



SZAKDOLGOZAT

Schewe Ervin

Üzemmérnök-informatikus

alapképzési szak

Kodolányi János Egyetem

Újmédia Kreatívipari Kar

Informatika Tanszék

**Adatvédelmi rendszer fejlesztése, képrögzítésre alkalmas
mobiltelefonokkal szemben**

*Development of a data protection system against mobile
phones capable of recording images*

Konzulens: Dr. Pitlik László

Készítette: Schewe Ervin (HOTGK2)

Üzemmérnök-informatikus
alapképzési szak

Budapest

2026

Tartalomjegyzék

Kivonat	7
Abstract	7
1 Bevezetés.....	8
1.1 A téma aktualitása	9
1.2 Problémafelvetés	9
1.2.1 Mobiltelefon-alapú fenyegetések elemzése	9
1.2.2 Mobiltelefonok detektálásának kihívásai.....	10
1.3 Célcsoportok.....	10
1.4 Hasznosság	11
1.5 Motiváció.....	15
1.6 A dolgozat célja és felépítése	16
1.6.1 Formai követelmények és jelölések	16
1.6.2 Korlátok	17
2 Szakirodalmi áttekintés	19
2.1 Információbiztonsági alapfogalmak	19
2.1.1 Az információbiztonság céljai	19
2.1.2 A tiltott adatszerzési módszerek áttekintése	20
2.2 Képfeldolgozás alapjai	21
2.2.1 Digitális képek reprezentációja.....	21
2.2.2 Előfeldolgozási lépések	24
2.3 Gépi tanulás és mélytanulás alapjai.....	24
2.3.1 Gépi tanulás fogalma	25
2.3.2 Mélytanulás és neurális hálózatok alapjai.....	26
2.3.3 Konvolúciós neurális hálózatok bemutatása	28
2.4 Objektumfelismerés a számítógépes látásban	29
2.4.1 A számítógépes látás szerepe.....	30

2.4.2 Objektumfelismerési módszerek áttekintése.....	30
2.4.3 YOLO alapú objektumfelismerés	32
2.5 Objektum–attribútum mátrix	34
2.6 A képzés tárgyainak és a szakdolgozati témának kapcsolata	35
2.6.1 Európai civilizáció és identitás	35
2.6.2 A jog szerepe a modern társadalmakban	35
2.6.3 Matematikai alapok.....	36
2.6.4 Adatszerkezetek és algoritmusok.....	36
2.6.5 Operációs rendszerek	36
2.6.6 Programozási alapelvek és módszertanok	37
2.6.7 Hálózatok és számítógép architektúrák	37
2.6.8 Kultúra, sport, munkahelyi jóllét	37
2.6.9 Elektronikus áramkörök.....	37
2.6.10 Az elektronika fizikai alapjai	37
2.6.11 Rendszermodellezés.....	37
2.6.12 Emberi viselkedés és kommunikáció.....	38
2.6.13 Programozás.....	38
2.6.14 Felhasználói interfészek és vizualizáció	38
2.6.15 Vezetési és vállalkozási ismeretek.....	38
2.6.16 Vállalati gazdaságtan	38
2.6.17 Komplex társadalomtudományi ismeretek	38
2.6.18 Adatbázisok.....	39
2.6.19 Szoftverüzemeltetés	39
2.6.20 Rendszertervezés.....	39
2.6.21 Informatikai védelem és biztonság	39
2.6.22 Szoftvertesztelés	39
2.6.23 Innovatív információs és kommunikációs technológiák az IT-biztonság kapcsán	40

2.6.24 Tudásmenedzsment az IT biztonság területén	40
2.6.25 Mesterséges intelligenciák az IT-biztonság területén	40
2.6.26 IT-biztonsági fejlesztések minőség- és projektmenedzsmentje	40
2.6.27 Mentori óra.....	41
2.6.28 Más COCO-orientált projektet tartalmazó szakdolgozat összevetése	41
3 Adatfeldolgozás és gyanúgenerálás módszerei, valamint a védelmi szoftver bemutatása ...	43
3.1 Gyanúgenerálás módszere Objektum-attribútum mátrix alkalmazásával	43
3.1.1 Az Objektum-attribútum mátrix alkalmazás képi adatok esetén	44
3.1.2 Adatok és módszerek az OAM alkalmazásához	44
3.1.3 Adatok előállítása, Image Converter szoftver bemutatása.....	44
3.1.4 Objektumok.....	45
3.1.5 Attribútumok.....	46
3.1.6 Elemzés	47
3.1.7 Eredmények	48
3.1.8 Objektum-attribútum-mátrix elemzése a ChatGPT használatával.....	56
3.2 Adatfeldolgozás és gyanúgenerálás objektumfelismerési modell alkalmazásával.....	58
3.2.1 A gyanúgenerálás folyamata.....	58
3.2.2 Mobiltelefon detektálás megvalósítása	58
3.2.3 Elemzés	59
3.2.4 Eredmények	60
3.3 Módszerek összehasonlítása	60
3.4 Védelmi szoftver tervezése és megvalósítása.....	61
3.4.1 A tervezett védelmi szoftver architektúrája	61
3.4.2 Működési folyamat	61
3.4.3 Rendszerkomponensek bemutatása, implementáció.....	64
3.4.4 Fejlesztői környezet	69
3.4.4.1 Hardver követelmények.....	69

3.4.4.2 Operációs rendszer.....	70
3.4.4.3 Programozási nyelv és futtatókörnyezet.....	70
3.4.4.4 Python könyvtárak.....	70
3.4.4.5 Integrált fejlesztői környezet.....	72
3.4.4.6 Projekt könyvtárszerkezet.....	72
3.4.4.7 Telepítés lépések.....	72
3.5 Tesztelés és esettanulmányok.....	73
3.5.1 Tesztelési környezet.....	73
3.5.2 Teljesítményelemzés.....	73
3.5.2.1 FPS teszt.....	73
3.5.2.2 CPU terhelés teszt.....	74
3.5.3 Kockázatelemzés és korlátok.....	74
3.5.3.1 Hamis pozitív események szemléltetése.....	75
3.5.3.2 Hamis negatív események szemléltetése.....	75
3.6 Forgatókönyvek.....	76
3.7 Teszt eredmények összegzése.....	76
4 Vita.....	76
5 Következtetések.....	78
6 Összefoglalás.....	78
7 Jövőkép.....	79
7.1 Jövőbeli fejlesztési irányok.....	79
7.1.1 Teljesítmény optimalizáció.....	79
7.1.2 GPU gyorsítás.....	79
7.1.3 fejlettebb mélytanulási modellek integrálása.....	80
7.1.4 Saját tanított modell.....	80
7.1.5 Többkamerás környezet támogatása.....	80
7.1.6 Hálózati naplózás fejlesztése.....	80

7.2 Detektáló naplózó és eseménykezelő rendszer fejlesztése	81
7.2.1 Detektáló és naplózórendszer fejlesztése	81
7.2.2 Riasztás és eseménykezelő alkalmazás fejlesztése	82
8 Mellékletek	85
8.1 Rövidítések jegyzéke	85
8.2 Ábrajegyzék	87
8.3 Fogalomjegyzék	88
8.4 Forrás és hivatkozás jegyzék	94
8.5 Forráskódok	96
8.5.1 Az Image To CSV program forráskódja és leírása	96
8.5.2 Objektumfelismerési teszt rendszer forráskódja	99
8.5.3 A Cellphoneblk v1.4 program forráskódja és leírása	100
8.6 Releváns LLM konverzációk és teljes szövege	105
8.6.1 Programkódok dokumentációjának megfogalmazása	106
8.6.2 Szakdolgozati szövegrészek nyelvi pontosítása	106
8.6.3 Egyes informatikai fogalmak rövid magyarázata	107
8.6.4 Fejlesztési környezetekkel kapcsolatos technikai kérdések tisztázása	108
8.6.5 LLM-alapú detektálás fejlesztés YOLO modell alkalmazásával	110

Kivonat

A szakdolgozat célja egy olyan IT biztonsági védelmi szoftver tervezése és prototípusának megvalósítása, amely képes a mobiltelefonokkal történő jogosulatlan adatszerzés felismerésére és kezelésére. A rendszer alapvető funkciói közé tartozik a felhasználói tevékenységek megfigyelése, gyanús helyzetekben riasztás és automatikus beavatkozás végrehajtása, valamint az események naplózása. A kutatás során különböző megközelítések kerülnek vizsgálatra a gyanúgenerálás megvalósítása érdekében. Kezdetben a képpontszintű adatok feldolgozása és elemzése Objektum–attribútum mátrix alkalmazásával, majd bemutatásra kerülnek a korszerű, mélytanulás-alapú objektumfelismerési módszerek, különös tekintettel a YOLO architektúrán alapuló detektáló modellekre. Továbbá a dolgozatban bemutatásra kerül egy prototípus rendszer fejlesztése, amely képes a mobiltelefonok valós idejű detektálására és az események rögzítésére, azonban éles környezetben történő alkalmazásához további fejlesztések szükségesek. A jövőbeli fejlesztési irányok a rendszer hatékonyságának és megbízhatóságának növelésére irányulnak.

Abstract

The aim of the thesis is to design and prototype an IT security protection software that is capable of detecting and handling unauthorized data acquisition using mobile phones. The basic functions of the system include monitoring user activities, issuing alerts and automatic intervention in suspicious situations, and logging events. During the research, different approaches are examined in order to implement suspicion generation. Initially, pixel-level data is processed and analyzed using an Object–Attribute Matrix, and then modern, deep learning-based object recognition methods are presented, with particular attention to detection models based on the YOLO architecture. Furthermore, the thesis presents the development of a prototype system that is capable of real-time detection of mobile phones and recording events, however, further developments are required for its application in a live environment. Future development directions are aimed at increasing the efficiency and reliability of the system.

1 Bevezetés

A digitalizáció előrehaladtával az információs rendszerekben tárolt adatok értéke folyamatosan növekszik, ezzel párhuzamosan pedig a tiltott adatszerzés módszerei is egyre kifinomultabbá válnak.

A diplomamunkám témája, egy olyan IT biztonsági szolgáltatás, illetve védelmi szoftver fejlesztésének a tervezése, amely képes lehet megakadályozni egy esetleges tiltott adatszerzést, egy képalkotásra képes mobileszközzel szemben. Legegyszerűbb és leggyorsabb adatszerzési módszerek egyike a fényképezés. Természetesen a digitális és analóg fényképezőgép, esetleg kamera is egy létező veszélyforrást jelenthet, de jelen esetben csak a képek rögzítésére alkalmas mobiltelefonokkal szembeni védelemre helyezném a hangsúlyt, amelyeken az adat könnyedén feldolgozható és tovább is küldhető.

A cél egy olyan szoftveres megoldás fejlesztése, amely lehetővé teszi a felhasználói tevékenységek megfigyelését és naplózását, valamint jogosulatlan adatszerzés gyanúja esetén riasztást és blokkolást hajthat végre, ezáltal növelve az adatvagyon védelmének hatékonyságát.

A munkám során azt vizsgálom, hogy a birtokunkban lévő adatvagyonból, illetve képekből, milyen módszerekkel tudunk összefüggéseket találni a gyanú generálásához, kezdetben a képpontadatok feldolgozásával, **Objektum-attributum mártix** alkalmazásával. (vö:[3.1 fejezet](#)) Ezt követően kitérek a jelenleg alkalmazott **objektumfelismerés** módszerére, amely a **számítógépes látás** (vö:[2.4 alfejezet](#)) egyik alapvető területe, melynek célja, hogy a számítógépek képesek legyenek azonosítani a képen, vagy videón szereplő objektumokat. Említésre kerül a rendszer alapját képező **gépi tanulás**, illetve a **mély tanulás** alapú képfeldolgozási eljárások, valamint a **konvolúciós neurális hálózat**, amelyre a detektálás folyamata épül. (vö: [2.3.3 fejezet](#))

A bemutatott prototípus szoftver jelenlegi formájában működőképes és alkalmas az elképzelt folyamatok szemléltetésére, azonban éles környezetben történő alkalmazásához további fejlesztések szükségesek, melyek részletezésére a jövőkép fejezetben kerül sor. (lásd. [11.1 fejezet](#))

1.1 A téma aktualitása

Az információbiztonság napjaink egyik kiemelt területe, mivel a vállalati adatvagyon stratégiai jelentőséggel bír. Az adatvesztés és jogosulatlan adateltulajdonítás, nem csupán anyagi károkat okozhat, hanem reputációs veszteséggel és jogi következményekkel is járhat. Míg a klasszikus informatikai támadások – például idegen adathordozók, hálózati behatolások vagy rosszindulatú szoftverek – ellen számos technikai védelem létezik, addig az úgynevezett oldalsó csatornás tiltott adatszerzési módszerek, (vö: [2.1.2 fejezet](#)) gyakran kevésbé kontrolláltak. Ezek a módszerek általában a rendszer működéséből származó, közvetett információkat használják fel az adatok megszerzésére. Bár a megjelenítők lefényképezése hagyományosan nem tartozik az oldalsó csatornás támadások közé, de értelmezésem szerint hasonló jellegű információszivárgást eredményezhet, mivel a rendszerből származó vizuális információt használ fel az adatok megszerzésére.

1.2 Problémafelvetés

A mai modern mobileszközök tulajdonképpen egy „miniszámítógépeknek” tekinthetők, melyek nagy számítási teljesítménnyel, fejlett kamerarendszerrel és folyamatos hálózati kapcsolattal rendelkeznek, ezáltal lehetővé téve a megjelenített információk gyors és diszkrét rögzítését, feldolgozását és továbbítását. Egyetlen fénykép elegendő lehet érzékeny adatok – például üzleti tervek, személyes adatok vagy hitelesítési információk – megszerzéséhez. A probléma különösen kiélezett olyan környezetekben, ahol több felhasználó fér hozzá ugyanahhoz a megjelenítőhöz, illetve, ahol a fizikai kontroll korlátozott.

1.2.1 Mobiltelefon-alapú fenyegetések elemzése

A mobileszközök elterjedése, jelentős hatással van az információbiztonságra. Az okostelefonok ma már szinte minden munkahelyen jelen vannak, és sok esetben elengedhetetlen eszközei a mindennapi kommunikációnak és munkavégzésnek. A mobiltelefon-alapú tiltott adatszerzés különösen nehezen észlelhető, mivel gyakran hétköznapi tevékenységnek tűnő műveletek mögé rejtőzik, például üzenetek olvasása során. A modern információs környezetben ezért egyre fontosabbá válik azoknak a módszereknek a fejlesztése, amelyek képesek felismerni a mobil eszközök jelenlétét és potenciálisan veszélyes használatát. A mobiltelefonok automatikus detektálása hozzájárulhat az információbiztonsági rendszerek hatékonyságának növeléséhez, különösen olyan környezetekben, ahol érzékeny adatok kezelése történik.

1.2.2 Mobiltelefonok detektálásának kihívásai

A mobiltelefonok automatikus felismerése számos technikai kihívást jelent. Az egyik legfontosabb nehézség az eszközök megjelenésének változatossága. A különböző gyártók és modellek eltérő méretű, formájú és színű készülékeket kínálnak, amelyek gyakran tokban, vagy egyéb kiegészítőkkel együtt jelennek meg. Ez jelentősen megnehezíti az egységes felismerést.

További kihívást jelent a mobiltelefonok használatának különböző módja. Az eszköz gyakran részben takart, illetve sok esetben a felhasználó kezében folyamatos mozgásban van. A különböző pozíciók miatt, a felismerő algoritmusnak képesnek kell lennie az objektum különböző nézőpontokból történő azonosítására.

A környezeti tényezők szintén befolyásolják a detektálás pontosságát. A fényviszonyok változása, az árnyékok, valamint a háttérben található hasonló objektumok mind nehezíthetik a felismerést. Emiatt a detektáló algoritmusnak robusztusnak kell lennie, és megbízható működést kell biztosítani különböző környezeti feltételek mellett.

A modern számítógépes látási (vö.:[2.4 fejezet](#)) megoldások, különösen a mélytanulás alapú objektumdetektáló algoritmusok, lehetőséget biztosítanak ezen kihívások kezelésére, mivel képesek nagy mennyiségű tanítóadat alapján megtanulni az objektumokra jellemző vizuális mintázatokat. A számítógépes látás fejlődése lehetővé teszi, hogy képi adatokból magas szintű információkat nyerjünk ki, így az ilyen jellegű tevékenységek automatizált módon felismerhetők.

1.3 Célcsoportok

A detektálására szolgáló védelmi szoftver több célcsoportot is érint, mivel a probléma egyszerre technológiai, szervezeti és társadalmi jellegű. A rendszer elsődleges felhasználói azok a szervezetek és intézmények, ahol érzékeny információk kezelése történik, és ahol a vizuális tiltott adatszerzés reális kockázatot jelent.

A vállalati szektor kiemelt célcsoportot képvisel, különösen azok a szervezetek, amelyek üzleti titkokkal, kutatás-fejlesztési eredményekkel, vagy pénzügyi adatokkal dolgoznak. Az ilyen környezetekben gyakran előfordul, hogy a munkavállalók, ügyfelek vagy látogatók, mobil eszközökkel rendelkeznek, amelyek alkalmasak bizalmas információk rögzítésére.

A **kormányzati és államigazgatási intézmények** szintén jelentős célcsoportot alkotnak, mivel ezekben a környezetekben gyakran kezelnek nemzetbiztonsági vagy egyéb érzékeny adatokat. A tiltott adatszerzés ilyen esetekben súlyos következményekkel járhat, ezért az ilyen rendszerek alkalmazása különösen indokolt.

Az **egészségügyi intézmények** esetében, nagy mennyiségű személyes és különleges adat kerül kezelésre, például betegadatok, vagy diagnosztikai információk. Az ilyen adatok illetéktelen rögzítése komoly adatvédelmi incidensekhez vezethet.

Az **oktatási intézmények** szintén fontos felhasználói körnek tekinthetők. Egyrészt a vizsgák tisztaságának megőrzése, másrészt az intézményi adatok védelme indokolja az ilyen rendszerek alkalmazását. A mobiltelefonok használata vizsgahelyzetben is visszaélésekhez vezethet, amelyeket egy automatizált detektáló rendszer képes lehet azonosítani.

Az **IT-biztonsági szakemberek és rendszergazdák** a rendszer közvetlen kezelői közé tartoznak. Számukra a szoftver egy olyan eszközt jelenthet, amely integrálható a meglévő biztonsági infrastruktúrába, és amely támogatja a fenyegetések felismerését és kezelését. A rendszer konfigurálása, monitorozása és karbantartása ezen szakemberek feladata lehet.

Végül az **általános felhasználók** is a célcsoport részét képezik, bár inkább közvetett módon. A rendszer jelenléte növeli a biztonságérzetet, és hozzájárul az adatok védelméhez. Ugyanakkor fontos szempont, hogy a rendszer működése megfeleljen a jogszabályi előírásoknak és az etikai elvárásoknak, különös tekintettel a megfigyelés és adatkezelés kérdésére.

Összességében a védelmi szoftver célcsoportjai széles spektrumot fednek le, és a rendszer hatékony alkalmazása érdekében, figyelembe kell venni az egyes felhasználói csoportok eltérő igényeit és környezetét.

1.4 Hasznosság

A védelmi szoftver fejlesztése több szempontból is jelentős hasznossággal bír, mind gyakorlati, mind tudományos értelemben.

A szoftver egyik legfontosabb gyakorlati haszna, hogy képes automatizált módon felismerni a potenciális tiltott adatszerzési helyzeteket, ezáltal csökkentve az emberi felügyelet szükségességét.

A rendszer hozzájárulhat a szervezetek biztonsági szintjének növeléséhez is. A vállalatok és intézmények számára lehetőséget biztosít arra, hogy olyan fenyegetéseket is kezeljenek, amelyek eddig nehezen voltak észlelhetők. Ez nemcsak a jogosulatlan adatszerzés kockázatát csökkenti, hanem növeli a biztonságtudatosságot is a szervezeten belül. További előnyt jelent, hogy a megoldás integrálható meglévő biztonsági rendszerekbe, például kamerarendszerekbe, vagy hozzáférés-ellenőrző megoldásokba. Ezáltal egy komplexebb, több rétegű védelmi architektúra alakítható ki, amely hatékonyabban képes kezelni a különböző típusú fenyegetéseket.

Tudományos szempontból a dolgozat hozzájárul a számítógépes látás és az információbiztonság határterületének fejlődéséhez. A vizuális tiltott adatszerzés detektálása egy viszonylag új kutatási irány, amely ötvözi a képfeldolgozást, a gépi tanulást és a biztonsági alkalmazásokat. A bemutatott módszerek és eredmények, alapot szolgáltathatnak további kutatásokhoz és fejlesztésekhez.

Összességében elmondható, hogy a védelmi szoftver nem csak konkrét gyakorlati problémára nyújt megoldást, hanem hozzájárul egy új, intelligens biztonsági megközelítés kialakításához, amely a jövő informatikai rendszereiben egyre nagyobb szerepet kaphat.

A rendszer gazdasági hasznosságának értékeléséhez, szükséges a fejlesztési és üzemeltetési költségek, valamint a potenciális megtakarítások számszerű becslése. Egy mobiltelefon-detektáló rendszer kialakítása több költségelemből áll, amelyek közé tartozik a **hardverbeszerzés**, a **szoftverfejlesztés**, valamint az **üzemeltetés költségei**.

A hardveroldalon a rendszer fejlesztésének megkezdéséhez – amely a későbbiekben a modell tanítását is magában foglalja – rendelkezünk kell egy **webkamerával**, valamint egy megfelelő számítási teljesítménnyel rendelkező **személyi számítógéppel** (PC). A hatékony működés érdekében a számítógép **grafikus processzorral** (GPU) is kell, hogy rendelkezzen, amely jelentősen felgyorsítja a **mélytanulási modellek tanítási** és futtatási folyamatát.

Egy jó minőségű, nagyfelbontású és magas képkockasebességgel (FPS) rendelkező webkamera például Logitech Brio 4K költsége körülbelül 45 000 - 60 000Ft között mozog. A képfeldolgozást és későbbiekben majd a tanítást (learning) is végző, megfelelő videokártyával rendelkező fejlesztői PC ára körülbelül 1 500 000 – 2 500 000 Ft lehet. Így egy ideális, – az

Az üzemeltetés során, figyelembe kell venni az energiafogyasztást is. Egy ilyen rendszerben működő fejlesztői munkaállomás teljesítményfelvétele függ az aktuálisan zajló folyamattól, mivel az átlagos munkavégzés során – kódolásnál, modell futtatásánál – jóval kevesebb, körülbelül 80 –150 watt a felvétel, mint a jóval időigényesebb tanítás közben, mikor a terhelés maximális, azaz 300-420 watt is lehet. A kamera fogyasztása körülbelül 4,5 watt, így a teljes rendszer fogyasztása körülbelül 350 wattot emészt fel óránként.

Ha a rendszerfejlesztés napi tíz órában működik közepes intenzitással, akkor az energiafogyasztás:

$$0,35 \text{ kW} \times 10 \times 15 = 52,5 \text{ kWh/óra}$$

Egy általános irodai munkára, illetve kódolásra való Laptop fogyasztása 60-65watt, amely biztosítja a fejlesztéshez való erőforrásokat is a felhőszolgáltatás által. Ebben az esetben, a hibrid rendszer teljes fogyasztása körülbelül 70 watt-ra csökkenthető, ezáltal az energiafogyasztás:

$$0,07 \text{ kW} \times 10 \times 15 = 10,5 \text{ kWh/óra}$$

Magyarországon a villamosenergia árát átlagosan 70 Ft/kWh értékkel számolva, a fejlesztés időszaka alatt a saját tulajdonú PC esetében az üzemeltetési költség nagyjából 4000 Ft. Amennyiben már egy hatalmas adatkészlettel rendelkező tanítást már felügyelet nélkül, napi 24 órában végezzük, akkor a fogyasztás nagymértékben megnőhet. A „non-stop tanítás 0-24 órában” a fejlesztés időszaka alatt akár 150 kWh is lehet, ami már jelentős tétel, kb. 10 000 – 12 000 Ft feletti összeg is lehet. Ezzel szemben a felhőszolgáltatás használata során az energiaköltség körülbelül. 700-800 Ft-ra tehető.

A becsült költségek összefoglalva:

	Saját tulajdonú PC	Laptop + Google Colab Pro, vagy Pro+
Költségtípus	Becsült érték	
Kamera	45 000 - 60 000Ft	
Számítógép	1 500 000 – 2 500 000 Ft	150 000 – 200 000Ft + 11,75EUR vagy 53,66 EUR/hó
Fejlesztési költség	2 250 000 Ft	
Energiafogyasztás	4 000 – 12 000 Ft	1 700 – 2 300 Ft
<i>Össz. min. fejlesztés költség becslése</i>	3 800 000 Ft	2 450 000 Ft

1. ábra, fejlesztési költségmodell

A teljes kezdeti beruházás minimum költsége így körülbelül 2,5 – 3,8 millió forint nagyságrendű lehet. Egyéb külső periféria, mint billentyűzet, egér és monitor összege nem került meghatározásra. Ha az elkészült és sikeres tesztfolyamatokat is teljesítő rendszer, akár egyetlen adatbiztonsági incidenst is megelőz, amelynek becsült költsége több millió forint is lehet, akkor bármely irányú **beruházás rövid időn belül megtérülhet**.

1.5 Motiváció

A dolgozat motivációját a munkám során szerzett tapasztalatok adták, amelyek során számos információbiztonsági intézkedéssel – például külső adathordozók használatának korlátozásával, vagy e-mailek külső postafiókokba történő továbbításának szabályozásával – találkoztam. Ugyanakkor a képrögzítésre alkalmas mobiltelefon által jelentett kockázatok kezelése jelenleg nem megoldott. Mivel a digitális adatvagyon védelme napjaink egyik kiemelt információbiztonsági kihívása, így indokoltá válik olyan új megközelítések vizsgálata, amelyek képesek ezen típusú fenyegetések felismerésére.

1.6 A dolgozat célja és felépítése

A szakdolgozat célja egy olyan szoftveres megoldás elméleti és gyakorlati megalapozása, amely képes a vizuális megfigyelésre, a gyanús események felismerésére, valamint az ezekre adott automatikus válaszokra. Tehát a cél, az adatok képrögzítésre alkalmas mobiltelefonokkal történő engedély nélküli adateltulajdonításának a megakadályozása egy számítógépen, háttérben futó szolgáltatással, minél kevesebb erőforrásigénnyel, mindez úgy, hogy a célszemély erről ne minden esetben szerezzen tudomást.

A dolgozat bemutatja, a megvalósításhoz szükséges képfeldolgozáshoz kötődő számítógépes látási módszereket és az ezekhez kapcsolódó technológiákat. A téma ismertetése során arra törekedtem, hogy a bemutatott elméleti háttér és a gyakorlati megoldások a terület iránt érdeklődő laikus olvasók számára is minél érthetőbb módon kerüljenek bemutatásra. A dolgozat első része az **információbiztonság** elméleti hátterét mutatja be, ezt követi a **képfeldolgozási** és **objektumfelismerési módszerek** ismertetése, majd egy tervezett rendszer architektúrájának és működésének, valamint az elkészített prototípus szoftver bemutatása.

1.6.1 Formai követelmények és jelölések

A szakdolgozat olvashatóságának és átláthatóságának biztosítása érdekében az alábbi formai követelmények kerültek alkalmazásra:

- Betűtípus:

A főszöveg betűtípusa Times New Roman, 12 pontos betűmérettel, a kódrészek betűtípusa Consolas típusú 10 pontos betűmérettel íródtak. A kereszthivatkozások és ábrák 10 pontos, míg a lábjegyzetek 8 pontos betűmérettel, dőlt formázással kerülnek megjelenítésre.

- Fejezetek és alfejezetek számozása:

A főfejezetek arab számozással kerülnek jelölésre (például: 1., 2., 3.). Az alfejezetek hierarchikus számozást követnek (például: 1.1, 1.1.1, 1.1.1.1). Az átláthatóság megőrzése érdekében, legfeljebb negyedik szint mélységű tagolás került alkalmazásra.

