ÓBUDAI EGYETEM
ÓBUDA UNIVERSITY

NEUMANN JÁNOS
INFORMATIKAI KAR

SZAKDOLGOZAT

OE-NIK	Hallgató neve:	**Dóczi László**
2025	Hallgató törzskönyvi száma:	**T009121/FI12904/N**

Óbudai Egyetem
Neumann János Informatikai Kar
Szoftvertervezés és -fejlesztés Intézet

SZAKDOLGOZAT FELADATLAP

Hallgató neve: **Dóczi László**
Törzskönyvi száma: T009121/FI12904/N
Neptun kódja: [redacted]

A dolgozat címe:

Automatikus belépő személy azonosító és aktivitás naplózó rendszer
Automatic Entry Person Identifier and Activity Logger System

Intézményi konzulens: Dr. Tusor Balázs
Külső konzulens:

Beadási határidő: 2024. december 15.

A záróvizsga tárgyai: Számítógép architektúrák
Mesterséges intelligencia specializáció

A feladat

A hallgató feladata egy olyan szoftver fejlesztése, amely lehetővé teszi, akár egy helyiség, akár egy épület belépési pontjának ellenőrzését. A fő cél nemcsak azonosítani a belépő személyeket, hanem rögzíteni és naplózni is a belépéseket. A rendszer arcfelismerés során fogadja el, vagy tagadja meg a belépést, melynek alapjául egy webalkalmazás fog szolgálni. Mindemellett visszajelez az adminisztrátornak, ha illetéktelen személy próbál meg belépni, ezzel létrehozva egy hatékony belépési és aktivitás naplózó rendszert.

A dolgozatnak tartalmaznia kell:

- bevezetést, szükségesség felmérését,
- hasonló rendszerek vizsgálatát és összehasonlítását,
- arcfelismerő rendszerek működésének összefoglalását,
- arcfelismerő rendszer felépítését,
- arcfelismerő rendszer működését,
- arcfelismerő rendszer tesztelését,
- arcfelismerő rendszer hatékonyságának és megbízhatóságának elemzését.

ÓBUDAI EGYETEM * Neumann János Informatikai Kar * 1. Budapest

Ph.

.....
Dr. habil. Vámossy Zoltán
intézetigazgató

A szakdolgozat elévülésének határideje: **2026. december 15.**
(ÓE HKR 54.§ (10) bekezdés szerint)

A dolgozatot beadásra alkalmasnak tartom:

....
külső konzulens intézményi konzulens

ÓBUDAI EGYETEM
ÓBUDA UNIVERSITY

Neumann János Informatikai Kar

HALLGATÓI NYILATKOZAT

Alulírott Dóczi László (I2CQ73) hallgató kijelentem, hogy a szakdolgozat / diplomamunka saját munkám eredménye, a felhasznált szakirodalmat és eszközöket azonosíthatóan közöltem. Az elkészült szakdolgozatomban / diplomamunkámban található eredményeket az egyetem és a feladatot kiíró intézmény saját céljára térítés nélkül felhasználhatja.

Budapest, 2024.05.13.

Dóczi László

.....

hallgató aláírása

ÓBUDAI EGYETEM
ÓBUDA UNIVERSITY

Neumann János Informatikai Kar

KONZULTÁCIÓS NAPLÓ

Hallgató neve: Dóczi László

Neptun Kód: [redacted]

Tagozat: Nappali

Telefon: [redacted]

Levelezési cím (pl.: lakcím): [redacted]

Szakdolgozat / Diplomamunka[1] címe magyarul:

Automatikus belépő személy azonosító és aktivitás naplózó rendszer

Szakdolgozat / Diplomamunka2 címe angolul:

Automatic Entry Person Identifier and Activity Logger System

Intézményi konzulens: Dr. Tusor Balázs

Külső konzulens:

Kérjük, hogy az adatokat nyomtatott nagy betűkkel írja!

Alk.	Dátum	Tartalom	Aláírás
1.	2024.03.20.	Szakdolgozat szerkezetének átbeszélése	Tusor Balázs
2.	2024.04.04.	Az arcfelismerő metódusok áttekintése	Tusor Balázs
3.	2024.04.10.	Eddigi haladás bemutatása	Tusor Balázs
4.	2024.04.18.	Általános konzultáció	Tusor Balázs

A Konzultációs naplót összesen 4 alkalommal, az egyes konzultációk alkalmával kell láttamoztatni bármelyik konzulenssel.

A hallgató a Szakdolgozat / Szakdolgozat I. / Szakdolgozat II. / Projektlabor 1 / Projektlabor 2 / Projektlabor 3 / Záródolgozati projekt / Diplomamunka I / Diplomamunka II / Diplomamunka III / Diplomamunka IV [3] tantárgy követelményét teljesítette, beszámolóra / védésre [4]bocsátható.

A konzulens által javasolt érdemjegy: .....5.....

Budapest, 2024.05.13.....

Tusor Balázs

Intézményi konzulens

[1] Megfelelő aláhúzandó!
[2] Megfelelő aláhúzandó!
[3] Megfelelő aláhúzandó!
[4] Megfelelő aláhúzandó!

ÓBUDAI EGYETEM
ÓBUDA UNIVERSITY

Neumann János Informatikai Kar

KONZULTÁCIÓS NAPLÓ

Hallgató neve: Dóczi László

Neptun Kód: [redacted]

Tagozat: Nappali

Telefon: [redacted]

Levelezési cím (pl.: lakcím): [redacted]

Szakdolgozat / Diplomamunka[1] címe magyarul:

Automatikus belépő személy azonosító és aktivitás naplózó rendszer

Szakdolgozat / Diplomamunka2 címe angolul:

Automatic Entry Person Identifier and Activity Logger System

Intézményi konzulens: Dr. Tusor Balázs

Külső konzulens:

Kérjük, hogy az adatokat nyomtatott nagy betűkkel írja!

Alk.	Dátum	Tartalom	Aláírás
1.	2024.09.23.	Az első félév ütemtervének áttekintése, célkitűzés	Tusor Balázs
2.	2024.10.15.	Haladás megbeszélése	Tusor Balázs
3.	2024.11.04.	A szakdolgozat formai követelményeinek tárgyalása	Tusor Balázs
4.	2024.11.26.	Végső javítások és beadás előtti teendők áttekintése	Tusor Balázs

A Konzultációs naplót összesen 4 alkalommal, az egyes konzultációk alkalmával kell láttamoztatni bármelyik konzulenssel.

A hallgató a Szakdolgozat / Szakdolgozat I. / Szakdolgozat II. / Projektlabor 1 / Projektlabor 2 / Projektlabor 3 / Záródolgozati projekt / Diplomamunka I / Diplomamunka II / Diplomamunka III / Diplomamunka IV [3] tantárgy követelményét teljesítette, beszámolóra / védésre [4]bocsátható.

A konzulens által javasolt érdemjegy: 5

Budapest, 2024.12.02.

Tusor Balázs

Intézményi konzulens

[1] Megfelelő aláhúzandó!
[2] Megfelelő aláhúzandó!
[3] Megfelelő aláhúzandó!
[4] Megfelelő aláhúzandó!

Kivonat

Az automatikus arcfelismerés jelentős szerepet játszik a modern biztonsági rendszerekben, mivel lehetővé teszi az arcok pontos és gyors azonosítását. A dolgozat célja egy olyan webalapú arcfelismerő és logoló rendszer fejlesztése, amely képes valós időben felismerni és rögzíteni az arcokat. A rendszer fejlesztéséhez a Django keretrendszert használtuk, a Convolutional Neural Network (CNN) modellt az arcfelismeréshez, valamint a PostgreSQL adatbázist az adatok tárolásához.

Az összefoglaló bemutatja a rendszer fejlesztési folyamatát, az alkalmazott technológiákat, valamint a teszteredményeket. Végezetül, javaslatokat tesz a továbbfejlesztési lehetőségekre, mint például a pontosság növelésére, új funkciók integrálására és a rendszer skálázhatóságának javítására.

Abstract

Automatic face recognition plays a significant role in modern security systems as it enables accurate and rapid identification of faces. The aim of this thesis is to develop a web-based face recognition and logging system capable of real-time face detection and recording. For the development of the system, the Django framework was used, the Convolutional Neural Network (CNN) model for face recognition, and the PostgreSQL database for data storage.

The summary presents the development process of the system, the technologies used, and the test results. Finally, it provides suggestions for further development opportunities, such as increasing accuracy, integrating new features, and improving the scalability of the system.

TARTALOMJEGYZÉK

1 BEVEZETÉS

1.1 A projekt bemutatása

A projekt célja egy olyan program kifejlesztése, amely lehetővé teszi egy helyiség, épület vagy akár telefonos/webes alkalmazások belépésének ellenőrzését. A fő cél nemcsak azonosítani a belépő személyeket, hanem rögzíteni és naplózni is a belépéseket. A rendszer arcfelismerés alapján fogadja el vagy tagadja meg a belépést. Mindemellett visszajelez az adminisztrátornak, ha illetéktelen személy próbál meg belépni, ezzel létrehozva egy hatékony belépési és aktivitás naplózó rendszert. A rendszer alapjául egy webalkalmazás fog szolgálni, melynek a szolgáltatásait az adminisztrátor a saját belépési adataival tud elérni. A webalkalmazás felhasználóbarát felületet biztosít, ahol az adminisztrátorok nyomon követhetik a belépési eseményeket, kezelhetik a felhasználói adatokat és beállíthatják a rendszer paramétereit.

1.2 A projekt célja

A projektem célja, hogy egy olyan programot hozzak létre, mely egy stabil beléptető rendszer alapjául szolgál. Képes legyen felismerni automatikusan, hogy a bemeneti, élő videón egy valós személyt láthatunk, vagy egy képet, melyet éppen maga elé tart valaki. A kiértékelést egy webes felületen láthatjuk majd, illetve a rendszergazda szintúgy egy ilyen felületen fogja látni, a logokat.

1.3 A projekt aktualitása

A rohamosan fejlődő világunkba, úgy érzem sosem tudhatjuk biztonságban értékeink, adataink, ezek alól nem kivételek a cégek, illetve a magánszemélyek sem. Egy ilyen rendszer nagyban elősegíti az épületeink, helységeink és akár weboldalaink biztonságát is, azzal, hogy kiszűri és megtagadja a belépést egy idegen, rosszindulatú emberrel szemben.

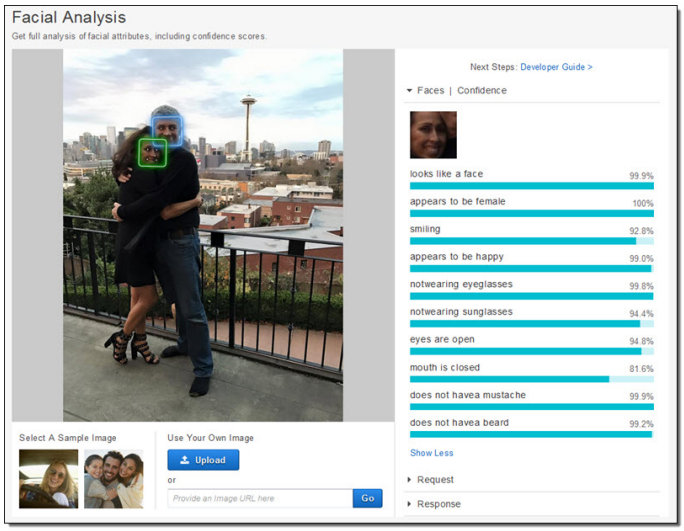
2 IRODALOMKUTATÁS

2.1 Hasonló rendszerek

A technikai fejlődés és a biztonsági igények növekedésével, egyre nagyobb érdeklődés mutatkozik az ilyen rendszerek iránt. Mindezek hatására, számos vállalat kiveszi a részét abból, hogy egy megbízhatóbb rendszert hozzanak létre vetélytársaiknál. Az alábbiakban a hasonló rendszerek összehasonlítása lesz a cél.

2.1.1 Amazon Rekognition

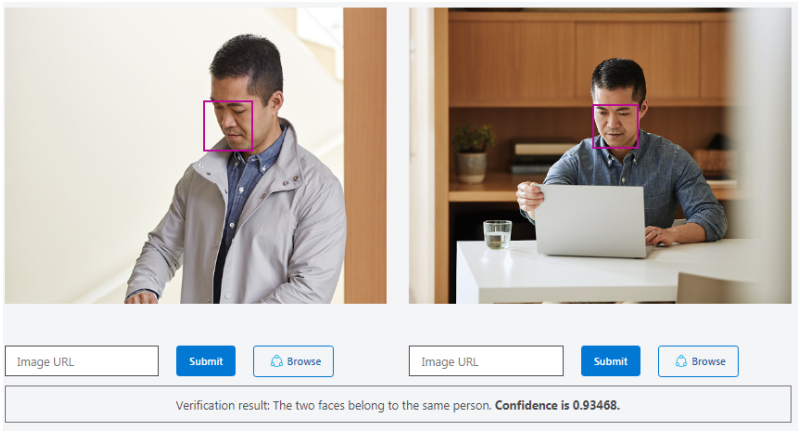
A számítástechnika egyik meghatározó szereplője, az Amazon is fejleszt egy hasonló rendszert. 2016-ban adták ki a saját fejlesztésű felhő alapú Amazon Rekognition [1] névre hallgató szoftverüket. Az Amazon Rekognition széleskörű lehetőséget biztosít, minden olyan cégnek, akik szeretnének élni a gépi tanulás előnyeivel, legyen szó akár képfelismerésről, akár videó analizálásról. A felhasználónak nem kell rendelkeznie semmilyen gépi tanulás tapasztalattal, mely imponáló lehet a cégek számára. Ezen felül a legkiemelkedőbb tulajdonsága, hogy elég magas szintű pontosságot lehet vele elérni. Mindezt képes kiterjeszteni, arc (2.1 ábra), objektum, jelenet és akár szöveg detektálásra is. Továbbá bizalomküszöb bevezetésével lehetséges állítani a minimális megbízhatósági szintet, mely befolyásolja a kapott eredményt. Magas biztonsági szintet igénylő helyzetekben, ahol a hamis észlelés kockázata magas, például: repülőtereken/határátkelőhelyeken, érdemes a küszöböt 99% körüli értékre állítani, míg egy általános felhasználáshoz elég lehet egy 80% körüli bizalomküszöb is. Az arcfelsimerő rendszerük felhasználásával képesek lehetünk azonosítani például diákokat, munkavállalókat csupán a személyazonosító okmányuk segítségével. Plusz funkciója, hogy képes felismerni az élőséget a médiatartalmakon, azaz ellenőrizni, hogy a valós ember látható-e, vagy csak egy kinyomtatott kép/maszk, ezen felül a felhasználó elláthatja ezeket a tartalmakat címkékkel, mely által kereshetővé teszi azokat.



2.1. ábra: Amazon Rekognition felülete [2]

2.1.2 Microsoft Azure AI

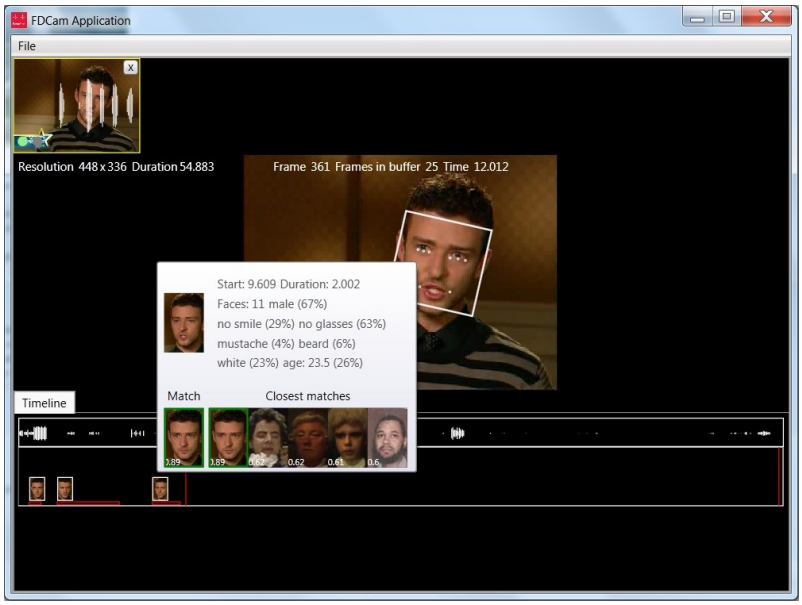
A Microsoft-nak is van egy megoldása a problémára, melyet Microsoft Azure AI Vision [3] néven találhatunk meg. Az Azure AI Vision szintúgy egy felhő alapú képfelismerő és azonosító rendszer, melyet specifikusan lehet arcfelsimerésre is használni (2.2 ábra). Magas szintű pontosság és megbízhatóság jellemzi. Összetettségének ellenére ez a rendszer is könnyen testre szabható különböző, egyéb beépített funkciókkal, például: élőség, érzelmek, életkor meghatározásával. A rendszer akár azt is megkérheti tőled, hogyha napszemüveget viselsz, akkor azt vedd le. Ennek ellenére ez a rendszer is érzékeny tud lenni a pontosságra, külső hibákból adódóan, mint például: extrém világítás, objektumok melyek eltakarják az arc bizonyos részeit stb. A biztonságot tekintve, itt is érdemes lehet beállítani bizalomküszöböt, melyekkel el lehet kerülni bizonyos későbbi komplikációkat.



2.2. ábra: Microsoft Azure AI felülete [4]

2.1.3 Betaface

A kevésbé ismert Betaface [5] névre hallgató cég is egy magas szintű, széles körben alkalmazható arcfelismerő rendszert kínál elénk (2.3 ábra). A rendszer számos funkciója lehetővé teszi a könnyebb testreszabhatóságot Windows és Linux operációs rendszereken. Ilyen funkciók lehetnek, például: arcprofil, életkor elemzés és akár nemzetiség felismerés. Mindezt sima képek/videók vagy élő videók alapján. A pontosságát a több mint 40 arcjellemző kiértékelésének köszönhetően kapja. Itt is találkozhatunk élőség vizsgálattal a nagyobb biztonság érdekében. Annak ellenére, hogy nem annyira ismert, mint az Amazon Rekognition vagy az Azure AI Vision, ez a rendszer is képes kezelni a nagy mennyiségű adatot és felhasználót. Könnyen integrálható egyedi rendszerekbe, legyen szó, alkalmazásról vagy webhelyről. Illetve számos API-t kínál a fejlesztők számára.

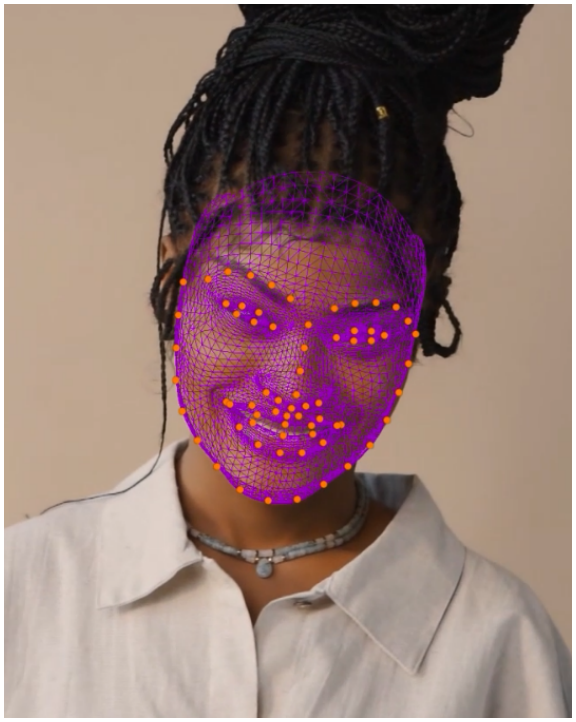


2.3. ábra: Betaface felülete [6]

2.1.4 Banuba

A Banuba [7], szintén egy sokoldalú, dinamikus rendszer, mely számos területen megállja a helyét, például szórakoztató iparban, digitális marketingben is, amit a különböző szolgáltatásainak köszönhet. Az arcdetektáláson és arcfelismerésen (2.4 ábra) kívül foglalkozik szem lekövetéssel, automatikus háttérkép változtatással, mesterséges intelligencia által vezérelt videó szerkesztővel és különböző virtuális Try-On filterekkel is. A fejlesztők számára számos API-t és SDK-t kínál, mely nagyban elősegíti a felhasználói élményt. Megannyi híres ügyfél dolgozik velük együtt, beleértve a Samsungot és a Guccit is. Rendszereik elérhetőek, Androidon, IOS-n, MacOS-n, Windowson és Linuxon is

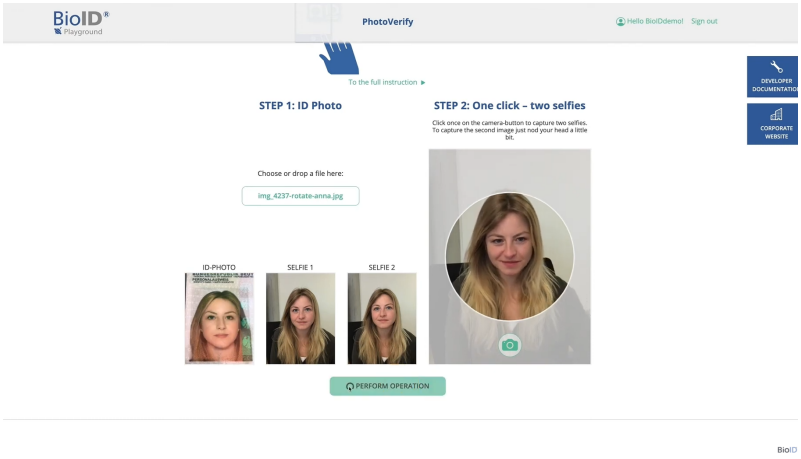
egyaránt. A neurális hálózatokat optimalizálták annak érdekében, hogy az alacsonyabb teljesítményű készülékeken is problémamentesen működjenek. Az arcfelismerésben jelentős szerepet játszó élőség felismerés itt sem maradt el. Itt is elég magas a pontosság, mely köszönhető annak, hogy az algoritmus képes eltávolítani bizonyos keretek között a virtuális zajt, megszüntetve a rázkódást.



