- zamos síkba került az érzékelővel	65
6.23. A sorokat jelölő középvonal és a ROI határvonalai	65
6.24. A határvonalak mentén maszkolt kukorica sor	66

Táblázatok jegyzéke

3.1. Felvétel összesítő táblázat	24
3.2. Felvételek összesítése terület fajtánként	25
4.1. Tiszta kukoricát tartalmazó, gyommentes felvételekből származó képkockák száma és eloszlása	28
4.2. Csak gyomot tartalmazó felvételekből származó képkockák száma és eloszlása	28
4.3. Gyomot és kukoricát vegyesen tartalmazó felvételekből származó képkockák száma és eloszlása	29
5.1. Teszteléshez használt csökkentett adathalmaz összetétele	35
5.2. Tesztelési halmazon mért pontosság CNN hálózattal	36
5.3. EfficientNet különböző változataihoz tartozó képfelbontások	37
5.4. EfficientNetB0 model kiegészített rétegekkel.	42
5.5. Hyperparameters	44
5.6. ReduceLROnPlateau paraméterek	45
6.1. Hyperparameter-ek a Mask R-CNN szegmentáció tanításhoz	54
6.2. Mask R-CNN detect() metódus visszatérési adatszerkezete.	55

Irodalomjegyzék

- [1] Amazone Smart Sprayer. <https://amazone.hu/hu-hu/agritechnica/neuheiten-details/ux-5201-smartsprayer-trailed-sprayer-amazone-1000296>.
- [2] App Store - GPS Speed applikáció. <https://apps.apple.com/hu/app/gps-speed/id344223674>.
- [3] GoPro7 Manual. https://gopro.com/content/dam/help/hero7-black/manuals/HERO7Black_UM_HU_REVA.pdf.
- [4] Make sense online annotation tool. <https://www.makesense.ai/>.
- [5] Dr. Sreenivas Bhattiprolu. Youtube:mask r-cnn install. https://youtu.be/Fu_km7FXyaU.
- [6] Borja Espejo-Garcia, Ioannis Malounas, Nikos Mylonas, Aikaterini Kasi-mati, and Spyros Fountas. Using EfficientNet and transfer learning for image-based diagnosis of nutrient deficiencies. *Computers and Electronics in Agriculture*, 196:106868, 2022.
- [7] Laszlo Fazekas. Tensorflow alapozó. <https://thebojda.medium.com/tensorflow-alapoz%C3%B3-d2d1ee97c9db> (letöltés ideje: 2022.08.04.).
- [8] Constanze Fetting. The European Green Deal, ESDN Report, December 2020, ESDN Office, Vienna. https://www.esdn.eu/fileadmin/ESDN_Reports/ESDN_Report_2_2020.pdf (letöltés ideje: 2022.11.12.).
- [9] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

- [10] Ahmed Gad. Git:ahmedfgad/mask-rcnn-tf2. <https://github.com/ahmedfgad/Mask-RCNN-TF2.git>.
- [11] S. Garrido-Jurado, R. Muñoz-Salinas, F.J. Madrid-Cuevas, and M.J. Marín-Jiménez. Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition*, 47(6):2280–2292, 2014.
- [12] Sebastian Haug and Jörn Ostermann. A crop/weed field image dataset for the evaluation of computer vision based precision agriculture tasks. In *ECCV Workshops*, 2014.
- [13] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.
- [14] Sepp Hochreiter, Yoshua Bengio, Paolo Frasconi, Jürgen Schmidhuber, et al. *Gradient flow in recurrent nets: the difficulty of learning long-term dependencies*. A field guide to dynamical recurrent neural networks, IEEE Press In, 2001.
- [15] Honghua Jiang, Chuanyin Zhang, Yongliang Qiao, Zhao Zhang, Wenjing Zhang, and Changqing Song. Cnn feature based graph convolutional network for weed and crop recognition in smart farming. *Computers and Electronics in Agriculture*, 174:105450, 2020.
- [16] Daniel Johnson. CNN Image Classification in Tensor-Flow with Steps & Examples. <https://www.guru99.com/convnet-tensorflow-image-classification.html> (letöltés ideje: 2022.08.04.).
- [17] Matterport. Git:matterport/mask_rcnn. https://github.com/matterport/Mask_RCNN.git.
- [18] Vasileios Moysiadis, Panagiotis Sarigiannidis, Vasileios Vitsas, and Adel Khelifi. Smart Farming in Europe. *Computer Science Review*, 39:100345, 2021.

- [19] Emerson Navarro, Nuno Costa, and António Pereira. A systematic review of IoT solutions for smart farming. *Sensors*, 20(15):4231, 2020.
- [20] Prajit Ramachandran, Barret Zoph, and Quoc V. Le. Swish: a self-gated activation function. *arXiv: Neural and Evolutionary Computing*, 2017.
- [21] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.
- [22] Rebecca Stone. Image segmentation using color spaces in OpenCV + Python. <https://realpython.com/python-opencv-color-spaces/> (letöltés ideje: 2022.08.03.).
- [23] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pages 6105–6114. PMLR, 2019.
- [24] Tao Tao and Xinhua Wei. A hybrid CNN–SVM classifier for weed recognition in winter rape field. *Plant Methods*, 18(1):29, Mar 2022.
- [25] TechVidvan. Detect Objects of Similar Color using OpenCV in Python. <https://techvidvan.com/tutorials/detect-objects-of-similar-color-using-opencv-in-python/> (letöltés ideje: 2022.08.03.).
- [26] Christian Versloot. machine-learning-articles/how-to-predict-new-samples-with-your-keras-model. <https://github.com/christianversloot/machine-learning-articles/blob/main/how-to-predict-new-samples-with-your-keras-model.md> (letöltés ideje: 2022.08.10.).
- [27] Christian Versloot. Maize whole plant image dataset. <https://zenodo.org/record/1002675#.Y3J4qsuZ0Uk>.
- [28] Tyr Wiesner-Hanks, Ethan L. Stewart, Nicholas Kaczmar, Chad DeChant, Harvey Wu, Rebecca J. Nelson, Hod Lipson, and Michael A. Gore.

Image set for deep learning: field images of maize annotated with disease symptoms. *BMC Research Notes*, 11(1):440, Jul 2018.