- Hivatkozások és kereszthivatkozások:

A fejezetekre történő hivatkozás, a következő formákban jelenik meg: (vö. 3.5.2 fejezet), (lásd 3.5.2 fejezet) vagy (3.5.2 fejezet). Az ábrákra való hivatkozás az (X. ábra) jelöléssel történik, amelyet az ábra rövid leírása követ. A forráshivatkozások a (Forrás: szerző, cím, év és megtekintés dátuma) formát követik.

- Hangsúlyozás és kiemelés:

Elsősorban a fejezetcímek és alcímek, tovább az alcímeken belül a fontosabb pontok és lépések, valamint az idegen kifejezések, kulcsszavak vagy fontos fogalmak előfordulásakor kerülnek félkövér formázással kiemelésre. Az idézetek dőlt betűtípussal íródtak.

- Rövidítések:

A technikai rövidítések nagybetűkkel jelennek meg (például: CPU, GPU, YOLO).

- Listák és felsorolások:

A pontozott listák szövegfolyamban történő felsorolások esetén, míg a számozott listák a sorrendet vagy lépéseket bemutató felsorolásoknál kerülnek alkalmazásra.

- Ábrák, táblázatok:

Minden táblázat rendelkezik felirattal, amely a táblázat tartalmának rövid, egyértelmű leírását tartalmazza.

1.6.2 Korlátok

A szakdolgozat elkészítése során felmerülő korlátozó tényezők a rendelkezésre álló erőforrások, különösen a megfelelő grafikus processzorral rendelkező videokártya hiánya, melyek behatárolták a nagyobb és pontosabb modellek alkalmazását. Emellett a saját tanítóadatok hiánya is korlátozta a detektálás pontosságának további növelését.

További korlátot jelentett az időkeret, amely nem tette lehetővé egy egyedi, célzottan tanított modell teljes körű kidolgozását és optimalizálását.

Fontos szempont volt továbbá, a rendelkezésre álló terjedelem betartása is. A számítógépes látás és az objektumfelismerés teljes, többrétegű és szerteágazó témakörének részletes

bemutatására nem kerül sor. A dolgozat elsősorban a fejlesztési folyamatra, a legfontosabb fogalmak ismertetésére, valamint a számítógépes látás módszereire – különös tekintettel az objektumfelismerésre – koncentrál. Az algoritmusok matematikai leírása, valamint a különböző modellek mélyebb architekturális részletei nem, vagy csak röviden kerülnek bemutatásra.

2 Szakirodalmi áttekintés

A szakirodalmi áttekintés célja, hogy bemutassa azokat az elméleti alapokat és technológiai megoldásokat, melyek a dolgozat témájának megértéséhez feltétlenül szükségesek. A vizuális tiltott adatszerzés detektálása egy összetett probléma, amely több tudományterület is magában foglal, többek között az **információbiztonságot**, a **képfeldolgozást**, a **gépi tanulást**, valamint a **számítógépes látáshoz** kapcsolódó területeket.

2.1 Információbiztonsági alapfogalmak

Az információbiztonságot és az IT-biztonságot, gyakran egymás szinonimájaként használják, azonban van köztük egy fontos különbség. Még az IT-biztonság vagy más néven **kiberbiztonság**, elsősorban a digitális eszközök, például számítógépek, hálózatok és az azokon tárolt adatok technikai védelmére fókuszál, addig az információbiztonság ennél tágabb fogalom.⁵ „Az információbiztonság azon eljárások, szabályok és eszközök összessége, amelyek célja, az adatok és információs rendszerek védelme a jogosulatlan hozzáférés, felhasználás, közzététel, módosítás vagy megsemmisítés ellen.”⁶

Ennek része a fizikai védelem, – például zárt helyiségek, – a jogi megfelelés – például az adatvédelmi szabályozások, mint a GDPR betartása –, (vö. 2.6.2 fejezet) valamint az emberi tényező kezelése, például a dolgozók biztonság tudatosági képzése is.

2.1.1 Az információbiztonság céljai

„Az információbiztonság alapvető modellje a CIA-triád, amely három kulcsfontosságú elem mentén határozza meg a rendszerek és adatok védelmét. Ez a mozaik szó, angol kezdőbetűkből áll össze és a klasszikus céljai, a bizalmasság (*Confidentiality*), sértetlenség (*Integrity*) és rendelkezésre állás (*Availability*).”⁷ A vizuális tiltott adatszerzés elsősorban a bizalmasságot veszélyezteti, mivel jogosulatlan személyek juthatnak hozzá érzékeny információkhoz.

⁵ André S: *IT-biztonság vs. információbiztonság – mi a különbség?* Dqs global.hu, 2022.11.12.
<https://www.dqsglobal.com/hu/szolgalatas/blog/it-biztonsag-vs.-informaciobiztonsag-mi-a-kueloenbseg> (megtekintve: 2026.04.10.)

⁶ Giesz I: *Szemelvények az információbiztonság területéről*, Egyetemi órai előadás, (2024.01.28.)

⁷ U.o., (2024.01.28.)

2.1.2 A tiltott adatszerzési módszerek áttekintése

Az tiltott adatszerzési módszerek, az egyszerű megtevesztéstől a legkifinomultabb hardveres megfigyelésig terjednek. Az adatszerzés történhet digitális és fizikai úton. „A leggyakoribb módszerek lehetnek:

- **Oldalsó csatornás (Side-channel) támadások;** melyek nem magát a szoftvert, hanem a hardver működésének a fizikai jeleit figyelik, például az időzítést és energiát. Így a műveletek sebességéből, vagy az áramfelvétel ingadozásából a titkos kulcsok visszakódolhatók.
- **Pszichológiai manipuláció (Social Engineering);** A támadók az emberi tényezőket használják ki adathalászati (Phising) módszerekkel, amelyekkel a jelszavakat hamis e-mail címekkel, vagy weboldallal csalnak ki.
- **Szoftveres és hálózati támadások;** melyek közül a legismertebb a Zsarolóvírus (Ransomware), amellyel az adatokat titkosítják, majd „váltásdíjat” követelnek értük, gyakran kiszivárogtatással fenyegetve.
- **Fizikai tiltott adatszerzés** módszernek tekinthető az adathordozók eltulajdonítása, vagy egyéb fizikai úton történő jogosulatlan adateltulajdonítás.
- **Vizuális (optikai) adatszerzés;** melyek közül a legismertebb a *„képernyő megfigyelése rejtett kamerákkal, vagy annak visszatükröződésének keresése a tárgyakon, – például szemüvegen, csillogó tárgyakon, poharakon, ablaküvegen –”*⁸ vagy akár a képernyő lefényképezése, melynek előnye az elkövető számára, az alacsony technikai belépési küszöb, ami azt jelenti, hogy nincs szükség különösebb szaktudásra, másodpercek alatt elvégezhető és nehezen detektálható. A védekezés ezért új megközelítéseket igényel, amelyek túlmutatnak a hagyományos informatikai biztonsági megoldásokon.

⁸ What Techniques Do Hackers Use to Steal Information?, The North East Business Resilience Centre (NEBRC) forrás: <https://www.nebrcentre.co.uk/what-techniques-do-hackers-use-to-steal-information/> (megtekintve:2026.02.08.)

2.2 Képfeldolgozás alapjai

A képfeldolgozási módszerek lehetővé teszik, hogy a számítógépek a képekből, hasznos információkat nyerjenek ki, például felismerjék az objektumokat, mintázatokat azonosítsanak, vagy különböző képi jellemzőket elemezzenek.

A digitális képfeldolgozás alapja a képek matematikai és numerikus reprezentációja, amely lehetővé teszi a képi információk számítógépes manipulációját és elemzését. A képfeldolgozás során, különböző műveletek alkalmazhatók, például **szűrés**, **zajcsökkentés**, **élkiemelés** és **szegmentálás**, amelyek célja a kép minőségének javítása és a további feldolgozási lépések előkészítése. „*A lineáris szűrők számos különböző képi transzformáció végrehajtására alkalmasak*”⁹

A következő fejezetben a digitális képek reprezentációja, valamint a képfeldolgozás során alkalmazott fontosabb előfeldolgozási lépések kerülnek bemutatásra. A fejezet célja, hogy ismertesse azokat az alapfogalmakat és módszereket, amelyek a későbbi számítógépes látás és objektumfelismerési eljárások megértéséhez szükségesek.

2.2.1 Digitális képek reprezentációja

A digitális képek **képpontokból (pixelekből)** álló mátrixként írhatók le, ahol minden képponthez egy adott intenzitás- vagy színérték tartozik. A képpont a digitális képfeldolgozás legkisebb megcímezhető és szerkeszthető egysége. Egy digitális kép valójában az eredeti, folyamatos képi információ mintavételezett változata, mely számos fontos paraméterrel, illetve tulajdonsággal rendelkezik. „*Ezek közül az egyik legismertebb a felbontás, amely a számhalmaz dimenzióit adja meg. Ez a paraméter a kép részletességét nagy mértékben meghatározza, azonban növelésével a kép tárolásához szükséges adatmennyiség is növekszik.*”¹⁰ Minél sűrűbb a mintavételezés, annál több pixel áll rendelkezésre, ami nagyobb felbontást és pontosabb képmegjelenítést eredményez.

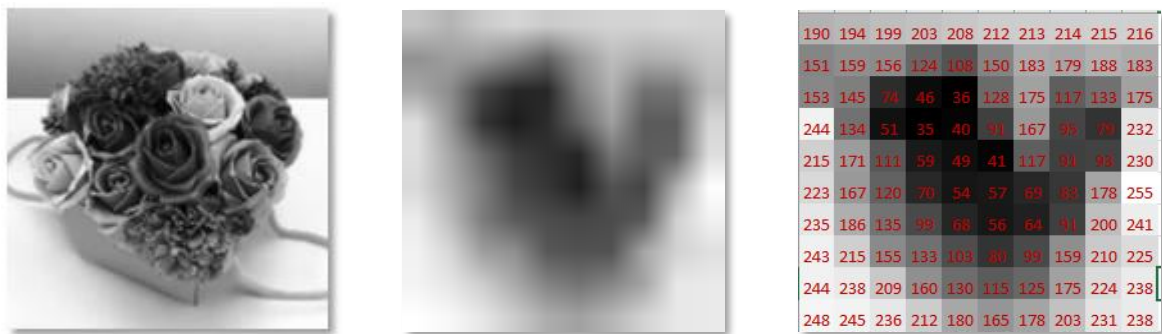
Számos alacsony szintű képfeldolgozó művelet, **szürkeárnyaltos (grayscale)** képeken kerül alkalmazásra. A szürkeárnyaltos képek esetében, minden pixel egyetlen intenzitásértékkel

⁹ Szeliski, R.: *Computer Vision: Algorithms and Applications (2nd Edition)*, (2022, 132.o.)

¹⁰ Szemenyei M. – Varnyú D.: *Deep Learning a Vizualis Informatikában*, BME jegyzet, (2023. 162.o.)

rendelkezik, amely a fényerősséget (intenzitást) határozza meg színinformációk nélkül. A szürkeárnyalatos képekben a pixelértékek **0 és 255 közötti intenzitástartományban** helyezkednek el, ahol a **0 érték a fekete színt, a 255 a fehér színt** jelöli, míg a köztes értékek a különböző szürkeárnyalatokat jelenítik meg. Ezeknek az intenzitásértékeknek a statisztikai és strukturális elemzése lehetőséget biztosít, különböző mintázatok és jellemzők felismerésére.

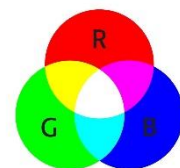
A digitális képek által hordozott információ mennyisége rendkívül nagy lehet. Egy 10×10 pixel méretű szürkeárnyalatos kép esetén, minden pixel 0 és 255 közötti értéket vehet fel, így az összes lehetséges képkombináció száma 256^{100} amely jól mutatja, hogy a digitális képek milyen hatalmas mennyiségű információt tartalmazhatnak.



2. ábra szürkeárnyalatos kép 10x10px-re való átméretezése.

A digitális képek megjelenítésében fontos szerepet játszanak a különböző **színmodellek**, amelyek meghatározzák, hogyan írhatók le és jeleníthetők meg a színek numerikus értékek segítségével. Ilyen színmodellek például az *RGB*, a *CMYK* vagy a *HSB*. Ezek a modellek különböző módszerekkel reprezentálják a látható színtartományt.

Az egyik leggyakrabban alkalmazott színmodell az **RGB (Red, Green, Blue)** modell. Ebben a rendszerben a vörös, a zöld és a kék alapszínek különböző intenzitású kombinációjával állítható elő a színek széles spektruma. Az RGB modell additív színkeverésen alapul, ami azt jelenti, hogy a három alapszín fényének összeadásával jönnek létre az egyes színek. A maximális intenzitás esetén fehér színt kapunk, míg a fény teljes hiánya fekete színt eredményez.

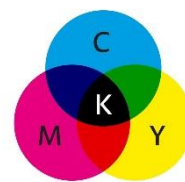


3. ábra, RGB színmodell

11

¹¹ Forrás: Wikimedia Commons – “RGB color model”, CC BY-SA 3.0

A **CMYK (Cyan, Magenta, Yellow, Key/Black)** színmodell a nyomtatásban legelterjedtebben alkalmazott modell. Ebben a rendszerben a **cián**, a **bíbor** és a **sárga** alapszínek kombinációjával állíthatók elő a különböző színek, amelyeket a **fekete** szín egészít ki a mélyebb tónusok és a pontosabb árnyalatok elérése érdekében. A CMYK modell szubtraktív színkeverésen alapul, ami azt jelenti, hogy a festékek a rájuk eső fény egy részét elnyelik (kivonják), és a visszavert fény határozza meg az észlelt színt. Minél több festéket viszünk fel a felületre, annál sötétebb színt kapunk. Ideális esetben a cián, bíbor és sárga színek teljes kombinációja fekete színt eredményezne, azonban a gyakorlatban ez nem tökéletes, ezért van szükség külön fekete festéket is alkalmazni.



4. ábra, CMYK színmodell

12

A **HSB (Hue, Saturation, Brightness)** – más néven **HSV (Hue, Saturation, Value)** – színmodell a színek leírására szolgáló, az emberi színérzékeléshez közelebb álló megközelítést alkalmaz. A modell három összetevő segítségével írja le a színeket. Az árnyalat (Hue), a telítettség (Saturation) és a fényerő (Brightness).



5. ábra, HSN/HSV színmodell

13

Az árnyalat a szín típusát határozza meg, amelyet általában fokban adnak meg a színek mentén (0–360°), például a vörös, zöld vagy kék színek pozíciójaként. A telítettség a szín tisztaságát fejezi ki, ahol az alacsony telítettség szürkés, fakó színt eredményez, míg a magas telítettség élénk, intenzív színt jelent. A fényerő a szín világosságát írja le, amely a teljes sötétségtől (fekete) a maximális világosságig terjed.

A HSB színmodellt elsősorban grafikai alkalmazásokban, képszerkesztő szoftverekben és felhasználói felületek tervezésében használják, mivel intuitív módon teszi lehetővé a színek kiválasztását és módosítását.

¹² Forrás: Wikimedia Commons – “CMYK color model”, CC BY-SA 3.0

¹³ Forrás: Wikimedia Commons – “HSV color space cylinder”, CC BY-SA

2.2.2 Előfeldolgozási lépések

„A képfeldolgozási rendszerekben az előfeldolgozás fontos szerepet játszik, mivel célja a képminőség javítása, a zaj csökkentése, valamint a releváns információk kiemelése.”¹⁴ Az előfeldolgozás során a nyers képadatok olyan módosításokon mennek keresztül, amelyek megkönnyítik a későbbi feldolgozási lépések, például az **objektumfelismerés** vagy **mintázatfelismerés** végrehajtását.

Az előfeldolgozás során alkalmazott leggyakoribb műveletek:

- **méretezés**; a képfeldolgozás egyik alapvető művelete, amely során a kép méretét – azaz felbontását – megváltoztatjuk. Ennek célja lehet a számítási igény csökkentése, a különböző képek egységes méretre igazítása, vagy hogy a kép megfeleljen egy adott algoritmus bemeneti követelményeinek, például a neurális hálózat által megkövetelt bemeneti méretre alakítása a hatékony feldolgozás érdekében.
- **normalizálás**; a képpontok értéktartományának egységesítését szolgálja. A képi adatok értékeit egy előre meghatározott tartományba, jellemzően 0 és 1 közé transzformáljuk, amely elősegíti a neurális hálózatok hatékonyabb és stabilabb működését.
- **szűrés**; a képpontok értékét egy meghatározott szűrő segítségével módosítjuk, figyelembe véve a környező pixelek értékeit. A szűrés célja lehet zajcsökkentés, élkimelés vagy a kép részleteinek kiemelése.
- **éldetektálás** célja, a képen található intenzitásváltozások – azaz élek – azonosítása. Az él, olyan terület a képen, ahol a pixelek intenzitása hirtelen megváltozik, például egy objektum és a háttér határán.

2.3 Gépi tanulás és mélytanulás alapjai

„A gépi tanulás egy olyan módszer, melyben a múltbéli esetek felhasználásával megtanítják a gépeket, a problémák megoldására vagy könnyebb kezelhetőségekre (pl.: elemző képesség, különféle technológiák) és így akár képesek helyettesíteni a tudástechnológust, vagy a szakértőt.”¹⁵ A gépi tanulás és a mélytanulás, a **mesterséges intelligencia** két fontos területe, amelyek lehetővé teszik a számítógépek számára, hogy adatokból tanuljanak és mintázatokat

¹⁴ Szeliski, R.: *Computer Vision: Algorithms and Applications (2nd Edition)*, (2022. 105–110. o.)

¹⁵ MIAU - *Gépi tanulás*, forrás: https://miau.my-x.hu/mediawiki/index.php/G%C3%A9pi_tanul%C3%A1s (2008.02.01)

ismerjenek fel. Ezek a módszerek különösen fontosak a számítógépes látásban, ahol a rendszerek képek és videók elemzésével végeznek különböző feladatokat, például objektumfelismerést vagy képosztályozást.

A gépi tanulás alapját a statisztikai módszerek és a mintafelismerési algoritmusok képezik, amelyek segítségével a rendszer képes általánosítani a tanulási adatokból és új adatokra is alkalmazni a megszerzett tudást. *„A modern alkalmazásokban gyakran mély neurális hálózatokat használnak, amelyek képesek komplex mintázatok felismerésére nagy mennyiségű adatból”*¹⁶.

2.3.1 Gépi tanulás fogalma

A gépi tanulás olyan módszerek összessége, amelyek lehetővé teszik a rendszerek számára, hogy explicit programozás nélkül tanuljanak, azaz nem szükséges pontosan lépésről lépésre meghatározni az algoritmust és a végrehajtandó utasításokat. A rendszer egy tanulási folyamat során képes felismerni a mintázatokat és szabályszerűségeket, majd ezek alapján döntéseket hozni vagy előrejelzéseket készíteni. *„A mintafelismerés célja az adatokban lévő szabályszerűségek automatikus felfedezése, majd ezek használata például osztályozásra.”*¹⁷ A gépi tanulási feladatok több kategóriába sorolhatók.

A leggyakoribb típusok közé tartozik;

- a **felügyelt tanulás**, ahol a tanító adatok tartalmazzák a helyes kimenetet,
- a **felügyelet nélküli tanulás**, amely esetében nincs adott kimenet, hanem a rendszer az adatok struktúráját próbálja felfedezni,
- valamint a **megerősítéses tanulás**, ahol a rendszer jutalmak alapján tanulja meg az optimális döntéseket. A fejlesztő nem azt írja elő, hogyan érje el a célt, hanem csak egy jutalmazási rendszert határoz meg például pontszámot ad a jó lépésért, levonást a rosszért.

¹⁶ I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA: MIT Press, 2016.

¹⁷ C. M. Bishop, *Pattern Recognition and Machine Learning*, (2006, 3.o.)

2.3.2 Mélytanulás és neurális hálózatok alapjai

A **mélytanulás (deep learning)** a gépi tanulás egyik speciális területe, amely többrétegű **neurális hálózatokat** alkalmaz az adatok feldolgozására. A „mély” kifejezés arra utal, hogy a modell több egymásra épülő rétegből áll, amelyek fokozatosan egyre összetettebb jellemzőket tanulnak meg az adatokból.

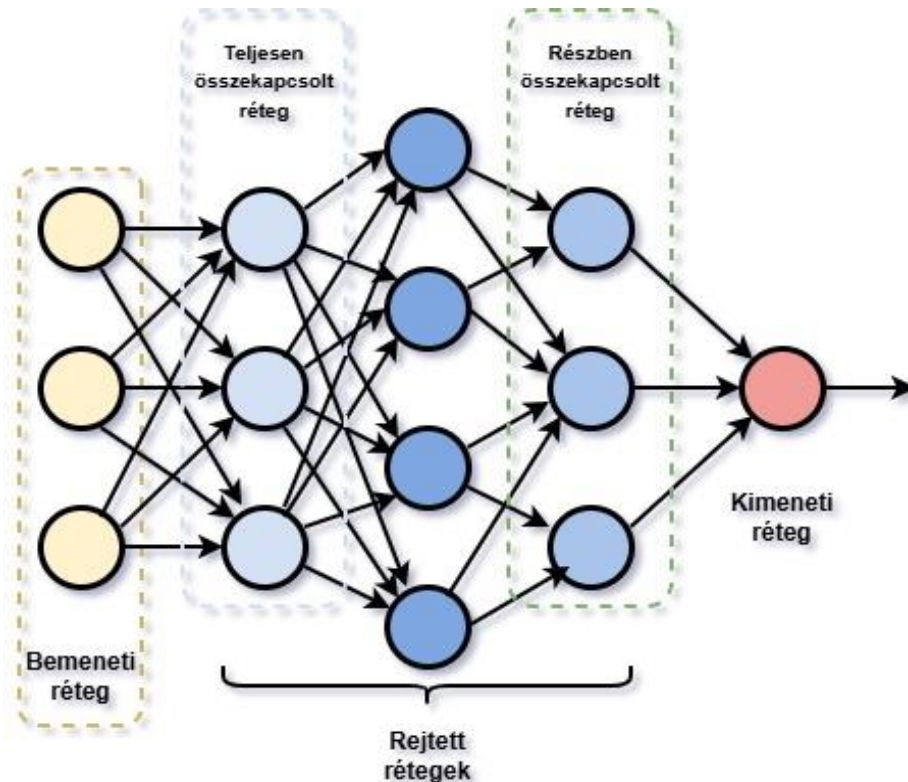
A neurális hálózatok három fő rétegből épülnek fel:

- **Bemeneti réteg (input layer)** fogadja az adatokat,
- **Rejtett rétegek (hidden layers)** az adatok feldolgozását végzik,
- **Kimeneti réteg (output layer)** amely a feldolgozás eredményét szolgáltatja.

A rejtett rétegek egymásra épülő, általában háromnál több – gyakran akár több száz – rétegből állnak, amelyek lehetővé teszik a hierarchikus adatfeldolgozást. A különböző rétegek eltérő szintű jellemzőket tanulnak meg az adatokból. Az első rétegek egyszerű mintázatokat, például éleket vagy textúrákat ismernek fel, míg a mélyebb rétegek összetettebb struktúrákat, például objektumok formáit és jellegzetességeit.

„A neurális hálózatok alapvető működése az emberi agy idegsejtjeinek működését modellezi. A hálózat csomópontokból (neuronokból) és ezek közötti súlyozott kapcsolatokból áll. A tanulási folyamat során a hálózat módosítja ezeket a súlyokat annak érdekében, hogy minimalizálja az előrejelzési hibát.”¹⁸ A neurális hálózatok **súlyai**, olyan numerikus paraméterek, amelyek meghatározzák a bemenetek hatását a kimenetre és a tanulási folyamat során folyamatosan módosulnak a modell pontosságának javítása érdekében.

¹⁸ Szeliski, R.: *Computer Vision: Algorithms and Applications (2nd Edition)*, (2022, 270.o, 291-296o, 315.o.)



6. ábra, Neurális hálózatok rétegei

Legfontosabb rejtett réteg típusok:

- **Teljesen összekapcsolt réteg (Fully Connected / Dense layer);**

Ebben a rétegben minden neuron kapcsolatban áll az előző réteg összes neuronjaival. Általános célú feldolgozásra használják, például osztályozási feladatok végén.

- **Részben összekapcsolt réteg (Partially connected layer);**

A részben összekapcsolt rétegben a neuronok, csak az előző réteg egy részhalmazával állnak kapcsolatban. Ez a megközelítés különösen a konvolúciós neurális hálózatokban jelenik meg, ahol a lokális kapcsolatok lehetővé teszik a hatékony jellemzőkinyerést.

- **Konvolúciós réteg (Convolutional layer);**

Képfeldolgozásban használatos réteg, amely szűrők (kernel) segítségével lokális mintázatokat, például éleket, textúrákat ismer fel. A konvolúciós neurális hálózatok alapját képezi. (vö.:[2.3.3 fejezet](#))

- **Pooling réteg;**

a bemeneti adatok méretét csökkenti, miközben megőrzi a fontos jellemzőket. Segít a számítási igény csökkentésében és a túlilleszkedés (overfitting) elkerülésében. A túlilleszkedés olyan jelenség, amikor a neurális hálózat a tanítóadatok sajátosságait túlzott mértékben megtanulja, így az új, korábban nem látott adatokon gyengébb teljesítményt nyújt.

- **Aktivációs réteg (Activation layer);**

Nemlineáris függvényt alkalmaz, például ReLU, amely lehetővé teszi a hálózat számára komplex mintázatok tanulását.

- **Rekurzív réteg (Recurrent layer – RNN, LSTM);**

Idősoros vagy szekvenciális adatok feldolgozására használják. A Rekurzív réteg figyelembe veszi az előző állapotokat is.

- **Dropout réteg;**

a tanítás során véletlenszerűen „kikapcsol” neuronokat, ezzel csökkentve a túlilleszkedést.

A mélytanulás különösen hatékony olyan feladatokban, ahol nagy mennyiségű adat áll rendelkezésre, például képfeldolgozásban, beszédfelismerésben vagy természetes nyelvfeldolgozásban.

2.3.3 Konvolúciós neurális hálózatok bemutatása

A konvolúciós neurális hálózatok (CNN - Convolutional Neural Network) a neurális hálózatok egy speciális típusát képviselik, amelyeket **kifejezetten képi adatok feldolgozására** fejlesztettek ki. A CNN-ek felépítése három fő rétegtípusból áll össze, amelyek egymásra épülve dolgozzák fel az információt a nyers pixelektől a felismerhető objektumokig.

1. Konvolúciós réteg (Convolutional Layer):

A konvolúciós réteg, kis méretű tanulható szűrőket (kernelek) alkalmaz, amelyek végig haladva a képen, különböző jellemzőket, például éleket, sarkokat vagy mintázatokat nyernek ki, majd matematikai szorzásokkal (diszkrét konvolúcióval) létrehoz egy **jellemzőteret (feature map)**.

2. Összevonó réteg (Pooling Layer):

A konvolúció után általában egy mintavételezési lépés következik, melynek feladata, csökkenteni az adatok méretét (felbontását) és a számítási igényt. A pooling rétegek a fontos információk megtartása mellett csökkentik az adatok méretét és növelik a modell robusztusságát a kisebb eltolásokkal vagy torzulásokkal szemben. A leggyakoribb a Max Pooling, ami egy kis ablakon belül csak a legnagyobb értéket tartja meg, a többit eldobja. Ezzel a hálózat „érzékletlenebbé” válik az eltolódásokra, azaz „ha egy tárgy kicsit arrébb van a képen, még felismeri”.

3. Teljesen összekapcsolt réteg (Fully Connected Layer)

A Teljesen összekapcsolt réteg, a hálózat végén található, melynek feladata az osztályozás. Az előző rétegekből kinyert jellemzőket egy hosszú számsorrá (vektorra) alakítja és minden bemenetet minden kimenettel összeköt (lásd. a hagyományos neurális háló), majd meghatározza, hogy milyen objektum található a képen, és megadja ennek valószínűségét.

4. Aktivációs függvény (Activation Function)

Bár az aktivációs függvény nem tekinthető önálló rétegnek, minden konvolúciós művelet után alkalmazásra kerül. *„A leggyakrabban használt aktivációs függvény a ReLU, amely elősegíti a nemlineáris kapcsolatok modellezését és a releváns jellemzők kiemelését. Enélkül a hálózat csak lineáris leképezések sorozatából állna, és nem lenne képes összetett mintázatok felismerésére.”*¹⁹

A konvolúciós neurális hálózatok, alapvető szerepet játszanak számos modern számítógépes látási alkalmazásban, például a képfelismerés, objektumdetektálás és az arcfelismerés területén.