2.4. ábra: Banuba egy tesztesete [8]

2.1.5 BioID

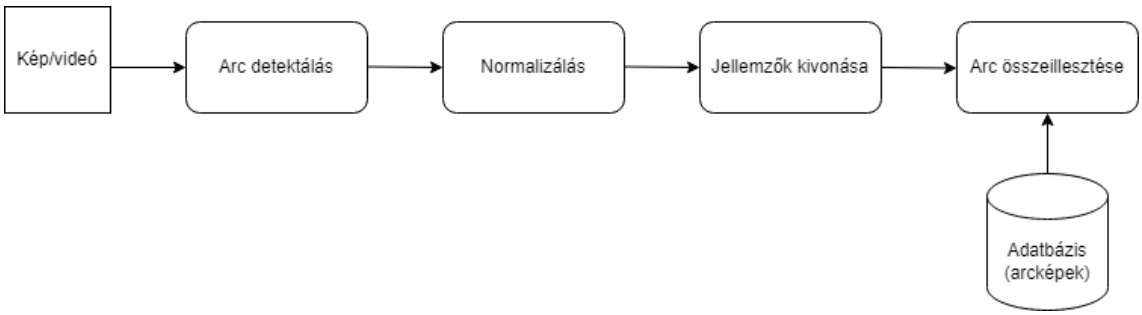
A BioID [9] is egy, a biometrikus azonosítás és az arcfelismerés terén kiemelkedő teljesítményt nyújtó rendszerek egyike. Az egyik legfontosabb jellemzője a fejlett élőfelismerési technológia alkalmazása, amelyek lehetővé teszik az azonosítás során a felhasználók jelenlétének valós idejű észlelését és ellenőrzését. Ez a funkció különösen fontos az üzleti szektorban és a pénzügyi szolgáltatásokban, ahol a biztonság és az azonosítás megbízhatósága kulcsfontosságú. A BioID további szolgáltatásai közé tartozik az arc alapú azonosítás és az azonosítás megerősítése további biometrikus jellemzők, például az ujjlenyomatok vagy a hang alapján. Ez a többfaktoros azonosítási megközelítés javítja a felhasználói azonosítás biztonságát és megbízhatóságát, minimalizálva a visszaélések kockázatát. A BioID rendszer (2.5 ábra) rendkívül rugalmas és könnyen integrálható más alkalmazásokba és platformokba. Számos API-t és SDK-t kínál a fejlesztőknek, amelyek lehetővé teszik a testreszabott megoldások fejlesztését az egyedi üzleti igényeknek megfelelően.



2.5. ábra: BioID felülete [10]

2.2 Az arcfelismerés folyamata

Az arcfelismerés egy olyan összetett folyamat, amely magába foglalja az arcdetektálás, arcszegmentáció és az azonosítás feladatkörét (2.6 ábra).



2.6. ábra: Arcfelismerés folyamata

2.2.1 Kezdetleges arcdetektálási algoritmusok

Az egyik legalapvetőbb lépés az arcfelismerésben az arcdetektálás. A detektálás magába foglalja az arc körvonalainak azonosítását, helyzetének meghatározását.

Az első ilyen algoritmus mely képes volt kimagasló pontossággal arcokat detektálni, két kutató nevéhez köthető. Paul Viola és Michael Jones 2001-ben megalkotta a Viola Jones algoritmust [11]. Az algoritmus alapja, hogy egy szürkeárnyalatos képen különböző méretű és helyzetű ablakokat vizsgál meg, keresve a specifikus képi jellemzőket.

Négy fő lépésből áll:

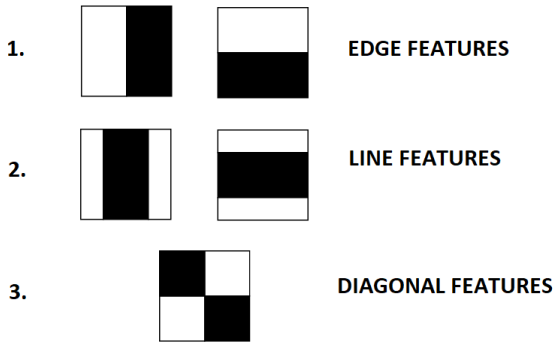
– Haar-jellemzők kiválasztása

– Integrális kép
– AdaBoost tanítás
– Kaszkád osztályzás

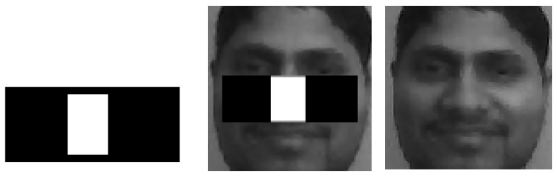
A Haar-jellemzők (2.7, 2.8 ábra) olyan alapvető mintázatok, amelyek segítik az algoritmust az arc különböző részeinek a felismerésében.

3 fajta ilyen jellemző van:

– Él jellemzők
– Vonal jellemzők
– Középpont jellemzők



2.7. ábra: Haar-jellemzők [12]



2.8. ábra: Haar-jellemzők használatban [13]

Ezek a jellemzők nem rögzített méretűek, szükség szerint méretezhetőek. Számításilag egyszerűek, melynek köszönhetően hatékonyak lehetnek valós idejű alkalmazásokban. Működésük alapja, hogy ezek segítségével kiszámíthatjuk a különbséget a szomszédos téglalap alakú régiókban lévő pixelösszegek között. Az él jellemzők élváltozást rögzítenek. Két egymás melletti téglalapot tartalmaznak ellentétes intenzitásértékekkel. A vonal jellemzők függőleges és vízszintes vonalakat észlelnek. Ezek három egymás melletti téglalapot tartalmaznak váltakozó intenztiás értékekkel. A középpont jellemzők pedig kettő vagy több téglalapot tartalmaznak változó intenzitásokkal, melyek képesek kihangsúlyozni a középpont és a környező területek közötti különbségeket.

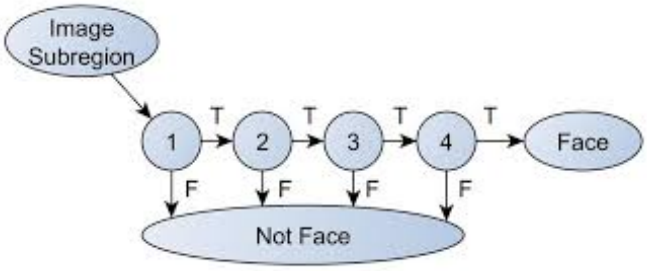
Ezen számítások felgyorsítása érdekében használatos az integrális kép [14]. Az integrálási képet (2.1) képlet alapján lehet kiszámolni.

$$I(x,y) = \sum_{\substack{x' \leq x \\ y' \leq y}} i(x',y') \tag{2.1}$$

Ahol i a szürkeárnyalatos kép és $I(x,y)$ az integrálási kép adott pontjának értéke, azaz az (x,y) pontban az összes olyan képpont összege, amelyek az (x,y) pont fölött és balra találhatóak, beleértve az (x,y) pontot is. Egy n pixelből álló kép esetén az integrál kép számításának időkomplexitása $O(n)$, azaz lineáris idejű.

A számításoknak köszönhetően tömérdek jellemző fog használatunkra állni, melyek közül csak néhány fontos az arcok azonosításához. Az objektumok/arcok felismerésének pontosságának javításához ezért AdaBoost algoritmust érdemes használni. Az AdaBoost képes kiválasztani és összevonni több gyenge osztályozót (pl. Haar jellemzők), hogy egy erős osztályozót hozzon létre.

Ennek ellenére sok adat marad meg, ami nem releváns detektálás szempontjából. Ezért egy kaszkád osztályozó [14] (2.9 ábra) segítségével szűrjük az adatokat a detektálás sebességének növelése érdekében. A kaszkád osztályzás lényege az, hogy az objektumfelismerést több lépcsőben végzi, ahol minden lépés egyre finomabb szűrőket alkalmaz az adatokra. A kaszkádok olyan sorrendben vannak felépítve, hogy először gyorsan kiszűrjék a nem érdekes régiókat, majd egyre részletesebben ellenőrizzenek minden lehetséges tartományt az objektumok azonosítására.



2.9. ábra: Kaszkád osztályozó működése [15]

Összességében ez egy egyszerűen implementálható arc/objektum detektáló modell, amely még mindig népszerű a valós idejű alkalmazásokban történő arcok felismerésére. Habár problémái vannak azon arcok azonosításával, amelyek takarva vannak, vagy nem megfelelően orientáltak.

A Dlib HOG (Histogram of Oriented Gradients) [16] is, egy a népszerű arcdetektá-

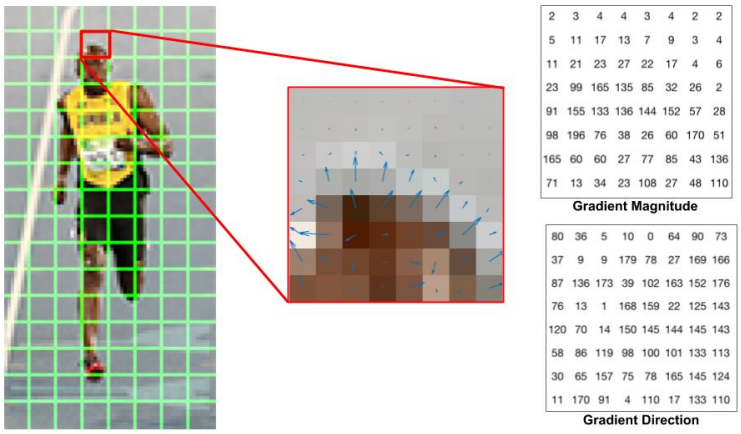
ló algoritmusok közé tartozik. Az algoritmus alapja, hogy a jellemzőket egy vektorba csoportosítjuk, majd egy osztályozó algoritmussal, például SVM(Support Vector Machine) segítségével megítéli, hogy az adott területen látható-e arc. Ez a metódus is szintúgy használható objektumdetektálásra is.

Az előzetes képfeldolgozást követően, további négy lépéssel érhetjük el a kívánt eredményt. Ezen fő lépések az alábbiak:

- Gradiensek kiszámítása
- HOG kiszámítása
- Hisztogram normalizálás
- SVM osztályozó feltanítása

Első lépésként a gradienseket kell kiszámítanunk (2.10 ábra), ami nem csak, hogy csökkenti a kép dimenzióját, hanem segít az élek és a fontosabb tulajdonságok kiszűrésében. Itt érdemes mind a horizontális és vertikális gradienseket kiszámítani a nagyobb pontosság érdekében.

Ezt követően A képet 8x8 cellákra osztjuk, melynek lényege, hogy a zajjal szemben ellenállóbb legyen. Ezekre a cellákra külön-külön kiszámítjuk a HOG-ot. Egy hisztogramot építünk a gradiens irányokból és a gradiens nagyságokból. Ebben az esetben mindkét területen 64(8*8) érték van.



2.10. ábra: Gradiensek reprezentálása [17]

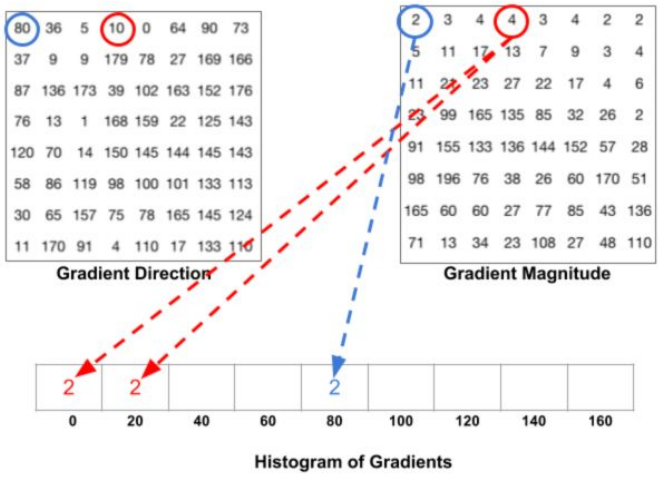
A hisztogram kategóriák a gradiens szögekkel vannak azonosítva. 0-180 fokig vannak a szögek felosztva, azaz 9 darab kategória van összesen (0-160 fokig 20 fokos lépésközzel) (2.11 ábra).

A HOG építésénél három különböző esetet veszünk figyelembe:

- A szög kisebb, mint 160 fok és nem két osztály között helyezkedik el, akkor ez a

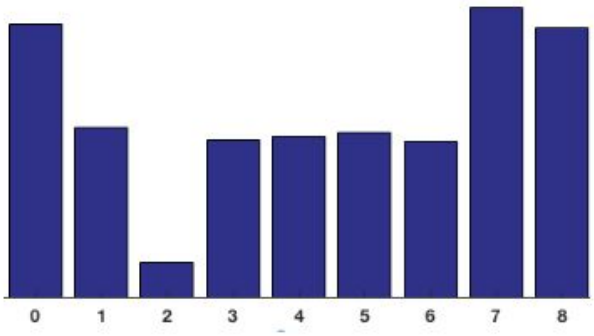
szög a szöghelyes kategóriához adódik hozzá

– A szög kisebb, mint 160 fok és két osztály között helyezkedik el, akkor ez a szög elfelezve a két közel álló kategóriához adódik hozzá
– A szög nagyobb mint 160 fok, akkor ez a szöget arányosan hozzáadjuk a 0 és a 160 fokos kategóriához is.



2.11. ábra: HOG felépítése [18]

Végezetül, minden 8x8 cellához fog tartozni egy hisztogram (2.12 ábra):



2.12. ábra: Hisztogram reprezentáció [19]

Utolsó lépések egyikeként, pedig normalizáljuk a hisztogramot. Annak érdekében, hogy a képet invariánssá tegyük, például a világításra.

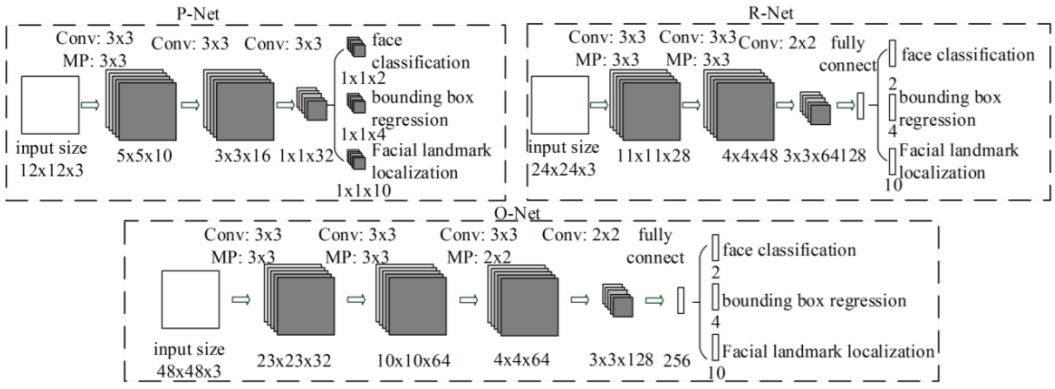
Ha pedig minden adat a rendelkezésünkre áll, feltaníthatjuk az SVM osztályozónkat.

Összességében ezt a modellt is könnyű implementálni, jól használható frontális és kicsit oldalra figyelő arcok detektálására is. Viszont hátrányként lehet említeni, hogy valós idejű alkalmazásnál elég lassú tud lenni.

Az iparág fejlődésével jelentek meg egyre kifinomultabb rendszerek. Így volt ez az MTCNN (Multitask Convolutional Neural Network) [11] algoritmussal is. Az algoritmust 2016-ban fejlesztették ki, amely neurális hálózatok sorozatát használja annak érdekében, hogy magas sebességgel és pontossággal tudja észlelni az arcokat.

Összesen 3 neurális hálózatból áll, melyek egy kaszkádban vannak összekapcsolva (2.13, 2.14 ábra):

- P-Net (Proposal Network)
- R-Network (Refine Network)
- O-Net (Output Network)



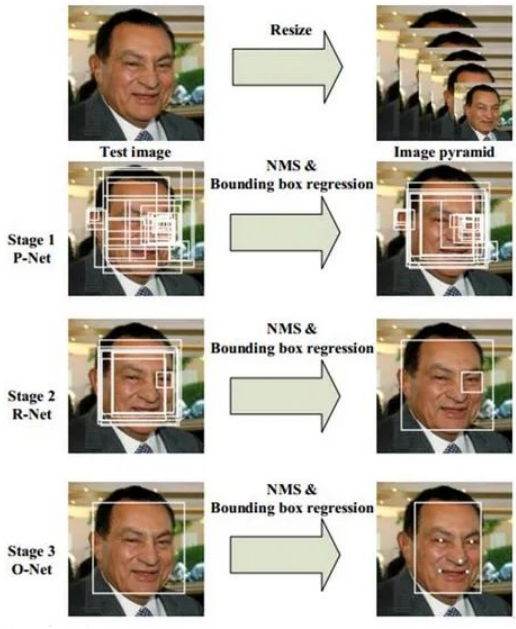
2.13. ábra: A 3 használt neurális hálózat felépítése [20]

Az első szakasza az algoritmusnak a Proposal Network. Célja, hogy gyorsan előállítson egy adathalmazt, amelyek potenciális jellemzőkkel bírnak, azaz arcokat tartalmazhatnak. Ezek a későbbiekben további finomításon fognak átesni.

Az algoritmus először a bemeneti képen egy sor konvolúciós szűrőt alkalmaz, annak érdekében, hogy ezeket a jellemzőket kiszűrje. Egy teljesen összekapcsolt réteg fogja ezt az tulajdonság halmazt feldolgozni, hogy megjósolja annak a valószínűségét, hogy az arc jelen van-e a kép egy bizonyos régióiban. Mindemellett az észlelt arc körüli határolókeret koordinátáit is visszaadja. A hálózatnak alacsony küszöbérték van beállítva, ami miatt sok téves pozitív eredményt adhat vissza, de direkt ilyen megengedő, hisz mindezt elég a későbbiekben finomítani.

Ezt követően a P-Net által generált adatokat átadjuk az R-Net neurális hálózatnak. Ez fogja finomítani az előzőekben generált határoló dobozokat. A bemeneti kép megfelelő régióit kivágja a határoló kereteknek megfelelően és egy sor összekapcsolt rétegen, keresztül osztályozza azokat arc és nem arc csoportokba, megszabadulva ezzel az alacsonyabb konfidencia szintű dobozoktól. Majd ezt a finomított határoló kereteket tartalmazó adathalmazt továbbítja.