2.4 Objektumfelismerés a számítógépes látásban

Az objektumfelismerés a számítógépes látás egyik legfontosabb feladata, amelynek célja a képeken vagy videókon található objektumok azonosítása és lokalizálása. A rendszer nemcsak azt határozza meg, hogy milyen objektumok találhatóak, hanem azt is, hogy azok hol helyezkednek el a képen.

¹⁹ Szeliski, R.: *Computer Vision: Algorithms and Applications (2nd Edition)* (2022, 272.o.)

2.4.1 A számítógépes látás szerepe

A számítógépes látás célja, hogy a számítógépek képesek legyenek a vizuális információ értelmezésére, feldolgozására és döntések meghozatalára.

A számítógépes látási rendszerek különböző algoritmusok segítségével képesek felismerni objektumokat, detektálni mozgást, vagy akár teljes jeleneteket értelmezni. A vizuális információ feldolgozása több lépésből állhat. Az első lépés általában a kép előfeldolgozása, amely során a rendszer csökkenti a zajt vagy normalizálja a képet. Ezt követi a jellemzők kinyerése, amely során a rendszer azonosítja a kép fontos vizuális elemeit, például éleket, textúrákat vagy alakzatokat. A végső lépésben egy tanulási algoritmus segítségével történik az objektumok felismerése, vagy osztályozása.

„A számítógépes látás számos modern technológia alapját képezi. Például az autonóm járművek, kamerák segítségével érzékelik a környezetet, a mobiltelefonok, biztonsági berendezések, arcfelismerő rendszereket alkalmaznak a felhasználók azonosítására, míg az ipari rendszerek, kamerák segítségével ellenőrzik a gyártási folyamat minőségét.”²⁰

2.4.2 Objektumfelismerési módszerek áttekintése

Az objektumfelismerési módszerek két fő kategóriába sorolhatók. A hagyományos képfeldolgozási módszerekre és modern, **mélytanulás-alapú (Deep-Learning)** megközelítésekre.

A hagyományos módszerek közé tartoznak a **sablonillesztésen (template matching)** alapuló eljárások, amelyek során a rendszer egy előre definiált mintát keres a képen. Egy másik fontos megközelítés a **kézzel tervezett jellemzők (hand-crafted features)** használata, ahol a fejlesztők olyan jellemzőket határoznak meg, amelyek jól leírják az objektumok tulajdonságait. Ilyen jellemzők például az **élek, sarkok vagy textúrák**. Ezek a módszerek sokáig meghatározó szerepet játszottak a számítógépes látásban, azonban gyakran korlátozott teljesítményt nyújtanak komplex vizuális környezetekben.

A modern objektumfelismerési rendszerek ezzel szemben a gépi tanulásra és mélytanulásra épülnek. Ezek a módszerek nagy mennyiségű adat felhasználásával képesek automatikusan megtanulni a releváns jellemzőket, így nem szükséges a jellemzők kézi tervezése. A neurális

²⁰ Szeliski, R.: *Computer Vision: Algorithms and Applications (2nd Edition)* (2022, 274.o.)

hálózatok, különösen a konvolúciós neurális hálózatok, jelentős előrelépést hoztak a képfeldolgozás területén, mivel képesek komplex mintázatok felismerésére és hatékonyan kezelik a nagy méretű képi adatokat.

A klasszikus modellek közé tartozik az egyik korai áttörést hozó módszer a **R-CNN (Region-based Convolutional Neural Network)** modell, amely régiókat generál a képen, majd azt követően azok külön osztályozásra kerülnek. Előnye, hogy nagyon pontos, de ezáltal lassú is. „Az R-CNN módszer egyik legfontosabb hátránya, hogy az összes régiójavaslaton külön-külön futtatjuk le a neurális hálót, ami pazarlás.”²¹ Az R-CNN módszert 2014-ben mutatták be, és a klasszikus képfeldolgozási technikákat kombinálta a mély neurális hálózatokkal. Az R-CNN továbbfejlesztett változata, a **Fast R-CNN** jelentős mértékben javította az eredeti modell sebességét. Míg az R-CNN minden egyes régiójavaslat esetén külön futtatja a konvolúciós neurális hálózatot, addig a Fast R-CNN a teljes képet egyszerre dolgozza fel, és csak ezt követően vonja ki a releváns jellemzőket a régiójavaslatokhoz.

Ezt a folyamatot a Region of Interest (RoI) pooling réteg teszi lehetővé, amely a jellemzőtérképen a régiójavaslatoknak megfelelő területekből fix méretű jellemzővektorokat (vö.:8.3 fejezet) állít elő. A kinyert jellemzők ezt követően teljesen összekapcsolt rétegekbe kerülnek, amelyek elvégzik az objektumok osztályozását, valamint a **határoló dobozok (bounding boxok)** pontosítását.

A Fast R-CNN jelentősen gyorsabb, mint az R-CNN, mivel a konvolúciós hálózatot elegendő egyszer lefuttatni a teljes képen. Ugyanakkor a régiójavaslatok előállítására továbbra is egy különálló, a hálózattól független folyamat, amely lassítja a teljes detektálási eljárást.

A modern objektumfelismerő modellek közé tartoznak az **SSD (Single Shot Multibox Detector)** és a **YOLO (You Only Look Once)** típusú neurális hálózatok, amelyek valós időben képesek objektumokat detektálni képeken vagy videókban. Ezek a modellek egyetlen neurális háló segítségével határozzák meg az objektumok osztályát és helyét a képen.

Az SSD modell több különböző felbontású **feature map** (a kép feldolgozott változata) rétegen végzi az objektumok detektálását, ezért hatékonyabban kezeli a különböző méretű objektumokat. A modell, előre definiált méretű és arányú **határoló horgony dobozokat**

²¹ Szemenyei M. – Varnyú D.: *Deep Learning a Vizualis Informatikában*, BME jegyzet, (2023. 77.o.)

(anchor boxes) alkalmaz, amelyek segítik a hálózatot a különböző méretű objektumok detektálásában.

A YOLO modell ezzel szemben a képek rácsokra osztásával határozza meg az objektumok helyét és osztályát, ami általában gyorsabb működést tesz lehetővé.

Az SSD a YOLO-hoz hasonlóan egylépéses (**Single-Stage**) detektálási módszer, amely gyors működést biztosít, ugyanakkor bizonyos esetekben pontosabb eredményeket nyújthat, különösen kisebb objektumok felismerése során.

2.4.3 YOLO alapú objektumfelismerés

A YOLO egy korszerű, mélytanulás alapú objektumdetektáló algoritmus, amely képes valós időben felismerni és lokalizálni a képeken található objektumokat. *„A módszer sajátossága, hogy az objektumdetektálást egyetlen neurális háló segítségével egy lépésben végzi el, így jelentősen gyorsabb a hagyományos, több lépésből álló megközelítéseknél.”*²²

A YOLO modellek a bemeneti képet egyszerre dolgozzák fel, és közvetlenül határozzák meg az objektumokhoz tartozó határoló dobozokat (bounding box) és az **osztályvalószínűségeket (class probabilities)**.

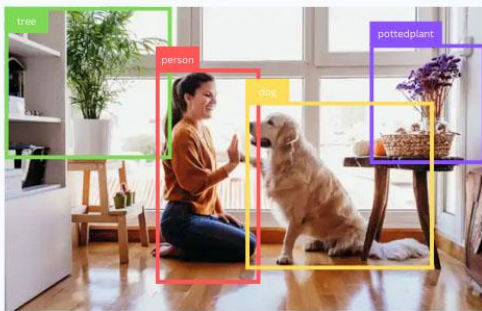
Az Ultralytics által fejlesztett legújabb YOLO modellek, például a YOLO11 többféle számítógépes látási feladatot is támogatnak, mint az objektumdetektálás, a szegmentáció vagy követés, továbbá különböző hardverplatformokon is optimalizáltak a gyors és hatékony működés érdekében

²² Ultralytics, “Ultralytics YOLO Documentation, forrás: <https://docs.ultralytics.com/> (2026)

YOLO algoritmus



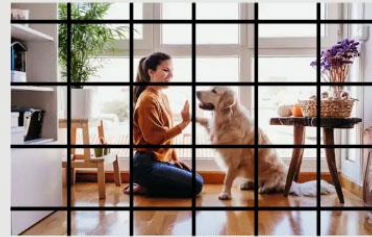
Eredeti kép



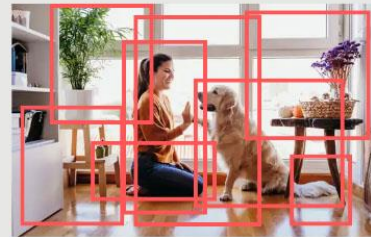
Végző kép osztályokkal

Novelis

Lépések



A kép felosztva $N \times N$ rácsra.
Ha egy objektum közepe egy rács cellába esik,
akkor az a cella felelős az objektum észleléséért.



Megkeresi az összes lehetséges határolódobozt.
A legmagasabb konfidenciájú (megbízhatósági
értékkel rendelkező) dobozokat részesíti előnyben.







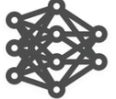
Létrehoz egy osztályvalószínűségi térképet
(Class Probability Map), hogy meghatározza,
melyik osztály felel meg a kiválasztott
határolódobozoknak.

7. ábra, YOLO algoritmus szemléltetése ²³

A YOLO11 a korábbi modellekhez hasonlóan többféle méretű és teljesítményű változatban érhető el, amelyek különböző kompromisszumot biztosítanak a feldolgozási sebesség és a detektálási pontosság között. A modellek közötti fő különbség a rétegek száma, a paraméterek mennyisége és a számítási igény.

²³ Kép forrása: Novelis (2023.10.05); <https://novelis.io/research-lab/computer-vision/> (letöltve: 2026.03.22)

YOLO11 modellváltozatok:

Modell		Jelölés	Jellemző
YOLO11n		nano	nagyon gyors, alacsonyabb erőforrásigény.
YOLO11s		small	gyors, kiegyensúlyozott.
YOLO11m		medium	jó pontosság.
YOLO11l		large	magas pontosság.
YOLO11x		extralarge	maximális pontosság, nagy erőforrásigény.

8. ábra, YOLO11 modellváltozatok

A YOLO algoritmus hatékonysága és sebessége miatt széles körben alkalmazható olyan rendszerekben, ahol fontos a gyors döntéshozatal, például felügyeleti rendszerekben, önvezető autóknál, vagy akár a vizuális tiltott adatszerzés detektálására szolgáló megoldásokban.

2.5 Objektum–attribútum mátrix

A Kodolányi János egyetemen²⁴ végzett tanulmányaim során ismerkedtem meg az *Objektum–Attribútum-Mátrix (OAM)*²⁵ fogalmával, amelynek segítségével az objektumok és azok tulajdonságai rendszerezett formában ábrázolhatók. A szakdolgozatban az OAM szemlélete is megjelenik, mint az adatok rendszerezésének és az objektumok jellemzőinek strukturált vizsgálatának egyik lehetséges megközelítése. Ez a megközelítés, lehetővé teszi az adatok strukturált elemzését, valamint a különböző jellemzők közötti kapcsolatok feltárását, amely szerepet játszhat a mintázatok felismerésében és az információk feldolgozásában.

²⁴ Kodolányi János Egyetem (2026); <https://www.kodolanyi.hu/>

²⁵ Hasonlóságelemzés COCO használatával, Oktatási segédanyag, 2013.08.24.

„Az objektum-attribútum-mátrix, azaz más szóval tanulási minta, melyben immár tradicionálisan a sorok az objektumok (esetek), az oszlopok ezek jellemzői (attribútumok, szempontok, változók stb.)”²⁶

„Az OAM azaz az Objektum-attribútum mátrix struktúra, amelynek célja az objektumok közötti hasonlóság meghatározásának támogatása azok attribútumainak alapján. Az OAM nem más, mint egy fajta tanulási minta. Az "objektum" lehet például egy dokumentum, egy kép, egy termék vagy bármilyen más entitás, amely rendelkezik attribútumokkal, vagyis jellemzőkkel, tulajdonságokkal. Egy Objektum-attribútum mátrixban az objektumokat sorok, az attribútumokat pedig oszlopok reprezentálják, ahol az egyes cellák értékei az adott objektum adott attribútumának jellemzői lehetnek, például bináris (jelenlét vagy hiány), numerikus vagy kategorikus értékek.”²⁷

2.6 A képzés tárgyainak és a szakdolgozati témának kapcsolata

A szakdolgozat témája több területet, illetve tudományágat is nagymértékben érint. A megvalósítás során számos informatikai, műszaki, matematikai és társadalomtudományi terület ismerete szükséges. Az alábbiakban bemutatásra kerül, hogy a képzés egyes tárgyai, milyen módon kapcsolódnak a témához.

2.6.1 Európai civilizáció és identitás

A tárgy hozzájárul a digitális társadalom működésének megértéséhez, különös tekintettel az adatvédelem és a magánszféra fontosságára. Az európai értékrend kiemelten kezeli az információk védelmét, amely közvetlenül kapcsolódik a vizuális tiltott adatszerzés problémájához. (vö:[2.1.2 fejezet](#))

2.6.2 A jog szerepe a modern társadalmakban

A jogi ismeretek alapvetőek az adatvédelem területén, mivel a vizuális tiltott adatszerzés személyes és üzleti adatok illetéktelen megszerzését jelenti.

²⁶ My-x Team - Avagy egy innovatív „ötlet-istálló” (Megtekintve: 2026. 03. 19.)

²⁷ Dr. Pitlik L.: *Üzemmérnök-Informatikai Kar, KJE, (2023. Egyetemi előadás.)*

Az olyan szabályozások, mint a *GDPR – General Data Protection Regulation* –,²⁸ meghatározzák az adatkezelés kereteit és közvetlen hatással vannak arra, hogy milyen módon lehet képfeldolgozási és megfigyelési rendszereket fejleszteni és alkalmazni. A tiltott adatszerzés Btk.422.§ megakadályozására szolgáló rendszereknek is meg kell felelniük ezeknek az előírásoknak.

2.6.3 Matematikai alapok

A matematikai háttér elengedhetetlen a képfeldolgozás és a gépi tanulás megértéséhez. A kombinatorika (vö. [3.1.1 fejezet](#)), lineáris algebra (mátrixműveletek), a valószínűségszámítás és az analízis, mind olyan eszközöket biztosítanak, amelyek nélkülözhetetlenek a **konvolúciós műveletekhez** és a **neurális hálózatok** működésének és tanításának leírásához. (vö. [2.3.2 fejezet](#))

2.6.4 Adatszerkezetek és algoritmusok

A hatékony adatkezelés és feldolgozás, kulcsfontosságú szerepet tölt be nagy mennyiségű képi adatok esetén. Az algoritmusok optimalizálása, szükséges a valós idejű detektálás megvalósításához. (vö.:[3.4.1 fejezet](#))

2.6.5 Operációs rendszerek

Az operációs rendszerek ismerete szükséges a hardver erőforrások – memória kezelés, párhuzamos feldolgozás – hatékony kihasználásához, különösen a grafikus feldolgozóegység, GPU-alapú számítások esetén. A mélytanulási modellek tanítása és futtatása jelentős számítási kapacitást igényel, amelynek kezelése operációs rendszerszintű optimalizációkat is feltételez. A fejlesztői környezet operációs rendszere a *Microsoft Windows 11*, melyen verziószáma: *23H2*. Azért esett a választás a Windows környezetre, mert az általam fellelt adatok alapján a 2026-os adatok szerint Magyarországon, az asztali számítógépek **több mint 75%-án** fut a Microsoft valamelyik rendszere.²⁹ (vö.:[3.4.4.2 fejezet](#))

²⁸ GDPR – General Data Protection Regulation; az Európai Unió adatvédelmi rendelete (European Parliament & Council, 2016).

²⁹ Statcounter Globalstats, (2026.03.15.)

2.6.6 Programozási alapelvek és módszertanok

A fejlesztés megvalósítása során, fontos a szoftverfejlesztés strukturált megközelítésének a biztosítása, melybe beletartozik a moduláris felépítés, a karbantarthatóság és a skálázhatóság. (vö.: 3.4.3 fejezet)

2.6.7 Hálózatok és számítógép architektúrák

A tantárgy segít megérteni az adatátvitel és a számítási infrastruktúra működését. A hálózati ismeretek nélkülözhetetlenek, ha a rendszer több eszköz között kommunikál, felhőben fut, vagy valós idejű adatfeldolgozást végez, így az adatok hatékony továbbítása és feldolgozása kulcsfontosságú. Az architektúra ismerete, segíti a teljesítmény optimalizálását. (vö.:7.1.6 fejezet)

2.6.8 Kultúra, sport, munkahelyi jóllét

A munkahelyi környezet és az emberi tényezők megértése, hozzájárul a biztonsági rendszerek megfelelő kialakításához. A tiltott adatszerzés gyakran emberi viselkedéshez kapcsolódik, például milyen helyzetben történik az adateltulajdonítás, milyen emberi tényezők játszhatnak benne szerepet, így ennek figyelembevétele elengedhetetlen. (vö.:1.4 fejezet)

2.6.9 Elektronikus áramkörök

A kamerák és mobil eszközök működésének alapját elektronikus áramkörök adják. Ezek ismerete segít megérteni a képalkotás fizikai hátterét.

2.6.10 Az elektronika fizikai alapjai

A képérzékelés fizikai folyamatai – például fényérzékelés, zajok keletkezése – közvetlen hatással vannak a képfeldolgozásra. A kamerák és szenzorok működése meghatározza a képek minőségét és zajosságát, amely közvetlen hatással van a feldolgozás eredményére. A valós képek zajjal terheltek, amelyek különböző szűrési módszerekkel kezelhetők. (vö.:2.2.2 fejezet)

2.6.11 Rendszermodellezés

A tiltott adatszerzés észleléséhez szolgáló rendszer több komponensből áll, amelyek együttműködéséhez szükséges a modellezés. A rendszer tervezése során architektúra készítésére, komponensek közti kapcsolatok meghatározása és rendszer szintű optimalizálásra van szükség. (vö.: 3.4.3 fejezet)

2.6.12 Emberi viselkedés és kommunikáció

Az Emberi viselkedés és kommunikáció tárgy szorosan kapcsolódik a dolgozat témájához, mivel a tiltott adateltulajdonítás gyakran emberi döntések és viselkedési minták eredménye. A rendszer tervezése során azt is figyelembe kell venni, hogy az emberek hogyan használják a mobil eszközöket. (vö.:[1.3 fejezet](#))

2.6.13 Programozás

A programozás ismerete a megvalósítás alapját képező készség, amely biztosítja az implementációhoz szükséges gyakorlati tudást. A képfeldolgozási algoritmusok és neurális hálózatok megvalósítása, programozási környezetben történik. (vö.:[6.1.3 fejezet](#))

2.6.14 Felhasználói interfészek és vizualizáció

A rendszer kimenetének megjelenítése fontos szerepet játszik, mivel annak értelmezhetőnek és felhasználóbarátnak kell lennie. Az objektumdetektálási eredmények vizuális megjelenítése kulcsfontosságú a rendszer használhatósága szempontjából. (vö.:[7.2.2 fejezet](#))

2.6.15 Vezetési és vállalkozási ismeretek

Mivel egy fejlesztés projektként kezelendő, így a tárgyban említett időmenedzsment, erőforrástervezés, kockázatkezelés ismeretei segítenek megérteni a technológiai megoldások üzleti környezetét. (vö.:[1.3 fejezet](#))

2.6.16 Vállalati gazdaságtan

Egy ilyen rendszer fejlesztése és bevezetése esetén, a gazdasági szempontokat is figyelembe kell venni, mint például az adatvesztés költségét, biztonsági beruházások megtérülését. (vö.:[1.4 fejezet](#))

2.6.17 Komplex társadalomtudományi ismeretek

Hozzájárulnak a technológia társadalmi hatásainak megértéséhez, különösen az adatvédelem és megfigyelés kérdésében.

2.6.18 Adatbázisok

Az adatbázisok tárgy biztosítja az adatok tárolásának és kezelésének alapjainak megértését. A projekt során, a tanítóadatok és a detektálási eredmények hatékony kezelése kulcsfontosságú szerepet játszik. Emellett a rendszer működése során keletkező naplóadatok tárolására, egy külön adatbázis kerül a későbbiek során alkalmazásra, amely támogatja az események nyomon követését és későbbi elemzését. (vö.:[7.2.1 fejezet](#))

2.6.19 Szoftverüzemeltetés

A rendszer telepítése és karbantartása, a folyamatos működés biztosítása, hibakezelése, a frissítések kezelése, mind az üzemeltetés területéhez tartozik. (vö.: [3.4.4.6 fejezet](#))

2.6.20 Rendszertervezés

A teljes rendszer architektúrájának tervezése, kialakítása és a rendszerek életciklusának kezelésének a támogatása, a legfőbb feladat egy jól működő alkalmazás létrehozásához. Szükséges a komponensek definiálása, adatáramlás megtervezése és a skálázhatóság biztosítása. (vö. [3.4 fejezet](#))

2.6.21 Informatikai védelem és biztonság

Közvetlenül kapcsolódik a dolgozat témájához, hiszen a vizuális tiltott adatszerzés egy specifikus biztonsági fenyegetés. A rendszer alkalmazása növelheti a biztonsági szintet, azonban hatékony működéséhez elengedhetetlen a megfelelő pontosság és megbízhatóság biztosítása. (vö.:[1.4 fejezet](#))

2.6.22 Szoftvertesztelés

A szoftvertesztelés célja, a rendszer működésének ellenőrzése és a hibák feltárása. A mobiltelefon-felismerő rendszer különböző környezeti feltételek mellett került tesztelésre, figyelembe véve a detektálás pontosságát és a teljesítményt.

A tesztelés során kiemelt szerepet kapott a hamis pozitív és hamis negatív eredmények vizsgálata, valamint a rendszer sebességének és erőforrás-felhasználásának elemzése. (vö.[3.5 fejezet](#))

2.6.23 Innovatív információs és kommunikációs technológiák az IT-biztonság kapcsán

Az IT-biztonság területén az innovatív információs és kommunikációs technológiák alkalmazása lehetővé teszi a hatékonyabb és automatizált védekezési megoldások kialakítását. A jelen projektben egy mesterséges intelligencián alapuló képfeldolgozó rendszer került alkalmazásra. Az ilyen technológiák hozzájárulnak a hagyományos biztonsági megoldások továbbfejlesztéséhez, azonban alkalmazásuk során, figyelembe kell venni a teljesítmény, a pontosság és a megbízhatóság közötti egyensúlyt. (vö.: [2.4.3 fejezet](#))

2.6.24 Tudásmenedzsment az IT biztonság területén

A fejlesztés során a különböző technológiák, módszerek és tapasztalatok rendszerezése és folyamatos bővítése, hozzájárult a rendszer fejlesztéséhez és optimalizálásához. A megszerzett tudás – például a modell viselkedésének megértése, a hibák elemzése és a teljesítmény javítása – lehetővé tette a rendszer működésének folyamatos finomhangolását, ezáltal növelve annak megbízhatóságát és hatékonyságát. (vö.: [3.5 fejezet](#))

2.6.25 Mesterséges intelligenciák az IT-biztonság területén

A mesterséges intelligencia szinte minden területen jelen van, például az egészségügyben a közlekedésben, az ipari automatizált gyártásban, sőt még a turisztikában is, ahol a „*mesterséges intelligencia alapú adatkezelési megoldások javítják az elméleti és gyakorlati eredményeket a turizmusban.*”³⁰ Az IT biztonság területén a mesterséges intelligencia lehetővé teszi az automatikus fenyegetések felismerést, mintázatok azonosítását, elemzését, így a tárgy közvetlenül kapcsolódik a mély tanulás alapú megoldásokhoz, amelyek a képfeldolgozás és objektumdetektálás alapját képezik. (vö.: [2.3 fejezet](#))

2.6.26 IT-biztonsági fejlesztések minőség- és projektmenedzsmentje

Az IT-biztonsági fejlesztések során fontos a minőség- és projektmenedzsment, amely biztosítja a rendszer megbízható működését. A jelen projektben egy valós idejű mobiltelefon-felismerő rendszer készül, amelynek fejlesztése során kiemelt szerepet kapott a teljesítmény és a pontosság vizsgálata. A tesztelések alapján a rendszer képes a mobiltelefonok felismerésére,

³⁰ Grotte, J., *Introducing the new concept of personalized digital tourism (PDT)*. Infocommunications Journal., (2024, Abstract.)

azonban éles környezetben további optimalizálás szükséges a megbízhatóság növeléséhez.
(vö.:[3.5.3 fejezet](#))

2.6.27 Mentori óra

A mentori óra támogatja a szakdolgozat elkészítésének folyamatát, lehetőséget adva a folyamatos konzultációra és szakmai visszajelzésre. A konzulens segíti a fejlesztés és kutatás irányát, a problémák megoldását, a szakdolgozat minőségének javítását. Ezúton szeretném megköszönni **Dr. Pitlik László** türelmét és segítségét, valamint szakmai támogatását, amellyel jelentős mértékben hozzájárult a dolgozat elkészítéséhez.

2.6.28 Más COCO-orientált projektet tartalmazó szakdolgozat összevetése

A fejezet célja a COCO (Component-based Object Comparison for Objectivity) módszertanra épülő, más hallgatók által készített elemzések összehasonlítása. Az összehasonlítás célja annak bemutatása, hogy a különböző tématerületek ellenére a COCO-alapú megközelítések milyen közös elemeket tartalmaznak, valamint miben térnek el egymástól, különös tekintettel az attribútumok kezelésére, a rangsorolási eljárásokra és az optimalizációs módszerekre. Az összevetéshez **Lehrer Zoltán** által készített „Rendszerfigyelési adatok log alapú sérülékenységvizsgálata és USB háttértár kockázatelemzése MI- alapú modellel” című szakdolgozata került kiválasztásra, mivel az előfeldolgozás azonos lépéseket tartalmaz, – például az adatok rangsorolása a további műveletek elvégzéséhez –, amely a COCO módszertan alapvető eleme. Azonban az adatgyűjtés, illetve az alkalmazott adatforrások és az előfeldolgozás utáni lépések és a gyanúgenerálás módja jelentős mértékben eltér.

(vö.: [https://miau.my-x.hu/miau/329/lz/Szakdolgozat_Lehrer_Zolt%
c3%a1n%20v9.docx](https://miau.my-x.hu/miau/329/lz/Szakdolgozat_Lehrer_Zolt%c3%a1n%20v9.docx))

Zoltán dolgozatában szereplő adatgyűjtés a MS Windows operációs rendszer teljesítményfigyelő eszközének (Performance Monitor) segítségével történik. Az adatok időalapú mérésekből származnak meghatározott időintervallumok mentén, így jól strukturált adathalmaz áll rendelkezésre. Az előfeldolgozás manuális lépéseket tartalmaz, amely során az adatok strukturált formába kerülnek.

A munkám során alkalmazott adatforrás rögzített fényképekből áll. Az előfeldolgozás automatizált módon történik egy saját fejlesztésű szoftver, az ImageToCsvConverter alkalmazásával, amely a képeket szürkeárnyaltos formára alakítja, majd a pixelértékeket numerikus adathalmazzá konvertálja.

A két dolgozat egyik legfontosabb eltérése az objektum és attribútum fogalmának értelmezésében jelenik meg. Az összehasonlító dolgozatban az objektumok időintervallumok, az attribútumok pedig rendszerparaméterek (CPU, memória stb.)

A munkám esetében az objektumok scenáriók (képek), az attribútumok pedig statisztikai jellemzők (szórás, átlag, medián stb.).

A kiválasztott dolgozatban a rangsorolás közvetlenül a mért értékek alapján történik az irányok (pozitív/negatív kockázat) meghatározása, még a munkám során a rangsorolást korreláció-alapú iránymeghatározás előzi meg, mely során a kapcsolat előjele határozza meg az attribútum viselkedését.

A feldolgozáshoz alkalmazott COCO modell eltérő módon kerül alkalmazásra. A kiválasztott dolgozat a COCO Y0 modellt alkalmazza, ezzel szemben az általam alkalmazott feldolgozás a COCO STD modellel történik előre meghatározott Y változóval, amelyet a küszöbérték-alapú döntés bevezetése követ.

A kiválasztott dolgozatban a kockázat közvetlenül a COCO becslésekből kerül levezetésre, míg a saját munkám során a gyanú egy küszöbérték és szabályrendszer alapján kerül meghatározásra.

A saját munkám és a vizsgált dolgozatokban alkalmazott megközelítések összehasonlítása alapján megállapítható, hogy míg a legtöbb dolgozat egy strukturált, jól értelmezhető, de egyszerűbb COCO-alapú elemzést valósít meg, addig az általam alkalmazott megközelítés egy komplexebb, több lépcsőből álló optimalizált módszertant alkalmaz. Eltérés figyelhető meg a gyanúgenerálás alkalmazásában, a küszöbérték-alapú döntéshozatalban, valamint a direkt és inverz mátrixok kombinálásával elért jelentős pontosságnövekedésben.

3 Adatfeldolgozás és gyanúgenerálás módszerei, valamint a védelmi szoftver bemutatása

A tervezett védelmi szoftver központi feladata, a mobiltelefonok megbízható felismerése. Ennek érdekében kulcsfontosságú a rendelkezésre álló képi adatok hatékony feldolgozása, valamint az ezekből származó információk értelmezése. Az adatfeldolgozást követően, a rendszer működésének következő lépése a gyanúgenerálás, amelynek során meghatározásra kerül, hogy a detektált objektumok és azok környezete alapján, fennáll-e jogosulatlan adateltulajdonítás lehetősége. A gyanúgenerálás folyamata, két eltérő megközelítés vizsgálatával történik.

Az egyik módszer az Objektum–attribútum mátrix alkalmazása, amely külső szoftveres támogatás nélkül, szabályalapú módon értékeli az objektumok jellemzőit. A másik megközelítés egy korszerű, mélytanulás alapú objektumfelismerési modell, amely képes a képi adatok automatikus feldolgozására és objektumok detektálására.

3.1 Gyanúgenerálás módszere Objektum-attribútum mátrix alkalmazásával

A következő fejezet azt mutatja be, hogy vajon mennyire lehet sikeres a gyanú megállapítására az Objektum-attribútum mátrix alkalmazása. Az attribútumok közötti összefüggések elemzésével a gyanús mintázatok azonosíthatók-e, amelyek potenciális adateltulajdonítási kísérletre utalhatnak. Fontos megjegyezni, hogy az Objektum-attribútum-mátrix nem alkalmas a valós idejű objektumdetektálásra.

A feladat az optikai eszközök esetén egy-egy állókép-értelmezési/elemzési kihívás kezelése, egyéb szoftveres megoldások nélkül. „Mind a játéktér mind pedig a hatásos megoldások száma végtelen, a mozgásterünk rendkívül sokszínű, például a felbontások különböző változatain, a képek minőségén és egyéb különböző aggregációkon keresztül kell meghatározni, melyik a leghatékonyabb megoldás az erőforrás figyelembevételével.” Így tulajdonképpen a folyamatot, az – ár- teljesítmény problémára – konvertálja, ahol az ár a felhasznált erőforrások tükrözése, a teljesítmény pedig a matematikai pontosság, hogy hány esetből mennyiszor detektálunk helyesen. A feladat, helyes gyanút generálni, ez a konkrét cél.”³¹

³¹ Dr. Pitlik L.: „Rendszermodellelés”, Üzemélnök-Informatikai Kar, KJE, (2024. 04 27. Egyetemi előadás.)

A megoldás során több kihívással állunk szemben, például, hogy segít-e a képek mennyisége és a felbontás minősége és intenzitástartománya a gyanús esemény felderítésében.

3.1.1 Az Objektum-attribútum mátrix alkalmazás képi adatok esetén

A képfeldolgozás során az objektum lehet egy képrészlet, vagy detektált alakzat, míg az attribútumok statisztikai módszerekkel meghatározott jellemzők, mint például az intenzitáseloszlás, élstruktúra vagy textúra. Egy tipikus objektumfelismerési feladat során a rendszer egy adott képen például, egy célterület felé néző emberalakot keres, majd az ember környezetében egy szögletes tárgyat, amelyen belül egy kör alakú struktúra található, azon belül pedig egy kisebb kört, amely optikai eszközre utalhat.

Az ilyen típusú felismerési feladatok során a lehetséges objektum- és jellemzőkombinációk száma véges, így a probléma kombinatorikai szempontból is értelmezhető. A kombinatorika – szó szerinti jelentése „kapcsolástan” – a matematika azon területe, amely egy véges halmaz elemeinek szabályok szerinti kiválasztásával, csoportosításával és sorrendbe rendezésével foglalkozik.

3.1.2 Adatok és módszerek az OAM alkalmazásához

A feldolgozáshoz használt képi adatok különböző valós események során rögzített fényképekből származnak, így az adatvagyon saját, tetszőleges és rugalmasan bővíthető. A szükséges adatok előállítására egy erre a célra fejlesztett „*ImageToCsvConverter*”³² nevű szoftver segítségével történt.

Az egyszerűsítés érdekében, szürkeárnyaltos színskála került alkalmazásra, amely az árnyalatok összevonásával tovább egyszerűsíthető. Az így kinyert adatok alkalmasak az Objektum-attribútum mátrix segítségével történő feldolgozásra.

3.1.3 Adatok előállítása, Image Converter szoftver bemutatása

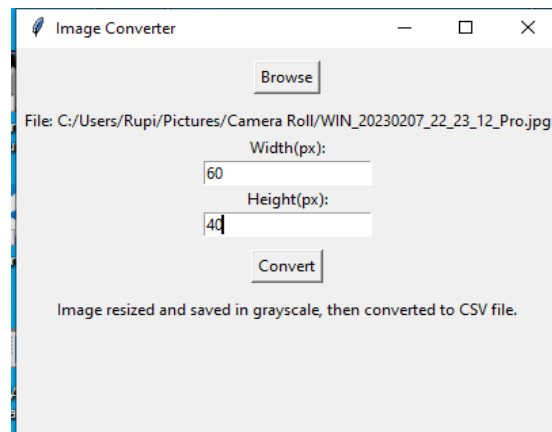
Az adatok előállítása az *ImageToCsvConverter* segítségével történt, amely egy tetszőleges képet alakít át szürkeárnyaltosra és azt egy tetszőleges méretre, képpont-koordináta megadása alapján. Majd a képpont adatokat, egy csv – Comma-Separated Values, azaz vesszővel elválasztott értékeket tartalmazó fájlban tárolja, ami később beimportálható egy táblázatkezelő,

³² Schewe E (2024). - *ImageToCsvConverter*:

például Microsoft Excel alkalmazásba. Az átalakított kép elmentésre kerül a csv fájl mellett, így a kimenet megtekinthető. Jelen verzióban a mentés helye nem definiálható, a fájl neve nem módosítható, és a már „**meglévő fájlok felülírásra kerülnek az aktuálisra!**”

A program működése:

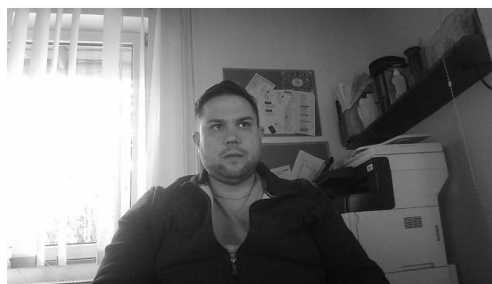
1. A Browse gomb megnyomásával hívható be az átalakításra váró képfájl;
2. A szélesség és a hosszúság pixelben (képpontokban) adható meg;
3. Convert gomb megnyomásával fut le a folyamat.



9. ábra, ImageToCsvConverter szoftver felülete

3.1.4 Objektumok

Az Objektum–attribútum mátrixban az objektumok a mátrix sorainak felelnek meg. A jelen esetben ezek a sorok különböző scenáriókat reprezentálnak, azaz összehasonlítható eseményekről készült fényképeket. A képekből származó, 0–255 közötti intenzitásértékek aggregált formában jelennek meg, amelyek számos valós jellemzőt és mintázatot reprezentálhatnak.



10. ábra, Saját előállított adat, illetve scenario gyanús esemény nélkül

Forrás: https://miau.my-x.hu/miau/330/sche/valos_OAM_v5_2.xlsx (1 lapfűl A1:XP360)

	A	B	C	D	E	F	G	H	I
1	Nyers OAM	Szórás	Maximum	Minimum	Átlag	Módusz	Medián	Y (Telefon: igen=2000, nem=1000)	Becslés
2	Mértékegység	intenzitástartomány	intenzitástartomány	intenzitástartomány	intenzitástartomány	intenzitástartomány	intenzitástartomány	Jóságpont	intenzitástartomány
3	Scenario01	82	255	4	117	254	94	1000	805
4	Scenario02	83	255	1	120	254	94	2000	807
5	Scenario03	83	255	7	119	254	94	1000	812
6	Scenario04	78	255	6	122	254	103	1000	819
7	Scenario05	83	255	6	121	254	98	1000	817
8	Scenario06	82	255	6	121	254	100	1000	819
9	Scenario07	84	255	0	118	254	94	1000	803
10	Scenario08	84	255	1	116	254	92	1000	803
11	Scenario09	86	255	4	121	254	100	1000	820
12	Scenario10	84	255	6	120	254	95	2000	814
13	Scenario11	83	255	6	121	254	97	2000	817
14	Scenario12	83	255	6	121	254	97	2000	817
15	Scenario13	84	255	3	120	254	98	1000	813
16	Scenario14	84	255	6	120	254	98	1000	817
17	Scenario15	83	255	6	121	254	97	2000	817
18	Scenario16	86	255	3	117	254	95	2000	811
19	Scenario17	83	255	5	120	254	98	1000	816
20	Scenario18	88	255	3	113	254	85	2000	799
21	Scenario19	79	255	7	113	254	91	1000	799
22	Scenario20	83	255	1	120	254	94	2000	807
23	Scenario21	83	255	1	120	254	94	2000	807
24	Scenario22	84	255	3	118	254	93	2000	807
25	Scenario23	84	255	4	120	254	98	1000	814
26	Scenario24	84	255	3	118	254	93	2000	807
27	Scenario25	82	255	7	122	254	101	2000	821
28	Korreláció	0.3	Nem értelmezhető	-0.2	0.0	Nem értelmezhető	-0.3	1.0	-0.1
29	Irány	0	0	1	0	0	1	0	1

13. ábra, Nyers OAM táblázata

Forrás: https://miau.my-x.hu/miau/330/sche/valos_OAM_v5_2.xlsx (OAM lapfűl A1:I29)

3.1.6 Elemzés

Az elemzés során első lépésben nem történik meg az értékek rangsorolása, hanem csak a nyers adatok és az attribútumok közötti összefüggések vizsgálat kerül előtérbe. A 29-es cella számú **irány** sor az arányosság irányát szemlélteti, amely a B, C, D, E, F, G oszlopok H oszloppal vett korrelációs értékei alapján kerül meghatározásra. Amennyiben a korreláció értéke kisebb mint 0, akkor az adott kapcsolat értéke 1, azaz minél kisebb a bemenő jel annál nagyobb a kimenő jel értéke is. Tehát az Excel korrelációs függvényének értelmezése alapján, a negatív kapcsolat esetében fordított arányosság, míg a pozitív kapcsolat esetében egyenes arányosság áll fenn.

A B és E oszlopok esetében egyenes arányosság figyelhető meg a H oszloppal, mivel az értékek növekedésével a H oszlop értéke is növekszik. Ez azt jelenti, hogy minél nagyobb a szórás és az átlag mértéke, annál magasabb a gyanú értéke is. Ennek alapján megállapítható, hogy az adott scenárió, illetve kép, gyanúsnak tekinthető és akár fennállhat a fényképezési szándék lehetősége.

A D, G és I oszlopok esetében fordított arányosság figyelhető meg a H oszloppal, vagyis minél kisebb az oszlop értéke, annál közelebb van a H oszlop értékéhez, ezáltal a gyanú mértéke is növekszik. A C és az F oszlop értékei nem értelmezhetők, mivel nem lehet másik eltérő értékhez viszonyítani. A folyamatban a scenáriónként kockázat megállapításához a Becslés került alkalmazásra a soronkénti összegek összeadásával.

A következő körben előállítjuk a nyers adatokból a rangsorokat a további elemzések elvégzéséhez.

Rangsorolt OAM Direkt	Szórás	Maximum	Minimum	Átlag	Módusz	Medián	Y (Telefon: igen=2000, nem=1000)	Becsles 2
Mértékegység	sorszám	sorszám	sorszám	sorszám	sorszám	sorszám	Jóságpont	sorszám
Scenario01	23	1	11	22	1	6	1000	11
Scenario02	12	1	2	14	1	6	2000	6
Scenario03	17	1	23	17	1	6	1000	11
Scenario04	25	1	15	1	1	25	1000	11
Scenario05	15	1	15	7	1	17	1000	9
Scenario06	21	1	15	3	1	22	1000	11
Scenario07	7	1	1	20	1	6	1000	6
Scenario08	4	1	2	23	1	3	1000	6
Scenario09	2	1	11	8	1	22	1000	8
Scenario10	8	1	15	12	1	12	2000	8
Scenario11	18	1	15	4	1	14	2000	9
Scenario12	18	1	15	4	1	14	2000	9
Scenario13	9	1	6	11	1	17	1000	8
Scenario14	11	1	15	10	1	17	1000	9
Scenario15	18	1	15	4	1	14	2000	9
Scenario16	3	1	6	21	1	12	2000	7
Scenario17	16	1	14	9	1	17	1000	10
Scenario18	1	1	6	24	1	1	2000	6
Scenario19	24	1	23	25	1	2	1000	13
Scenario20	12	1	2	14	1	6	2000	6
Scenario21	12	1	2	14	1	6	2000	6
Scenario22	5	1	6	18	1	4	2000	6
Scenario23	10	1	11	13	1	17	1000	9
Scenario24	5	1	6	18	1	4	2000	6
Scenario25	22	1	23	2	1	24	2000	12
Korreláció	-0.21	Nem értelmezhető	-0.23	-0.04	Nem értelmezhető	-0.27	1.00	-0.41

14. ábra, Rangsorolt OAM táblázata

Forrás: https://miau.my-x.hu/miau/330/sche/valos_OAM_v5_2.xlsx (OAM lapfűl A34:I61)

A rangsorolt OAM oszlop a felső adatokkal ellátott táblázat sorszámok szerinti megjelenítését tartalmazza. A sorszámok a nyers adatokból képződnek, melyek egy rendezettebb állapotot tükröznek, de önmagukban nem tekinthetők végleges eredménynek.

3.1.7 Eredmények

A rangsorolásnál elmondható, hogy a becslések jelen esetben nem vezettek konkrét eredményre.

A következő művelet elvégzéséhez a „COCO (component-based object comparison for objectivity / objektivitást támogató komponens-alapú objektum összehasonlítás) hasonlóságelemzést végző algoritmuscsalád alkalmazására kerül sor.”³³A különböző elemzőmodulok közül a COCO_STD (standard) került alkalmazásra, mivel a cél a gyanúgenerálás. „A COCO_STD a COCO- alapú hasonlóságelemzés standard modulja, mely keretében kell, hogy legyen valós Y-változó például ár, mely az X-változók lépcsős függvényeként kerül közelítő jelleggel felépítésre, a hasonlóságelemzés komponensalapú logikája szerint (például ár/teljesítmény elemzések során).”³⁴

³³ COCO forrás: <https://miau.my-x.hu/myx-free/coco/> (2026.02.12)

³⁴ My-x Team - Avagy egy innovatív „ötlet-istálló”, 2014 (2026.03.28.)

A rangsorolást követően az objektumhoz Y értéket rendelünk, melyben **2000 jószágpontokkal** jutalmazzuk a **gyanús**, és **1000 ponttal** a **gyanútlan** eseményeket. A COCO modell ezek alapján végez el számításokat és ad egy becslési értéket minden objektumnak.



15. ábra, COCO Standard felülete

Forrás: https://miau.my-x.hu/myx-free/coco/beker_std.php

A lefuttatás során kapott eredmény alapján nem egyértelmű, hogy a 25 darab kép, azaz szcenárió közül mely gyanúsítható, illetve menthető fel a gyanú alól a delta oszlopok értékei alapján.

Rangsorolt OAM Direkt	Szórás	Maximum	Minimum	Átlag	Módusz	Medián	Becslés	Tény+0	Delta	Delta/Tény
Mértékegység	Sorszám	Sorszám	Sorszám	Sorszám	Sorszám	Sorszám	Jószágpont	Érték	Érték	Érték
Scenario01	0	0	0	0	0	1013.7	1013.7	1000	-13.7	-1.37
Scenario02	0	0	0	1013.7	0	1013.7	2027.4	2000	-27.4	-1.37
Scenario03	0	0	0	0	0	1013.7	1013.7	1000	-13.7	-1.37
Scenario04	0	0	0	1520.5	0	0	1520.5	1000	-520.5	-52.05
Scenario05	0	0	0	1013.7	0	0	1013.7	1000	-13.7	-1.37
Scenario06	0	0	0	1013.7	0	0	1013.7	1000	-13.7	-1.37
Scenario07	0	0	0	0	0	1013.7	1013.7	1000	-13.7	-1.37
Scenario08	506.8	0	0	0	0	1013.7	1520.5	1000	-520.5	-52.05
Scenario09	506.8	0	0	1013.7	0	0	1520.5	1000	-520.5	-52.05
Scenario10	0	0	0	1013.7	0	1013.7	2027.4	2000	-27.4	-1.37
Scenario11	0	0	0	1013.7	0	1013.7	2027.4	2000	-27.4	-1.37
Scenario12	0	0	0	1013.7	0	1013.7	2027.4	2000	-27.4	-1.37
Scenario13	0	0	0	1013.7	0	0	1013.7	1000	-13.7	-1.37
Scenario14	0	0	0	1013.7	0	0	1013.7	1000	-13.7	-1.37
Scenario15	0	0	0	1013.7	0	1013.7	2027.4	2000	-27.4	-1.37
Scenario16	506.8	0	0	0	0	1013.7	1520.5	2000	479.5	23.98
Scenario17	0	0	0	1013.7	0	0	1013.7	1000	-13.7	-1.37
Scenario18	1013.7	0	0	0	0	1013.7	2027.4	2000	-27.4	-1.37
Scenario19	0	0	0	0	0	1013.7	1013.7	1000	-13.7	-1.37
Scenario20	0	0	0	1013.7	0	1013.7	2027.4	2000	-27.4	-1.37
Scenario21	0	0	0	1013.7	0	1013.7	2027.4	2000	-27.4	-1.37
Scenario22	506.8	0	0	0	0	1013.7	1520.5	2000	479.5	23.98
Scenario23	0	0	0	1013.7	0	0	1013.7	1000	-13.7	-1.37
Scenario24	506.8	0	0	0	0	1013.7	1520.5	2000	479.5	23.98
Scenario25	0	0	0	1520.5	0	0	1520.5	2000	479.5	23.98

16. ábra, COCO STD modell által kapott eredmények

Forrás: https://miau.my-x.hu/miau/330/sche/valos_OAM_v5_2.xlsx (COCO Rang Direkt&Inverz lapfűl

A88:K114)

A szabályrendszer alapján a Maximum, Minimum és Módusz oszlopok figyelmen kívül hagyhatók a monotonitásuk miatt, viszont láthatjuk, hogy a Szórás, Átlag és a Medián oszlopok

információkat tartalmaznak a Delta oszlop értékei alapján. Mivel a gyanú fennáll, ezáltal a gyanúgenerálás módszerét alkalmazzuk. „A gyanú fogalma nem más, mint a becslési hiba szinonimája. Minden gyanús, ami a rendelkezésre álló adatok alapján nem magyarázható. Természetesen kellően sok magyarázó tényező esetén és relatíve kevés esetszám mellett szinte minden mindig teljes mértékben magyarázható. Hiába igaz az, hogy a tényleges befolyásoló tényezők hiányában áll elő a becslési hiba, ettől még adott szempontrendszer mellett keletkező tény-becslés eltérés információértékkel bír, hiszen éppen azt keresik a modellek, vajon adott szempontmennyiséggel az Y mekkora aránya magyarázható. A gyanú az anti-diszkriminációs modellek esetén az a becslési hiba, ami semmilyen módon nem redukálható, hiszen a figyelembe vett változókon túl a modellező nem tud és nem is kíván más befolyásoló tényezőket figyelembe venni. Így a gyanú az ilyen modellekben a minden(ki) másként egyforma elv megsértésének mértékét fejezi ki.”³⁵

A feldolgozást követően kapott elemzési eredmények alapján, a becslült és a tényleges értékek összevetésre kerülnek a jóságpontokkal. Az összehasonlítás az Excel alkalmazás kimutatás funkciójának segítségével történik, amely lehetővé teszi a szabályrendszer, illetve küszöbérték meghatározását amely alapján megállapítható, hogy mely értékek helyezkednek el a jóságpontok közelében, azaz mely szcenáriók tekinthetők gyanúsak, illetve melyek menthetők fel a gyanú alól.

Darabszám /Becslés	Y =		Végösszeg
Becslési érték	1000	2000	
1013.7	10		10
1520.5	3	4	7
2027.4		8	8
Végösszeg	13	12	25
Küszöbérték = 1500	1500 fölött Y =2000		
	1500 alatt Y =1000		

17. ábra, Szabály meghatározása a becslé és tény értékei alapján színekkel

Forrás: https://miau.my-x.hu/miau/330/sche/valos_OAM_v5_2.xlsx (COCO Rang Direkt&Inverz lapfűl L6:O12)

A becslési értékek alapján megállapítható, hogy azok az esetek tekinthetők gyanúsak, amelyek becslés szerinti értékei aránya közelebb helyezkedik el a **2000**-es jóságponthoz, míg azok az esetek, amelyek értékei az **1000**-es jóságponthoz állnak közelebb, nem tekinthetők gyanúsak. Az értékek áttekinthetősége és összehasonlíthatósága érdekében az elemzés során két

³⁵ My-x Team - Avagy egy innovatív „ötlet-istálló”, 2014, (2026.03.28., 17.o.)

különböző színek kerül alkalmazásra. A kimutatásból megállapítható, hogy a küszöb értéke megközelítőleg **1500** becslési érték.

A következő műveletben a COCO modell által szolgáltatott eredményekből származó becslési érték összevetésre kerül a rangsorolt Objektum–attribútum mátrix adataival, lehetővé téve annak meghatározását, hogy mely értékek milyen irányban térnek el a megállapított küszöbérték alapján. A Becslés oszlop egyértelműen szemlélteti, hogy a modell számításai alapján mely scenáriók, illetve az azokhoz tartozó képek tekinthetők a gyanú alól felmentettnek, és mely esetekben merül fel a mobiltelefonnal történő fényképezés szándéka.

Mértékegység	Sorszám	Sorszám	Sorszám	Sorszám	Sorszám	Sorszám	Jóságpont	Érték	
Rangsorolt OAM Direkt	Szórás	Maximum	Minimum	Átlag	Módusz	Medián	Y (Telefon: igen=2000, nem=1000)	Becslés Direkt	Eredmények
Scenario01	23	1	11	22	1	6	1000	1013.7	IGAZ
Scenario02	12	1	2	14	1	6	2000	2027.4	IGAZ
Scenario03	17	1	23	17	1	6	1000	1013.7	IGAZ
Scenario04	25	1	15	1	1	25	1000	1520.5	HAMIS
Scenario05	15	1	15	7	1	17	1000	1013.7	IGAZ
Scenario06	21	1	15	3	1	22	1000	1013.7	IGAZ
Scenario07	7	1	1	20	1	6	1000	1013.7	IGAZ
Scenario08	4	1	2	23	1	3	1000	1520.5	HAMIS
Scenario09	2	1	11	8	1	22	1000	1520.5	HAMIS
Scenario10	8	1	15	12	1	12	2000	2027.4	IGAZ
Scenario11	18	1	15	4	1	14	2000	2027.4	IGAZ
Scenario12	18	1	15	4	1	14	2000	2027.4	IGAZ
Scenario13	9	1	6	11	1	17	1000	1013.7	IGAZ
Scenario14	11	1	15	10	1	17	1000	1013.7	IGAZ
Scenario15	18	1	15	4	1	14	2000	2027.4	IGAZ
Scenario16	3	1	6	21	1	12	2000	1520.5	IGAZ
Scenario17	16	1	14	9	1	17	1000	1013.7	IGAZ
Scenario18	1	1	6	24	1	1	2000	2027.4	IGAZ
Scenario19	24	1	23	25	1	2	1000	1013.7	IGAZ
Scenario20	12	1	2	14	1	6	2000	2027.4	IGAZ
Scenario21	12	1	2	14	1	6	2000	2027.4	IGAZ
Scenario22	5	1	6	18	1	4	2000	1520.5	IGAZ
Scenario23	10	1	11	13	1	17	1000	1013.7	IGAZ
Scenario24	5	1	6	18	1	4	2000	1520.5	HAMIS
Scenario25	22	1	23	2	1	24	2000	1520.5	HAMIS

18. ábra, Rangsorolt OAM értékeinek összevetése a becslés értékeivel

Forrás: https://miau.my-x.hu/miau/330/sche/valos_OAM_v5_2.xlsx (COCO Rang Direkt&Inverz lapfűl A6:J32)

Az eredmények összegzése során megállapítható, hogy a 25 vizsgált 12 gyanús (2000 jóságpont) és 13 darab gyanútlan (1000 jóságpont) eseményből, a COCO STD modell 15 esetet jelölt meg gyanús, és 10-et gyanútlannak.

Az eredmények oszlop az összevetések eredményét tükrözi, azaz, hogy mely scenárió esetében IGAZ és mely esetben HAMIS a szabály, illetve **küszöbérték szerinti megállapítás**.

Eredmények összegzése		
A COCO STD becslése a 25 darab scenárióból		
Darabszám:	Eredmény:	Százalékos arány
21	IGAZ	84%
4	HAMIS	16%

19. ábra Eredmények összegzése

Forrás: https://miau.my-x.hu/miau/330/sche/valos_OAM_v5_2.xlsx (COCO Rang Direkt&Inverz lapfűl L21:N25)

Az összehasonlítás alapján a modell becslése az esetek 84%-ában helyesnek, míg 16%-ában tévesnek bizonyult.

Ahhoz, hogy a kapott eredményt hitelesnek tekintsük, elkészítésre került az Objektum-attribútum-mátrix inverz rangsorolt változata, amely a direkt OAM értékeinek ellentétes iránya, amely úgy állítható elő, ha az egyes cellák értékeit 26-ból kivonjuk. Ez a megközelítés egyszerűbb és gyorsabb művelet, mint a megfelelő ellentétes irányokkal való önálló sorszám-függvény írása.

Ha az így előállított inverz rangsorolt mátrix eredménye a COCO STD modell futtatását követően ellentétes irányban megegyezik a direkt mátrix eredményével (vö: [Lehrer Zoltán szakdolgozatának 50. oldalán szereplő 10. „Kockázat becslés kimutatás”](#) ábra alapján), abban az esetben a kapott eredményt hitelesnek tekinthetjük.