Az algoritmus utolsó fázisa nagyon hasonló az előző kettőhöz, de ebben az esetben több felügyelettel próbálja meg a háló az arcrégiókat azonosítani. Különös figyelmet fordít az öt arcjellegzetesség helyzetének meghatározására, amelyet kimenetként is vissza fog adni a határoló keretek koordinátáival együtt.



2.14. ábra: Az MTCNN algoritmus fázisai [20]

Az előbbi megoldásokkal szemben az MTCNN kimagasló pontossággal rendelkezik. Mellette szól, hogy valós idejű arcdetektálásra is alkalmas, illetve, hogy különböző helyzetekből, például oldalról is képes egy arcot felismerni. Hátrányaként említhető az, hogy a feltanítása időigényesebb a komplexitásából eredően, az előző modellekhez képest.

Idővel megjelentek a konvolúciós neurális hálózatok, amelyek hatékonyabbnak bizonyultak az arcdetektálásban és az arcfelismerésben is egyaránt. Hatékonyságát főképp annak köszönheti, hogy képes csökkenteni a dimenziót és osztályozóként tanítható.

Végezetül a 2.1 táblázat a fentebb említett és vizsgált arcdetektáló algoritmusokról ad egy átfogó képet:

2.1. táblázat: Összehasonlító táblázat a vizsgált algoritmusok között [20]

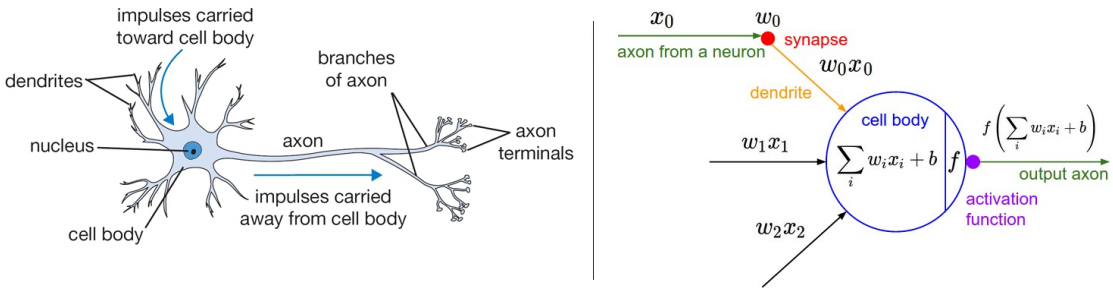
Algoritmusok	**Előny**	**Hátrány**
Viola James	Könnyű implementálás, Valós idejű arcfelismerés	Oldalra néző arcokkal nem működik, Sok hamis predikció
Dlib HOG	Könnyű implementálás, Bizonyos mértékig tudja kezelni az oldalra néző arcokat	Bizonyos szint felett nem tudja felismerni az oldalra néző arcokat, Lassú valós idejű arcfelismerés
MTCNN	Magas pontosság, Hatékony, Valós idejű arcfelismerés	A feltanítása sok időbe kerül

2.3 Neurális hálózatok

A neurális hálózatok olyan rendszerek, amelyek az emberi agy működését utánozzák. Képesek tanulni és önálló döntéseket hozni. Ezek az algoritmusok az adatokban rejlő összefüggéseket fedezik fel, és alkalmazzák azokat az új problémák megoldására.

2.3.1 Neurális hálózatok alapja

A neurális hálózat rétegei és csomópontjai nagyon hasonlítanak az emberi agy struktúrájához. Az agyban az idegsejtek kapcsolatban állnak egymással, és impulzusokat továbbítanak egymás között, ami hasonlít a neurális hálózatokban a neuronok közötti kapcsolatokhoz és az információ áramlásához (2.15 ábra). Ebből kifolyólag tudunk új dolgokat megtanulni, ahogy azt a háló is teszi. A neurális hálózatok képesek tanulni az adatokból és az ismételt visszacsatolásokból, illetve tudnak alkalmazkodni a változó körülményekhez. Fő célja, hogy felismerjék a mintázatokat és összefüggéseket az adatokban.



2.15. ábra: Idegsejt és egy matematikai neuron összehasonlítása [21]

A neuronok felépítése matematikai képlettel egyszerűen felírható. Mindehhez először is kell egy függvény (2.2) mely a súlyozott összeget számolja ki, ahol y a kimenetet, x_i a bemeneti jelek erősségét, a w_i a súlyokat és a b az eltolást (bias) jelenti.

$$y = \sum_{i=1}^{\infty} x_i w_i + b \tag{2.2}$$

Ezt követően pedig egy aktivációs függvény (2.3) (f) segítségével megkapjuk a neuronunk kimenetét (Y).

$$Y = f\left(\sum_{i=1}^{\infty} x_i w_i + b\right) \tag{2.3}$$

Azt aktivációs függvények általánosságban 2 csoportba oszthatóak [22]:

- Lineáris aktivációs függvény
- Nem lineáris aktivációs függvény

Amikor lineáris aktivációs függvényt használsz minden rétegben, az eredményül kapott hálózat ugyanolyan egyszerű és lineáris lesz. Ez azt jelenti, hogy a neurális háló nem lesz képes olyan bonyolult mintákat vagy kapcsolatokat megtanulni, amelyekre a valóélet-beli problémák általában igényt tartanak. Emiatt általában nem a lineáris aktivációs függvényeket használják a neurális hálózatokban, hanem inkább a nem lineáris aktivációs függvényeket.

A nem lineáris aktivációs függvények, mint például a ReLU (Rectified Linear Unit), Sigmoid vagy Tanh, képesek a bemeneti adatok nemlineáris transzformációjára, ami lehetővé teszi a hálózat számára, hogy különböző bonyolultabb mintákat és összefüggéseket tanuljon meg. Ezáltal a hálózat hatékonyabban tud reprezentálni és feldolgozni a valós világban előforduló változatos és összetett adatokat.

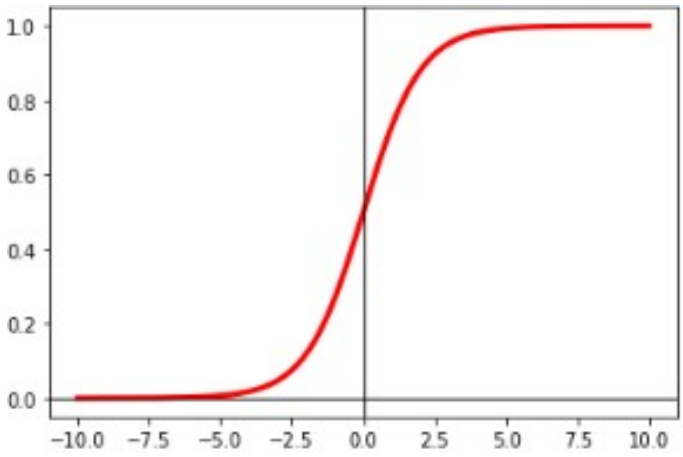
Az egyik legtöbbet használt aktivációs függvény a neurális hálózatok építése során az úgynevezett sigmoid függvény (2.16 ábra). A sigmoid függvény segít normalizálni bármely bemenet kimenetét a 0 és 1 közötti tartományban. Az aktiváló funkció fő célja, hogy a kimenetet vagy a becsült értéket az adott tartományban tartsa, ami jó hatásfokot eredményez.

Egyenlete (2.4):

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{2.4}$$

Az x értéke minél kisebb, a sigmoid függvény értéke annál közelebb lesz 0-hoz. Ha

pedig az x értéke egyre nagyobb lesz, akkor a sigmoid értéke annál közelebb lesz az 1-hez.



2.16. ábra: Sigmoid függvény [23]

Előnyök:

- Az egyik legjobb normalizált függvény
- Egyértelmű előrejelzést ad (pl: osztályozásnál)

Hátrányok:

- Hajlamos a "Vanishing Gradient" problémára
- Mindig pozitív értékeket ad vissza
- Számításigényes függvény (exponenciális)

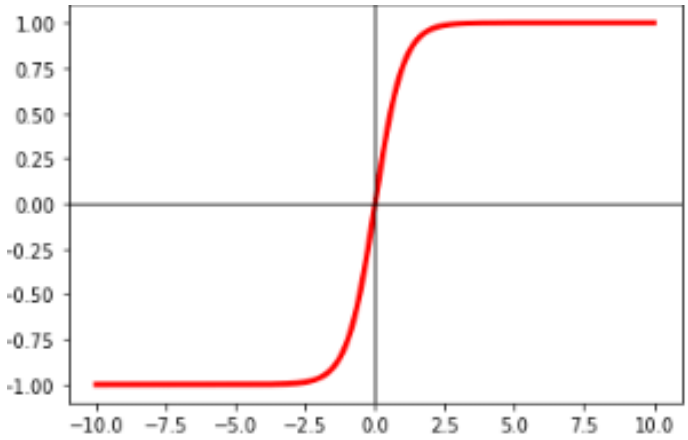
A "Vanishing Gradient" probléma egy olyan jelenség, amely akkor jelentkezik, amikor a gradiens (lehetséges változás) túl kicsi és gyakran nullához közelít az alacsonyabb rétegekben visszaterjesztés (backpropagation) során. Amikor a gradiens túl kicsi vagy nullához közelít, a súlyokat nem fogja megfelelően frissíteni a tanulási algoritmus, ami azt eredményezi, hogy az adott réteg nem fog megfelelően tanulni, és a hálózat által elért eredmények nem javulnak, sőt akár romolhatnak is.

A másik népszerű aktivációs függvény a Tanh (tangens hyperbolicus) (2.17 ábra). Nagyon hasonlít a sigmoid függvényhez, hiszen mindkettő az S-alakú függvények közé tartozik, amelyek korlátok közé szorítják a bemeneti értékeket.

Egyenlete (2.5):

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{2.5}$$

Ellenben a Sigmoid aktivációs függvénnyel valamivel hatékonyabb, mert ennek az aktivációs függvénynek az értékkészlete nagyobb, mint a sigmoid aktivációs függvényé.



2.17. ábra: Tanh függvény [23]

Előnyök:

– Szélesebb értékkészlet, a sigmoiddal ellentétben

Hátrányok:

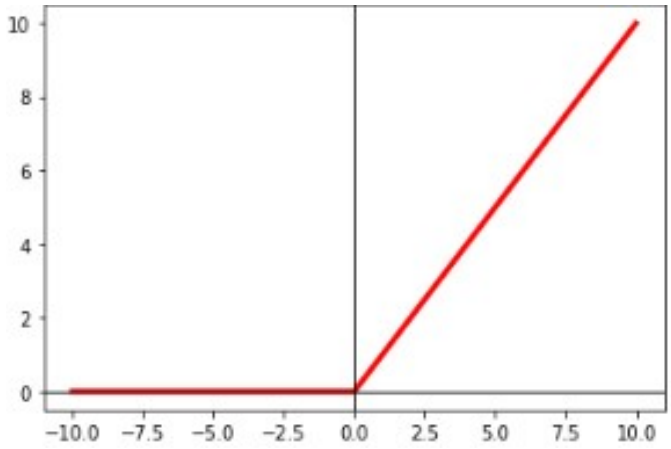
– Hajlamos a "Vanishing Gradient" problémára
– Számításigényes függvény (exponenciális)

Utoljára pedig szintúgy az egyik legismertebb aktivációs függvényről, a ReLU-ról lesz szó (2.18 ábra). A ReLU-t nem használták korábban főként azért, mert nem volt differenciálható a nullapontban. A kutatók inkább differenciálható aktivációs függvényeket, mint például a sigmoid és a tanh függvényeket részesítették előnyben. Azonban ma már megállapították, hogy a ReLU az egyik legjobb aktivációs függvény a mély tanuláshoz.

Egyenlete (2.6):

$$f(x) = \max(0, x) \tag{2.6}$$

Sikerességét annak köszönheti, hogy a sigmoid és a tanh aktivációs függvény hiányosságait ezzel a függvénnyel egész jól át lehet hidalni.



2.18. ábra: Tanh függvény [23]

Előnyök:

- Nincs "Vanishing Gradient" probléma
- Gyorsabb más függvényekhez képest

Hátrányok:

- Minden negatív érték azonnal nullává változik

Utóbbit könnyen orvosolhatjuk a ReLU aktivációs függvény különböző változatainak segítségével, például leaky ReLU-val.

Az aktivációs függvények közötti hasonlóságok és különbségek szemléltetésére a 2.2 táblázat ad jobb rálátást:

2.2. táblázat: Függvények összehasonlítása

Aktivációs függvény	Sigmoid	Tanh	ReLU
Értékkészlet	0-tól 1-ig	-1-től 1-ig	0-tól végtelenig
Vanishing gradient probléma	Van	Van	Nincs
Pontosság	Jó	Nagyon jó	Legjobb
Gyorsaság	Lassú	Lassú	Gyors

Ezeken felül érdemes több aktivációs függvényt is kipróbálni, mert a feladat szabja meg igazából, hogy melyik lesz a legoptimálisabb a kívánt eredmény eléréséhez. Illetve a különböző rétegek funkciója is meghatározza, hogy éppen mit érdemes rajtuk használni.

2.3.2 Neurális hálózatok felépítése

A neurális hálózatok általában több rétegből állnak (2.19 ábra). Ezek a rétegek neuronokból épülnek fel. Az egyes neuronok összeköttetésben vannak az előző és a következő rétegek neuronjaival. Ezeket az összeköttetéseket súlyoknak nevezzük, és azok adják a hálózat tanulható paramétereit.

A neurális hálózatok általában a következő rétegeket tartalmazzák [24]:

- Bemeneti réteg
- Rejtett rétegek
- Kimeneti réteg

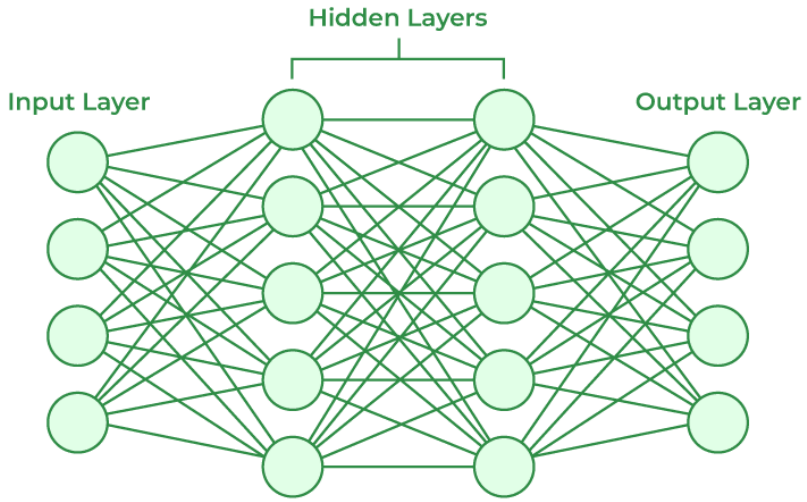
A bemeneti réteg a hálózat első rétege, amely fogadja a bemeneti adatokat. A bemenetek lehetnek például képek, hangok, szövegek vagy egyéb formátumok. Az arcfelismerés során a bemeneti formátum lehet videó illetve kép is. Általában ezek a pixelértékek 2D vagy 3D tenzorok formájában érkeznek be, attól függően, hogy a képek színesek vagy fekete-fehérek. Például egy színes kép esetén a bemeneti réteg csak egy 3D tenzort fogadhat el, ahol a magasság, a szélesség és a színcsatornák (például a vörös, zöld és kék) is megjelennek.

Fontos megjegyezni, hogy az arcfelismerési rendszerekben a bemeneti réteg gyakran része az előfeldolgozási folyamatnak, amely magában foglalhatja például a képek méretezését, forgatását vagy normalizálását is, hogy a bemeneti adatokat megfelelően fel lehessen dolgozni a hálózat számára.

A rejtett rétegek a bemeneti rétegek után következik. Minél több rejtett réteget adunk hozzá a hálózathoz, annál mélyebbnek nevezzük azt. Ezek a rétegek felelősek azért, hogy kinyerje a hálózat a bemeneti képekben rejlő fontos jellemzőket és mintákat. Az arcfelismeréshez használt neurális hálózatokban a rejtett rétegek számát, valamint a rétegekben lévő neuronok számát az adatok bonyolultságától és az alkalmazott modell architektúrájától függően kell kialakítani.

Például, ha a feladat nagyon összetett, ilyen például az arc különböző jellemzőinek felismerése, mint az ajkak, szemek, orr stb., akkor lehet, hogy több és mélyebb rejtett rétegre van szükség a hálózatban annak érdekében, hogy ezeket a részleteket hatékonyan kinyerje és reprezentálja.

Legvégül pedig a kimeneti réteg következik. Ez a hálózat utolsó rétege, amely a kívánt kimenetet generálja. A kimeneti réteg struktúrája függ attól, hogy a hálózatot milyen feladatra tervezték, mit várunk el eredményként. Például egy osztályozási feladat esetén a kimeneti réteg lehet egy softmax aktivációs függvény, amely valószínűségi eloszlást ad az egyes osztályokra.



2.19. ábra: Neurális hálózatok felépítése [24]

2.4 Neurális hálózatok használata az arcfelismerésben

A neurális hálózatok elterjedésével jelentek meg újabb és újabb megoldások, melyek jelentős előrelépést hoztak ezen a téren. Talán a legtöbbet használt mélytanulási módszerek közé tartoznak a konvolúciós neurális hálózatok (CNN-ek), amelyeket arcfelismerésre is előszeretettel használnak. A mélytanulási módszerek fő előnye, hogy nagy mennyiségű adatokkal taníthatók. Ily módon ellen tudnak állni különböző típusú zajoknak/zavaró tényezőknek, mint például világítás, arckifejezés, póz, életkor. Azonban a mélytanulási algoritmus előnye egyben a hátránya is, hisz nagyon nagy adathalmazzal kell őket tanítani, amelyek elegendő variációt tartalmaznak ahhoz, hogy általánosíthatók legyenek a még nem látott mintákra. Szerencsére több nagyszabású, arcokat tartalmazó adatbázis jelent meg az évek alatt a nyilvánosság számára.

Az egyik legelterjedtebb megközelítés az, hogy az arcfelismerési problémát osztályozási feladatként kezeljük. Ebben az esetben a tanító halmaz minden egyes alanya egy osztályt képvisel. A modell a tanítás során a bemeneti tanító halmazon megtanulja azokat a jellemzőket, amik alapján később osztályozni tud a számára ismeretlen képeken. A tanítás első szakaszának befejezése után a modellt tovább lehet finomítani különböző technikákkal annak érdekében, hogy a jellemzőket optimalizáljuk a konkrét alkalmazási területre. Például A CNN finomhangolásával, különböző veszteségfüggvények használatával.

Az az ötlet, hogy neurális hálózatokat használjunk arcfelismerésre, nem új keletű dolog. Egy korai módszert, amely valószínűségi döntésalapú neurális hálózaton (PBDNN) [25] alapult, 1997-ben javasolták arcfelismerésre. Az arcfelismerő PBDNN egy telje-

sen összekapcsolt alhálóra oszlott minden tanító objektum esetében, hogy csökkentse a rejtett egységek számát és elkerülje a túltanulást. A végső eredmény elérése érdekében két PBDNN-t tanítottak fel intenzitás és él jellemzők felhasználásával, és azok kimeneteit kombinálták. Egy másik korai módszer a Self-Organising Map (SOM) [25] és egy konvolúciós neurális hálózat kombinációját javasolta. A Self-Organising Map egy olyan neurális hálózat, amelyet felügyelet nélküli módon tanítanak fel, és amely a bemeneti adatokat egy alacsonyabb dimenziós térbe vetíti. Mindemellett megőrzi a bemeneti tér helyzeti tulajdonságait, azaz az eredeti térben a közeli bemenetek a kimeneti térben is közel lesznek egymáshoz.

A fent említett módszerek egyike sem ért el áttörést, főként a neurális hálózatok alacsony kapacitása és a tanításhoz rendelkezésre álló viszonylag kis adathalmazok miatt. Csak akkor terjedtek el a mélytanulási módszerek az arcfelismerés kapcsán, amikor ezeket a modelleket felskálázták és nagy mennyiségű adattal tanították. Az első ilyen meghatározó algoritmus a Facebook DeepFace volt. Az egyik első olyan CNN-alapú megközelítés volt az arcfelismerés területén, amely nagy kapacitású modellt használt fel. Sikeressége megnyilvánul abban, hogy 97,35%-os pontosságot ért el az LFW (Labeled faces in the wild) publikus adathalmazon, amelyet kifejezetten arcfelismerésre hoztak létre. A háló felépítése során softmax veszteségfüggvényt használtak a kimeneti rétegen, hogy a kimenet egy valószínűségi eloszlás legyen.

Ezen hálózatok sikerességéhez elengedhetetlen a jó minőségű és rendelkezésünkre álló nagy adathalmaz. Ezek közül talán a legelterjedtebbek [26]:

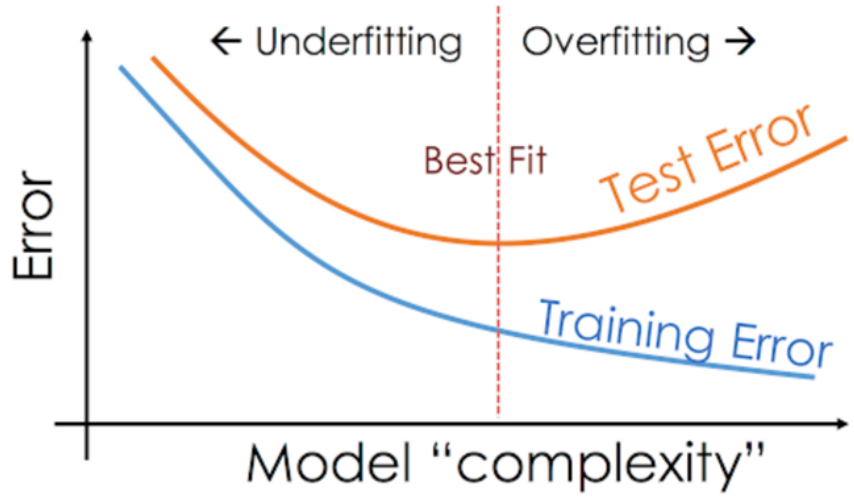
- Digi-Face 1M
- CelebFaces
- VGG Face2
- UMDFaces
- MS-Celeb-1M
- iQIYI-VID
- IMDB-Wiki

Ezeket az arcokat tartalmazó adathalmazokat a 2.3 táblázat mutatja be részletesen:

2.3. táblázat: Publikus arcokat tartalmazó adathalmazok összehasonlítása [26]

Adathalmaz	Képek száma	Identitások száma	Megjelenési év
Digi-Face 1M	1,2 millió	110 000	2022
CelebFaces	202 599	10 177	2015
VGG Face2	3,31 millió	9131	2018
UMDFaces	367 888	8277	2017
MS-Celeb-1M	10 millió	100 000	2016
iQIYI-VID	600 000	5000	2018
IMDB-Wiki	524 230	100 000	2015

A nagy tanító adathalmazokra azért van szükségünk, hogy a modell túltanulását el tudjuk kerülni. Ha egy hálózat túltanul (overfitting) (2.20 ábra), akkor nem képes helyes predikciót adni a még nem látott teszt adathalmazunk legtöbb elemére, mivel nagyon erősen rátanult a tanító adathalmaz jellemzőire, azaz más adatokra már nem olyan rugalmas. Szinte kiváló teljesítményt nyújt a tanító adatokon, de a tesztelési adatokat nem tudja megfelelően azonosítani. Általánosságba elmondható, hogy a CNN-ek pontosabbá válnak az osztályozás során, ha az egyes osztályokhoz tartozó minták száma nagy. Ennek az ellentéte az alultanulás (underfitting) (2.20 ábra). Ennek eredményeként a modell nem képes megfelelően elsajátítani a jellemzőket, nem tudja azokat jól általánosítani. Gyenge teljesítményt nyújt mind a tanító, mind a tesztelési adatokon.



2.20. ábra: Túltanulás és alultanulás szemléltetése [27]

Az arcfelismerés során érdemes minél nagyobb adathalmazzal dolgozni. Fontos kiemelni, hogy minél több arcképpel rendelkezik az egyes alanyokról az adatbázisunk, annál nagyobb pontosságot tudunk elérni a későbbi tesztelések során.

Bizonyos tanulmányokban azt vizsgálták, hogy a "szélesebb" vagy a "mélyebb" adathalmazok a jobbak a tanítás során. A "szélesebb" azt jelentette, hogy azon van a hangsúly, hogy minél több ember arcával tanítsák fel a hálózatot. Ezzel ellentétben a "mélyebbnél" pedig azon volt a hangsúly, hogy egy személyhez több arckép legyen társítva. Az derült ki a tanulmányból, hogy a "szélesebb" adathalmazok jobban teljesítettek. Ennek oka az lehet, hogy a "szélesebb" adathalmaz több különböző jellemző variációkat tartalmaznak, ezért azok jobban általánosíthatóak a még nem látott adatokra.

Az arcfelismerés technológiák jelentős átalakuláson mentek keresztül az elmúlt években. A korábbi tradicionális megoldásokat mára már felváltották a konvolúciós neurális hálózatokra (CNN-ekre) épülő mélytanulási technikák. Ezek a CNN-alapú rendszerek ipari szabvánnyá váltak, mivel lényegesen nagyobb pontosságot érnek el, mint a korábbi módszerek. Továbbá, a CNN-ekre épülő arcfelismerő rendszerek egyszerűen skálázhatók, így a pontosság tovább növelhető a tanító adathalmazok méretének növelésével vagy akár a különböző aktivációs függvények testreszabásával.

2.5 Keretrendszerek áttekintése a megvalósításhoz

2.5.1 Arcfelismeréshez használatos keretrendszerek

A TensorFlow [28] talán az egyik legelterjedtebb nyílt forráskódú szoftverkönyvtár, amelyet gépi tanulást és mesterséges intelligenciát használó alkalmazások fejlesztésére használnak. A Google Brain csapata fejlesztette ki 2015-ben. A TensorFlow célja, hogy megkönnyítse a gépi tanulási modellek létrehozását, feltanítását és telepítését különböző környezetekben, mint például szervereken, számítógépeken és mobil eszközökön.

A TensorFlow egyik fő előnye a rugalmas felépítése, amely lehetővé teszi a különböző szintű absztrakciók használatát. Az alacsonyabb szintű API-k segítségével részletes finomhangolást végezhetünk, míg a magasabb szintű API-k, mint a Keras, egyszerűbbé teszik a modellek gyors fejlesztését. A keretrendszer széles körű támogatást és integrációt biztosít, beleértve a különböző gépi tanulási és mélytanulási modelleket, mint például a neurális hálózatok, konvolúciós neurális hálózatok (CNN), és rekurzív neurális hálózatok (RNN). Emellett könnyedén integrálható más gépi tanulási könyvtárakkal és eszközökkel, mint például a scikit-learn, a NumPy és a Pandas.

A TensorFlow skálázhatósága kiemelkedő, mivel hatékonyan futtatható különböző hardvereken, beleértve a CPU-kat, GPU-kat és TPU-kat (Tensor Processing Unit), amik lehetővé teszik a modellek hatékony tanítását nagy adathalmazokon is.

A TensorFlow főbb komponensei közé tartozik a TensorFlow Core API, amely alacsony szintű eszközöket és funkciókat biztosít a gépi tanulási algoritmusok létrehozásá-

hoz és optimalizálásához. A magas szintű, felhasználóbarát Keras API egyszerűsíti a neurális hálózatok építését és tanítását, támogatva a gyors modell tervezést. A TensorBoard egy vizualizációs eszköz, amely segíti a gépi tanulási modellek fejlesztését és optimalizálását a tanulási folyamat során generált adatok grafikus megjelenítésével. A TensorFlow Extended pedig a gépi tanulási munkafolyamatok kezelésére szolgál, beleértve az adatelőkészítést, a modelltréninget, a telepítést és a monitorozást.

A TensorFlow gyakorlati alkalmazásai közé tartozik a képfelismerés és képosztályozás, amely lehetővé teszi különböző objektumok azonosítását és kategorizálását képekben, a természetes nyelv feldolgozás (NLP), amely szövegelemzésre, nyelvi modellek készítésére és fordításra használható, valamint az idősoros előrejelzés, amely különböző időbeli adatminták elemzésére és előrejelzésére alkalmas.

A TensorFlow előnyei közé tartozik a nagy közösségi támogatás és az aktív fejlesztés, a részletes dokumentáció és a számos online forrás. Azonban hátrányai közé sorolható a komplexitás, illetve a Windows operációs rendszer támogatásának a hiányossága. Összefoglalva, a TensorFlow egy rendkívül erős és sokoldalú eszköz a gépi tanulás és a mesterséges intelligencia területén, amely széles körű alkalmazási lehetőségeket kínál a kutatók és a fejlesztők számára.

A másik legnépszerűbb nyílt forráskódú gépi tanulási könyvtár a PyTorch [29], amelyet a Facebook AI Research fejlesztett ki, és 2016-ban adtak ki. Gyorsan népszerűvé vált a kutatók és fejlesztők körében, mivel rugalmas és könnyen használható keretrendszert kínál a mélytanulási modellek fejlesztéséhez és telepítéséhez. A PyTorch célja, hogy a kutatók és mérnökök számára hatékony eszközt biztosítson a gyors modellek készítéséhez és a modellek feltanításához.

A PyTorch egyik legnagyobb előnye a flexibilitás. A PyTorch megkönnyíti a más Python könyvtárakkal való integrációt, mint például a NumPy, a SciPy és a Pandas. Mindemellett a TensorFlow-hoz képest sokkal egyszerűbb a kezdők számára. Számos oktatóanyag, dokumentáció és online forrás áll rendelkezésre, amelyek segítik a felhasználók munkáját.

A PyTorch alapvető adatstruktúrája a tensor, amely hasonló a NumPy tömbjeihez, de GPU-n is futtatható. Ez lehetővé teszi a nagy számítási igényű műveletek gyors végrehajtását. A torch.nn modul előre definiált rétegeket és funkciókat biztosít a neurális hálózatok építéséhez, egyszerűsítve a hálózatok létrehozását és tanítását. A torch.optim modul különböző optimalizálási algoritmusokat kínál, mint például a Stochastic Gradient Descent (SGD), Adam és RMSprop, amelyek segítik a modellek hatékony tanítását. Az adatkezelési eszközök, mint a DataLoader és Dataset osztályok, megkönnyítik a nagy adathalmazokkal való munkát.

A PyTorch gyakorlati alkalmazásai közé tartozik a képfelismerés és képosztályozás, ahol kiválóan alkalmazható képfeldolgozási feladatokra, mint például objektumfelismerés és osztályozás konvolúciós neurális hálózatok (CNN) használatával. Emellett ideális szövegelemzési feladatokra is, mint például nyelvi modellezés, szövegosztályozás és gépi fordítás különböző rekurzív neurális hálózatok (RNN) és transzformátorok segítségével. Alkalmas továbbá időbeli adatok elemzésére és előrejelzésére, például pénzügyi előrejelzések és érzékelő adatok elemzése esetén.

Hátrányai között említhető, hogy kevesebb ipari felhasználással bír, mint a TensorFlow, bár ez gyorsan változik. Illetve nagy méretű modellek esetén memória és teljesítményproblémák merülhetnek fel.

Összefoglalva, a PyTorch egy erős és rugalmas gépi tanulási könyvtár, amely ideális választás a kutatók és fejlesztők számára a gyors prototípus-készítéshez és a mélytanulási modellek fejlesztéséhez. A Python-barát szintaxis különösen vonzóvá teszi azok számára, akik könnyen kezelhető és hatékony eszközt keresnek a gépi tanulás projektükhöz.

2.5.2 Webalkalmazáshoz használatos keretrendszer

A webalkalmazások fejlesztéséhez számos keretrendszer áll rendelkezésre, amelyek megkönnyítik és felgyorsítják a fejlesztési folyamatot.

A Django [30] egy magas szintű Python webkeretrendszer, amelyet a gyors fejlesztés és a tiszta, pragmatikus tervezés szem előtt tartásával fejlesztettek ki. Első kiadása 2005-ben jelent meg, és azóta az egyik legnépszerűbb keretrendszerré vált a webfejlesztés világában.

A Django jól skálázható, így képes nagy forgalmú weboldalak kiszolgálására is. Számos nagy weboldal és szolgáltatás használja a Django-t, például a Pinterest és az Instagram, bizonyítva ezzel a keretrendszer teljesítményét és megbízhatóságát.

A Django MVT (Model-View-Template) architektúrát követ, amely hasonló az MVC (Model-View-Controller) mintához. Az MVT komponensei közé tartozik a Model, amely az adatbázis struktúráját és az adatok kezelését definiálja, a View, amely az üzleti logikát és a felhasználói kérések kezelését definiálja, és a Template, amely a felhasználói felületet definiálja és dinamikus HTML oldalakat generál.

A Django fejlesztési folyamata általában a projekt létrehozásával és az alkalmazások létrehozásával kezdődik, majd a modellek definiálásával, migrációk készítésével, nézetek és URL-ek készítésével, sablonok létrehozásával, végül pedig az adminisztrációs felület testreszabásával folytatódik.

A Django Object-Relational Mapping (ORM) rendszere lehetővé teszi, hogy a fejlesz-

tők Python kód segítségével kezeljék az adatbázisokat. Az ORM támogatja a különböző adatbázisokat, mint például a PostgreSQL, MySQL, SQLite és Oracle, és lehetővé teszi a komplex lekérdezések és műveletek egyszerű végrehajtását.

A Django rugalmassága, gazdag funkcionalitása és erős közösségi támogatása miatt ideális választás a modern webalkalmazások fejlesztéséhez. Akár egyszerű blogot, akár komplex vállalati alkalmazást szeretnél fejleszteni, a Django biztosítja a szükséges eszközöket és keretrendszert a sikeres projektek megvalósításához.

2.5.3 Adatbázishoz használatos keretrendszerek

Az arcfelismerés során kulcsfontosságú a nagy mennyiségű arckép hatékony tárolása és kezelése. A PostgreSQL és a MongoDB keretrendszerek rendelkeznek olyan tulajdonságokkal és képességekkel, amelyek ideálissá teszik őket az arcfelismerési projektekben való felhasználásra.

A PostgreSQL [31] egy nyílt forráskódú, nagy teljesítményű relációs adatbázis-kezelő rendszer, amely híres a megbízhatóságáról, skálázhatóságáról és fejlett adatbázis-funkcióiról. Az arcfelismerési projektekben a PostgreSQL több szempontból is előnyös választás lehet.

Először is, a PostgreSQL támogatja a komplex adatstruktúrák és relációk kezelését, ami lehetővé teszi az arcképekhez kapcsolódó metaadatok hatékony tárolását. Például, egy arcképhez társíthatunk olyan információkat, mint a felhasználó azonosítója, az arckép készítésének dátuma és helyszíne, valamint az arcfelismerési algoritmus által generált jellemzők.

Másodszor, a PostgreSQL fejlett keresési és indexelési képességei lehetővé teszik az arcképek gyors és hatékony lekérdezését. Az adatbázis támogatja a különböző típusú indexeket, mint például a B-tree, GIN és GiST indexeket, amelyek segítségével optimalizálhatók a komplex keresési műveletek, mint például a hasonlósági keresés az arcfelismerés során.

Továbbá, a PostgreSQL támogatja a JSON és JSONB adattípusokat, amelyek segítségével strukturálatlan adatokat is tárolhatunk és kezelhetünk. Ez különösen hasznos lehet az arcfelismerési projektekben, ahol az arcképekhez kapcsolódó jellemzők változó formátumúak lehetnek.

A PostgreSQL kiválóan skálázható és támogatja a nagy teljesítményű adatbázis-kezelést. Nagy mennyiségű adat esetén is megbízhatóan teljesít, így ideális választás lehet nagy méretű arcfelismerési rendszerek számára.

A MongoDB [32] egy nyílt forráskódú, dokumentumorientált NoSQL adatbázis-kezelő

rendszer, amely rugalmas adatmodellje és horizontális skálázhatósága révén kiváló választás az arcfelismerési projektekhez. A MongoDB különösen alkalmas nagy mennyiségű és változatos szerkezetű adatok kezelésére, ami az arcfelismerési alkalmazásokban gyakori.

A MongoDB dokumentumorientált adatmodellje lehetővé teszi, hogy az arcképeket és az azokhoz kapcsolódó metaadatokat egyetlen dokumentumban tároljuk. Ez a megközelítés egyszerűsíti az adatok kezelését és lehetővé teszi a rugalmas adatstruktúrák használatát. Például egy arcképdokumentum tartalmazhatja az arcképet, az azonosítókat, a jellemzővektort és bármilyen egyéb releváns információt egyetlen JSON-szerű struktúrában.

A MongoDB skálázhatósága szintén nagy előnyt jelent. Könnyedén növelhetjük az adatbázis kapacitását és teljesítményét. Ez különösen fontos nagy méretű arcfelismerési rendszerek esetén, ahol hatalmas mennyiségű adatot kell kezelni és gyorsan lekérdezni.

Emellett a MongoDB támogatja a térinformatikai adatkezelést, amely lehetővé teszi az arcképekhez kapcsolódó helyszíni adatok hatékony kezelését. Ez különösen hasznos lehet olyan arcfelismerési alkalmazásokban, ahol a helyszíni információk is relevánsak, például biztonsági rendszerekben vagy közlekedési megfigyelő rendszerekben.

A MongoDB továbbá támogatja a beágyazott dokumentumokat és tömböket, ami lehetővé teszi a komplex adatszerkezetek tárolását és lekérdezését. Ez a rugalmasság lehetővé teszi az arcfelismerési algoritmusok által generált összetett adatstruktúrák hatékony kezelését.

2.6 Irodalumkutatás összegzése

Az arcfelismerésen alapuló beléptető rendszerek egyre növekvő népszerűségnek örvendenek a biztonsági technológiák területén. Az irodalomkutatásomban áttekintettem a hasonló rendszerek fejlődését és alkalmazásait, kezdve a kezdetleges arcdetektáló algoritmusoktól egészen a modern neurális hálózatokig.

Az arcdetektálás korai algoritmusai, mint például a Viola-Jones detektor, alapvető szerepet játszottak az arcfelismerés fejlődésében. Ezek az algoritmusok főként egyszerűbb, de hatékony módszerekkel dolgoztak, amelyek lehetővé tették az arcok azonosítását valós időben. Azonban a korlátaik miatt csak alapvető szintű pontosságot értek el.

A neurális hálózatok megjelenése forradalmasította az arcfelismerés területét. A mély tanulási technikák, különösen a konvolúciós neurális hálózatok (CNN-ek), lehetővé tették az arcfelismerés pontosságának és hatékonyságának drámai növelését. A modern rendszerek képesek a legapróbb arcvonások felismerésére is, ami jelentősen javítja a rendszer

megbízhatóságát és biztonságát.

Az irodalomkutatás során számos keretrendszert és eszközt is áttekintettem, amelyek segítik a neurális hálózatok alapú arcfelismerő rendszerek fejlesztését. Ezek közé tartoznak olyan nyílt forráskódú könyvtárak, mint a TensorFlow és a PyTorch, amelyek egyszerűsítik a mély tanulási modellek fejlesztését és integrálását. Illetve különböző adatbázis keretrendszereket és webalkalmazás keretrendszereket tekintettem meg, amelyek elengedhetetlenek egy sikeres webalkalmazás fejlesztéséhez.

Összességében az arcfelismerésen alapuló beléptető rendszerek hatékonysága és megbízhatósága jelentősen nőtt a neurális hálózatok alkalmazásával. A kutatásom célja az volt, hogy bemutassam e technológia fejlődését és a modern fejlesztési eszközök használatát, amelyek lehetővé teszik a hatékony és biztonságos beléptető rendszerek létrehozását.

3 KÖVETELMÉNY SPECIFIKÁCIÓ

3.1 Általános követelmények

3.1.1 Rendszer áttekintése

A rendszer célja egy arcfelismerésen alapuló beléptető rendszer létrehozása, amely képes valós időben azonosítani a felhasználókat és engedélyezni a belépést a meghatározott felületre.

3.1.2 Rendszer környezete

A rendszer webalapú alkalmazásként működik, amelyhez a felhasználók böngészőn keresztül férhetnek hozzá. Az webalkalmazás alapját a Django keretrendszer fogja szolgáltatni HTML5, CSS3 és JavaScript felhasználásával, míg a mögöttes logikát megvalósító mesterséges intelligencia modell Tensorflow segítségével fog létrejönni.