Mértékegység	Sorszám	Sorszám	Sorszám	Sorszám	Sorszám	Sorszám	Jóságpont	Érték	
Rangsorolt OAM Inverz	Szórás	Maximum	Minimum	Átlag	Módusz	Medián	Y (Telefon: igen=2000, nem=1000)	Becslés Inverz	Eredmények
Scenario01	3	25	15	4	25	20	1000	2055.6	HAMIS
Scenario02	14	25	24	12	25	20	2000	1541.7	IGAZ
Scenario03	9	25	3	9	25	20	1000	1541.7	HAMIS
Scenario04	1	25	11	25	25	1	1000	1027.8	IGAZ
Scenario05	11	25	11	19	25	9	1000	1027.8	IGAZ
Scenario06	5	25	11	23	25	4	1000	1541.7	HAMIS
Scenario07	19	25	25	6	25	20	1000	1541.7	HAMIS
Scenario08	22	25	24	3	25	23	1000	1541.7	HAMIS
Scenario09	24	25	15	18	25	4	1000	1027.8	IGAZ
Scenario10	18	25	11	14	25	14	2000	1541.7	IGAZ
Scenario11	8	25	11	22	25	12	2000	1541.7	IGAZ
Scenario12	8	25	11	22	25	12	2000	1541.7	IGAZ
Scenario13	17	25	20	15	25	9	1000	1027.8	IGAZ
Scenario14	15	25	11	16	25	9	1000	1027.8	IGAZ
Scenario15	8	25	11	22	25	12	2000	1541.7	IGAZ
Scenario16	23	25	20	5	25	14	2000	1541.7	IGAZ
Scenario17	10	25	12	17	25	9	1000	1027.8	IGAZ
Scenario18	25	25	20	2	25	25	2000	1541.7	IGAZ
Scenario19	2	25	3	1	25	24	1000	2055.6	HAMIS
Scenario20	14	25	24	12	25	20	2000	1541.7	IGAZ
Scenario21	14	25	24	12	25	20	2000	1541.7	IGAZ
Scenario22	21	25	20	8	25	22	2000	1541.7	IGAZ
Scenario23	16	25	15	13	25	9	1000	1541.7	HAMIS
Scenario24	21	25	20	8	25	22	2000	1541.7	IGAZ
Scenario25	4	25	3	24	25	2	2000	2055.6	IGAZ

20. ábra, Rangsorolt Inverz OAM értékeinek összevetése a becslés értékeivel

Forrás: https://miau.my-x.hu/miau/330/sche/valos_OAM_v5_2.xlsx (COCO Rang Direkt&Inverz lapfűl Q6:Z32)

A következő lépésben meghatározásra kerülnek a jóságpontok, a küszöbérték és az inverz rangsorolás értékei alapján.

Darabszám / Becslés	Y =		
Becslési érték	1000	2000	Végösszeg
1027.8	6		6
1541.7	5	11	16
2055.6	2	1	3
Végösszeg	13	12	25
Küszöbérték = 1500	1500 fölött Y =2000		
	1500 alatt Y =1000		

21. ábra, Küszöbérték meghatározása az inverz rangsorolás értékei alapján színekkel

Forrás: https://miau.my-x.hu/miau/330/sche/valos_OAM_v5_2.xlsx (COCO Rang Direkt&Inverz lapfűl

AB7:AE12)

Az összegvetés során megállapítható, hogy a 25 vizsgált eseményből, a modell 19 esetet jelölt meg gyanúsként, és 6-ot gyanútlanak.

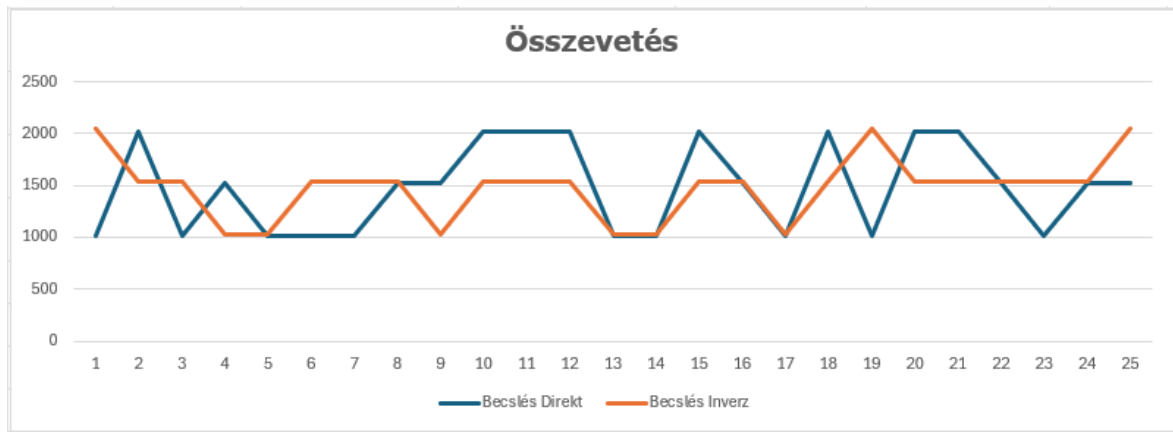
Eredmények összegzése		
A COCO STD becslése a 25 darab scenárióból		
Darabszám:	Eredmény:	Százalékos arány
18	IGAZ	72%
7	HAMIS	28%

22. ábra Inverz mátrix eredmények összegzése

Forrás: https://miau.my-x.hu/miau/330/sche/valos_OAM_v5_2.xlsx (COCO Rang Direkt&Inverz lapfűl AB21:AD25)

Az Inverz OAM-al történő összehasonlítás alapján a modell becslése az esetek 72%-ában helyesnek, míg 28%-ában tévesnek bizonyult.

A már említett esetben, ha a két Objektum-attribútum mátrix eredményeit összevetve egymás tükörképeit kapjuk, akkor a kapott eredmény hitelesnek tekinthető.



23. ábra, Rangsorolt Objektum-attribútum-mátrix direkt és inverz becslés értékének kimutatása

Forrás: https://miau.my-x.hu/miau/330/sche/valos_OAM_v5_2.xlsx (COCO Rang Direkt&Inverz lapfűl I45:O42)

A két diagram összevetése során megállapítható, hogy nem figyelhető meg egyértelmű szabályszerűség, ami a modell alacsony találati arányaira vezethető vissza, amelyet a diagrammokban megjelenő szimmetria hiánya is alátámaszt.

Végül elemzésre kerül a Rangsorolt Objektum-attribútum mátrix direkt és inverz összevonásának elemzése a COCO STD modell használatával, majd azt követően a becslések értékének összevetése és kimutatása.

Mértékegység	Direkt	Direkt	Direkt	Direkt	Direkt	Direkt	Inverz	Inverz	Inverz	Inverz	Inverz	Inverz	Jóságpont	Érték	
Rangsorolt OAM+Inverz	Sorszám	Sorszám	Sorszám	Sorszám	Sorszám	Sorszám	Sorszám	Sorszám	Sorszám	Sorszám	Sorszám	Sorszám	Y (Telefon: Igen=2000,	Becslés	Eredmények
Scenario01	Szórás	Maximum	Minimum	Átlag	Módusz	Medián	Szórás	Maximum	Minimum	Átlag	Módusz	Medián			
Scenario01	23	1	11	22	1	6	3	25	15	4	25	20	1000	1013.7	IGAZ
Scenario02	12	1	2	14	1	6	14	25	24	12	25	20	2000	2027.4	IGAZ
Scenario03	17	1	23	17	1	6	9	25	3	9	25	20	1000	1267.1	IGAZ
Scenario04	25	1	15	1	1	25	1	25	11	25	25	1	1000	1520.5	HAMIS
Scenario05	15	1	15	7	1	17	11	25	11	19	25	9	1000	1013.7	IGAZ
Scenario06	21	1	15	3	1	22	5	25	11	23	25	4	1000	1013.7	IGAZ
Scenario07	7	1	1	20	1	6	19	25	25	6	25	20	1000	1267.1	IGAZ
Scenario08	4	1	2	23	1	3	22	25	24	3	25	23	1000	1267.1	IGAZ
Scenario09	2	1	11	8	1	22	24	25	15	18	25	4	1000	1013.7	IGAZ
Scenario10	8	1	15	12	1	12	18	25	11	14	25	14	2000	2027.4	IGAZ
Scenario11	18	1	15	4	1	14	8	25	11	22	25	12	2000	2027.4	IGAZ
Scenario12	18	1	15	4	1	14	8	25	11	22	25	12	2000	2027.4	IGAZ
Scenario13	9	1	6	11	1	17	17	25	20	15	25	9	1000	1267.1	IGAZ
Scenario14	11	1	15	10	1	17	15	25	11	16	25	9	1000	1013.7	IGAZ
Scenario15	18	1	15	4	1	14	8	25	11	22	25	12	2000	2027.4	IGAZ
Scenario16	3	1	6	21	1	12	23	25	20	5	25	14	2000	1520.5	IGAZ
Scenario17	16	1	14	9	1	17	10	25	12	17	25	9	1000	1013.7	IGAZ
Scenario18	1	1	6	24	1	1	25	25	20	2	25	25	2000	2027.4	IGAZ
Scenario19	24	1	23	25	1	2	2	25	3	1	25	24	1000	1013.7	IGAZ
Scenario20	12	1	2	14	1	6	14	25	24	12	25	20	2000	2027.4	IGAZ
Scenario21	12	1	2	14	1	6	14	25	24	12	25	20	2000	2027.4	IGAZ
Scenario22	5	1	6	18	1	4	21	25	20	8	25	22	2000	1520.5	IGAZ
Scenario23	10	1	11	13	1	17	16	25	15	13	25	9	1000	1013.7	IGAZ
Scenario24	5	1	6	18	1	4	21	25	20	8	25	22	2000	1520.5	IGAZ
Scenario25	22	1	23	2	1	24	4	25	3	24	25	2	2000	1013.7	IGAZ

24. ábra, Rangsorolt Objektum-attribútum-mártix direkt és inverz összevonásának elemzése a COCO STD modell használatával

Forrás: https://miau.my-x.hu/miau/330/sche/valos_OAM_v5_2.xlsx (COCO Rang Direkt+Inverz lapfűl A6:P33)

Továbbá meghatározásra kerülnek a jóságpontok a COCO STD által megállapított értékei alapján.

Darabszám / Becslé	Y =		
Becslési érték	1000	2000	Végösszeg
1013.7	8		8
1267.1	4		4
1520.5	1	4	5
2027.4		8	8
Végösszeg	13	12	25
Küszöbérték = 1500	1500 fölött Y=2000		
	1500 alatt Y=1000		

25. ábra, Küszöbérték meghatározása a Direkt és Inverz összevonásával

Forrás: https://miau.my-x.hu/miau/330/sche/valos_OAM_v5_2.xlsx (COCO Rang Direkt+Inverz lapfűl S8:V14)

Az így kapott eredmények alapján megfigyelhető, hogy a két mátrix egyesítésével végrehajtott modellfuttatás során, jelentősen javulás érhető el a találati arány tekintetében.

Eredmények összegzése		
A COCO STD becslése a 25 darab scénárióból		
Darabszám:	Eredmény:	Százalékos arány
24	IGAZ	96%
1	HAMIS	4%

26. ábra, Direkt és Inverz mátrix összevont eredmények összegzése

Forrás: https://miau.my-x.hu/miau/330/sche/valos_OAM_v5_2.xlsx (COCO Rang Direkt+Inverz lapfűl

S22:U26)

A két Objektum-attribútum mátrix egyesítése során a modell becslése alapján az esetek **96%**-ában helyesnek, és csak **4%**-ában bizonyult tévesnek. Ezzel a módszerrel a találatok szempontjából jelentős javulás érhető el.

3.1.8 Objektum-attribútum-mátrix elemzése a ChatGPT használatával

Az objektum–attribútum mátrix elemzések során a kiértékelés és eredmények támogatására mesterséges intelligencián alapuló nyelvi modell került alkalmazásra. A modell irányt mutatott az adatok közötti összefüggések feltárásában, valamint a szabályrendszer kialakításának támogatásában.

A következő feladatot szeretném megoldani: A feltöltött táblázatban az OAM lapon található A1 és I29 cella adatok alapján, készítsd el a becslést, melyel határozd meg mely sorok tekinthetők gyanúsak és melyek nem. A gyanúgenerálás jóságpontjai a H oszlopban találhatóak, a gyanús események 2000 a gyanútlan események pedig 1000 pontokkal lettek jutalmazva. A becslésen kívül más megoldást is alkalmazhatsz.



Megnéztem az OAM lapot, és a feladat logikája alapján ez egy klasszikus kétosztályos (gyanús / nem gyanús) probléma, ahol a H oszlop pontszámai már egyfajta „tanító jelként” működnek:

- 2000 pont → gyanús
- 1000 pont → nem gyanús

Alap megfigyelés

A pontszámok eloszlása alapján:

- medián \approx 1000
- felső kvartilis \approx 2000

Ez nagyon erősen azt sugallja, hogy a rendszer már diszkrét kategóriákat használ.

✓ Egyszerű és működő becslés

A legegyszerűbb (és ebben az esetben teljesen indokolt) becslés egy küszöb alapú osztályozás:

Szabály:

- ha pont ≥ 1500 → gyanús
- ha pont < 1500 → nem gyanús

Eredmény:

- Gyanús sorok: 24 db
- Nem gyanús sorok: 69 db

📊 Miért jó ez a becslés?

- A 1000 és 2000 pontok között természetes választóvonal a 1500
- A pontok nem folytonosak, hanem kvázi két klasztert alkotnak
- Ez gyakorlatilag egy optimális Bayes-döntési határ, ha nincs átfedés

🔴 Alternatív (jobb) megközelítések

Ha nem csak a H oszlopot akarod használni:

1. Klaszterezés (unsupervised)

- pl. K-means ($k=2$)
- automatikusan szétválasztja:
 - alacsony pont → nem gyanús
 - magas pont → gyanús

2. Logisztikus regresszió

Ha más oszlopok (pl. szórás, intenzitás stb.) is számítanak:

- bemenet: többi oszlop
- cél: H oszlopból képzett címke

3. Z-score alapú anomália detektálás

- extrém értékek → gyanús
- jól működik, ha a pontszám nem lenne diszkrét

⚠ Fontos észrevétel az A1 és I29 cellákra

- A1 = "Nyers OAM" → inkább metaadat (nem numerikus)
- I29 = 1 → valószínűleg egy paraméter vagy jelölő

👉 Ezek önmagukban nem adnak erős prediktív információt, ezért a valódi döntéshez a H oszlop kulcsfontosságú.

27. ábra Beszélgetés részlete a ChatGPT nagy nyelvi modellel

3.2 Adatfeldolgozás és gyanúgenerálás objektumfelismerési modell alkalmazásával

Ebben a fejezetben azt vizsgáljuk, hogy egy objektumfelismerési modell, milyen hatékonysággal képes különböző szimulált helyzetekben felismerni a mobiltelefonokat.

3.2.1 A gyanúgenerálás folyamata

A gyanúgenerálás folyamata a képfeldolgozás és az objektumdetektálás eredményeinek elemzésén alapul. A rendszer célja, hogy automatikusan azonosítsa a kameraképen megjelenő mobiltelefonokat, és azok észlelése esetén megfelelő jelzést generáljon.

A gyanúgenerálásra szolgáló rendszer Python programozási nyelven került implementálásra, a modell futtatásához szükséges könyvtárak alkalmazásával. A működés során a rendszer a bemeneti képkockákat feldolgozza, majd a detektálási eredmények alapján eldönti, hogy indokolt-e gyanújelzés generálása. A folyamat több egymást követő lépésből áll.

A folyamat első lépéseként a rendszer a kamera által szolgáltatott képkockát beolvassa és előfeldolgozza. Ezt követően a képkocka a detektáló modell bemenetére kerül, amely elemzi a képet és meghatározza az azon található objektumokat.

A modell, minden detektált objektumhoz hozzárendeli annak osztályát, valamint az objektumot körülvevő határoló doboz (bounding box) koordinátáit. Amennyiben a detektált objektum a mobiltelefon osztályba sorolható, a rendszer gyanúeseményt generál. A jelzés lehet például egy figyelmeztetés, vagy naplóbejegyzés.

3.2.2 Mobiltelefon detektálás megvalósítása

A detektálási folyamat megvalósításához az alábbi komponensek szükségesek: (vö.: [3.4.3 fejezet](#))

- Integrált fejlesztői környezetre, (például: VsCode)
- Python programozási és könyvtár környezet telepítésére,
- Egy működő webkamerára,
- És a YOLO modell-fájlra.

A detektáló modell futtatásához az Ultralytics YOLO implementáció kerül alkalmazásra, míg a kamera képkockáinak kezeléséhez az OpenCV könyvtár modul biztosította a szükséges funkciókat.

A teljes gyanúgenerálási folyamat az alábbi lépésekből áll:

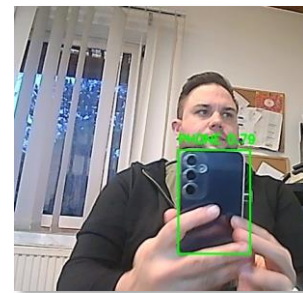
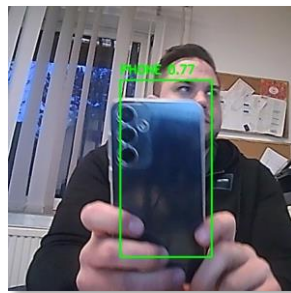
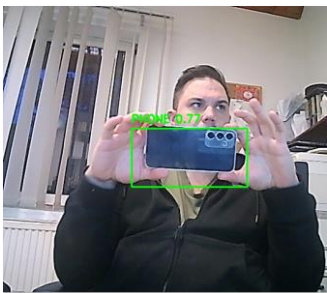
1. A kamera képkockájának beolvasása,
2. A képkocka továbbítása az objektumdetektáló modell számára,
3. A detektált objektumok osztályának és pozíciójának meghatározása,
4. A mobiltelefon kategóriába tartozó objektumok azonosítása,
5. Gyanújelzés generálása mobiltelefon észlelése esetén határoló dobozzal.

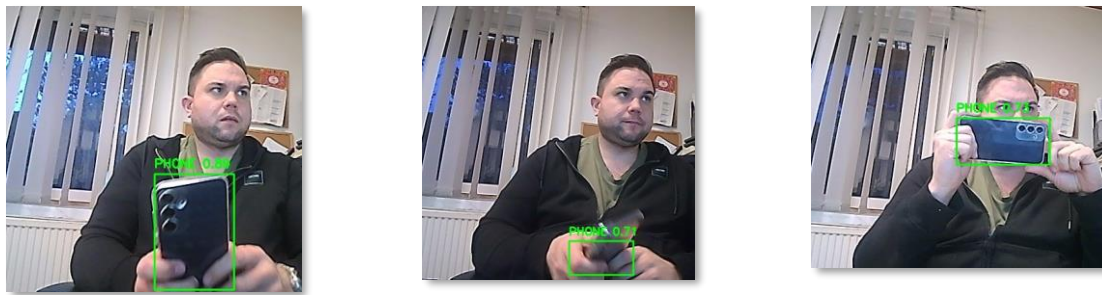
3.2.3 Elemzés

A modell működésének ellenőrzése, valós idejű kameraképen történt. A rendszer a bemeneti képkockákon objektumdetektálást végzett, és az eredményeket határoló dobozok (bounding boxok) formájában jelenítette meg.

A tesztelés során a következő paraméterek kerültek beállításra:

- képméret: 640 px,
- confidence küszöb: 0.85–0.93,
- többszörös frame megerősítés a stabil detektálás érdekében.





28. ábra, Sikeres detektálás reprezentációja

3.2.4 Eredmények

A modell alkalmazása során megállapítható volt, hogy a módszer lehetővé teszi egy olyan automatikus rendszer kialakítását, amely valós időben képes a mobiltelefonok detektálására, valamint az ehhez kapcsolódó események rögzítésére.

3.3 Módszerek összehasonlítása

A gyanúgenerálás során alkalmazott két megközelítés eltérő elven működik.

Az Objektum–attribútum mátrix egy szabályalapú megközelítés, amely az objektumok előre definiált jellemzőit vizsgálja. Előnye, hogy nem igényel nagy számítási kapacitást és alkalmazása jól átlátható. Ugyanakkor, mivel csak egy előre meghatározott szabályok alapján képes döntéseket hozni, így nem kezeli a változatos, vagy váratlan helyzeteket. A szürkeárnyalatos képekből származó adatok nagymértékben korlátozzák a detektálás pontosságát, mivel a színinformáció hiánya megnehezíti az objektumok megbízható azonosítását. Az egyszerű teszt, illetve a 25 darab kép adatainak feldolgozása során megállapítható, hogy a módszer önmagában nem alkalmas a pontos és azonnali objektumdetektálásra.

Ezzel szemben a mélytanulás alapú objektumfelismerés, jelen esetben a YOLO modell egy tanuló algoritmusra épül, amely automatikusan képes a képi adatokból jellemzőket kinyerni és új mintázatokat felismerni. Előnye a nagyobb pontosság és rugalmasság, különösen komplex környezetben. Hátránya viszont a magasabb számítási igény, valamint a tanításhoz szükséges nagy mennyiségű adat.

3.4 Védelmi szoftver tervezése és megvalósítása

A fejezet, egy tervezett védelmi szoftver felépítését és megvalósítását ismerteti. Bemutatásra kerül a rendszer architektúrája, működési folyamata, valamint a megvalósítás során alkalmazott technológiák. A cél egy hatékony, egyszerű és átlátható, mobiltelefon-detektáláson alapuló védelmi megoldás bemutatása.

3.4.1 A tervezett védelmi szoftver architektúrája

A rendszer működése egy folyamatos adatfeldolgozási ciklusra épül, amely során a bemeneti képi adatok feldolgozása több egymást követő lépésben történik. Az architektúra kialakításánál fontos szempont volt a valós idejű működés biztosítása, és az alacsony erőforrásigényre törekvés.

Az architektúra fő lépései egy lineáris feldolgozási folyamatot alkotnak, amely az alábbi műveletekből áll:

1. Adatgyűjtés,
2. Képfeldolgozás és objektumfelismerés,
3. Szűrés és validáció,
4. Esemény naplózása,
5. Reakciókezelés.

Ez a felépítés lehetővé teszi a rendszer skálázhatóságát és a későbbi bővíthetőséget, például további detektálási, elemzési, vagy riasztási funkciók integrálásával. (vö.: [7.2 fejezet](#))

3.4.2 Működési folyamat

A rendszer valós időben elemzi a kameraképet és amennyiben felismeri a mobiltelefon jelenlétét, eseményt generál, majd a gyanús események naplózásra kerülnek.

A folyamat elindítása után a Python programnyelv használatával vezéreljük a webkamerát és a valós idejű kép rögzítését.

Az objektum érzékelése egy előre betanított modell, jelen esetben a YOLO11n (nano) modell használatával történik, amely a modell legkevésbé erőforrásigényes változata.

Ha a gyanúgenerálás, azaz a mobiltelefon felismerés sikeres, akkor a riasztási folyamat megkezdődik:

1. Létrehozásra kerül a helyi tárolón;

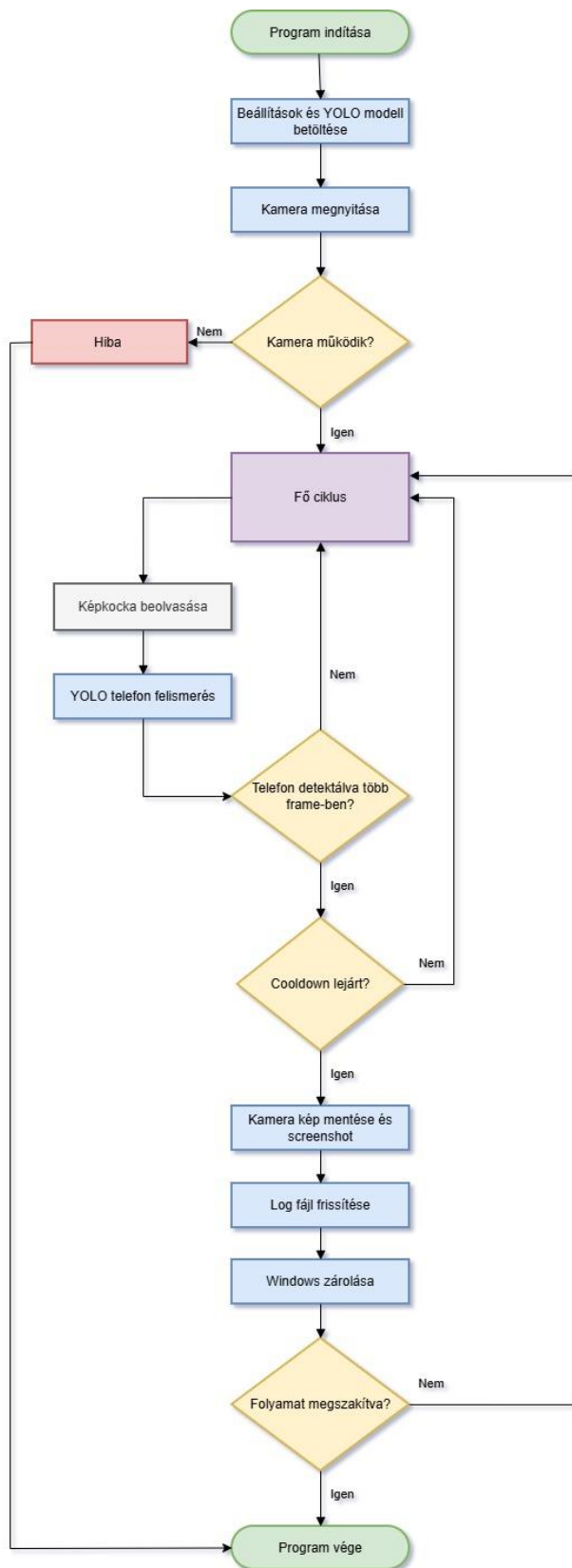
- kép a gyanúba keveredett felhasználóról,
- kép a fényképezni kívánt képernyőről,
- naplófájl, mely az esemény időpontját, a bejelentkezett felhasználónevet és a munkaállomás nevét tartalmazza.

A naplófájlok lehetnek ideiglenes fájlok, melyek csak azon ideig tárolódnak a helyi munkaállomáson, amíg továbbításra nem kerülnek a vezérlőgépnek, illetve kiszolgálónak. Amennyibe a hálózati kapcsolat nem megfelelő, úgy egy megadott időközönként a küldési ciklus mindaddig ismétlődhet, még a fájlok továbbítása sikeres nem lesz. (lásd.:[7.1.16 fejezet](#))

2. Az eltulajdonítani kívánt képernyőn lévő adatok elrejtésre kerülnek, illetve a számítógép zárolása megtörténik.

3. Ezután a folyamat visszatér a megfigyeléshez.

A rendszer megszakítható a szoftver kezelője által.



29. ábra, Tervezett védelmi szoftver folyamatábrája

3.4.3 Rendszerkomponensek bemutatása, implementáció

A rendszer több egymással együttműködő komponensből épül fel, amelyek együttesen biztosítják a mobiltelefonok detektálását és a rendszer megfelelő működését. A moduláris felépítés lehetővé teszi a rendszer áttekinthetőségét, valamint megkönnyíti az egyes komponensek fejlesztését és karbantartását.

Az **adatgyűjtő** komponens felelős a bemeneti adatok biztosításáért. Egy webkamera segítségével folyamatosan képeket rögzít a megfigyelt környezetről, amely a további feldolgozás alapját képezi.

Az **képfeldolgozó és objektumfelismerő** komponens a rendszer központi eleme, amely egy YOLO alapú neurális hálózat segítségével elemzi a képeket. Az elemzést előtt a nyers képi adatok átalakításra kerülnek, amely során megtörténik a képek átméretezése és normalizálása, hogy azok megfeleljenek a neurális hálózat bemeneti követelményeinek.

A **szűrés és validációs** komponens elemzi a detektálási eredményeket, és meghatározza, hogy az adott esemény jelent-e biztonsági kockázatot. A döntés különböző feltételek alapján történhet, például az objektum jelenlétének időtartama, vagy gyakorisága alapján.

Eseménygeneráló komponens végzi az események naplózását. Amennyiben a feltételek teljesültek, úgy megtörténik a kamerakép készítése, képernyőkép lementése és az esemény naplózása szöveges fájlba.

A **reakciókezelő** komponens felelős a rendszer válaszlépéseiért. Ez magában foglalhatja a riasztások generálását, vagy egyéb automatikus intézkedések végrehajtását.

A **megjelenítő** komponens a detektálás fejlesztésének a támogatására került implementálásra. A felhasználói visszajelzés érdekében, a rendszer grafikus ablakban jeleníti meg a feldolgozott videóképet. A megjelenítés opcionális, és a program megfelelő részének módosításával kikapcsolható.

A komponensek működését egy háttérben futó **vezérlő folyamat (fő ciklus)** biztosítja, amely koordinálja az adatáramlást és a feldolgozás lépéseit.