3.2 Funkcionális követelmények

3.2.1 Felhasználói hitelesítés

- A rendszernek képesnek kell lennie az arcok felismerésére és azonosítására.
- A rendszernek lehetővé kell tennie a felhasználók regisztrációját, ahol ahol a saját arcképeiket feltölthetik és elmenthetik a későbbi beléptetéshez.
- Sikeres azonosítás esetén a rendszernek engedélyeznie kell a belépést.

3.2.2 Adatbázis kezelés

- A rendszernek biztonságos adatbázist kell használni az adatok tárolására, mert érzékeny adatok kerülnek benne eltárolásra.
- Az adatbázisnak képesnek kell lennie nagy mennyiségű adat tárolására és gyors elérésére.

3.2.3 Értesítések és naplózás

- A rendszernek naplóznia kell minden belépési kísérletet, beleértve a sikeres és sikertelen azonosításokat is.
- A rendszernek értesítéseket kell küldenie a rendszergazdáknak a szokatlan aktivitá-

sokról vagy az esetleges biztonsági incidensekről.

3.3 Nem funkcionális követelmények

3.3.1 Biztonság

- A rendszernek biztosítania kell az adatbiztonságot.
- A rendszernek meg kell felelnie a GDPR és más releváns adatvédelmi előírásoknak.

3.3.2 Felhasználói élmény

- A rendszer felhasználói felületének könnyen használhatónak kell lennie.

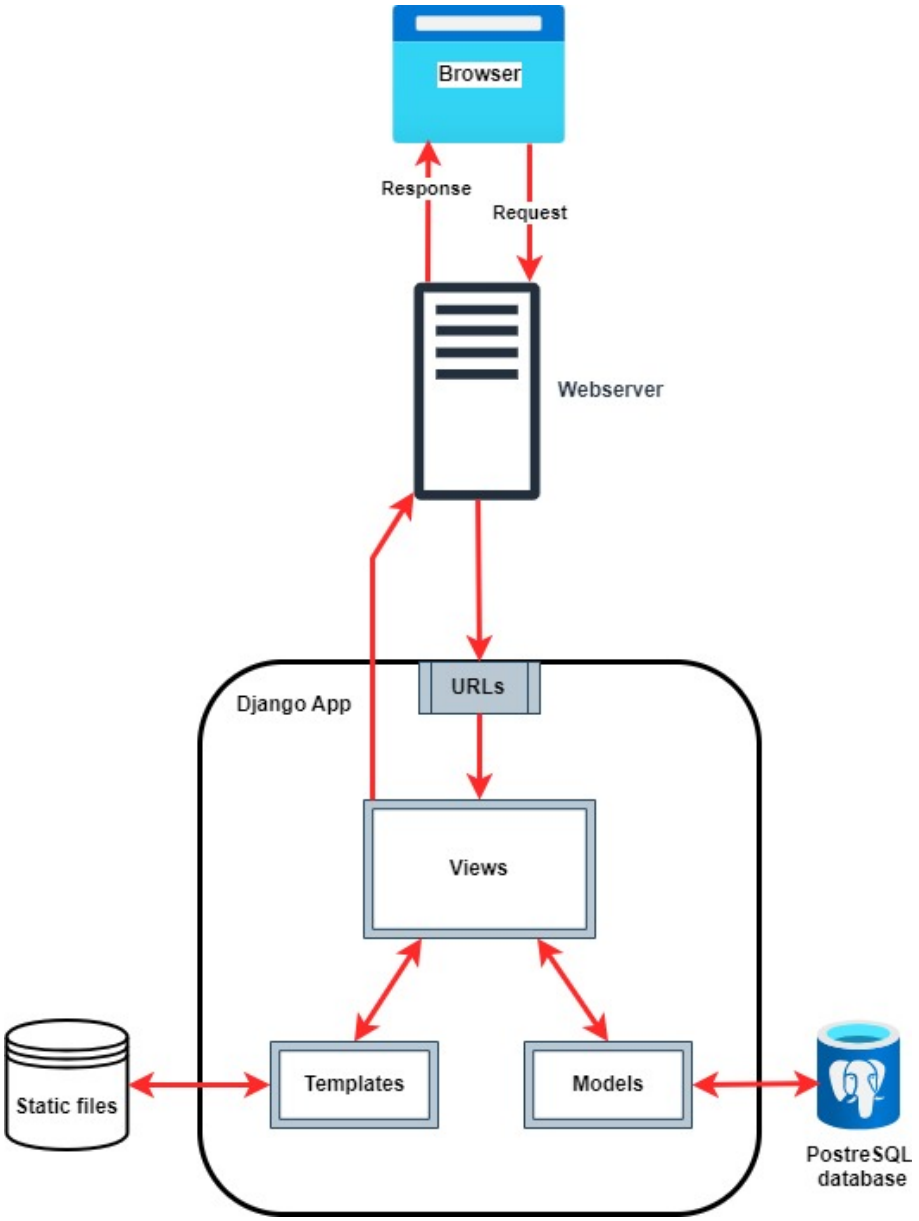
3.3.3 Karbantarthatóság és bővíthetőség

- A rendszernek átláthatónak kell lennie, hogy könnyen karbantartható és bővíthető legyen.

4 TERVEZÉS

4.1 Rendszerterv

A rendszer több rétegből áll (4.1 ábra), hogy biztosítsa a skálázhatóságot, megbízhatóságot és biztonságot.



4.1. ábra: Rendszerterv

Felhasználói felület (Frontend): A felhasználók böngészőből érhetik el az alkalmazást, ahol regisztrálhatnak a felhasználónevükkel, jelszavukkal és képeikkel. A rendszerbe

kétféleképpen lehet belépni, az első módszer a megszokott felhasználónév-jelszó páros, a második pedig az arcfelismerésen alapuló belépés. A felület HTML5, CSS3 és JavaScript segítségével épül fel.

Kiszolgáló réteg (Backend): A backend rész Python és Django keretrendszer által készül el. A Django kezeli a felhasználói kéréseket, feldolgozza az adatokat, és biztosítja a kommunikációt az adatbázissal és az arcfelismerő modellel. Mindemellett a Django beépített admin felületet is biztosít a rendszergazdák számára.

Arcfelismerő modell: Neurális hálózat, a TensorFlow és Keras könyvtárak segítségével történik az arcok elemzése és azonosítása. Kulcsfontosságú a rendszert tekintve, egy megbízható modell létrehozása. Ehhez a feladathoz Konvolúciós Neurális Hálózatokat (CNN) fogok használni, mivel ezek különösen hatékonyak képfeldolgozási és mintafelismerési feladatokban. A tanítás és a finomhangolás végett előre tanított modelleket is kipróbálok, például VGG16-t, ResNet-t vagy InceptionV3-t.

Adatbázis: Az regisztráció során megadott adatokat és a naplózott eseményeket PostgreSQL adatbázisban tárolom majd. A PostgreSQL adatbázis biztonságos és gyors hozzáférést biztosít a szükséges adatokhoz.

4.2 Frontend

A frontend az alkalmazás azon része, amely biztosítja az interaktív felületeket, beleértve a regisztrációs, belépési, dokumentumkezelő oldalakat. Ez a réteg felelős az adatok vizuális megjelenítéséért, valamint a felhasználói interakciók kezeléséért.

4.2.1 Technológiák használata

- HTML és CSS: A vizuális elemek, például a regisztrációs, bejelentkezési oldal, a dokumentumkezelési felület és a profilképekhez tartozó oldalak struktúrájának és stílusának megvalósítására szolgál.
- JavaScript: A dinamikus elemek, például az űrlapok validációja, interaktív gombok kezelésére és az arcfelismerésen alapuló bejelentkezésnél a kamera előhívására szolgál.

4.2.2 Funkciók

- Regisztráció és bejelentkezés: A felhasználók, felhasználónevük és jelszavuk, illetve arcképeik megadásával regisztrálhatnak. Az arckép feltöltése egyszerű, "drag-and-drop" vagy fájlkiválasztási lehetőséggel működik. Bejelentkezéskor hagyomá-

nyos felhasználónév-jelszó páros használatával vagy arcfelismerésen alapuló hitelesítéssel léphetnek be.
- Főoldal, dokumentumok megjelenítése: A felhasználók feltölthetik, megtekinthetik és törölhetik dokumentumaikat. A felület egy egyszerű táblázatos megjelenítést használ, amely listázza a fájlok nevét, feltöltési dátumát, azt, hogy ki töltötte fel. Ezen felül, ha a dokumentumot feltöltő felhasználó munkamenetében vagyunk, akkor látható egy törlés gomb is a dokumentum neve mellett.
- Profilkép módosítása: A felhasználók egy külön végponton tölthetik fel vagy törölhetik profilképüket, amely valós időben le is frissül.

4.2.3 Felhasználói élmény

- Valós idejű hibakezelés: Az űrlapok kitöltése közben a rendszer azonnal visszajelzést ad a hibás mezőkről (pl. helytelen jelszó megadása).

4.3 Backend

A backend az alkalmazás legfontosabb eleme, amely biztosítja a logikát, az adatok feldolgozását és tárolását, valamint a rendszer biztonságát is ez szolgáltatja. A program Django keretrendszerre épül, amely gyors, átlátható fejlesztést, könnyű adatkezelést és erős biztonsági funkciókat kínál.

4.3.1 Technológiák használata

- Beépített ORM: Az objektum-relációs leképezés megkönnyíti az adatbázis-kezelést, és biztosítja a SQL-injekció elleni védelmet is.

4.3.2 Megvalósítandó modulok

- Hitelesítés modul: Ez a modul kezeli a felhasználók azonosítását. A jelszavas bejelentkezéshez a rendszer a jelszavakat hash-elve tárolja el az adatbázisban. Az arcfelismerésen alapuló bejelentkezés során a backend a pillanatnyi arcképet továbbítja a feltanított modellnek. A modell elkészít nekünk egy embeddinget, ami igazából egy vektor, ezt az embeddinget pedig összehasonlítjuk a már eltárolt arcjellemzőkkel.
- Dokumentumkezelési modul: A dokumentumok feltöltését, törlését és megtekintését kezeli. A jogosultságokat szigorúan ellenőrzi, hogy csak a fájl tulajdonosa végezhesse el a törlést.

– Naplózás modul: Minden felhasználói tevékenységet rögzít, beleértve a regisztrációkat, bejelentkezéseket és dokumentumkezelési műveleteket. Az események az esemény létrehozásának idejével és a felhasználó azonosítójával kerülnek tárolásra az adatbázisban.

4.3.3 Biztonság

– SQL-injekció elleni védelem: A Django ORM használata minimalizálja a SQL-injekció kockázatát.
– Adattitkosítás: Az érzékeny adatokat, jelen esetben például a jelszavakat, hash-elve tárolja a rendszer.

4.4 Példa a backend és a frontend közötti kapcsolatra

Az alábbi példák bemutatják a backend és a frontend közötti kapcsolat főbb lépéseit:

1. példa (Bejelentkezés arcfelismeréssel): A felhasználó beküldi az arcképét a frontend felületen. Ezt követően a frontend továbbítja a képet a backendhez egy API-híváson keresztül. A backend feldolgozza a képet a neurális háló segítségével, és kiszámítja az embedding vektort. Az embedding vektort összehasonlítja az adatbázisban tárolt többi, már ismert embeddinggel. Ha a hasonlóság meghaladja a küszöbértéket, a rendszer sikeres bejelentkezést állapít meg. Az eseményt naplózza a rendszer, azaz létrehozza a hozzá tartozó rekordokat az adatbázisban.

2. példa (Regisztráció): A felhasználó regisztrál a webalkalmazáson keresztül, feltölti az arcképeit, illetve beírja a regisztrációhoz szükséges adatokat, mint például a felhasználónevét és jelszavát. Az adatait a frontend továbbítja a backendhez. A backend mindemellett feldolgozza a képet, majd az arcfelismerő modell kinyeri az arc jellemzőit és eltárolja az adatbázisban a felhasználó többi regisztrációs adatával együtt.

3. példa (Profil kezelés): Miután a felhasználó bejelentkezett megjelenik a főoldal, ahonnan át lehet kattintani egy gomb segítségével a profilunkra, ahol a képeinket tudjuk törölni, illetve új képet is fel tudunk tölteni. Például a törlésnél, ha az adott képet kiválasztjuk a frontend felületen, amit törölni szeretnénk, akkor a frontend elküldi a backendnek az adott képhez tartozó kép elérési útját, majd ezt követően a backend törli az adatbázisból a képet, mely azonnal eltűnik a frontend felületről is.

4. példa (Dokumentum kezelés): Miután a felhasználó bejelentkezett megjelenik a főoldal, amit mindenki elér a bejelentkezett felhasználók közül. Itt jelennek meg az emberek által feltöltött dokumentumok lista formájában. Az adott dokumentumot csak az a felhasználó tudja törölni, aki feltöltötte azt. Egy törlés kérésnél az adatbázisból kinyeri a

backend, hogy az adott dokumentumhoz melyik felhasználó tartozik, ha ez megegyezik akkor a dokumentum törlésre kerül, ami automatikusan frissíti a frontendet és eltűnik a dokumentum a felületről és az adatbázisból egyaránt.

4.5 Adatbázis-terv

Az adatbázis tervezése során olyan szerkezetet alakítottam ki, amely hatékonyan támogatja a rendszer működését, miközben biztosítja az adatok biztonságát és könnyű elérhetőségét. Az adatbázisban négy fő táblát definiáltam (4.2 ábra):

Felhasználók (Users): Ez a tábla tárolja a rendszerbe beregisztrált felhasználók adatait, mint például a neveket, a titkosított jelszavakat és a profilképek elérési útjait. Az id mező egyedi azonosítóként szolgál, amely más táblákban is hivatkozható.

Dokumentumok (Documents): Ez a tábla a feltöltött dokumentumokat kezeli, beleértve azok metaadatait, például a fájl nevét, a feltöltés dátumát és a feltöltő felhasználót. Az user_id mező a Users tábla id mezőjéhez kapcsolódik, így biztosítva a fájlok tulajdonjogának nyomon követését.

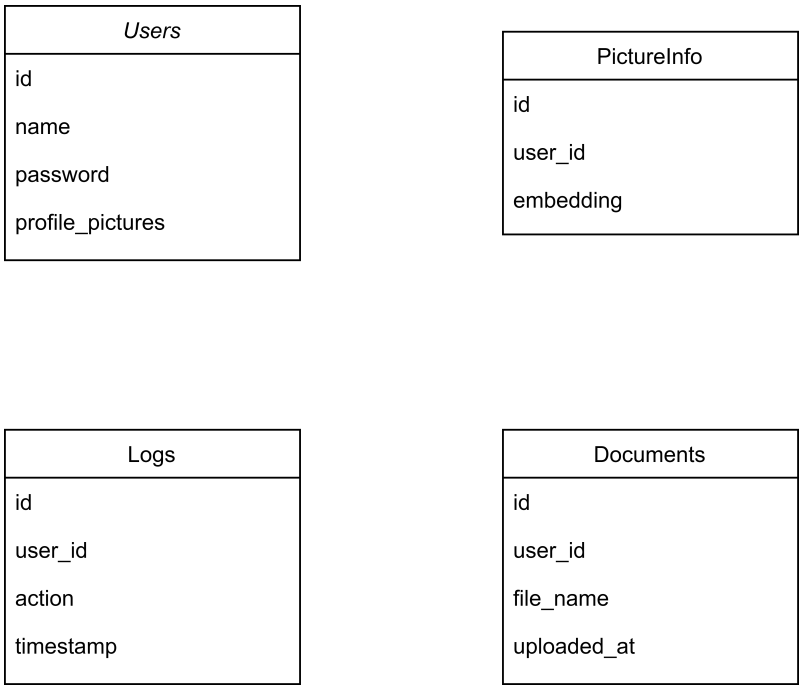
Naplózás (Logs): A rendszer minden eseményt naplóz, beleértve a regisztrációkat, bejelentkezéseket, dokumentum feltöltéseket és törléseket. A tartalmazza az esemény létrejöttének az idejét, típusát és az eseményt létrehozó felhasználót.

Arckép Információk (PictureInfo): Ez a tábla a feltöltött képeket kezeli. A tábla eltárolja a felhasználókhoz tartozó képek adatai, azaz az embeddingeket, melyeket majd a hálózat fog létrehozni regisztrációkor minden egyes képre. Az user_id mező itt is a Users tábla id mezőjéhez kapcsolódik, így biztosítva, hogy a képek a megfelelő felhasználóhoz legyenek rendelve.

Az adatbázis táblák között lévő kapcsolatot a 4.4 táblázat mutatja be:

4.4. táblázat: Táblák kapcsolatai és attribútumai

Tábla	**Kapcsolat**	**Attribútumok**
Users	-	id, name, password, profile-pictures
Documents	Users (user-id)	id, user-id, file-name, uploaded-at
Logs	Users (user-id)	id, user-id, action, timestamp
PictureInfo	Users (user-id)	id, user-id, embedding



4.2. ábra: Táblák

4.6 További tervek

A későbbiekben az alap program létrehozása után, további kiegészítő tervek megvalósítására kerül sor.

Rendszer Optimalizálás: Az arcfelismerési sebesség és pontosság növelése, valamint a rendszer teljesítményének javítása.

Felhasználói Felület Fejlesztése: A frontend felület további finomítása, hogy még felhasználóbarátabb legyen.

4.7 API végpontok

4.7.1 Felhasználókezelés

Felhasználó regisztrációja (/api/register): A regisztrációs végpont lehetővé teszi az új felhasználók létrehozását a rendszerben. A regisztrációhoz meg kell adni a felhasználónevet, a jelszót és a profilhoz tartozó arcképeket. Sikeres regisztráció esetén a rendszer 201 Created választ ad vissza, míg hiba esetén 400 Bad Request üzenetet küld.

Bejelentkezés (/api/login): A bejelentkezési végpont kétféle autentikációt támogat: hagyományos felhasználónév-jelszó párossal, vagy arckép-alapú azonosítással lehet belépni

a rendszerbe. A sikeres bejelentkezés eredménye 200 OK, míg sikertelen hitelesítés esetén 401 Unauthorized válasz érkezik.

Kijelentkezés (/api/logout): Ez a végpont lezárja a felhasználói munkamenetet, a token érvénytelenítésével, azaz kilépteti a felhasználót. Ehhez szükséges a felhasználói token. Sikeres kijelentkezés esetén 200 OK üzenetet kapunk.

4.7.2 Dokumentumkezelés

Főoldal és dokumentumok listázása (/api/home): Bejelentkezett felhasználók számára elérhető végpont, amely visszaadja a feltöltött dokumentumok listáját. A kérés során érvényes token azonosítóra van szükség. Siker esetén 200 OK válasz érkezik, hiba esetén pedig 401 Unauthorized.

Új dokumentum feltöltése (/api/documents/upload): Ezen végpont segítségével új fájlokat lehet feltölteni a rendszerbe. A post kérés részeként meg kell adni a feltöltendő fájlt, mindehhez érvényes token azonosító kell. Sikeres feltöltés esetén a rendszer 201 Created választ ad, míg hibás fájl vagy rossz paraméterek esetén 400 Bad Request üzenetet küld.

Dokumentum törlése (/api/documents/<doc_id>): A megadott azonosítójú dokumentum törlésére szolgál. A delete kérés részeként meg kell adni a törölni kívánt file azonosítóját, mindehhez szintúgy érvényes token azonosító kell. Sikeres törlés esetén 200 OK, míg jogosultság hiányában 403 Forbidden válasz érkezik.

4.7.3 Profilkép kezelése

Profilkép feltöltése (/api/profile): A felhasználók ezen a végponton keresztül tölthetik fel új arcképeiket. A post kérés részeként meg kell adni a feltöltendő fájlt, amihez érvényes token azonosító is fog kelleni. Sikeres feltöltés esetén 201 Created, hiba esetén pedig 400 Bad Request választ kapunk.

Profilkép törlése (/api/profile/delete): A profilkép eltávolítását teszi lehetővé. A kérés során csak az érvényes token azonosító szükséges. A sikeres törlést a rendszer 200 OK válasszal igazolja, jogosultság hiányában pedig 403 Forbidden válasz érkezik.

4.7.4 Összegző táblázat

A 4.5 táblázat bemutatja a rendszer API végpontjait, azok célját, a HTTP-metódusokat, valamint a szükséges paramétereket és a várható válaszokat.

4.5. táblázat: API végpontok, kapcsolataik és működésük

Végpont	**Leírás**	**Metódus**	**Paraméterek**	**Válasz**
/api/register	Felhasználó regisztrációja	POST	username, password, pictures	201 Created vagy 400 Bad Request
/api/login	Bejelentkezés	POST	username, password vagy arckép alapján	200 OK vagy 401 Unauthorized
/api/logout	Felhasználói munkamenet lezárása	POST	Token azonosító az engedélyezett felhasználótól	200 OK
/api/home	Feltöltött dokumentumok listázása, főoldal a belépés után	GET	Token azonosító	200 OK (JSON fájlok listája) vagy 401 Unauthorized
/api/documents/upload	Új dokumentum feltöltése	POST	file (feltöltött fájl)	201 Created vagy 400 Bad Request
/api/documents/-<doc_id>	Dokumentum törlése	DELETE	Dokumentum azonosító (URL útvonalon)	200 OK vagy 403 Forbidden
/api/profile-picture	Profilkép feltöltése	POST	image (feltöltött kép)	201 Created vagy 400 Bad Request
/api/profile-picture/delete	Profilkép törlése	DELETE	Token azonosító	200 OK vagy 403 Forbidden

5 FEJLESZTÉS

5.1 Fejlesztési kihívások és döntéshozatal

A webalkalmazás fejlesztése során számos technikai kihívással szembesültem, amelyek alapos tervezést, döntéshozatalt és problémamegoldást igényeltek. A következőkben részletesen bemutatom a fejlesztési folyamat főbb szakaszait, a felmerült problémákat és az azokra adott megoldásokat.

5.1.1 Arcfelismerő modell létrehozása

A projekt központi eleme az arcfelismerésre támaszkodó hitelesítés. Kezdetben az előre feltanított modellek kipróbálásával indultam neki a feladatnak. Ki akartam próbálni őket, milyen megbízhatóak egy arcfelismerési folyamat során, hogy legyen viszonyítási alapom. Az általam kipróbált modellek magas szintű pontosságot biztosítottak, mindemellett könnyű volt őket feltanítani. Viszont a végső célom egy saját hálózat megalkotása volt. Egy egyedi beállítású, a regisztrált személyeket jól megkülönböztető modellt szerettem volna létrehozni, ami hatékonyan alkalmazható és integrálható egy Django-alapú webalkalmazásban.

A következő lépésben egy egyszerű CNN (Convolutional Neural Network) modellt próbáltam ki. A CNN-ek széles körben használtak különböző képfelismerési feladatokban, ezért logikusnak tűnt, hogy egy egyszerűbb architektúrával induljak neki a feladatnak, mielőtt bonyolultabb megoldásokat alkalmaznék. A CNN-ek tanítása viszonylag gyorsan sikerültek, és a kezdeti tesztek azt mutatták, hogy a modell képes volt felismerni és osztályozni az arcokat. Azonban, ahogy a rendszer felhasználói bázisának növekedését szimuláltam, egyre világosabbá vált, hogy az egyszerű CNN nem képes megfelelő pontosságot nyújtani. Minél több volt a regisztrált felhasználó, annál nehezebben küzdött meg az emberek azonosításával. Mindezt adat-augmentáció segítségével se tudtam kiküszöbölni. Ezért döntöttem úgy, hogy egy olyan architektúrát választok, amely kimondottan az arcok közti különbségek felismerésére képes.

Ez a probléma késztetett arra, hogy egy teljesen új megközelítést válasszak. Ekkor találtam rá a triplet loss alapú siamese modell koncepciójára, amely kifejezetten az olyan feladatokhoz készült, ahol a képpárok hasonlóságának vizsgálatán túl figyelembe kell venni az osztályok közötti különbségeket is. Ez a modell három bemenetet vár: egy anchor képet, amely a referenciaként szolgál, egy pozitív képet, amely ugyanahhoz az emberhez tartozik, mint az anchor kép, és egy negatív képet, amely egy teljesen eltérő személyhez tartozik. A hálózat célja, hogy a pozitív kép és az anchor kép közötti távolságot minimalizálja, miközben maximalizálja az anchor és a negatív kép közötti távolságot.

A modell bevezetése nem volt zökkenőmentes. Az egyik legnagyobb kihívást a megfelelő adathalmaz előkészítése jelentette. Mivel a modell nem egyesével vagy páronként, hanem hármasával dolgozza fel a képeket, ezért ez azt jelentette, hogy a tanító adathalmazomat jelentősen át kellett alakítanom, hogy tripleteket generálhassak belőlük, azaz egy anchor képhez pozitív és negatív képpárt társítsak. Ez a folyamat időigényes volt, hiszen ügyelni kellett arra, hogy megfelelő mennyiségű tripletet tudjak előállítani, de ez ne menjen a modell tanítási idejének a rovására.

A modellek létrehozásához a Keras és TensorFlow könyvtárak mellett döntöttem, mivel ezek széleskörű modell konfigurációt enged meg.

A Siamese modell architektúrája (5.1 ábra):

A modell összesen három bemenetet fogad:

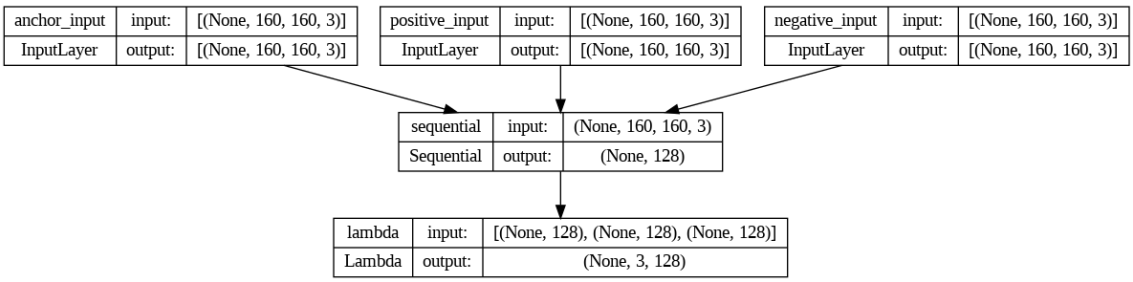
- Anchor kép: Ez a referenciaként szolgáló kép.
- Pozitív kép: Egy olyan kép, amely ugyanahhoz a személyhez tartozik, mint az anchor.
- Negatív kép: Egy olyan kép, amely egy másik személyhez tartozik.

Ezek alapján a bemenetek három különálló InputLayer segítségével kerülnek a modellbe.

Ezeket a bemeneteket egy közös Sequential modell dolgozza fel, amely a képeket embeddingekké alakítja. A Sequential modell architektúrája több rétegből áll, beleértve a konvolúciós rétegeket, a pooling rétegeket és egy Dense réteget, amely a végső 128-dimenziós embeddinget generálja.

A kimeneti embeddingek a következőképpen alakulnak:

Az anchor, a pozitív és a negatív bemenetek embeddingjei egy közös Lambda rétegen keresztül kerülnek összevonásra. Ez egy (None, 3, 128) alakú kimenetet eredményez, amelyben a három kép embeddingje egyetlen tensorban vannak eltárolva.



5.1. ábra: Siamese model

A Sequential modell architektúrája (5.2 ábra):

A Sequential modell belső struktúráját az alábbi rétegek alkotják:

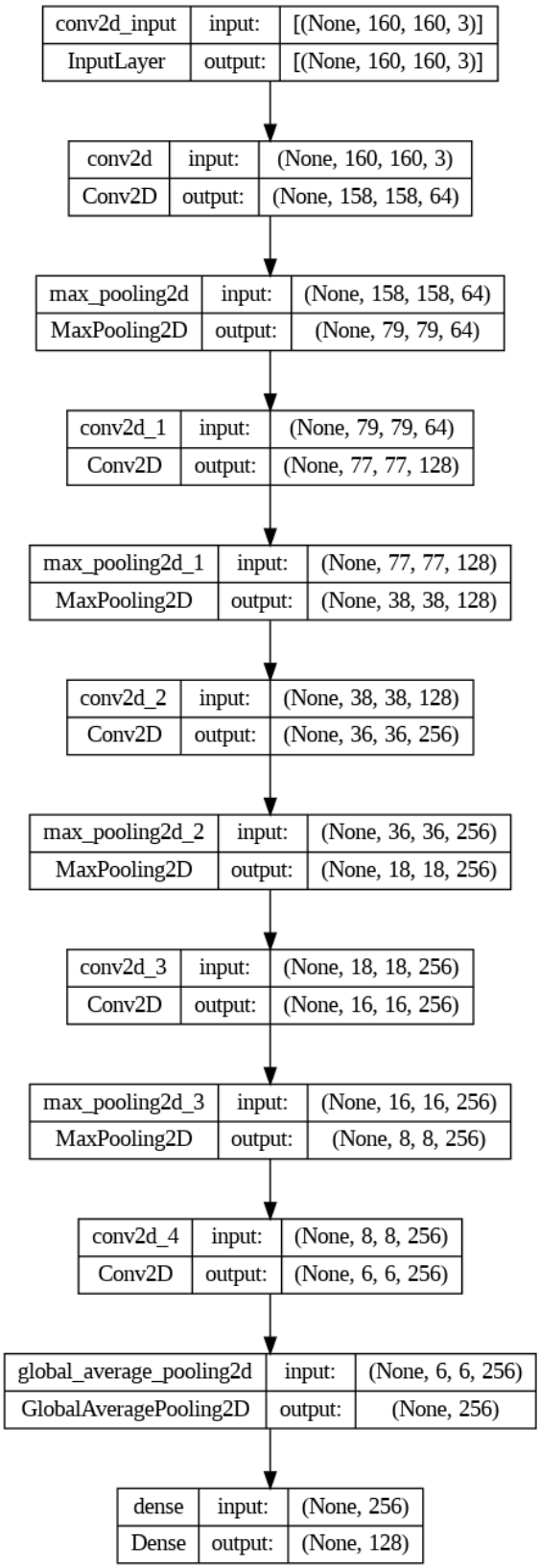
- Konvolúciós rétegek: Az első réteg 64 szűrőt használ, amely 3x3-as kernelmérettel dolgozik. Ez a réteg az alapvető vizuális mintázatokat, például éleket és textúrákat tanul meg. Az ezt követő rétegek szűrőszáma növekszik (128, majd 256), amelyek egyre komplexebb mintázatokat tanulnak meg.
- Max pooling rétegek: Minden konvolúciós réteget követően max pooling rétegek csökkentik a bemenet méretét és kiválasztják a legdominánsabb jellemzőket.
- Global Average Pooling: A hálózat végén a Global Average Pooling réteg a jellemzőket egy 256-dimenziós vektorba sűríti.
- Dense réteg: Az utolsó réteg 128 kimeneti neuront tartalmaz, amely a képek végső embeddingjét állítja elő.

A modell tanítása: A modell tanítási folyamata a triplet loss függvényen alapul. Ez a veszteségfüggvény arra kényszeríti a modellt, hogy az anchor és a pozitív kép közötti embeddingek távolsága kisebb legyen, mint az anchor és a negatív kép közötti távolság.

A triplet loss célja, hogy a modell, az azonos személyek embeddingjeit közel hozza egymáshoz, míg az eltérő személyekét távol tartsa egymástól.

A végső modellhez Adam optimalizálót használtam, amely segít beállítani a modell paramétereit a jobb teljesítmény érdekében.

A modell tesztelése: Végezetül a tesztelés során a modell teljesítményét egy külön teszthalmazon kiértékeltem, amely különböző fényviszonyokat és arckifejezéseket tartalmazott.



5.2. ábra: Sequential model

5.1.2 Adathalmaz kiválasztása

A modell hatékonyságának egyik kulcsa a megfelelő adathalmaz kiválasztása volt, ami az arcfelismerési rendszer pontosságát és általánosíthatóságát nagymértékben befolyásolta. Az arcfelismeréshez nem csak nagy mennyiségű, hanem kellően változatos adathalmazra volt szükségem. Ez azt jelentette, hogy a képeknek különböző szögekből, különböző fényviszonyok mellett és eltérő hátterekkel kellett készülniük ahhoz, hogy a modell valóban stabilan és megbízhatóan működjön. Ezen felül azt is kellett mérlegelnem, hogy egy sokszínű, de személyenként kevés képet tartalmazó adathalmazt vagy egy kevesebb egyeddel rendelkező, de személyenként sokkal több képet tartalmazó adathalmazzal dolgozzak.

A hálózat tanításához végül úgy döntöttem, hogy az adatokat úgy szervezem, hogy sok különböző emberről szerepeljen legalább néhány kép, ezzel lehetővé téve, hogy a modell ne általánosítson, azaz meg tudja különböztetni a személyek közötti különbségeket. Ez segít a Siamese hálózatnak abban, hogy megbízható távolságmetrikát adjon meg, amely hasonló arcokat közelebb hoz egymáshoz, míg az eltérő arcokat távol tartja egymástól.

Adatelőkészítés:

- Haar Cascades algoritmus használata az arcok kinyeréséhez a képből
- Az adatok normalizálása és méretezése (160x160 pixeles képek).
- Hármas (anchor, pozitív, negatív) példák generálása a tanításhoz.
- A tanítás során adat-augmentáció használata (például forgatás, zaj hozzáadása), hogy a modell különböző körülmények között is megfelelően tanulja meg a kép jellemzőket.

Emellett, a nagyobb mélység elérése érdekében néhány személyről több képet is generáltam adat-augmentáció segítségével, hogy a hálózat felismerje ugyanazon személy arcának különböző változatait (például eltérő szögek, megvilágítás és arckifejezések esetén). Az adat-augmentáció során használtam különböző rotációkat, nagyításokat, eltolásokat, fényerő és kontraszt módosításokat is. Így ez a fajta adatkészlet hozzájárult ahhoz, hogy a modell jobban általánosítson, és stabilabban különböztesse meg a személyeket, akár eltérő körülmények között is.

5.1.3 Webalkalmazás backendje

A webalkalmazás backendjének fejlesztése a projekt egyik legfontosabb és legösszetettebb része volt. A cél egy stabil, biztonságos és skálázható rendszer kialakítása volt, amely megannyi beépített funkciót kínál. Az egyik legfontosabb szempont viszont az volt, hogy könnyen integrálható legyen a modell. Ezeket mérlegelve a Django keretrendszert válasz-

tottam, mivel jól strukturált környezetet biztosít a fejlesztéshez, és számos olyan funkciót kínál, amelyek a projekt számára ideálisak voltak.

A Django keretrendszer egyik legnagyobb előnye, hogy letisztult és jól dokumentált struktúrával rendelkezik, ami megkönnyítette a fejlesztési folyamatot. Az egész keretrendszer MVC (Model-View-Controller) architektúrára épül. Az MVC architektúra alkalmazásával különválasztottam az adatmodelleket, a logikákat és a felhasználói felületeket, ami átláthatóvá és könnyen karbantarthatóvá tette a kódot. A Django emellett beépített hitelesítési és jogosultságkezelési rendszerrel rendelkezik, amelyeket kiterjesztettem az alkalmazás igényeinek megfelelően.

A backend fejlesztésének egyik legkritikusabb része az arcfelismerő modell integrációja volt. A hagyományos felhasználónév-jelszó alapú hitelesítés mellett az alkalmazás lehetőséget biztosít arra, hogy a felhasználók arcfelismerés segítségével is beléphessenek. Az arcfelismerő modell egy .h5 kiterjesztésű fájlként van beimportálva az alkalmazásba, amit a háttérben használ az arcok embeddingjeinek kiszámítására.

Amikor a felhasználó a bejelentkezési folyamat során lefotózza az arcát, az alkalmazás a háttérben elő készíti a képet a modell számára, méretezéssel és normálással. Az előkészített képet a modell feldolgozza, ezt követően pedig kiszámolja a képhez tartozó embeddinget, amely az arcot, egy egyedi, numerikus reprezentációként írja le egy sokdimenziós térben. Ezután a rendszer a kiszámolt embeddinget összehasonlítja az adatbázisban tárolt összes, már eddig ismert embeddinggel, koszinusz-hasonlóság (cosine similarity) segítségével.

A koszinusz-hasonlóság értéke azt fejezi ki, hogy a feltöltött kép embeddingje és az adatbázisban található képek embeddingjei mennyire helyezkednek el egy irányba a sokdimenziós térben. Ha az összehasonlítás során a legmagasabb koszinusz-hasonlósági érték meghaladja az előre definiált küszöbértéket, akkor a rendszer úgy ítéli meg, hogy a feltöltött kép megegyezik az adott felhasználóhoz társított valamely, már megadott képével. Ez esetben a bejelentkezést sikeresnek tekinti, és a felhasználót beengedi a rendszerbe. Ha pedig a hasonlóság nem éri el a küszöbértéket, a rendszer a bejelentkezési kísérletet elutasítja.

5.1.4 Adatbázis kiválasztása

Kezdetben az alkalmazás SQLite adatbázissal működött, amely gyors és könnyen integrálható megoldás volt a fejlesztés kezdeti szakaszában. Ahogy azonban a projekt komplexebbé vált, az SQLite adatbázis korlátai is egyre inkább felszínre kerültek. A rendszer összetettebb lett, amelyhez egy skálázhatóbb és egy jobb teljesítményű adatbázisra volt szükség. Így döntöttem a PostgreSQL bevezetése mellett, amely megbízható adatkeze-

lést, párhuzamos feldolgozási lehetőségeket és fejlett adatbiztonsági funkciókat biztosított. Mindemellett fontos szempont volt, hogy könnyen integrálható legyen a Django keretrendszerrel.

Az adatbázis szerkezetét úgy alakítottam ki, hogy az megfeleljen az alkalmazás funkcionális igényeinek. Az alábbi főbb táblákat hoztam létre:

- Felhasználók: Ez a tábla tartalmazza a regisztrált felhasználók alapvető adatait, mint például a nevüket és jelszavukat.
- Arcképek: A felhasználókhoz társított arcképeket és személyes adatokat tartalmazza, beleértve a képek elérési útjait és metaadatait.
- Dokumentumok: Ez a tábla a feltöltött dokumentumok metaadatait tárolja, beleértve a fájl nevét, a feltöltő felhasználónak az azonosítóját, a feltöltés dátumát és a fájl elérési útját.
- Logok: A bejelentkezési eseményeket, sikertelen próbálkozásokat és az egyéb felhasználói műveleteket rögzíti.

Az adatbázis-migráció során fontos volt, hogy az SQLite-ban tárolt adatokat zökkenőmentesen át tudjam mozgatni a PostgreSQL-be. Ehhez a Django beépített migrációs eszközeit használtam, amelyek lehetővé tették az adatok könnyű exportálását és integrálását az új adatbázisba.

A PostgreSQL-re való áttérés további előnyökkel is járt. A rendszer teljesítménye jelentősen javult. Az adatokhoz való hozzáférés gyorsabbá vált, ami különösen fontos volt a bejelentkezési folyamatnál, ahol az arcfelismerési modellnek gyorsan hozzá kellett férnie a felhasználók arcképeihez rendelt leíró vektorokhoz.

Egy másik jelentős kihívást a rendszerhez kapcsolódó adatbiztonság és adatkezelés jelentette. Mivel az alkalmazás érzékeny adatokat kezel, fontos volt, hogy ezek az adatok megfelelően védve legyenek. A PostgreSQL erős adatbiztonsági funkciói, például az adattitkosítás támogatása, kulcsfontosságúak voltak ebben. A titkosítás lehetőséget adott arra, hogy az adatbázisba történő íráskor az érzékeny adatokat titkosítva tároljam, biztosítva, hogy az adatok biztonságban tudhassam.

5.1.5 Dokumentumok kezelése

A dokumentumkezelési funkciók megvalósítása során szintén szembe kellett néznem egy fontos adatbiztonsági problémával. A legnagyobb probléma a fájlok biztonságos tárolásának és hozzáférésének szabályozása volt. A fájlokat az alkalmazás által generált egyedi azonosítókkal láttam el, amelyek biztosították, hogy az URL-ek ne legyenek könnyen kitalálhatók, és illetéktelen hozzáférések ne történhessenek.

5.1.6 Egyéb biztonsági intézkedések

A Django beépített biztonsági funkciói nagyban hozzájárultak a megfelelő védelem kialakításához. Az egyik legfontosabb ilyen eszköz a CSRF-védelem. Ez biztosítja azt, hogy az alkalmazásban végrehajtott műveletek csak a felhasználó által indított valódi kérések során valósulhassanak meg. Minden űrlap automatikusan kap egy egyedi CSRF-tokent, amelyet a szerver ellenőriz. Így megelőzve az olyan támadásokat, amelyek során egy támadó adatokat próbálna küldeni az alkalmazás szerverének.