Komponensek	Implementáció
<p>Adatgyűjtés</p> <p>A rendszer az OpenCV könyvtár VideoCapture osztályát használja a kamera elérésére.</p> <p>Feladata:</p> <ol style="list-style-type: none"> 1. kamera inicializálása, 2. képkockák folyamatos beolvasása, 3. frame továbbítása a feldolgozó modulnak. 	<pre>cap = cv2.VideoCapture(0) ret, frame = cap.read()</pre>
<p>Képfeldolgozás</p> <p>A képfeldolgozás a modell segítségével történik, feladata, a kamerából érkező képkockák előkészítése a neurális háló számára. Az erőforrás csökkentése érdekében a bemeneti kép mérete is csökkentésre kerül.</p> <p>A jelenlegi érték csökkenti a számítási igényt, különösen CPU használata esetén.</p>	<pre>imgsz=320</pre>
<p>Objektumfelismerés</p> <p>A rendszer legfontosabb eleme, amely az Ultralytics YOLOv11n implementációját használja, amelynek feladata:</p> <ul style="list-style-type: none"> • YOLO modell betöltése 	<pre>MODEL = "yolo11n.pt"</pre>

- Objektumdetektálás végrehajtása

A modell minden detektált objektumhoz visszaadja:

- osztályazonosítót
- confidence értéket
- bounding box koordinátákat

Szűrés és validáció

A neurális hálók hajlamosak hamis pozitív detektálásokra, ezért a rendszer több szűrést alkalmaz.

A validáció feltételei:

- Confidence küszöb

Csak a megfelelően biztos detektálások kerülnek feldolgozásra.

- Bounding box méret

Kiszűri a túl kicsi vagy zajos detektálásokat.

- Osztályszűrés

A modell több mint 80 különböző objektumosztály felismerésére képes.

```
results = model(frame, imgsz=320)
```

```
for r in results:  
    for box in r.bboxes:
```

```
        cls = int(box.cls[0])
```

```
        conf = float(box.conf[0])
```

```
        x1, y1, x2, y2 = map(int, box.xyxy[0])
```

```
CONF_THRESHOLD = 0.88
```

```
MIN_BOX_SIZE = 80
```

```
if cls == 67
```

Telefon osztály azonosítója a 67-es érték. A paraméter megadásával a detektálás során a modell kizárólag a mobiltelefonok felismerésére fókuszál, így csökkenthető a felesleges detektálások száma és növelhető a feldolgozás hatékonysága.

- Frame stabilizáció modul

Egyetlen frame detektálása nem elegendő egy esemény kiváltásához, ezért a rendszer egymást követő képkockákat vizsgál. Ha a telefon legalább **három** egymást követő képkockában jelen van, akkor a detektálás valószínűleg tekinthető. Ezzel jelentősen csökkenthetjük a hamis riasztások számát.

```
REQUIRED_FRAMES = 3
```

A rendszer biztosítja, hogy rövid időn belül ne történjen túl sok riasztás.

```
COOLDOWN = 15
```

Ehhez egy **cooldown** (lehülés) mechanizmus kerül alkalmazásra, mely szerint, ha egy esemény megtörtént, a rendszer 15 másodpercig nem indít újabb riasztást.

Eseménygeneráló

Esemény esetén a rendszer dokumentálja a történeteket.

A következő adatok kerülnek mentésre:

- Kamerakép:

```
cv2.imwrite(cam_img, frame)
```

Amely az OpenCV könyvtár egyik alapvető függvényének használatával kerül mentésre.

- Képernyőkép

A képernyőkép a PyAutoGUI könyvtár segítségével készül.

- Log fájl

A log fájl tartalmazza az esemény időpontját, a bejelentkezett felhasználó és munkaállomás neveket.

Reakciókezelő

A rendszer végső reakciója a számítógép azonnali zárolása.

A folyamat Windows API segítségével történik.

Megjelenítő modul (csak az események megjelenítéséhez)

A felhasználói visszajelzés érdekében **kizárólag a fejlesztési műveletekhez**, a rendszer grafikus ablakban jeleníti meg a feldolgozott videóképet. A kód futásának elhagyásával a kamerakép megjelenítése megszüntethető. például:

```
screenshot = pyautogui.screenshot()  
screenshot.save(screen_img)
```

```
USERNAME = getpass.getuser()  
WORKSTATION = socket.gethostname()
```

Példa:

```
PHONE DETECTED 20260408_09_37 | USER:  
Marika | WORKSTATION: Humán08
```

```
ctypes.windll.user32.LockWorkStation()
```

```
cv2.imshow("YOLO Phone Detection",  
frame)
```

```
#cv2.imshow("YOLO Phone Detection",  
frame)
```

A megjelenítés tartalmazza:

- detektált objektum bounding box,
- confidence érték,
- objektum címke.



30. ábra Részlet egy naplóképről

3.4.4 Fejlesztői környezet

A jelen fejezetben, csak a védelmi szoftver fejlesztéséhez szükséges hardver és szoftverelemek kerülnek ismertetésre. A modell fejlesztéséhez szükséges hardver elemek a hasznosság fejezetben kerülnek bemutatásra. (vö.: [1.4 fejezet](#))

3.4.4.1 Hardver követelmények

A program valós idejű képfeldolgozást végez, amelyhez szükségesek bizonyos hardverkövetelmények teljesítése.

Számítógép minimum követelmények:

- 64 bites processzor (Intel i5 vagy hasonló)
- 8 GB RAM
- webkamera
- legalább 50 GB szabad tárhely

Ajánlott konfiguráció:

- NVIDIA GPU (Csak az NVIDIA GPU támogatja a CUDA-t) (vö.:[8.1.2 alfejezet](#))
- legalább 16 GB RAM

GPU használatával a neurális háló **gyorsabban fut.**

Webkamera minimum követelménye:

- Felbontás: legalább 720p, (ideálisan 1080p)
- Képkockasebesség: minimum 20–30 FPS
- Csatlakozás: USB (stabil adatátvitel)
- Képminőség: megfelelő fényviszonyok mellett tiszta kép.

A tesztelésekhez és fejlesztéshez használt kamerák típusai:

- Alcor AWA-1080 (1080p, 30FPS)
- Logitech QuickCam C500 (1024p, 30FPS)

3.4.4.2 Operációs rendszer

A program fejlesztése Microsoft Windows 11 operációs rendszeren történt.

A kódban szereplő utasítás: `ctypes.windll.user32.LockWorkStation()`, a Windows API (vö.: 8.3 fejezet) egyik függvényét hívja meg, amely a számítógép zárolását végzi el. Emiatt a program közvetlenül csak Windows rendszer alatt működik.

Linux vagy Apple macOS operációs rendszer alatt a zárolási mechanizmust külön kell implementálni.

3.4.4.3 Programozási nyelv és futtatókörnyezet

A program Python programozási nyelven történő fejlesztés miatt, szükség volt a Python interpreter telepítésére. Az interpreter felelős a programkód végrehajtásáért, valamint a külső könyvtárak betöltéséért.

Ajánlott verzió: Python **3.9 – 3.12**.

3.4.4.4 Python könyvtárak

A program több külső könyvtárat használ, amelyek külön telepítést igényelnek. Ezek a könyvtárak, speciális funkciókat biztosítanak és modulként használhatóak a fejlesztések során.

A működéshez szükséges főbb könyvtárak:

OpenCV: Az OpenCV (Open Source Computer Vision Library) egy széles körben használt képfeldolgozó könyvtár. A programban kezeli:

- a webkamera megnyitását;
- a képkockák beolvasását;
- a téglalapok kirajzolását;
- a képek mentését;
- a videó megjelenítését.

Ultralytics YOLOv11: Ez a könyvtár tartalmazza a *YOLO*³⁶*v11* objektumfelismerő neurális hálót, melynek feladata:

- tárgyak felismerése a képen,
- objektumok osztályozása,
- bounding box koordináták visszaadása,
- felismerési valószínűség számítása.

PyAutoGUI: A PyAutoGUI egy automatizációs könyvtár, amely lehetővé teszi a felhasználói felület vezérlését. Ebben a fejlesztésben a képernyőkép készítésének a szerepét látja el.

További könyvtárak:

Time: A Time könyvtár modul az idő és dátum mérése szolgál.

Ctypes: Szerepe a Windows funkciók futtatása. Ebben az implementációban a számítógép zárolását végzi el.

Os: modell alkalmazásával fájlkezelő műveletek végezhetőek el.

Getpass, Socket a felhasználó nevének lekérdezéséhez, még a Socket a számítógép nevének a lekérdezéséhez került implementálásra.

Datetime: pedig az időbélyeg generálására szolgál ebben a fejlesztésben.

Ezek a könyvtárak együttesen, lehetővé teszik a rendszer hatékony működését.

³⁶ Ultralytics, "Ultralytics YOLO Documentation." <https://docs.ultralytics.com/> (2026)

3.4.4.5 Integrált fejlesztői környezet

A fejlesztés során a *Visual Studio Code* alkalmazás került használatba, mivel az egy könnyen kezelhető, széles körben elterjedt fejlesztői környezet, amely számos programozási nyelvet és kiegészítőt támogat. Emellett egyetemi tanulmányaim során is ezt a fejlesztői környezetet alkalmaztam, így annak használata számomra már jól ismert volt.

Az alkalmazás többek között biztosítja:

- a szintaxis kiemelést,
- hibakeresést,
- kódkiegészítést,
- integrált terminált.

3.4.4.6 Projekt könyvtárszerkezet

A projekt az alábbi struktúrát követi:

```
Cellphoneblck_project
├── cellphoneblck_vxy.py
├── yolov1 ln.pt
├── captures
└── logs.txt
```

A captures mappába kerülnek:

- kamera képek
- képernyőképek

A logs.txt fájl tartalmazza az események naplózását.

3.4.4.7 Telepítés lépések

A Python telepítése a saját weboldaláról letöltött .exe fájl futtatásával végezhető el.³⁷ A telepítés során fontos bejelölni az "Add Python to PATH" opciót, mert így a Python parancs a parancssorból is elérhető.

³⁷ Python, <https://www.python.org/downloads/> (letöltve, 2026.02.23)

Az össze szükséges Python könyvtár telepítése, néhány parancsból elvégezhető;

pl.: `pip install ultralytics opencv-python pyautogui numpy ...`

3.5 Tesztelés és esettanulmányok

A tesztelés célja annak vizsgálata, hogy a rendszer milyen hatékonysággal képes felismerni a mobiltelefon jelenlétét és a potenciális tiltott adatszerzési szándékot.

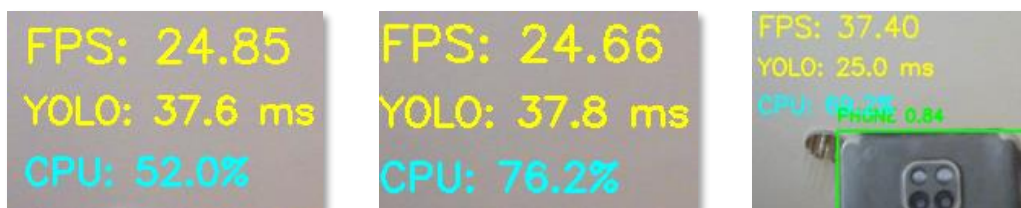
A mobiltelefon felismerő rendszer fejlesztése során fontos a teljesítménymérés, amely segít a rendszer hatékonyságának és működésének értékelésében. Különösen lényeges a feldolgozási sebesség és az erőforrás-felhasználás vizsgálata, mivel ezek befolyásolják a valós idejű működést.

3.5.1 Tesztelési környezet

A szoftver tesztelése kontrollált irodai és otthoni környezetben történt, különböző fényviszonyok és távolságok mellett.

3.5.2 Teljesítményelemzés

A mérés során elsősorban a Frames Per Second (FPS) azaz képkocka/másodperc és a processzor Central Processor Unit (CPU) terhelés kerül elemzésre. Az FPS a rendszer gyorsaságát, míg a CPU terhelés erőforrásigényét mutatja. Ezek alapján megállapítható, hogy a rendszer mennyire stabil és hogy szükséges-e további optimalizálás.



31. ábra YOLO11n modell teszteredményei

3.5.2.1 FPS teszt

A feldolgozás során, átlagosan valós időben 25–40 FPS között működött miközben magas felismerési arányt biztosított. Az FPS a tesztelésnél azért kulcsfontosságú, mivel azt mutatja meg, hogy a rendszer milyen gyorsan dolgozza fel a képkockákat valós időben. Ha alacsony az

FPS, akkor a modell későn észlelhet és kihagyhat eseményeket. Az FPS érték a rendszer valós idejű működésének mérőszáma, amely meghatározza a feldolgozás sebességét és **közvetlen hatással van a detektálás reakcióidejére és stabilitására.**

3.5.2.2 CPU terhelés teszt

A processzor terhelés a rendszer erőforrás-kihasználtságát jelzi, amely hatással van a feldolgozási sebességre, a stabil működésre és a valós idejű alkalmazhatóságra. A mérések során a processzor terhelése nagyon magas értéket mutatott, még a **YOLO11 nano** modell használata esetén is, ami a rendszer jelentős erőforrás-igényére utal.

A processzor terhelés csökkentése érdekében több kísérlet is alkalmazásra került. A bemeneti kép mérete, a feldolgozási gyakoriság és a modell komplexitása került optimalizálásra, amely ugyan csökkentette az erőforrásigényt, azonban bizonyos mértékben hatással volt a felismerés pontosságára és a rendszer reakcióidejére.

A Teszt során az alkalmazásra kerülő terhelést csökkentő paraméterek:

- `FRAME_SKIP = 6` – megadásával a modell csak minden 6-dik frame-en futott, de így a reakcióidő lassabb volt.
- `YOLO_INTERVAL = 1.5` – a minimum időt szabja meg két futás között.
- `IMG_SIZE = 320` – a modellbe kerülő bemeneti kép csökkentésével a hamis negatív esetek száma növekedett.

3.5.3 Kockázatelemzés és korlátok

A rendszer fő kockázata a **hamis pozitív és hamis negatív esetek** detektálása, amely a modell fejlesztésével és finomhangolással csökkenthető.

Az eredmények elemzése rámutatott arra, hogy bizonyos esetekben más objektumok is mobiltelefonként kerülhetnek detektálásra, ami a modell további finomhangolását teszi szükségessé. A rendszer kialakítása során indokolt a hamis pozitív találatok tolerálása, mivel ezek manuális felülvizsgálat során hatékonyan kiszűrhetők, szemben a hamis negatív esetekkel, amelyek **biztonsági kockázatot jelenthetnek.**

3.5.3.1 Hamis pozitív események szemléltetése

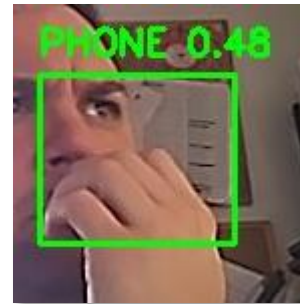
A hamis pozitív esetek részben csökkenthetők a modell finomhangolásával, valamint a confidence küszöb megfelelő megválasztásával. A küszöbérték meghatározza, hogy a rendszer milyen megbízhatósági szint felett fogad el egy detektálást, ezáltal befolyásolja a detektálás pontosságát és a téves riasztások számát.



Az eredmény 72% eséllyel mobiltelefon, vagy hasonló tárgy, mégis csak egy pénztárca. A modell fejlesztésre szorul.



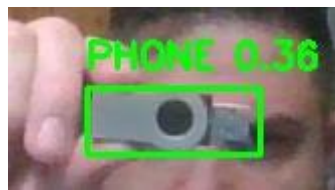
Kevesebb Frame beállítás.



Alacsony Confidence küszöb.



Alacsony Confidene küszöb



Alacsony Confidene küszöb és a Frame box mérete túl kicsi.



Alacsony Confidene küszöb.

32. ábra, Hamis pozitív esetek esetleges okai

3.5.3.2 Hamis negatív események szemléltetése

A hamis negatív esemény olyan eset, amikor a rendszer nem ismeri fel a valójában jelen lévő objektumot. A detektálás során ez azt jelenti, hogy az adott objektum – például egy mobiltelefon – szerepel a képen, azonban a modell nem rendel hozzá határoló dobozt (bounding box), így a rendszer nem generál jelzést.

Jellemző okok:

- az objektum részben takarásban van,
- kedvezőtlen fényviszonyok,
- kis méretű az objektum a képen,
- a modell nem látott hasonló példát a tanítás során,
- túl magas confidence küszöb.

3.6 Forgatókönyvek

A vizsgálat során több forgatókönyv került szimulálásra. Különböző típusú készülékek, részben takart eszköz, gyors mozgás, különböző távolság, valamint különböző fényviszonyok alkalmazása.

3.7 Teszt eredmények összegzése

Az eredmények azt mutatták, hogy a modell a legtöbb esetben sikeresen detektálta a mobiltelefonokat, azonban éles környezetben még a jellemzően maximális pontosságú YOLO11x modell további fejlesztések nélkül sem alkalmas egy hatékony védelmi szoftver komponenseként való alkalmazásra. A hamis pozitív és hamis negatív detektálások számának csökkentése érdekében szükséges egy egyedi, **célzottan tanított modell** alkalmazása, amely azonban időigényes folyamat.

4 Vita

A védelmi szoftver fejlesztése során a legfontosabb kérdés a gyanúgenerálás, azaz az objektumfelismerés megbízhatósága és hatékonysága. A módszerek alkalmazása során, központi problémát jelent az alakfelismerés.

Az Objektum-attribútum mátrix esetében a hatékonyságot nem kizárólag a képméret vagy a felbontás határozza meg, hanem a képen megjelenő objektumok összképe. Felmerül a kérdés, hogy nagyobb felbontású, így nagyobb adatállománnyal rendelkező képek, valóban biztosítanak-e többletinformációt a detektálás szempontjából, illetve, hogy ez a többlet milyen mértékben járul hozzá a felismerés pontosságának növeléséhez, valamint, hogy mennyi és milyen változatosságú scenárió szükséges a jóval megbízhatóbb felismeréshez.

A YOLO modell alkalmazása során a kérdés az, hogy a feldolgozás során a képek felbontásán túl, mekkora és milyen összetételű tanítóadatra van szükség a minél pontosabb felismerés eléréséhez.

5 Következtetések

Az objektumdetektálás tekintetében az Objektum-attribútum mátrix esetében, a vizsgálat során felhasznált 25 darab szürkeárnyalatos kép alapján, lehetőség nyílt a gyanú generálására a képpontadatok értékeinek COCO modell alapján történő becsléseinek elemzésével. Az értékek segítségével közelítő becslés adható arra vonatkozóan, hogy mely esetek tekinthetők gyanúsak. Az eredmények azt mutatják, hogy még viszonylag egyszerű adatrepresentáció esetén is kimutathatók bizonyos mintázatok, amelyek a gyanú meghatározásához felhasználhatók. Viszont ezek az eredmények nem tekinthetők kellően pontosnak, mivel az intenzitástartomány értékei önmagukban nem elegendők annak egyértelmű meghatározásához, hogy a képen valóban mobiltelefon szerepel, vagy egy ahhoz hasonló tárgy.

Feltételezhető, hogy nagyobb felbontású képek esetén még pontosabb eredmények érhetőek el, mivel ezek több vizuális információt tartalmaznak. Ennek köszönhetően a gyanúgenerálás folyamata részletesebben elemezhető lenne.

Ezzel szemben a YOLO alapú megközelítés, amely egy mélytanulási modellre épül, képes a képi adatokból automatikusan jellemzőket kinyerni és komplex mintázatokot felismerni. Ennek köszönhetően **nagyobb pontosságot és jobb képességet biztosít**, különösen változatos és valós környezetben. Hátránya azonban a magasabb számítási igény, valamint a tanításhoz szükséges nagy mennyiségű **annotált** adat. A YOLO modell az Objektum-attribútum-mátrix-szal ellentétben alkalmas a valós idejű objektumdetektálásra.

6 Összefoglalás

A szakdolgozat átfogó képet alkotott az objektumdetektálásról, egy szoftver nélküli módszerről és egy modell-alapú objektumfelismerést alkalmazó egyszerű biztonsági védelmi szoftver fejlesztéséről, amely hatékony megoldást kínálhat a mobiltelefonos képalkotáson alapuló adateltulajdonítás problémájára.

A prototípus szoftver fejlesztésének befejezését követően, a rendszer még nem lesz képes teljeskörűen ellátni adatvédelmi funkcióját, ugyanakkor alkalmas lehet a jogosulatlan adateltulajdonításra utaló események és bizonyítékok rögzítésére a zároló folyamat alkalmazása nélkül.

Az alkalmazás saját adataink védelmére is felhasználható, mivel előfordulhat, hogy munkavégzés során – különösen érzékeny adatok kezelése esetén – egy jogosulatlan személy a tudtunk nélkül próbál adatot eltulajdonítani, például a háttérből vagy egy pillanatnyi távollétünkben. Ilyen esetekben a rendszer által végrehajtott zárolási folyamat hatékony védelmi mechanizmust jelenthet.

7 Jövőkép

A rendszer jelenleg prototípus szinten működik, ezért további fejlesztések szükségesek a megbízhatóság és a funkcionalitás növelése érdekében.

7.1 Jövőbeli fejlesztési irányok

Fontos irány a detektálási pontosság javítása nagyobb és változatosabb adatállomány felhasználásával, valamint a modell finomhangolása.

7.1.1 Teljesítmény optimalizáció

A rendszer jelenleg is több optimalizációs technikát alkalmaz:

- **kisebb input kép**
imgsz=320
- **bounding box szűrés**
MIN_BOX_SIZE
- **frame megerősítés**
REQUIRED_FRAMES
- **esemény cooldown** (ami alatt a funkció nem használható újra)
COOLDOWN

Ezek együttesen biztosítják a stabilabb és hatékonyabb működést.

7.1.2 GPU gyorsítás

A neurális háló futtatása grafikus feldolgozó egység használatával, jelentősen növelheti a sebességet. A kód `results = model(frame, device="cuda")` formára történő módosításával, a modell futtatása a grafikus feldolgozóegységre (GPU) kerül, amely az NVIDIA CUDA

technológiát használja. Ennek köszönhetően a neurális hálózat működése jelentősen gyorsabb a CPU-n történő végrehajtáshoz képest.

Fontos azonban megjegyezni, hogy a CUDA kizárólag NVIDIA gyártmányú grafikus kártyák esetében érhető el, így a GPU-alapú gyorsítás csak ilyen hardverkörnyezetben alkalmazható.

7.1.3 fejlettebb mélytanulási modellek integrálása

A rendszer továbbfejleszhető fejlettebb mélytanulási modellek integrálásával, például a jelenleg legújabb YOLO26-os modell használatával.

7.1.4 Saját tanított modell

Ha pontosabb modellt szeretnénk használni, akkor azt saját adatok gyűjtésével és egy mélytanulási modell betanításával érhetjük el, – például mobiltelefon-használatot ábrázoló képeken, – ezáltal lehetővé téve a célzott és pontosabb detektálást. Ilyen mélytanulási modell a **TensorFlow keretrendszerre épülő neurális hálózat**, amely lehetővé teszi a különböző mélytanulási modellek – például a már említett konvolúciós neurális hálózatok – hatékony tanítását és alkalmazását a képfeldolgozási és objektumfelismerési feladatokban.

7.1.5 Többkamerás környezet támogatása

A többkamerás rendszer alkalmazása növeli a megfigyelés hatékonyságát, mivel különböző nézőpontokból is rögzíthetők az események. Ez lehetővé teszi a detektálási pontosság javítását, valamint csökkenti annak esélyét, hogy egy objektum a takarás, vagy kedvezőtlen látószög miatt ne kerüljön felismerésre.

A többkamerás környezet további előnye, hogy az egyes kamerák által szolgáltatott információk kombinálhatók, amely lehetőséget ad az események pontosabb rekonstruálására, valamint a gyanús tevékenységek megbízhatóbb azonosítására.

7.1.6 Hálózati naplózás fejlesztése

A hálózati naplózás fejlesztése különösen fontos, ha a rendszert éles, vállalati környezetben kívánjuk alkalmazni, mivel ezáltal a riasztások kezelése és a naplózás központilag hatékonyabban valósítható meg. A prototípus szoftver fejlesztése folyamatban van, amely része a naplókezelő rendszer fejlesztésének, amelynek rövid bemutatására a következőkben kerül sor.

7.2 Detektáló naplózó és eseménykezelő rendszer fejlesztése

A fejlesztés motivációját az adta, hogy több munkaállomás egyidejű megfigyelése esetén szükségessé válik egy központi eseménykezelő rendszer kialakítása is. Egy ilyen rendszer lehetővé teszi az adatok egységes tárolását és kezelését, valamint a riasztások központi kezelését is.

A projekt ezen irányú fejlesztése elkezdődött és jelenleg is folyamatban van.

A rendszer két fő komponensből áll: egy detektáló és naplózó alrendszerből, valamint egy megjelenítő és eseménykezelő egységből. A komponensek közötti adatkapcsolatot egy közös SQLite3 adatbázis biztosítja, amely lehetővé teszi a valós idejű eseményfeldolgozást és a későbbi visszakereshetőséget. Ezzel a szöveges naplófájl kiváltásra kerül.

7.2.1 Detektáló és naplózórendszer fejlesztése

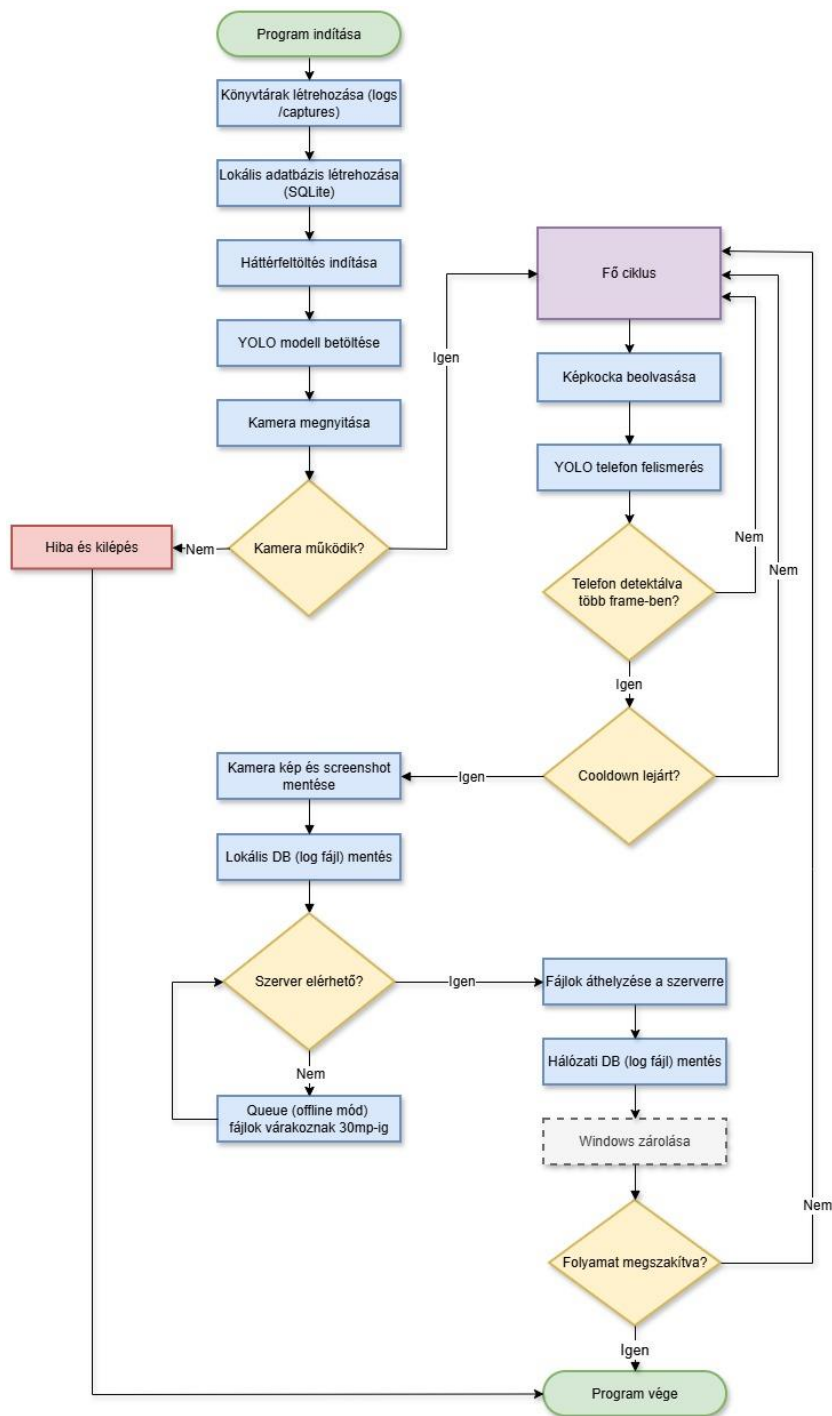
Az architektúra fő elemei az alábbi komponensekkel egészültek ki:

A **lokális adattárolás** folyamatban, a rendszer minden detektált eseményt elsőként lokálisan rögzít, amely biztosítja az adatok megőrzését hálózati hiba, vagy kapcsolatkimaradás esetén. A program az **SQLite3 Python könyvtárral** egészült ki.

A **hálózati szinkronizáció** komponens **subprocess** könyvtár segítségével folyamatosan ellenőrzi a szerver elérhetőségét **ping** rendszerszintű művelettel, és annak rendelkezésre állása esetén feltölti az eseményeket az **shutil** modul felhasználva a központi adatbázisba.

Queue alapú késleltetett feltöltés kiegészítése arra szolgál, hogy amennyiben a szerver nem érhető el, a rendszer a lokálisan tárolt adatokat ideiglenesen megőrzi, majd a kapcsolat helyreállása után automatikusan feltölti azokat.

Aszinkron feldolgozás során a hálózati és szinkronizációs műveletek külön háttérszálon futnak, így a detektálási folyamat nem áll meg, továbbá a rendszer folyamatos működése is biztosított. Ehhez a **threading** modul került alkalmazásra.



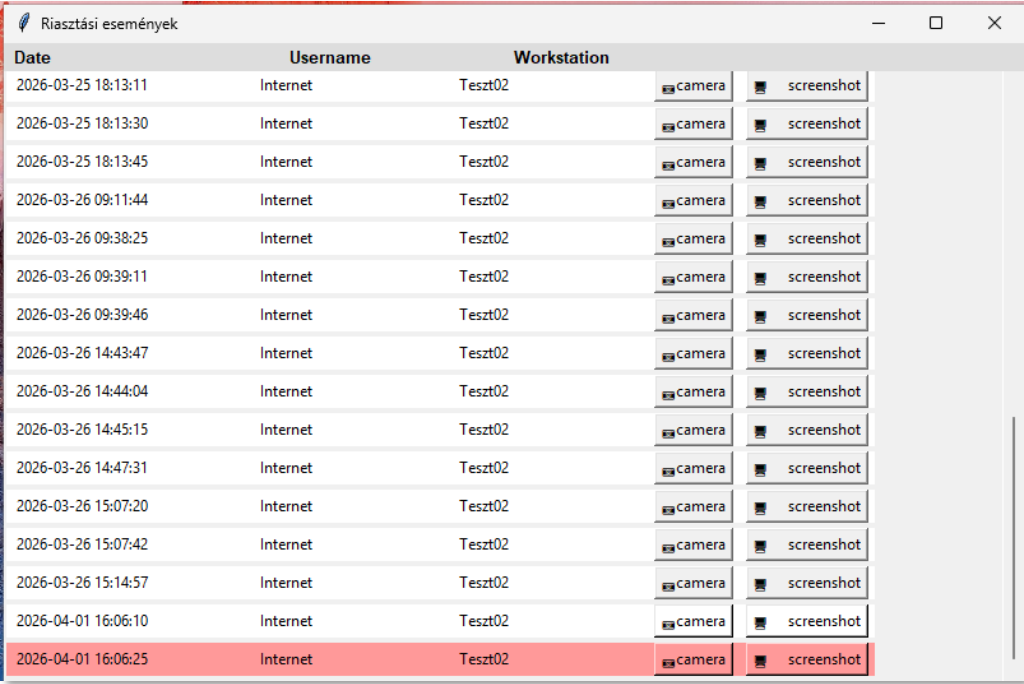
33. ábra Detektáló és naplózó rendszer folyamatábrája.

7.2.2 Riasztás és eseménykezelő alkalmazás fejlesztése

Az eseménykezelő alkalmazás működése az adatbázis inicializálásával kezdődik, amely során létrehozza a szükséges táblát, ha az még nem létezik. Ezt követően a rendszer lekérdezi a korábban rögzített eseményeket, eltárolja azok azonosítóit, majd megjeleníti őket a grafikus felhasználói felületen riasztás, illetve villogás nélkül. A program ezután a **tkinter** könyvtár

segítségével létrehozza a felhasználói felületet, amely egy fejlécből és egy görgethető listából áll, ahol megjelennek az események.

A működés során a rendszer időzített módon, meghatározott időközönként lekérdezi az adatbázist és ellenőrzi, hogy jelentek-e meg új események. Amennyiben új rekordot talál, azt hozzáadja a felülethez és vizuálisan kiemeli piros színű villogó megjelenítéssel. A villogás addig folytatódik, amíg a felhasználó rá nem kattint az adatra, vagy meg nem nyitja a hozzá tartozó képeket.



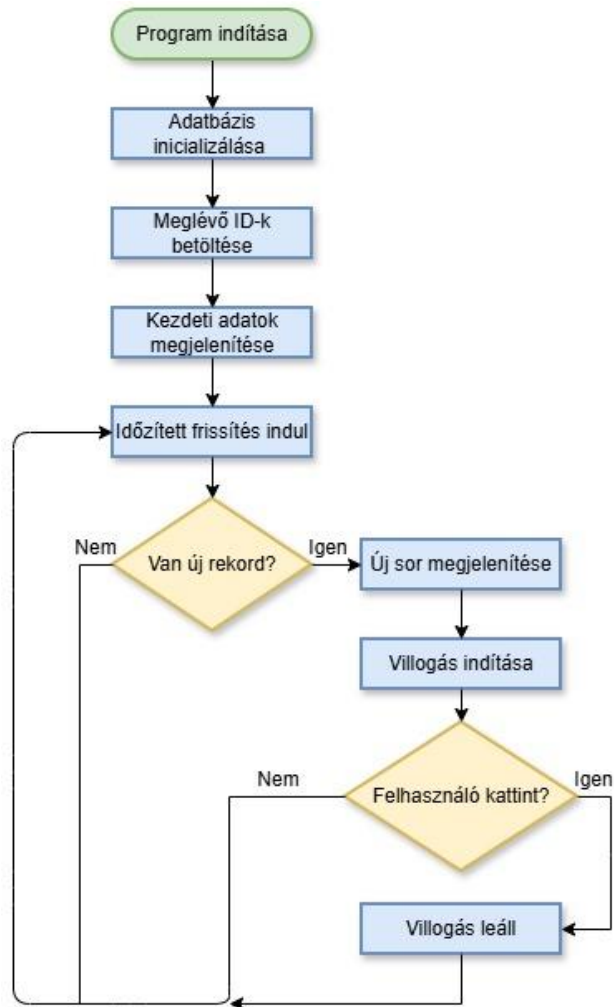
Date	Username	Workstation	camera	screenshot
2026-03-25 18:13:11	Internet	Teszt02	camera	screenshot
2026-03-25 18:13:30	Internet	Teszt02	camera	screenshot
2026-03-25 18:13:45	Internet	Teszt02	camera	screenshot
2026-03-26 09:11:44	Internet	Teszt02	camera	screenshot
2026-03-26 09:38:25	Internet	Teszt02	camera	screenshot
2026-03-26 09:39:11	Internet	Teszt02	camera	screenshot
2026-03-26 09:39:46	Internet	Teszt02	camera	screenshot
2026-03-26 14:43:47	Internet	Teszt02	camera	screenshot
2026-03-26 14:44:04	Internet	Teszt02	camera	screenshot
2026-03-26 14:45:15	Internet	Teszt02	camera	screenshot
2026-03-26 14:47:31	Internet	Teszt02	camera	screenshot
2026-03-26 15:07:20	Internet	Teszt02	camera	screenshot
2026-03-26 15:07:42	Internet	Teszt02	camera	screenshot
2026-03-26 15:14:57	Internet	Teszt02	camera	screenshot
2026-04-01 16:06:10	Internet	Teszt02	camera	screenshot
2026-04-01 16:06:25	Internet	Teszt02	camera	screenshot

34. ábra Eseménykezelő felülete

A felhasználó számára adott a lehetőség, a listában szereplő korábbi mentett kamera és képernyőképek megnyitására, amely során a rendszer ellenőrzi a fájlok létezését és hiba esetén visszajelzést ad. A program működése ciklikus, így az adatlekérdezés, a megjelenítés és a frissítés, folyamatosan ismétlődik a futás teljes ideje alatt biztosítva, az események valós idejű követését és kezelését.

A demonstrációs videó elérhetősége:

<https://miau.myx.hu/miau/330/sche/phone%20detect%20demo2.mp4>



35. ábra, Riasztás és eseménykezelő folyamatábrája.

A rendszernek a fejlesztése során a mesterséges intelligencián alapuló nyelvi modell segítségével kívül, szakmai segítség is rendelkezésemre állt, amely hozzájárult a fejlesztés megvalósításához. A tesztelések során több olyan problémával is szembesültem, amelyek megoldása egyelőre még nem vezetett érdemben eredményre, ezért a fejlesztés ezen részének részletes bemutatása nem képezi a dolgozat fő fejezeteinek részét.

8 Mellékletek

A mellékletek fejezet olyan kiegészítő információkat tartalmaz, amelyek szorosan kapcsolódnak a dolgozat témájához, azonban terjedelmük vagy részletességük miatt nem kerültek be a fő szövegbe. A mellékletek célja, hogy az olvasó számára további technikai részleteket biztosítsanak a bemutatott megoldás reprodukálásához. A mellékletekben megtalálhatók a fejlesztés során használt programkódok és konfigurációs beállítások.

8.1 Rövidítések jegyzéke

Rövidítés	Teljes kifejezés / Jelentés
------------------	------------------------------------

AI	Artificial Intelligence (Mesterséges Intelligencia)
-----------	---

ANN	Artificial Neural Network (Mesterséges neurális hálózat)
------------	--

CNN	Convolutional Neural Network (Konvolúciós neurális hálózat)
------------	---

COCO	Component-based object comparison for objectivity (Hasonlóságelemző modell család)
-------------	--

OAM	Objektum-attribútum mátrix (Egyfajta tanulási minta)
------------	--

CPU	Central Processing Unit (Központi feldolgozóegység)
------------	---

CSV	Comma-Separated Values (Vesszővel tagolt szöveges fájlformátum)
------------	---

CV	Computer Vision (Számítógépes látás)
-----------	--------------------------------------

DL	Deep Learning (Mélytanulás)
-----------	-----------------------------

FPS	Frames Per Second (Képkockasebesség)
GPU	Graphics Processing Unit (Grafikus processzor)
GUI	A GUI a Graphical User Interface (Grafikus Felhasználói Felület)
JPG/JPEG	Joint Photographic Experts Group (Fényképeszeti Szakértők Egyesült Társasága)
ML	Machine Learning (Gépi tanulás)
OpenCV	Open Source Computer Vision Library (A legnépszerűbb programozói könyvtár)
ReLU	Rectified Linear Unit (Aktivációs függvény a neurális hálózatokban)
RGB	Red, Green, Blue (A digitális képek alapszín-modellje)
ROI	Region of Interest (Érdeklődésre számot tartó terület a képen)

8.2 Ábrajegyzék

1. ábra, fejlesztési költségmodell	15
2. ábra, szürkeárnyaltos kép 10x10px-re való átméretezése.	22
3. ábra, RGB színmodell	22
4. ábra, CMYK színmodell.....	23
5. ábra, HSN/HSV színmodell	23
6. ábra, Neurális hálózatok rétegei	27
7. ábra, YOLO algoritmus szemléltetése	33
8. ábra, YOLO11 modellváltozatok	34
9. ábra, ImageToCsvConverter szoftver felülete	45
10. ábra, Saját előállított adat, illetve scenario gyanús esemény nélkül	45
11. ábra, Előállított adat, illetve scenario gyanús eseménnyel.....	46
12. ábra, Előállított adat, illetve scenario kiragadott részletének nagyítása	46
13. ábra, Nyers OAM táblázata	47
14. ábra, Rangsorolt OAM táblázata	48
15. ábra, COCO Standard felülete	49
16. ábra, COCO STD modell által kapott eredmények	49
17. ábra, Szabály meghatározása a becslé és tény értékei alapján színekkel.	50
18. ábra, Rangsorolt OAM értékeinek összevetése a becslés értékeivel	51
19. ábra, Eredmények összegzése	52
20. ábra, Rangsorolt Inverz OAM értékeinek összevetése a becslés értékeivel.....	53
21. ábra, Küszöbérték meghatározása az inverz rangsorolás értékei alapján színekkel ...	53
22. ábra Inverz mátrix eredmények összegzése	54
23. ábra, Rangsorolt Objektum-attribútum-mátrix direkt és inverz becslés értékének kimutatása	54
24. ábra, Rangsorolt Objektum-attribútum-mátrix direkt és inverz összevonásának elemzése a COCO STD modell használatával	55
25. ábra, Küszöbérték meghatározása a Direkt és Inverz összevonásával	55
26. ábra, Direkt és Inverz mátrix összevont eredmények összegzése	55
27. ábra, Beszélgetés részlete a ChatGPT nagy nyelvi modellel	57
28. ábra, Sikeres detektálás reprezentációja.....	60

29. ábra, Tervezett védelmi szoftver folyamatábrája	63
30. ábra, Részlet egy naplóképről	69
31. ábra, YOLO11n modell teszteredményei	73
32. ábra, Hamis pozitív esetek esetleges okai	75
33. ábra, Detektáló és naplózó rendszer folyamatábrája.	82
34. ábra, Eseménykezelő felülete	83
35. ábra, Riasztás és eseménykezelő folyamatábrája.....	84

8.3 Fogalomjegyzék

A fogalomjegyzék célja, hogy röviden bemutassa a dolgozatban előforduló legfontosabb szakkifejezéseket és technikai fogalmakat.

Fogalom	Definíció / Magyarázat
Aggregálás	olyan művelet, amikor több adatpontból összesített, áttekinthető, vagy feldolgozott eredményt hozunk létre. Ez lehet matematikai, statisztikai vagy informatikai művelet is.
Annotált adat	Az annotált adat olyan információ (szöveg, kép, videó), amelyet utólagos címkékkal, magyarázatokkal vagy metaadatokkal láttak el. Ez a folyamat a gépi tanulási algoritmusok tanításához elengedhetetlen, mivel struktúrát ad a nyers adatoknak, lehetővé téve a mesterséges intelligencia számára azok értelmezését. (hu.shaip.com 2026.04.10.)
API	Application Programming Interface; alkalmazások közötti programozási interfész
ChatGPT	A ChatGPT a Generative Pre-trained Transformer angol rövidítése, ami egy nagy nyelvi modellt takar, amelyet az

OpenAI fejlesztett ki, és képes emberi szintű szöveget generálni és párbeszédet folytatni.

CMYK	A CMYK színmodell egy szubtraktív színkeverési rendszer, amelyet elsősorban a nyomdaiparban alkalmaznak. A modell négy alapszínre épül: cián (C), bíbor (M), sárga (Y) és fekete (K). A színek előállítása a fény elnyelésén alapul, így a különböző festékek kombinációjával jönnek létre a kívánt árnyalatok.
CNN (Convolutional Neural Network)	Konvolúciós neurális hálózat, amely képfeldolgozási feladatokra optimalizált mélytanulási modell, és képes a képekből automatikusan jellemzőket kinyerni.
COCO (Common Objects in Context) Dataset	A COCO adatkészlet egy hatalmas méretű, nyílt forráskódú adatbázis, amelyet a mesterséges intelligencia, azon belül is a gépi látás (computer vision) fejlesztésére és tesztelésére használnak.
COCO (Component based Object Comparison for Objectivity):	hasonlóságelemzés. Optimalizálás keretében minden tulajdonság esetén minden rangsorszámhoz hozzárendelünk egy konkrét értéket (lépcsőt), amennyivel az adott tulajdonság hozzájárul a függő változó értékéhez.
COCO STD	a COCO modul alapváltozata, amely objektumok ár-teljesítmény számításra képes, Y-ként minden esetben szükséges valós árat megadni.
Computer Vision (Számítógépes látás)	A mesterséges intelligencia területe, amely lehetővé teszi a számítógépek számára a képi információk értelmezését és feldolgozását.

CPU	Central Processing Unit; központi feldolgozóegység.
CUDA (Compute Unified Device Architecture)	Az NVIDIA által fejlesztett párhuzamos számítási platform és programozási modell. Lehetővé teszi, hogy a grafikus processzor (GPU) ne csak grafikai feladatokat végezzen, hanem általános számításokat is. A mélytanulási algoritmusok, például a neurális hálózatok tanítása során nagyon sok matematikai műveletet kell végrehajtani, amelyeket a GPU a CPU-nál sokkal gyorsabban tud feldolgozni.
Deep Learning (Mélytanulás)	A gépi tanulás egyik ága, amely többrétegű neurális hálózatok segítségével képes komplex mintázatok felismerésére nagy mennyiségű adatból.
Éldetektálás	A digitális képfeldolgozás egyik alapvető technikája, amely a képeken belüli jelentős intenzitásváltozásokat (kontrasztkülönbségeket) azonosítja.
GDPR (General Data Protection Regulation)	Általános Adatvédelmi Rendelet.
GPU	Graphics Processing Unit magyarul grafikus feldolgozóegység vagy grafikus processzor, a számítógép videokártyájának központi egysége, amely a képek, videók, 3D modellek és animációk megjelenítéséért felel.
GUI	A GUI a Graphical User Interface (Grafikus Felhasználói Felület) rövidítése. Olyan számítógépes felület, amely ikonok, ablakok, menük és képek segítségével teszi lehetővé a felhasználók számára az eszközökkel való interakciót, a szöveges parancsok begépelése helyett.
HSB	A HSB (Hue, Saturation, Brightness) színmodell egy intuitív színábrázolási rendszer, amely a színeket három komponens

segítségével írja le. A *Hue* (színárnyalat) a szín típusát jelöli, a *Saturation* (telítettség) a szín intenzitását, míg a *Brightness* (fényerő) a világosságát határozza meg. A modell előnye, hogy közelebb áll az emberi színérzékeléshez, ezért gyakran használják képfeldolgozásban és grafikai alkalmazásokban.

Információbiztonság	Az információk védelmének rendszere, amely biztosítja az adatok bizalmasságát, sértetlenségét és rendelkezésre állását.
ISO/IEC 27001	Információbiztonsági irányítási szabvány (ISO, 2022).
IT-biztonság (kiberbiztonság)	Az információbiztonság részterülete, amely a digitális rendszerek, számítógépek és hálózatok technikai védelmével foglalkozik.
Jellemzővektor	Angolul (feature vector) egy olyan többdimenziós numerikus vektor, amely egy adott objektum vagy jelenség releváns tulajdonságait reprezentálja matematikai formában. Az informatikában és a gépi tanulásban ez az alapvető egység, amellyel az algoritmusok "értelmezik" a valós világ adatait.
Képpont (Pixel)	A digitális kép legkisebb egysége, amely egy adott színt és fényerőt hordoz.
KJE	Kodolányi János Egyetem.
Konvolúció	Matematikai művelet, amely során egy szűrő (kernel) végighalad a képen, hogy kiemelje a fontos jellemzőket (pl. éleket, sarkokat).
Machine Learning (Gépi tanulás)	A mesterséges intelligencia egyik területe, amely olyan algoritmusokat használ, amelyek képesek adatokból tanulni és azok alapján előrejelzéseket vagy döntéseket hozni.

OAM	Objektum Attribútum Mátrix, azaz egyfajta tanulási minta.
Objektumdetektálás	A számítógépes látás egyik feladata, amely során egy képen vagy videóban azonosításra kerülnek az objektumok és azok helye.
Pixel	A szó az angol picture element (kb. képelem) kifejezésből származik, ahol a picture szó a pix rövidített alakban szerepel.
Python	A Python egy rendkívül népszerű, magas szintű, általános célú programozási nyelv, amelyet egyszerű, olvasható szintaxisa miatt kezdőknek is ajánlanak.
Reputációs veszteség	A reputációs veszteség egy vállalat, márka vagy személy hírnevének, tekintélyének csökkenése, amely bizalomvesztéshez és jelentős anyagi kárral járó üzleti hatásokhoz vezet.
RGB	Az RGB színmodell egy additív színmodell, amelyben a fény vörös, zöld és kék alapszíneit különböző módon adják össze, hogy a színek széles skáláját reprodukálják. A modell neve a három additív alapszín, a piros, a zöld és a kék angol kezdőbetűiből származik (red, green, blue).
Szegmentálás	Az a folyamat, amikor a képet részekre bontják, és minden egyes képponthez (pixel) hozzárendelnek egy kategóriát (pl. mi az autó és mi az út).
Tiltott adatszerzés, adateltulajdonítás	A tiltott adatszerzés (Btk. 422. §) olyan bűncselekmény, amely személyes adat, magán- vagy üzleti titok jogosulatlan megszerzésére irányul technikai eszközökkel, titkos megfigyeléssel, küldemények felbontásával vagy információs rendszer (internet, számítógép) megsértésével.
Training (Betanítás)	Az a folyamat, amely során a modellt rengeteg címkézett képpel látják el, hogy megtanulja felismerni a vizuális jellemzőket.

Vizuális tiltott adatszerzés	Olyan adateltulajdonítási módszer, amely során a támadó képi rögzítéssel – például fényképezéssel vagy videófelvétellel – szerez meg információkat.
Windows API	A Windows API (más néven Win32 API) a Microsoft Windows operációs rendszerek natív alkalmazásprogramozási felülete. Ez egy gyűjtemény, amely funkciókat, interfészeket és protokollokat biztosít a fejlesztők számára, hogy közvetlenül kommunikálhassanak az operációs rendszerrel és annak erőforrásaival.
YOLO (You Only Look Once)	Valós idejű objektumdetektáló algoritmus, amely egyetlen neurális háló segítségével képes a képen található objektumok felismerésére és lokalizálására.
Zajszűrés	Képfeldolgozási lépés, amelynek célja a digitális kép szemcsésességének vagy hibáinak eltávolítása a jobb felismerhetőség érdekében.

8.4 Forrás és hivatkozás jegyzék

A források jegyzéke az alábbi kategóriák szerint kerültek kialakításra:

Kód	Leírás	Mennyiség/darab
T01	új, angol, cikk, nem KJE-releváns	2
T02	új, angol, cikk, KJE-releváns	1
T03	új, angol, weboldal, nem KJE-releváns	2
T04	új, angol, weboldal, KJE-releváns	1
T05	új, nem angol, cikk, nem KJE-releváns	2
T06	új, nem angol, cikk, KJE-releváns	1
T07	új, nem angol, weboldal, nem KJE-releváns	1
T08	új, nem angol, weboldal, KJE-releváns	1
T09	régi, angol, cikk, nem KJE-releváns	3
T10	régi, angol, cikk, KJE-releváns	1
T11	régi, angol, weboldal, nem KJE-releváns	1
T12	régi, angol, weboldal, KJE-releváns	2
T13	régi, nem angol, cikk, nem KJE-releváns	1
T14	régi, nem angol, cikk, KJE-releváns	1
T15	régi, nem angol, weboldal, nem KJE-releváns	1
T16	régi, nem angol, weboldal, KJE-releváns	1

T07 – Árukereső

Forrás: <https://www.arukereso.hu/>
(Megtekintve: 2026.02.15.)

T13 – André Saeckel: Dqs global.hu, 2022.11.12.,

Forrás: <https://www.dqsglobal.com/hu/szolgalattas/blog/it-biztonsag-vs.-informaciobiztonsag-mi-a-kueloenbseg>
(Megtekintve: 2026.04.10.)

T09 – Christopher M. Bishop: *Pattern Recognition and Machine Learning*. New York: Springer, 2006.

Forrás: <https://github.com/Benlau93/Data-Science-Curriculum/blob/master/Bishop-Pattern-Recognition-and-Machine-Learning-2006.pdf>
(Megtekintve: 2026.03.10)

T16 – Dr. Pitlik László. COCO online elemző szolgáltatás weblapja,

Forrás: <https://miau.myx.hu/myx-free/coco>
(Megtekintve: 2023. 10. 22.)

T04 – Dr. Pitlik, László. (n.d.). *CT 00*. Forrás: Miau Wiki

Forrás: https://miau.myx.hu/mediawiki/index.php/CT_00
(Megtekintve: 2026.03.28)

- T05 – Giesz István: Szemelvények az információbiztonság területéről, Egyetemi órai előadás, (2024.01.28.)
- T02 – Grotte, Judit: *Introducing the new concept of personalized digital tourism (PDT)*. Infocommunications Journal. 2024.
 Forrás: https://www.infocommunications.hu/documents/169298/5015279/ICJ_SpecIssCogInfo_2025_02_8.pdf
 (Megtekintve: 2026.03.28)
- T13 – HWS.hu: Fóti Marcell 2019. 11. 12.
 Forrás: <https://www.hws.hu/hirek/61088/yolo-v3-algoritmus-gepi-latas-ai.html>,
 (Megtekintve: 2026.03.22.)
- T16 – Hasonlóságelemzés COCO használatával, Oktatási segédanyag, 2013.08.24.
 Forrás: https://miau.my-x.hu/temp/tananyag/ginf/coco_demo.pdf
 (Megtekintve: 2026.03.15)
- T11 – Ian Goodfellow, Yoshua Bengio, and Aaron Courville: *Deep Learning*. Cambridge, MA: MIT Press, 2016.
 Forrás: <https://www.deeplearningbook.org/>
 (Megtekintve: 2026. 02. 24.)
- T10 – Miau.my-x.hu, i.e. an Innovative „Idea-Breeding-Farm”, 2014,
 Forrás: https://miau.my-x.hu/miau/196/My-X%20Team_A5%20fuzet_EN_jav.pdf
 (Megtekintve: 2026. 03. 14.)
- T14 – MIAU - Gépi tanulás
 Forrás: https://miau.my-x.hu/mediawiki/index.php/G%C3%A9pi_tanul%C3%A1s
 (Megtekintve: 2026. 03. 19.)
- T12 – Miau.my-x.hu, GYIK, 2008-2012,
 Forrás: https://miau.my-x.hu/myx-free/index_en.php3?x=s01
 (Megtekintve: 2026.03.27)
- T06 – My-x Team - Avagy egy innovatív „ötlet-istálló”
 Forrás: https://miau.my-x.hu/miau/196/My-X%20Team_A5%20fuzet_HU_jav.pdf
 (Megtekintve: 2026. 03. 19.)
- T01 – OpenCV, “OpenCV Documentation.”
 Forrás: <https://docs.opencv.org/>
 (Megtekintve: 2026. 02. 22.)
- T08 – Kodolányi János Egyetem weboldala
 Forrás: <https://www.kodolanyi.hu/>
 (Megtekintve: 2026.06.15)

- T09 – Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi (2016): *You Only Look Once: Unified, Real-Time Object Detection* CVPR, 2016.
(Megtekintve: 2026. 02. 22.)
- T09 – Szeliski, Richard. (2022): *Computer Vision: Algorithms and Applications (2nd Edition)*, 2022.
Forrás: <https://szeliski.org/Book/> (Letöltve: 2025. 01. 13.)
- T05 – Szemenyei Márton – Varnyú Dóra: *Deep Learning a Vizuális Informatikában*, BME, 2023. (Letöltve: 2025. 12. 16.)
- T03 – Ultralytics, “Ultralytics YOLO Documentation.”
Forrás: <https://docs.ultralytics.com/models/yolo11/>
(Megtekintve: 2026.03.04.)
- T01 – What Techniques Do Hackers Use to Steal Information?
Forrás: <https://www.nebrcentre.co.uk/what-techniques-do-hackers-use-to-steal-information/>
(Megtekintve: 2026. 03. 07.)
- T03 – YOLO implementáció (Ultralytics):
Forrás: <https://github.com/ultralytics/ultralytics> (Megtekintve: 2026. 02. 04.)

8.5 Forráskódok

Ebben a fejezetben a fejlesztések során készült forráskódok és részletes leírásaik szerepelnek.