A jelszavak kezelésére a Django szintén egy beépített funkciót kínál. Az alkalmazás a bejelentkezési és regisztrációs folyamatai során a jelszavakat nem egyszerű szövegként tárolja el az adatbázisban, hanem a Django automatikusan titkosítotja azokat, modern PBKDF2 algoritmust alkalmazva. Ez azt jelentette, hogy nem kellett manuálisan gondoskodnom a jelszavak biztonságos tárolásáról, mivel a Django ezt a folyamatot teljesen automatizálta.

Az adatbázisba történő mentést, a Django ORM (Object-Relational Mapping) rendszere végezte. Ez a megközelítés nemcsak gyors és hatékony volt, de egy esetleges SQL-injekciós támadás kockázatát is jelentősen csökkentené.

5.1.7 Valós idejű alkalmazás monitorozás

A Sentry bevezetése jelentős lépés volt a hibakezelés és a valós idejű alkalmazás monitorozás érdekében. Mivel az alkalmazásom érzékeny adatokat kezel, mint például az arcképek és jelszavak, rendkívül fontos volt, hogy gyorsan felismerjem és kezeljem az esetleges hibákat vagy teljesítménybeli problémákat.

A Sentry lehetővé teszi, hogy valós időben figyeljem az alkalmazásban felmerülő hibákat. Részletes információkat nyújt a hiba okáról, előfordulási helyéről és előzményeiről. Ez különösen hasznos az adminisztrátor számára, mivel így az alkalmazás teljesítményére és stabilitására vonatkozó adatokat könnyen elérheti és nyomon követheti. Amikor egy hiba bekövetkezik, a Sentry részletes naplót készít, amely tartalmazza az érintett kódrészletet, a hibához kapcsolódó változókat, valamint a kiváltó események sorozatát. Ez a fejlesztés során is kifejezetten jól jött, hisz lehetővé tette, hogy pontosan megértsem, mi vezetett az adott hibához.

A Sentry másik nagy előnye, hogy nemcsak az észlelt hibákat jelzi, hanem figyeli a teljesítményadatokat is, például a válaszidőket. Ez lehetővé teszi, hogy az alkalmazás optimalizálása során pontos adatokat lássak arról, hogy mely részek igényelnek javítást, illetve hol van szükség további optimalizálásra.

5.1.8 Naplózás

Szükség volt még egy naplózó könyvtárra is, mely lehetővé teszi az adminisztrátornak azt, hogy lássa ki, mikor, mit csinál a weboldalunkon. A felhasználói aktivitás, mint például a belépések és regisztrációk logolása különösen fontos, szintúgy az alkalmazás és a benne tárolt adatok biztonsága érdekében. A logok segítenek abban, hogy a felhasználói interakciókat és esetleges problémákat időben észleljük.

A belépések és regisztrációk logolása külön figyelmet igényel, mivel ezek az események közvetlenül kapcsolódnak a felhasználók személyes adataihoz. Minden egyes bejelentkezéskor, valamint a sikeres vagy sikertelen regisztrációk során rögzíti a rendszer az esemény időpontját, a felhasználó azonosítóját, a használt IP-címet és az alkalmazott hitelesítési módszert. A sikeres belépések mellett a sikertelen belépési kísérletek is naplózásra kerülnek, annak érdekében, hogy a rendszer adminisztrátora észlelhesse a potenciális biztonsági fenyegetéseket.

A rendszer minden más fontos eseményt is logol, például a felhasználók profiljának módosításait, a dokumentumok feltöltését és törlését. Mivel az alkalmazásban érzékeny adatokat kezelünk, különösen fontos volt, hogy minden adatkezelési művelet, például az arcok feltöltése vagy törlése, alaposan dokumentálva legyen.

Összefoglalva, a program fejlesztése során sikerült egy olyan erős alapot létrehoznom, amely támogatja az összes elvárt alap funkciókat, miközben biztosítja a biztonságot és a skálázhatóságot. Az alkalmazás a Django keretrendszer és a PostgreSQL adatbázis erejére támaszkodva nemcsak a jelenlegi igényeket elégíti ki, hanem rugalmasan bővíthető a jövőbeli fejlesztések során is.

6 TESZTELÉS

A fejlesztési folyamat fontos szakasza volt a tesztelés, amely során meggyőződtem arról, hogy az alkalmazás az elvárásoknak megfelelően működik. A tesztelés célja nemcsak az volt, hogy bizonyítékot szolgáltassak a fejlesztés sikerességéről, hanem az is, hogy az esetleges hibákat feltárjam és kijavítsam. A tesztelési folyamatot több szinten végeztem, beleértve a manuális, funkcionális, unit és biztonsági teszteket. Az alábbiakban részletesen bemutatom a különböző tesztelési szakaszokat és eredményeket.

6.1 Regisztráció

A regisztrációs funkció az egyik legfontosabb eleme az alkalmazásnak. A rendszer biztosítja, hogy csak érvényes adatokkal rendelkező felhasználók regisztrálhassanak. A tesztek során manuális és funkcionális tesztelést végeztem, amelyek során az e-mail címek formátuma és a jelszavak erőssége ellenőrzésre került, ezen felül a regisztrációs űrlap kitöltésére vonatkozó követelményeket is ellenőriztem. A rendszer megfelelően validálta a bemeneti adatokat, és csak az elvárásoknak megfelelő regisztrációkat fogadta el. A regisztrációs események naplózásra kerültek, ami lehetővé teszi az adminisztrátor számára, hogy nyomon kövesse az új felhasználók aktivitását.

6.2 Bejelentkezés

A bejelentkezési funkciók különböző módjait (jelszóval, illetve arcfelismeréssel) különkülön teszteltem.

- Jelszóval történő bejelentkezés: A hagyományos felhasználónév és jelszó párossal történő hitelesítés szintén sikeresen működött. A rendszer megfelelően kezelte a helyes és helytelen jelszavakat, a nem kitöltött bejelentkezési űrlapot, valamint a sikertelen bejelentkezési kísérleteket is naplózta. Ez különösen fontos a biztonsági szempontok miatt.
- Arcfelismeréssel történő bejelentkezés: Az arcfelismerési modell a koszinusz-hasonlóság alapján végzi a hitelesítést. A tesztelés során a rendszer a tesztadatokon 95,31%-os pontosságot ért el, ami biztosította a felhasználók azonosítását az esetek döntő többségében. A tesztek során ellenőriztem, hogy a modell 16 felhasználóval egész megbízható különbséget tesz az azonos és különböző személyek között. (A modell teszteléséről és kiértékeléséről a későbbiekben még írok) Az arcfelismerésen alapuló bejelentkezések sikeresen naplózásra kerültek, beleértve a bejelentkezések idejét és a hitelesítések típusát.

6.3 Közös felület elérése

A közös felület hozzáférésének szabályozása kiemelkedően fontos volt az adatvédelem érdekében. A funkcionális tesztek során ellenőriztem, hogy a közös felület, kizárólag bejelentkezett felhasználók számára legyen elérhető. A rendszer minden hitelesítés nélküli hozzáférési kísérletet blokkolt, és megfelelő átirányítást biztosított a bejelentkezési oldalra. Ez a funkció biztosította, hogy az adatokhoz illetéktelen felhasználók ne férhessenek hozzá.

6.4 Dokumentumkezelés

- Dokumentum feltöltés: A rendszer megfelelően kezelte a fájlok feltöltését. A sikeres feltöltési események naplózásra kerültek, beleértve a feltöltő felhasználó azonosítóját, a fájl nevét és a dokumentum feltöltésének idejét. Az adatbázisban sikeresen létrejöttek a hozzájuk tartozó rekordok.
- Dokumentum törlés: Csak a fájl tulajdonosa törölhetett dokumentumokat, amelynek a jogát a rendszer megfelelően érvényesített. Illetve az adatbázisban is sikeresen törlésre kerültek. A törlési események szintén naplózva lettek, amik segítenek az esetleges anomáliák nyomon követésében.

6.5 Arcképek kezelése a profil oldalon

Az arcképek kezelését manuálisan teszteltem, beleértve a képek feltöltését és törlését. A rendszer automatikusan méretezte és mentette a feltöltött képeket. A törlést követően pedig eltűnt a kép a felhasználói felületről, illetve az adatbázisban sem lehetett már megtalálni. Minden profilképpel kapcsolatos esemény (feltöltés vagy törlés) naplózásra került, biztosítva a teljes körű visszakövethetőséget.

6.6 Felhasználói aktivitás naplózása

A naplózási rendszer egyik legfontosabb feladata, hogy rögzítse a felhasználói aktivitásokat, beleértve a regisztrációkat, bejelentkezéseket, dokumentum feltöltéseket és törléseket. A tesztek során ellenőriztem, hogy minden ilyen esemény naplózásra kerüljön. A naplók tartalmazták az esemény típusát, a felhasználó azonosítóját és az idejét az esemény létrejöttének.

6.7 Biztonság

A biztonsági tesztek során különös figyelmet fordítottam az SQL-injekció elleni védelemre. A rendszer minden rosszindulatú próbálkozást blokkolt. A Django ORM mechanizmusa megelőzte a káros parancsok végrehajtását. Emellett a nem megfelelő fájlformátumok kezelése szintén sikeres volt, ezzel minimalizálva a rendszer kitettségét a potenciális fenyegetésekkel szemben.

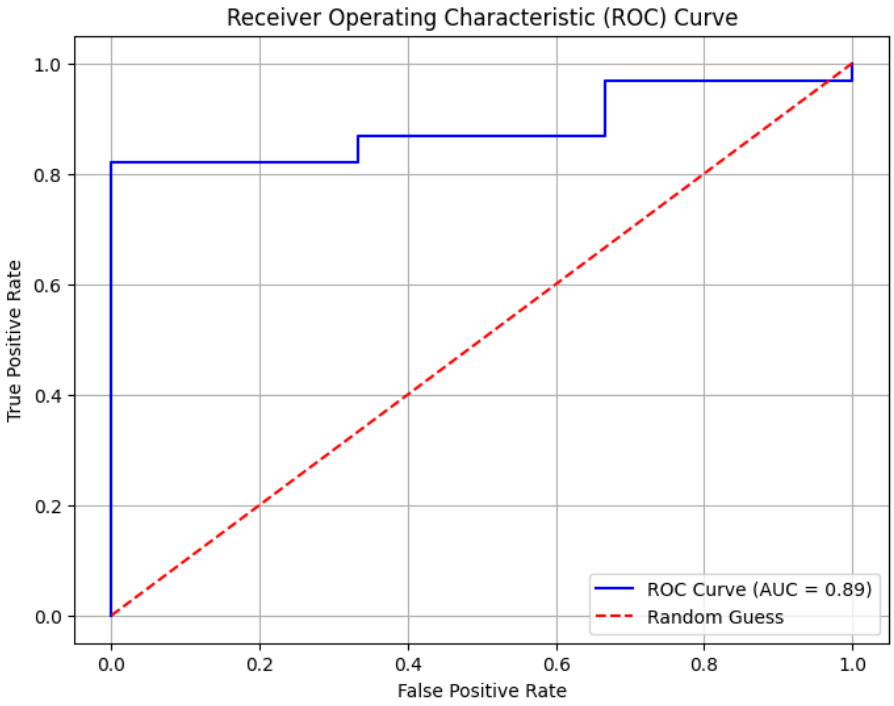
6.8 A modell kiértékelése

A szakdolgozat kritikus eleme, hogy a fejlesztett modell és rendszer megbízhatósága bizonyítható legyen. Ehhez különböző tesztelési eljárásokat kell alkalmazni, amelyek meggyőzően igazolják, hogy az elkészült megoldás megfelelően működik. A következőkben bemutatom a modell kiértékelésére alkalmazott metrikákat és a használt eszközöket.

A tesztelés során a modellt egy 16 személyt ábrázoló, 244 képből álló adatállomány alapján értékeltem, melyből embeddingeket generált a modell. A modell predikcióját 16 új személy, 64 képén végeztem el, ahol az volt a cél, hogy ezek a képek az ismert embeddingekhez kapcsolhatók legyenek.

6.8.1 ROC görbe és AUC érték

Az eredmények közül az egyik legfontosabb az ROC (Receiver Operating Characteristic) görbe (6.1 ábra), amely azt mutatja, hogyan változik az igaz-pozitív mutató (True Positive Rate) és a hamis-pozitív mutató (False Positive Rate) különböző küszöbértékek mellett. A modell AUC (Area Under Curve) értéke 0,89, ami nagyon jó teljesítményt jelez. Az AUC közel van az 1-hez, ami azt mutatja, hogy a modell nagy valószínűséggel képes helyesen megkülönböztetni a különböző személyeket. Az ROC görbe (kék vonal) lényegesen a véletlenszerű találgatást reprezentáló piros szaggatott vonal (AUC = 0,5) fölött helyezkedik el, ami azt igazolja, hogy a modell megbízhatóan működik az osztályozási feladatok során.



6.1. ábra: ROC Görbe

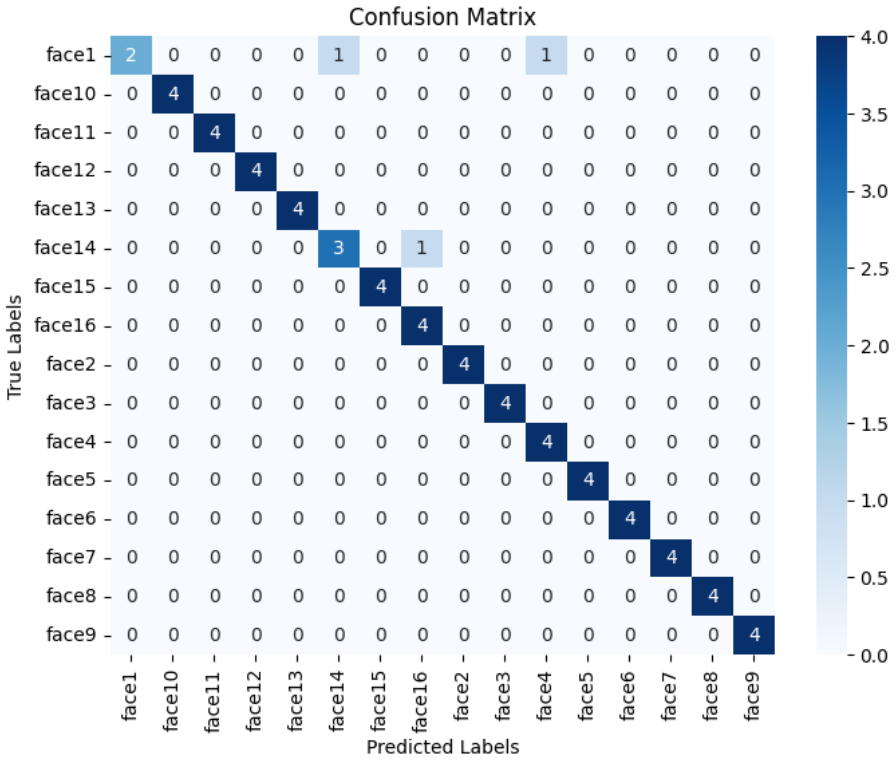
6.8.2 Pontosság (Accuracy), Precision, Recall, F1-Score

- Accuracy: Az összes tesztelt kép 95,31%-át helyesen osztályozta a modell, ami általánosságban jó teljesítményt tükröz (6.2 ábra).
- Precision: Az osztályozás során az adott személyhez tartozó előrejelzett képek nagy arányban helyesek voltak. Ez arra utal, hogy a modell kevés hamis pozitív esetet generál (6.2 ábra).
- Recall: A modell az egyes személyek képeit túlnyomó részt helyesen ismerte fel, bár néhány esetben előfordult, hogy egy-egy kép nem kapcsolódott az adott személyhez (hamis negatív) (6.2 ábra).
- F1-Score: A precision és a recall harmonikus átlaga, kiegyensúlyozottan tükrözi a modell teljesítményét. Az értékek azt igazolták, hogy a rendszer hatékonyan működik mind a pontos, mind a teljes körű felismerés szerint (6.2 ábra).

6.2. ábra: Kiértékelési mutatók

6.8.3 Igazságmátrix (Confusion Matrix)

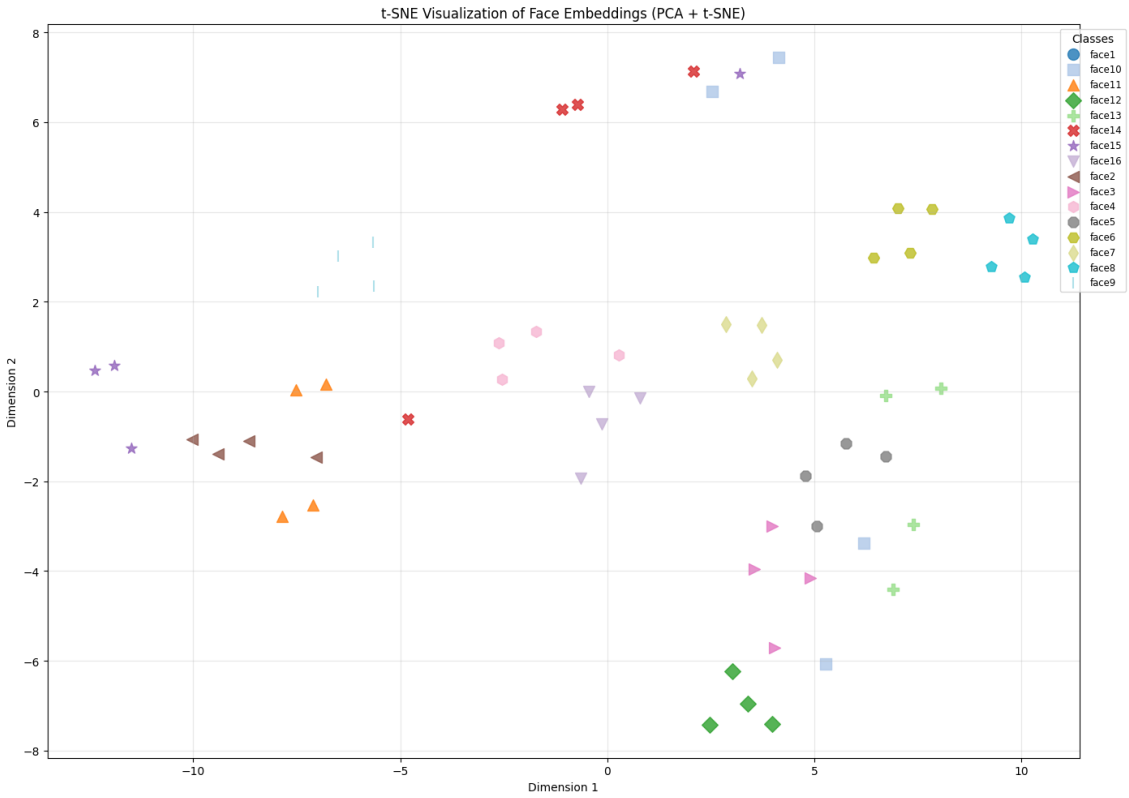
A igazságmátrix (6.3 ábra) részletesen bemutatta, hogy mely személyek között történtek téves osztályozások. Az eredmények azt mutatták, hogy a személyek túlnyomó többségét helyesen osztályozta a modell. Azonban néhány esetben megfigyelhető volt a téves osztályzás, amikor két különböző személy embeddingjei hasonlóvá váltak.



6.3. ábra: Igazságmátrix

6.8.4 Embeddingek kétdimenziós megjelenítése

Az embeddingek 2D-s ábrázolása (t-SNE) jól szemléltette (6.4 ábra), hogy az egyes személyek csoportjai elkülönülnek egymástól a kétdimenziós térben. A csoportosulások igazolják, hogy a modell sikeresen azonosította a különböző személyeket. Az átfedések száma minimális volt, de néhány átfedés jelezte, hogy az embeddingek további optimalizálásával tovább lehetne növelni a rendszer pontosságát.



6.4. ábra: Embeddingek vizualizációja

6.9 Tesztelés összegzése

A 6.6 táblázat összefoglalja az elvégzett összes tesztesetet, a tesztesetekhez tartozó alkalmazási módszereket és eredményeket. Összességében kijelenthető, hogy az összes, általam megvizsgált teszteset sikeres volt. Különféle tesztelési módszereket alkalmaztam, beleértve a manuális tesztelést is. A tesztesetekhez használt szövegek (jelszó, felhasználónév), illetve képek/fileok mind saját forrásból kerültek ki, ez alól kivétel a modell kiértékelése.