8.5.1 Az Image To CSV program forráskódja és leírása

A program egy grafikus felületű Python alkalmazás, amely egy tetszőlege képfájlt tetszőlege méretre alakít, majd a kép pixelértékeit CSV fájlba menti.

```
import tkinter as tk
from tkinter import filedialog
from PIL import Image
import pandas as pd
import numpy as np

def browse_file():
    global file_path
    file_path = filedialog.askopenfilename()
    if file_path:
        filename_label.config(text=f"File: {file_path}")

def convert_image():
    if file_path:
        # Open the image and convert to grayscale
        img = Image.open(file_path).convert('L')
```

```

# Get specified pixel dimensions from entry widgets
try:
    new_width = int(width_entry.get())
    new_height = int(height_entry.get())
except ValueError:
    status_label.config(text="Invalid dimensions. Please enter valid integers.")
    return

# Resize the image with specified pixel dimensions
img = img.resize((new_width, new_height))

# Convert image data to numpy array
pixel_data = np.array(img)

# Save the pixel data to a CSV file
df = pd.DataFrame(pixel_data)
df.to_csv('image_pixel_data.csv', index=False, header=False)

# Update the status
status_label.config(text="Image converted and saved to CSV.")

# Create the main window
root = tk.Tk()
root.title("Image Converter")

# File selection
file_path = ""
browse_button = tk.Button(root, text="Browse", command=browse_file)
browse_button.pack(pady=10)

# Display selected filename
filename_label = tk.Label(root, text="File: ")
filename_label.pack()

# Entry widgets for specified pixel dimensions
width_label = tk.Label(root, text="Width:")
width_label.pack()
width_entry = tk.Entry(root)
width_entry.pack()

height_label = tk.Label(root, text="Height:")
height_label.pack()
height_entry = tk.Entry(root)
height_entry.pack()

# Convert button
convert_button = tk.Button(root, text="Convert", command=convert_image)
convert_button.pack(pady=10)

# Status label
status_label = tk.Label(root, text="")
status_label.pack()

# Run the GUI loop
root.mainloop()

```

Szöveges kódmagyarázat:

- **tkinter könyvtár** a grafikus felület létrehozásához szükséges,
- **filedialog modul** lehetővé teszi, hogy a felhasználó fájlböngésző ablak segítségével válasszon ki egy képfájlt.
- **PIL könyvtár Image modulja** a képek megnyitására és módosítására szolgál.

- **pandas könyvtár** segítségével az adatokat könnyen el lehet menteni CSV formátumba.
- **numpy könyvtár** a képpixelék mátrixként való kezelését teszi lehetővé.
- **browse_file** nevű **függvény** a Browse gomb megnyomásakor fut le. Ez a függvény megnyit egy fájlválasztó ablakot, ahol a felhasználó kiválaszthat egy képfájlt.
- **file_path** **változó** kiválasztott fájl elérési útját tárolja. Ha a felhasználó kiválasztott egy fájlt, akkor a program a grafikus felületen megjeleníti annak nevét vagy elérési útját egy szövegmezőben.
- **convert_image** **függvény** a Convert gomb megnyomásakor kerül végrehajtásra. Először ellenőrzi, hogy a felhasználó kiválasztott-e már képfájlt. Ha igen, akkor a;
- **PIL könyvtár** segítségével a program megnyitja a képet, majd *szürkeárnyalatossá* alakítja. Ezután a program kiolvassa a grafikus felületen található *Width* és *Height* mezőkben megadott értékeket, amelyek az *új kép szélességét* és *magasságát adják meg pixelben*. Ha ezek az értékek nem számok, akkor a program *hibaüzenetet jelenít meg*.
- Ha a megadott méretek érvényesek, a program átméretezi a képet a megadott pixelméretre. Ezután a képet **numpy tömbbé** alakítja, amely egy kétdimenziós mátrixként tartalmazza a kép pixelértékeit.
- A **numpy tömbből** egy **pandas DataFrame** készül, amelyet a program *CSV fájlba ment image_pixel_data.csv* néven. A mentés során a program nem ír ki sor- vagy oszlopneveket.
- A **tkinter** könyvtár segítségével a program végén *létrejön a grafikus felület*. A főablak címe „Image Converter”. Az ablak tartalmaz egy *Browse gombot* a kép kiválasztásához, egy szövegmezőt a kiválasztott fájl megjelenítésére, *két beviteli mezőt a kép új szélességének és magasságának megadásához*, valamint egy *Convert gombot, amely elindítja a kép feldolgozását*. Emellett egy státuszmező is található, amely a program működésével kapcsolatos üzeneteket jeleníti meg.
- **root.mainloop()** ciklus indítja el a program futását, amely biztosítja, hogy a grafikus felület működjön, és a program reagáljon a felhasználói műveletekre, például a gombokra kattintásra.

8.5.2 Objektumfelismerési teszt rendszer forráskódja

A telefon felismerésének teszteléséhez, a YOLO11s-es modell került felhasználásra.

```
import cv2
from ultralytics import YOLO

# -----
# Beállítások
# -----

MODEL = "yolo11s.pt"
CONF_THRESHOLD = 0.8
MIN_BOX_SIZE = 30

# -----
# Modell betöltése
# -----

model = YOLO(MODEL)

# -----
# Kamera
# -----

cap = cv2.VideoCapture(1)

if not cap.isOpened():
    print("Kamera hiba")
    exit()

# -----
# F6 ciklus
# -----

while True:

    ret, frame = cap.read()

    if not ret:
        break

    results = model(frame, imgsz=640)

    for r in results:
        for box in r.boxes:

            cls = int(box.cls[0])
            conf = float(box.conf[0])

            # 67 = cell phone (COCO dataset)
            if cls == 67 and conf > CONF_THRESHOLD:

                x1, y1, x2, y2 = map(int, box.xyxy[0])

                width = x2 - x1
                height = y2 - y1

                if width > MIN_BOX_SIZE and height > MIN_BOX_SIZE:

                    cv2.rectangle(frame, (x1, y1), (x2, y2), (0, 255, 0), 2)

                    cv2.putText(frame,
                                f"PHONE {conf:.2f}",
                                (x1, y1 - 10),
                                cv2.FONT_HERSHEY_SIMPLEX,
                                0.6,
                                (0, 255, 0),
                                2)

                    cv2.imshow("YOLO Phone Detection", frame)

                    if cv2.waitKey(1) & 0xFF == ord("q"):
                        break

cap.release()
cv2.destroyAllWindows()
```

8.5.3 A Cellphoneblk v1.4 program forráskódja és leírása

Az alfejezet bemutatja a szoftver forráskódját és a detektálási folyamat lépései, beleértve az adatok betöltését, a modell inicializálását, valamint a detektálási eredmények feldolgozását és megjelenítését.

A tesztelés során történő kísérleteket, optimalizálásokat nem tartalmazza.

1. A kód elején először a Python könyvtárak kerülnek betöltésre

```
import cv2
import time
import ctypes
import pyautogui
import os
import getpass
import socket
from datetime import datetime
from ultralytics import YOLO
```

A könyvtárak nevei és a programban betöltött funkciói:

- **cv2:** Kamera kezelése és képfeldolgozása
- **time:** Idő és dátum mérése
- **ctypes:** Windows funkciók futtatása (pl. PC zárolás)
- **pyautogui:** screenshot készítés
- **os:** fájlkezelés
- **getpass:** felhasználó név lekérdezés
- **socket:** gépnév lekérdezés
- **datetime:** időbélyeg generálása
- **ultralytics YOLO:** objektumfelismerő AI modell

2. Beállítások

```
MODEL = "yolov11n.pt"
CONF_THRESHOLD = 0.8
MIN_BOX_SIZE = 30
REQUIRED_FRAMES = 3
COOLDOWN = 15
```

- MODEL: használt YOLO modell.
- CONF_THRESHOLD: minimum felismerési bizonyosság.
- MIN_BOX_SIZE: minimális objektum méret.
- REQUIRED_FRAMES: hány egymást követő framre-van szükség.
- COOLDOWN: riasztások közti várakozás, amely csökkentheti a téves riasztásokat.

```
SAVE_FOLDER = "captures"
LOG_FILE = "logs.txt"

USERNAME = getpass.getuser()
WORKSTATION = socket.gethostname()

os.makedirs(SAVE_FOLDER, exist_ok=True)
```

- A SAVE_FOLDER változó megadja a könyvtár nevét ahová a kamera képek és screenshotok kerülnek.
- A LOG_FILE változó megadja a logfájl nevét és kiterjesztését.
- USERNAME és MACHINE a felhasználó és gép nevét adja vissza.
- **os.makedirs** művelet futtatásakor, létrejön a **captures mappa**, ha viszont már létezik, nem hozza újra létre.

3. YOLO modell betöltése

```
model = YOLO(MODEL)
```

Betölti a neurális hálót, amely képes felismerni az objektumokat.

4. Kamera inicializálása

```
cap = cv2.VideoCapture(0)
```

```
if not cap.isOpened():
    print("Kamera hiba")
    exit()
```

```
phone_frames = 0
last_detection = 0
```

Megnyitja az alapértelmezett webkamerát. (Az első kamera a 0 a második kamera az 1 es értéket képviseli) Ha a kamera nem érzékelhető, vagy hibás, akkor a program „Kamera hiba” hibaüzenetet ad.

5. Fő ciklus

```
while True:
```

Ez egy végtelen ciklus, ami folyamatosan olvas be képet a kameráról, lefuttatja az AI modellt, majd ellenőrzi van-e mobiltelefon a képen.

6. Kamerakép beolvasása

```
ret, frame = cap.read()
```

A ret változó visszaadja, hogy sikerült-e képet olvasni, a frame változó pedig az aktuális kameraképet adja vissza.

7. YOLO detektálás

```
results = model(frame, imgs=320)
```

A Neurális háló elemzi a képet, majd visszaadja:

- bounding box koordinátákat
- objektum típust
- confidence értékét

```
imgs=320
```

A YOLO a kamera képét 320×320 pixelre átméretezve dolgozza fel.

8. Találatok feldolgozása

```
for r in results:  
    for box in r.bboxes:
```

A ciklus végigmegy minden detektált objektumon.

9. Objektum osztály ellenőrzése

```
cls = int(box.cls[0])
```

Az objektum kategóriáját adja vissza, ha a cls==67 akkor mobiltelefon.
(A COCO dataset 80 objektumot tartalmaz, közülük a 67-es számú a telefon.)

11. Confidence ellenőrzése

```
conf = float(box.conf[0])
```

Ez a modell bizonyossága, például a 0.92 = 92%-ig biztos.

12. Bounding box méret

```
x1, y1, x2, y2 = map(int, box.xyxy[0])
```

(x1,y1) bal felső sarok

(x2,y2) jobb alsó sarok

Megadja a telefon helyét (koordinátáit) a képen.

13. Téves riasztás szűrés

```
if width > MIN_BOX_SIZE and height > MIN_BOX_SIZE:
```

Kis objektumokat kizárja, például a távoli pixeleket, zajokat.

14. Több frame megerősítés

```
phone_frames += 1
```

Csak akkor történik riasztás, ha több egymás utáni képkockában is jelen van a mobiltelefon, ezáltal csökkentve a téves riasztást.

15. Riasztás

```
ctypes.windll.user32.LockWorkStation()
```

Meghív egy Windows API-t, amellyel megtörténik a számítógép lezárása.

16. Bizonyíték mentés

```
cv2.imwrite(cam_img, frame)
```

Egy darab kamerakép mentése.

17. Screenshot (képernyőkép)

```
screenshot = pyautogui.screenshot()
```

Ment egy képernyőképet.

18. Naplózás

```
with open(LOG_FILE,"a") as f:  
    f.write(...)
```

Bejegyzés kerül a log fájlba.

19. Cooldown

```
now - last_detection > COOLDOWN
```

Megakadályozza, hogy a rendszer *folyamatosan riasszon*. (A megadott érték másodperce vonatkozik.)

20. Program vége

```
cap.release()  
cv2.destroyAllWindows()
```

Bezárja a kamerát és az OpenCV ablakot.

8.6 Releváns LLM konverzációk és teljes szövege

A szakdolgozat elkészítése során, bizonyos feladatok támogatására, **nagy nyelvi modellek (Large Language Models – LLM)** is felhasználásra kerültek. Az ilyen rendszerek képesek természetes nyelven megfogalmazott kérdésekre válaszolni, valamint segítséget nyújtani programkódok értelmezésében, dokumentáció készítésében és különböző informatikai problémák megoldásában.

Tehát „Az LLM-eket MINDEN szakdolgozathoz használni kell (kvázi minden releváns aspektushoz: tervezés, kódolás stb.)”³⁸

Az LLM-ek használata, elsősorban támogató jellegű szerepet töltött be és nem helyettesítette a szakirodalmi kutatást, a saját fejlesztési munkát vagy az önálló elemzést. A modellek segítségével elsősorban a következő feladatok kerültek támogatásra:

- programkódok dokumentációjának megfogalmazása,
- szakdolgozati szövegrészek nyelvi pontosítása,
- egyes informatikai fogalmak rövid magyarázata,
- fejlesztési környezetekkel kapcsolatos technikai kérdések tisztázása.

Fontos megjegyezni, hogy az LLM által generált válaszok minden esetben ellenőrzésre kerültek, és a szakdolgozat végleges tartalma, a saját értelmezésem és feldolgozásomon alapult.

Az alábbiakban a dolgozat elkészítése során relevánsnak tekintett LLM-alapú konverzációk kivonata kerül bemutatásra.

³⁸ Dr. Pitlik, L. (n.d.). *CT 00*. Forrás: Míau Wiki: https://miau.myx.hu/mediawiki/index.php/CT_00 (megtekintve: 2026.03.28)

8.6.1 Programkódok dokumentációjának megfogalmazása

```
fogalmazd meg a programkód dokumentációját: results =  
model(frame, imgsz=320)
```

Íme egy **programkód-dokumentáció jellegű megfogalmazás**, amit szakdolgozatban vagy kódkommentként is használhatsz:

A `results = model(frame, imgsz=320)` utasítás a betanított objektumdetektáló modell futtatását végzi el a bemeneti képkockán. A `frame` paraméter tartalmazza a kamera által rögzített aktuális képet, amely a feldolgozás bemeneteként szolgál. Az `imgsz=320` paraméter meghatározza a modell által használt bemeneti kép méretét, amely hatással van a feldolgozás sebességére és pontosságára. A modell futtatásának eredménye a `results` változóban kerül eltárolásra, amely tartalmazza a detektált objektumok adatait, például azok pozícióját, kategóriáját és a felismerés megbízhatósági értékét.

OpenAI ChatGPT-5.3, 2026.02.23.

8.6.2 Szakdolgozati szövegrészek nyelvi pontosítása

Prompt: Fogalmazd át a következő mondatot a szakdolgozathoz: Gyanús esemény esetén a rendszer automatikusan blokkolhatja a megjelenítést, vagy riasztást küldhet az illetékes személyek számára.

ChatGPT-5.3: A rendszer képes automatikus reakciót kiváltani, amennyiben gyanús eseményt észlel. Ilyen esetben a szoftver különböző védelmi intézkedéseket hajthat végre.

Az egyik lehetőség a **megjelenítés blokkolása**, amely megakadályozza az érzékeny információk további megjelenítését a képernyőn. Emellett a rendszer **riasztást küldhet az illetékes személyek számára**, például rendszergazdák vagy biztonsági operátorok részére.

OpenAI ChatGPT-5.3, 2026.03.09.

ez így jó? Azonban az éldetektálás általában nem kerül külön lépésként használatba a mélytanulási modell alkalmazása során.

Igen 🍌 **tartalmilag jó**, csak érdemes egy kicsit finomítani a megfogalmazást.

Javított változat (ajánlott):

Azonban az éldetektálás általában nem kerül külön lépésként alkalmazásra a mélytanulási modellek használata során.

OpenAi ChatGPT-5.3,2026.03.17.

8.6.3 Egyes informatikai fogalmak rövid magyarázata

ez így helyes: módszer alapját képező gépi tanulás, illetve mélytanulás képfeldolgozási eljárásai, valamint a konvolúciós neurális hálózat (Convolutional Neural Networks – CNN) amelyen a rendszer működése alapul.

Igen, ez a megfogalmazás **szakmailag helyes** és logikusan épül fel. Jól tükrözi a technológiai hierarchiát: a tágabb keretet (gépi tanulás), az azon belüli módszertant (mély tanulás) és a konkrét technikai megvalósítást (CNN).

Egy apró stilisztikai javaslat a még gördülékenyebb szövegért:

„...a rendszer alapját képező **gépi tanulási** és **mély tanulási** képfeldolgozási eljárások, valamint a **konvolúciós neurális hálózat** (CNN), amelyre a detektálás folyamata épül.”

Miért pontos ez így?

- A **mély tanulás** adja a többrétegű intelligenciát.
- A **CNN** végzi a tényleges képi elemzést (szűrést).
- A **YOLO** pedig az a specifikus algoritmus, amely ezt a CNN-t képessé teszi a valós idejű tárgykeresésre.

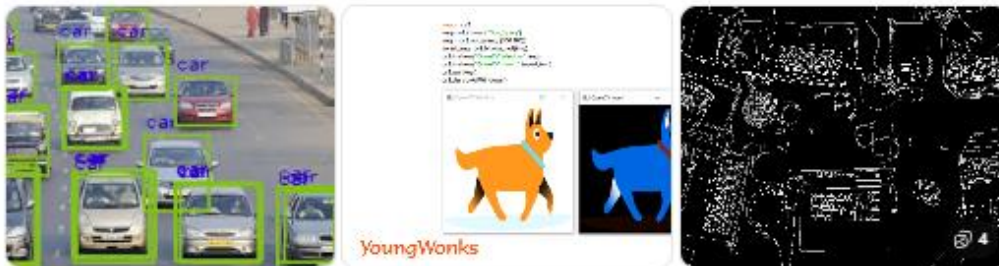
Google Gemini 1.5 Flash modell 2026.03.13.

8.6.4 Fejlesztési környezetekkel kapcsolatos technikai kérdések tisztázása

A mobiltelefon felismerő szoftver fejlesztéséhez mely Python könyvtárak szükségesek?

Egy mobiltelefon-felismerő szoftver (pl. kamera képen felismeri, hogy van-e telefon, vagy milyen típusú) fejlesztéséhez Pythonban főleg gépi látás (computer vision) és deep learning könyvtárakra lesz szükséged. Az alábbiak a leggyakrabban használtak.

1. Képfeldolgozás – alap könyvtárak



Legfontosabb könyvtár:

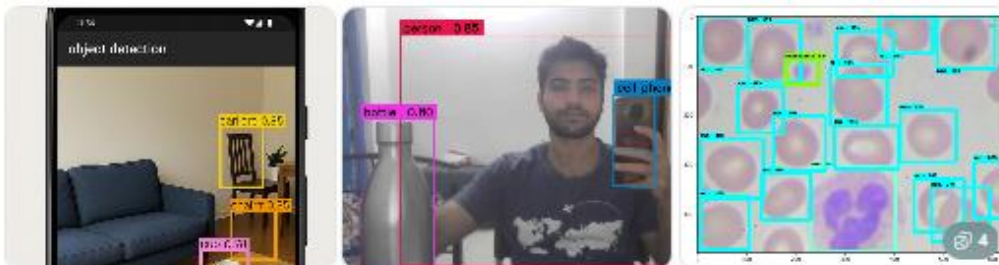
- **OpenCV**
 - Kamera kezelése
 - képfeldolgozás
 - objektumdetektálás
 - képfeldolgozó algoritmusok

Python telepítés:

```
↔ Bash 📄  
pip install opencv-python
```

Ezzel tudod pl. a kameraképet beolvasni és feldolgozni.

2. Mélytanulás – objektumfelismerés



Ha mobiltelefon objektum felismerést akarsz (AI modell):


Leggyakoribb keretrendszerek

- [PyTorch](#)
- [TensorFlow](#)

Telepítés:

```
↔ Bash   
pip install torch torchvision
```


vagy

```
↔ Bash   
pip install tensorflow
```

Objektumdetektáló modellek


- [YOLOv8](#) – nagyon népszerű
- [Faster R-CNN](#)
- [SSD](#)

YOLO telepítés:

```
↔ Bash   
pip install ultralytics
```

3. Adatkezelés és matematikai könyvtárak

Ezek szinte minden ML projekthez kelleneek:

```
↔ Bash   
pip install numpy pandas matplotlib scikit-learn
```

Fontos könyvtárak:

- [NumPy](#) – mátrix műveletek
- [Pandas](#) – adatkezelés
- [Matplotlib](#) – grafikonok
- [scikit-learn](#)

4. Képadatok kezelése

Hasznos extra könyvtárak:

```
↔ Bash   
pip install pillow albumentations 
```

- Pillow – képfájl kezelés
- Albumentations – tanító adatok bővítése

☑ Minimum stack mobiltelefon felismerő AI-hoz:

```
opencv-python
ultralytics (YOLO)
torch vagy tensorflow
numpy
pillow
```

💡 Fejlesztési workflow tipikusan:

1. Kamera kép beolvasása → OpenCV
2. Kép feldolgozása → NumPy
3. Modell tanítása → PyTorch / TensorFlow
4. Objektum detektálás → YOLO
5. Eredmény megjelenítés → OpenCV

OpenAI ChatGPT-5.3,2026.03.07

8.6.5 LLM-alapú detektálás fejlesztés YOLO modell alkalmazásával

A melléklet a dolgozat során felhasznált, ChatGPT mesterséges intelligencia alapú nyelvi modell által generált releváns információegységek beszélgetését (promptok és válaszok) tartalmazza.

Kérdés: Az alábbi kódban javítsd ki, azt a problémát, hogy a gépnév és a bejelentkezett felhasználó is kerüljön bele a txt fájlba.

Válasz: Megoldjuk 👍 — csak ki kell egészíteni a kódot a felhasználónév és gépnév lekérésével, majd beleírni a logba.

☑ **JAVÍTÁS**

◆ 1. IMPORTOK kiegészítése

Tedd be a tetejére:

```
import getpass
import socket
```

◆ 2. Felhasználó + gép lekérése

A beállítások alá:

```
USERNAME = getpass.getuser()
```

```
MACHINE = socket.gethostname()
```

◆ 3. LOG ÍRÁS módosítása

Cseréld le ezt:

```
with open(LOG_FILE, "a") as f:
```

```
    f.write(f'{{timestamp}} PHONE DETECTED\n')
```

erre:

```
with open(LOG_FILE, "a", encoding="utf-8") as f:
```

```
    f.write(f'{{timestamp}} | USER: {{USERNAME}} | MACHINE: {{MACHINE}} | PHONE  
DETECTED\n')
```

KÉSZ KÓDRÉSZ (csak a fontos rész)

```
import getpass
```

```
import socket
```

```
USERNAME = getpass.getuser()
```

```
MACHINE = socket.gethostname()
```

```
...
```

```
with open(LOG_FILE, "a", encoding="utf-8") as f:
```

```
    f.write(f'{{timestamp}} | USER: {{USERNAME}} | MACHINE: {{MACHINE}} | PHONE  
DETECTED\n')
```


EREDMÉNY (logs.txt)

20260406_142355 | USER: janos | MACHINE: PC-01 | PHONE DETECTED

EXTRA (ajánlott)

Ha később akarod feldolgozni (pl. logelemző):

```
f.write(f'{{timestamp}};{{USERNAME}};{{MACHINE}};PHONE\n')
```

 így CSV-szerű lesz

JAVÍTÁS

◆ 1. IMPORTOK kiegészítése

Tedd be a tetejére:

```
import getpass
```

```
import socket
```

◆ 2. Felhasználó + gép lekérése

A beállítások alá:

```
USERNAME = getpass.getuser()
```

```
MACHINE = socket.gethostname()
```

◆ 3. LOG ÍRÁS módosítása

Cseréld le ezt:

```
with open(LOG_FILE,"a") as f:
```

```
    f.write(f'{{timestamp}} PHONE DETECTED\n')
```

erre:

```
with open(LOG_FILE, "a", encoding="utf-8") as f:
```

```
    f.write(f'{{timestamp}} | USER: {{USERNAME}} | MACHINE: {{MACHINE}} | PHONE  
DETECTED\n")
```

KÉSZ KÓDRÉSZ (csak a fontos rész)

```
import getpass
```

```
import socket
```

```
USERNAME = getpass.getuser()
```

```
MACHINE = socket.gethostname()
```

```
...
```

```
with open(LOG_FILE, "a", encoding="utf-8") as f:
```

```
    f.write(f'{{timestamp}} | USER: {{USERNAME}} | MACHINE: {{MACHINE}} | PHONE  
DETECTED\n")
```


EREDMÉNY (logs.txt)

```
20260406_142355 | USER: janos | MACHINE: PC-01 | PHONE DETECTED
```

EXTRA (ajánlott)

Ha később akarod feldolgozni (pl. logelemző):

```
f.write(f'{{timestamp}};{{USERNAME}};{{MACHINE}};PHONE\n")
```

 így CSV-szerű lesz

Kérdés 2: A kódban túl sok téves riasztás tapasztalható a detektálás során. Egészítsd ki a kódot validálással, hogy megszűnjön a probléma, én a frame késleltetést alkalmaznám.

Válasz: A gondod teljesen jogos: a mostani kódodban van már frame-alapú logika (phone_frames), de hiányzik pár kritikus stabilizáló elem, ezért még mindig jöhetnek téves riasztások.

A legnagyobb probléma jelenleg:

- nincs definiálva REQUIRED_FRAMES
- minden detektálás ugyanannyit “ér”
- nincs “memória” a bizonytalan frame-ekre (1 rossz frame = reset)

Ezt finomítjuk úgy, hogy ne legyen túl érzékeny, de ne is legyen lomha.

✓ 1. Add hozzá ezt a beállítást

A Beállítások részhez:

```
REQUIRED_FRAMES = 4    # ennyi egymást követő frame kell  
MAX_MISSES = 2        # ennyi hibás frame még belefér
```

🧠 2. Okosabb frame-validáció (nem nulláz azonnal)

Cseréld le ezt a részt:

```
if detected_this_frame:  
    phone_frames += 1  
else:  
    phone_frames = 0
```

👉 Erre:

```
if detected_this_frame:  
    phone_frames += 1  
else:  
    if phone_frames > 0:  
        phone_frames -= 1 # lassú visszaesés, nem azonnali reset
```

📌 Ez nagyon fontos:

- egy-egy hibás frame NEM törli az egész detektálást
 - jelentősen csökkenti a flicker miatti hibákat
-

🎯 3. Confidence stabilizálás (kulcs a false positive ellen)

Most csak azt nézed, hogy `conf > 0.8`. Ez nem elég.

Adj hozzá egy “összegző” confidence logikát:

Felül:

```
confidence_sum = 0
```

A detektálásnál:

```
if detected_this_frame:  
    confidence_sum += conf  
else:  
    confidence_sum *= 0.5 # lassú felejtés
```

4. Riasztás finomítása

Cseréld ezt:

```
if phone_frames >= REQUIRED_FRAMES and now - last_detection > COOLDOWN:
```

👉 Erre:

```
avg_conf = confidence_sum / max(phone_frames, 1)
```

```
if (phone_frames >= REQUIRED_FRAMES and  
    avg_conf > 0.75 and  
    now - last_detection > COOLDOWN):
```

Majd reseteld:

```
confidence_sum = 0
```

5. Extra stabilizálás (bounding box szűrés)

Ez gyakran okoz false pozitívat:

👉 Add hozzá:

```
ASPECT_RATIO_MIN = 0.3  
ASPECT_RATIO_MAX = 3.0
```

És a box ellenőrzéshez:


```
aspect_ratio = width / height if height != 0 else 0
```

```
if (width > MIN_BOX_SIZE and height > MIN_BOX_SIZE and  
    ASPECT_RATIO_MIN < aspect_ratio < ASPECT_RATIO_MAX):
```

Mit javítottunk valójában?

Ezzel a rendszered:

- ✓ nem ugrik egyetlen hibás frame-re
 - ✓ több frame-en át stabil detektálást vár
 - ✓ kiszűri a “villanó” false pozitívokat
 - ✓ figyelembe veszi a confidence trendet
 - ✓ nem túl lassú
-

 **Őszinte megjegyzés**

A cls == 67 (telefon) YOLO-nál híresen zajos osztály, szóval:

- még ezzel is lehetnek hibák
 - a legjobb boost: ROI (pl. csak képernyő környéke)
-

 **Ha még stabilabb kell**

Szint 2 megoldások:

- tracking (pl. bytetrack)
 - csak akkor riaszt, ha ugyanott van a box több frame-en át
 - motion + detection kombinálása
-