6.6. táblázat: Tesztesetek, céljaik és eredményeik

Teszteset	**Cél**	**Tesztelési módszer**	**Eredmény**	**Megjegyzés**
Regisztráció validációval	Ellenőrizni, hogy a felhasználók helyesen tudnak regisztrálni.	Manuális és funkcionális teszt	Sikeres	Az e-mail formátuma és a jelszó erőssége megfelelően ellenőrizve, az esemény naplózva.
Bejelentkezés arcfelismerés-sel	Ellenőrizni, hogy a felhasználók arcfelismeréssel be tudnak-e jelentkezni.	Funkcionális teszt	Sikeres	A koszinusz-hasonlósági küszöb megfelelően működik, a bejelentkezés naplózva.
Bejelentkezés jelszóval	Ellenőrizni a hagyományos hitelesítést.	Manuális és funkcionális teszt	Sikeres	A jelszókezelés és validáció működik, az esemény naplózva.
Közös felület elérése hitelesítés nélkül	Biztosítani, hogy a közös felület csak bejelentkezés után legyen elérhető.	Funkcionális teszt	Sikeres	Hitelesítés nélküli hozzáférés blokkolva, megfelelő átirányítás működik.

Teszteset	Cél	Tesztelési módszer	Eredmény	Megjegyzés
Dokumentum feltöltése	Ellenőrizni, hogy a felhasználók dokumentumokat tudnak-e feltölteni.	Manuális és funkcionális teszt	Sikeres	A feltöltött fájlok megfelelően mentésre kerülnek, a feltöltési esemény naplózva.
Dokumentum törlése	Ellenőrizni, hogy a felhasználók törölhetik-e saját fájljaikat.	Funkcionális teszt	Sikeres	Csak a fájl tulajdonosa törölheti a dokumentumot, a törlés naplózva.
Profilkép feltöltése	Ellenőrizni, hogy a felhasználók képeket tölthetnek fel a profiljukhoz.	Manuális teszt	Sikeres	A képek méretezése és mentése megfelelően működik, az esemény naplózva.
Profilkép törlése	Ellenőrizni, hogy a felhasználók törölhetik-e a profilképüket.	Funkcionális teszt	Sikeres	A törlés után a kép helyett alapértelmezett kép jelenik meg, az esemény naplózva.
Felhasználói aktivitás naplózása	Ellenőrizni, hogy a rendszer rögzíti-e a felhasználói beavatkozásokat.	Manuális teszt	Sikeres	Minden esemény (regisztráció, bejelentkezés, dokumentumkezelés) pontosan rögzítésre került.

Teszteset	Cél	Tesztelési módszer	Eredmény	Megjegyzés
Arcfelismerési modell pontossága	Az arcfelismerési modell helyességének mérése különböző képekkel.	Unit teszt	Sikeres	A modell pontossága teszt adatokkal 95,31-os volt.
SQL-injekció elleni védelem	Ellenőrizni, hogy az adatbázis-lekérdezések biztonságosak-e.	Manuális teszt	Sikeres	Az ORM megakadályozza a káros SQL parancsokat.

7 ÉRTÉKELÉS

A szakdolgozatom keretén belül egy Django alapú webalkalmazást készítettem, ahol a felhasználó arcfelismerést is tud használni a webalkalmazásba való belépéshez. Az alkalmazás célja egy modern, biztonságos és kényelmes belépési lehetőség megvalósítása volt, emellett dokumentumkezelési funkciókat is biztosít. A rendszer továbbá adminisztrációs és monitorozási eszközökkel is rendelkezik, hogy a működése átlátható és könnyen felügyelhető legyen.

Az egyik legfontosabb funkció az arcfelismerés alapú belépés, amit egy gondosan megtervezett és alaposan letesztelt siamese modell valósít meg. Ez lehetővé teszi a felhasználók számára, hogy arcukkal azonosítsák magukat, ami egy kényelmes és korszerű megoldás. Azonban a tesztek során kiderült, hogy az arcfelismerési modell sajnos még nem teljesen hibátlan. Bizonyos helyzetekben, különösen akkor, ha a felhasználók száma egyre csak növekszik, pontatlanságok léphetnek fel az autentikáció során. Ezek a hibák rávilágítanak arra, hogy a modell továbbfejlesztése, finomhangolása és optimalizálása szükséges lehet a nagyobb pontosság érdekében, esetlegesen egy bizonyos számú felhasználó után a modell újratanítás is szóba jöhet. Azok számára, akik nem szeretnék ezt a funkciót használni, hagyományos felhasználónév és jelszó alapú belépési lehetőséget is biztosít az alkalmazás. Az új felhasználók regisztrációja szintén egyszerűen elvégezhető, melynek hatására az ehhez szükséges adatbázis bejegyzések automatikusan létrejönnek.

Sikeres bejelentkezés után a felhasználók egy közös felületre kerülnek, ahol dokumentumokat tölthetnek fel és törölhetnek. Ez a funkció különösen hasznos lehet egy kisebb cégnél az információk áramlásában és az együttműködés elősegítésében. Mindezek mellett a fejlesztés során kiemelt figyelmet fordítottam arra, hogy a felhasználók a profiljukhoz tartozó képeket is könnyedén kezelhessék. Ez a funkció lehetőséget biztosít számukra, hogy saját képeket töltsenek fel a profiljukhoz, azokat bármikor módosíthassák, vagy ha szükséges, törölhessék. Ez a megoldás személyre szabottabb felhasználói élményt is kínál. Az adatok biztonságos tárolását PostgreSQL alapú adatbázis-kezeléssel oldottam meg, amely megbízhatóságot és jó skálázhatóságot biztosít, így nagyobb terhelés esetén is megfelelően működhet az alkalmazásunk.

A rendszerbe épített logolási funkció minden bejelentkezési/regisztrációs eseményt és felhasználói műveletet rögzít, így az adminisztrátor könnyen nyomon követheti az alkalmazás működését. Ez a funkció kiemelten fontos a biztonság szempontjából, hiszen lehetőséget ad a rendszer eseményeinek monitorozására és az esetleges problémák gyors azonosítására. Mindezt kibővítettem Sentry segítségével, hogy ne csak a felhasználó szintű logokat lássam, hanem a rendszerszintű logolást is. A Sentry használata eredetileg nem szerepelt a tervben, de jelentős kiegészítést nyújtott, különösen a hibakezelés és a rendszer stabilitásának monitorozása terén.

Összességében a projekt sikeresnek mondható. Az elkészült rendszer megfelel az elvárásoknak. Mindemellett a beépített többletfunkció, azaz a Sentry integráció, hosszú távon is kifizetődő lehet. A jövőben az arcfelismerő modell magasabb szintű optimalizálása és a felhasználói élmény finomhangolása tovább növelheti az alkalmazás használhatóságát.

8 TOVÁBBFEJLESZTÉSI LEHETŐSÉGEK

Bár az alkalmazás jelenlegi állapotában már jól működik, és számos hasznos funkcióval rendelkezik, a fejlesztés során több olyan területet is azonosítottam, amely további fejlesztésekkel jelentős előrelépést hozhatna. Ezek a továbbfejlesztési lehetőségek célzott módon növelnék a funkcionalitást, a felhasználói élményt, valamint a rendszer biztonságát.

Az egyik legfontosabb továbbfejlesztési irány az arcfelismerési modell fejlesztése lenne. Bár a jelenlegi siamese modell alapvetően jól működik, a pontatlanságokon kéne még javítani. Ennek megoldására javasolt lenne a modell rendszeres újratanítása friss arcképekkel. Ezzel elérve azt, hogy a modell alkalmazkodjon a változó körülményekhez. Emellett korszerűbb arcfelismerési algoritmusok kipróbálása tovább növelhetné a pontosságot. Különösen fontos lenne olyan optimalizációk kipróbálása, amelyek rossz fényviszonyok között is megbízható eredményt biztosítanak, így a felhasználók még kényelmesebben használhatnák a rendszert.

A dokumentumkezelési funkciók kibővítése szintén jelentős fejlesztési potenciált hordoz. Például egy intelligens keresőmotor integrálása is nagy előrelépés lehetne, amely a dokumentumok címében, tartalmában vagy metaadataiban keresve gyorsan megtalálja a szükséges fájlokat. Egy másik hasznos kiegészítés a dokumentumok nyilvános vagy privát státuszának beállítási lehetősége, amely lehetővé tenné a felhasználók számára, hogy személyes irataikat biztonságban tudhassák, míg más fájlokat megoszthatnának a többiekkel. Ezzel párhuzamosan egy részletesebb jogosultságkezelési rendszert is be lehetne vezetni. A dokumentumokhoz és a profilokhoz különböző hozzáférési szinteket lehetne rendelni, például olvasási vagy szerkesztési jogokat.

Végül, de nem utolsósorban, a rendszer biztonságának és teljesítményének fejlesztése is kiemelt fontosságú. A kétfaktoros hitelesítés bevezetése, például SMS-kód vagy e-mailes visszaigazolás formájában, tovább növelhetné a belépések biztonságát.

9 ÖSSZEGZÉS

9.1 Magyar nyelvű összefoglaló

A szakdolgozatom egy arcfelismerő alapú bejelentkezési rendszer fejlesztését mutatja be, amely a hagyományos hitelesítési eljárások alternatívájaként kínál biztonságos és felhasználóbarát megoldást. A fejlesztési folyamat egyik legfontosabb tanulsága az volt, hogy a biztonság és a felhasználói élmény közötti egyensúly kritikus jelentőségű egy ilyen rendszer esetében.

A projekt során használt technológiák lehetővé tették a gyors prototípus készítést, valamint az arcfelismerő modell iteratív fejlesztését. Az alkalmazás alapját egy Siamese neurális hálózat képezi, amely képes az arcok összehasonlítására és azonosítására. A rendszer funkcionalitása magában foglalja a hagyományos belépésen kívül, a profilképek feltöltését és szerkesztését, valamint a dokumentumok megosztását és kezelését. Az alkalmazásban a PostgreSQL adatbázis felelős az ezekhez szükséges adatok megbízható tárolásáért.

A bejelentkezés utáni interfész lehetőséget ad a felhasználóknak dokumentumok megosztására, feltöltésére és törlésére, valamint a profilképek módosítására. Az alkalmazás hibamentes működését és stabilitását Sentry integrációval ellenőriztem, amely valós idejű hibakövetést biztosított a fejlesztés során.

A projekt megvalósítása során különös hangsúlyt fektettem a felhasználói élményre és az adatbiztonságra. Kiemelt figyelmet igényelt, hogy az arcfelismerő algoritmus magas pontossággal működjön. A Siamese modell kiválasztása hatékonynak bizonyult, mivel képes volt kis mennyiségű tanítóadat mellett is megbízható eredményeket produkálni. A fejlesztési folyamat során azonosítottam a technológia korlátait is. Az arcfelismerés pontossága bizonyos körülmények között például gyenge fényviszonyok vagy arcmaszkok használata esetén csökkenhet.

A szakdolgozat zárásaként megállapítottam, hogy a rendszer hatékonyan ötvözi a modern technológiát a praktikus felhasználói igényekkel. Bár az arcfelismerés nem tökéletes, a fejlesztés során szerzett tapasztalatok megerősítettek abban, hogy az ilyen rendszerek a jövőben egyre fontosabb szerepet játszanak majd a digitális biztonság területén. További fejlesztési irányként az algoritmus robusztusságának növelését és a felhasználói adatok védelmének további erősítését javasoltam. A rendszer éles környezetben történő tesztelése és a felhasználói visszajelzések integrálása szintén hozzájárulhat a projekt fejlődéséhez.

9.2 English Summary

My thesis presents the development of a facial recognition-based login system that offers a secure and user-friendly alternative to traditional authentication methods. One of the most important lessons from the development process was that the balance between security and user experience is critically significant in such a system.

The technologies used during the project enabled rapid prototyping and iterative development of the facial recognition model. The core of the application is a Siamese neural network capable of comparing and identifying faces. The system's functionality includes, beyond traditional login, the ability to upload and edit profile pictures, as well as share and manage documents. A PostgreSQL database ensures the reliable storage of data required for these operations.

The post-login interface allows users to share, upload, and delete documents, as well as modify their profile pictures. To ensure the application's seamless operation and stability, I integrated Sentry for real-time error tracking throughout the development process.

During the project implementation, I placed particular emphasis on user experience and data security. Ensuring high accuracy for the facial recognition algorithm was a key focus. The Siamese model proved to be an efficient choice as it delivered reliable results even with a small amount of training data.

Throughout the development process, I also identified the limitations of the technology. The accuracy of facial recognition can decrease under certain conditions, such as poor lighting or when users wear face masks.

In conclusion, the thesis demonstrates that the system effectively combines modern technology with practical user needs. While facial recognition is not flawless, the experience gained during development reinforced my belief that such systems will play an increasingly important role in digital security in the future. As future development directions, I proposed improving the robustness of the algorithm and strengthening the protection of user data. Testing the system in a live environment and integrating user feedback will also contribute to the project's growth.

10 IRODALOMJEGYZÉK

[1] *Amazon Rekognition*, https://aws.amazon.com/rekognition/, Utoljára megtekintve: 2024.04.01.

[2] *Amazon Rekognition facial analysis in use*, https://aws.amazon.com/blogs/aws/amazon-rekognition-image-detection-and-recognition-powered-by-deep-learning/, Utoljára megtekintve: 2024.04.01.

[3] *Microsoft Azure AI Vision*, https://azure.microsoft.com/en-us/products/ai-services/ai-vision, Utoljára megtekintve: 2024.04.01.

[4] *Microsoft Azure AI face recognition in use*, https://learn.microsoft.com/en-us/answers/questions/453331/face-api-facial-recognition-microsoft-azure-cognit, Utoljára megtekintve: 2024.04.01.

[5] *Betaface*, https://www.betaface.com/wpa/, Utoljára megtekintve: 2024.04.02.

[6] *FDCamStream demo app tracked face classification and recognition*, https://www.betaface.com/wpa/index.php/demo-gallery, Utoljára megtekintve: 2024.04.02.

[7] *Banuba*, https://www.banuba.com/, Utoljára megtekintve: 2024.04.03.

[8] *Banuba Face detection AR SDK*, https://www.banuba.com/ai-face-detection-sdk?utm_source=adwords&utm_medium=ppc&utm_campaign=Brand+Top+Search+Results&utm_term=banuba, Utoljára megtekintve: 2024.04.03.

[9] *BioID*, https://www.bioid.com/, Utoljára megtekintve: 2024.04.03.

[10] *BioID demo facial recognition*, https://www.bioid.com/playground/, Utoljára megtekintve: 2024.04.03.

[11] A. Jadhav, S. Lone, S. Matey, T. Madamwar és P. S. Jakhete, „Survey on Face Detection Algorithms”, *Information Technology Department*, 6. évf., 291–297. old., 2021. febr., ISSN: 2456-2165.

[12] P. Viola és M. Jones, „Rapid object detection using a boosted cascade of simple features”, 1. évf., I–I. old., 2001. DOI: 10.1109/CVPR.2001.990517.

[13] R. A. Nihal és N. Broti, „Design and Control of a Humanoid Robot for Medical Purpose”, 2020. jan. DOI: 10.13140/RG.2.2.21271.19364/1.

[14] I. Arrieta-Arellano, F. López-Orozco, J. Hernández-Hernández és J.-G. Ruiz-Ruiz, „HCI based on eye movements for unlocking mobile devices”, *Avances en Interacción Humano-Computadora*, 6. évf., 6. old., 2021. nov. DOI: 10.47756/aihc.y6i1.78.

[15] W. M. Sanjaya, D. Anggraeni, K. Zakaria, A. Juwardi és M. Munawwaroh, „The design of face recognition and tracking for human-robot interaction”, 315–320. old., 2017. DOI: 10.1109/ICITISEE.2017.8285519.

[16] A. Babu, S. Nair és K. Sreekumar, *Driver's Drowsiness Detection System Using Dlib HOG*, P. Karuppusamy, I. Perikos és F. P. García Márquez, szerk. Singapore: Springer Singapore, 2022, 219–229. old.

[17] V. K. Divyavarshini, N. Govind, A. Vasudevan, G. Chamundeeswari és S. Prasanna Bharathi, *Vehicle Recognition Using CNN*, S. S. Dash, S. Das és B. K. Panigrahi, szerk. Singapore: Springer Singapore, 2021, 671–690. old.

[18] N. El abady, M. Taha és H. Zayed, „Text-Independent Algorithm for Source Printer Identification Based on Ensemble Learning”, *Computers, Materials & Continua*, 73. évf., 1417–1436. old., 2022. máj. DOI: 10.32604/cmc.2022.028044.

[19] A. Hernández, C. Castejón, J. GARCIA-PRADA, I. Padrón és G. Marichal, „Wavelet Packets Transform processing and Genetic Neuro-Fuzzy classification to detect faulty bearings”, *Advances in Mechanical Engineering*, 11. évf., 168781401983118. old., 2019. aug. DOI: 10.1177/1687814019831185.

[20] K. Zhang, Z. Zhang, Z. Li, S. Member és Y. Qiao, „Computer Vision and Pattern Recognition”, *IEEE (Institute of Electrical and Electronics Engineers)*, 1–5. old., 2016. ápr.

[21] M. Gul, M. A. Kalam, M. M. Abbas és tsai., „Multi-objective-optimization of process parameters of industrial-gas-turbine fueled with natural gas by using Grey-Taguchi and ANN methods for better performance”, *Energy Reports*, 6. évf., 2394–2402. old., 2020. nov. DOI: 10.1016/j.egyr.2020.08.002.

[22] A. D. Rasamoelina, F. Adjailia és P. Sinčák, „A Review of Activation Function for Artificial Neural Network”, 281–286. old., 2020. DOI: 10.1109/SAMI48414.2020.9108717.

[23] I. Kandel és M. Castelli, „Transfer Learning with Convolutional Neural Networks for Diabetic Retinopathy Image Classification. A Review”, *Applied Sciences*, 10. évf., 2021. old., 2020. márc. DOI: 10.3390/app10062021.

[24] Y. Xu, „Machine Learning With Echo State Networks”, 2020. márc.

[25] D. Saez-Trigueros, L. Meng és M. Hartnett, „Face Recognition: From Traditional to Deep Learning Methods”, *CoRR*, abs/1811.00116 évf., 2018.

[26] M. You, X. Han, Y. Xu és L. Li, „Systematic evaluation of deep face recognition methods”, *Neurocomputing*, 388. évf., 144–156. old., 2020, ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2020.01.023.

[27] J. Guerrero, M. Coltelli, M. Marsella, A. Celauro és J. Antonio, „Convolutional Neural Network Algorithms for Semantic Segmentation of Volcanic Ash Plumes Using Visible Camera Imagery”, *Remote Sensing*, 14. évf., 4477. old., 2022. szept. DOI: 10.3390/rs14184477.

[28] *Tensorflow*, https://www.tensorflow.org/, Utoljára megtekintve: 2024.11.20.

[29] *Pytorch*, https://pytorch.org/, Utoljára megtekintve: 2024.09.10.

[30] *Django*, https://www.djangoproject.com/, Utoljára megtekintve: 2024.11.18.

[31] *PostreSQL*, https://www.postgresql.org/, Utoljára megtekintve: 2024.11.10.

[32] *MongoDB*, https://www.mongodb.com/, Utoljára megtekintve: 2024.09.10.

11 ÁBRAJEGYZÉK

12 TÁBLAJEGYZÉK

13 RÖVIDÍTÉSEK

CNN	Convolutional Neural Network
SVM	Support Vector Machine
AI	Artificial Intelligence
HOG	Histogram of Gradients
MTCNN	Multitask Convolutional Neural Network
P-Net	Proposal Network
R-Network	Refine Network
O-Net	Output Network
Tanh	Tangens hyperbolicus
ReLU	Rectified linear unit
PBDNN	Paired-Bayesian-Decision Neural Network
SOM	Self-Organising Map
LFW	Labeled faces in the wild
RNN	Recurrent Neural Network
API	Application Programming Interface
CPU	Central Processing Unit
GPU	Graphics Processing Unit
TPU	Tensor Processing Unit
NLP	Natural Language Processing
MVC	Model-View-Controller
MVT	Model-View-Template
ORM	Object-Relational Mapping