Gépi Látás

KÉZZEL ÍRT KARAKTERFELISMERÉS

Látrányi Péter Krisztián | VEC2Ao |

Tartalomjegyzék

Bevezetés

Az írásfelismerés fontossága többek között abban rejlik, hogy az írásunk szinte olyan pontossággal azonosít minket, mint az ujjlenyomatunk vagy a DNS-ünk, sőt, esetenkéntjobban is, hiszen még az egypetéjű ikrekre sem jellemző, hogy tökéletesen azonos lenne azírásképük, míg ez a DNS-szerkezetükre és az ujjlenyomatukra teljesül.

A kézírást már régóta használják azonosítási célokra az informatikában is. Az első aláírás-felismerő rendszert 1965-ben fejlesztették ki, ezt pedig további fejlesztések követték. Az aláírások felismerése azonban még nem okoz akkora nehézséget, mint a karakterfelismerés, mivel az ember aláírásában az azonosságokat vizsgáljuk és nem törődünk azzal, hogy ténylegesen milyen betűkből áll össze a név. Amikor azonban az egyes betűket kell felismernünk, figyelembe kell vennünk, hogy emberenként, sőt, még az egyes embereknél időben is szembetűnő eltérések tapasztalhatók az írásképben. Ezt figyelembe véve pedig nem állíthatjuk, hogy egy betű kinézetét egyetlen vagy akár egy tucat mintapélda alapján megtudjuk határozni, és ezek segítségével egy leírt karaktert valamilyen osztályba be tudunk sorolni.

KARAKTERFELISMERÉS NEHÉZSÉGEI:

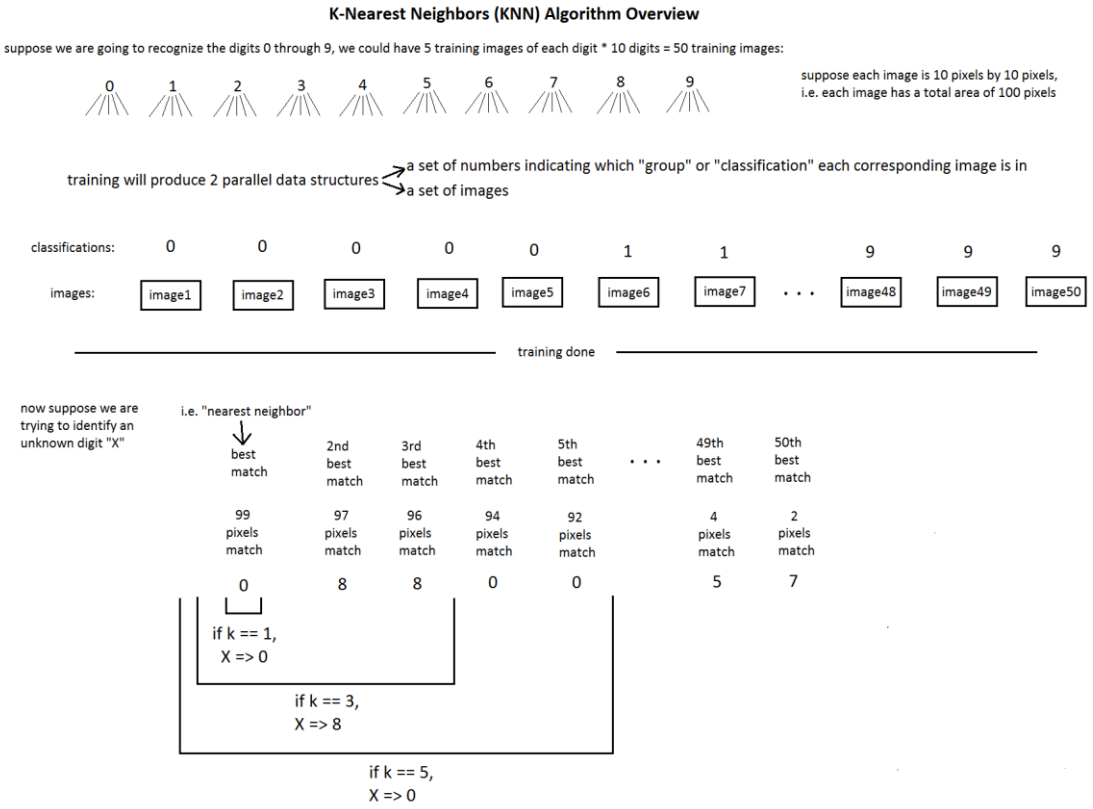
A karakterfelismerést nehezíti:

- „Ugyanazon” karakterek emberenként meglehetősen különbözőek lehetnek méretben, alakban és stílusban, emellett még ugyanazon ember esetén is megfigyelhetők bizonyos eltérések a különböző időpillanatokban leírt karakterek között.
- Minden más képpel egyetemben a karaktereket, illetve szövegeket tartalmazó képek esetén is számolnunk kell a képalkotás – digitalizálás – folyamán, illetve az egyéb okokból a képre került **zajokkal és egyéb képhibákkal** (pl. elmosódás).
- A karakterek kinézetének nincsenek megszeghetetlen, tudományosan lefektetett alapszabályai. Ennek megfelelően csak minták alapján tudunk valamiféle szabályosságot megállapítani az egyes karaktertípusok kinézetével kapcsolatban.

A fejlesztendő programot Pythonban fogom fejleszteni a NumPY és a Scikit-Learn csomagok felhasználásával.

A program működési elvét a KNN algoritmus határozza meg.

KNN ALGORITMUS MŰKÖDÉSE[2]:



1. Euklideszi távolság számítása a test adatpont és a betanított adatok között.
2. A kiszámított távolság rendezése növekvő sorrendben
3. A legközelebbi k szomszédos elemek megtalálása a rendezett tömbben
4. Leggyakoribb osztály keresés a k szomszédos elemek között
5. A prediktált osztály visszaadása
6. Megegyezőség vizsgálata, pontossági szám alapján, hogy meggyőződjünk arról, hogy a predikció jó-e, avagy sem.

Mi az a Python?

Röviden szeretném bemutatni ebben a fejezetben, hogy mi is a python programozási nyelv. A jellemzést a magyar Wikipédia oldalról hivatkoztam, mivel ennél jobban véleményem szerint nem lehet megfogalmazni hétköznapi nyelven a python sajátosságait.

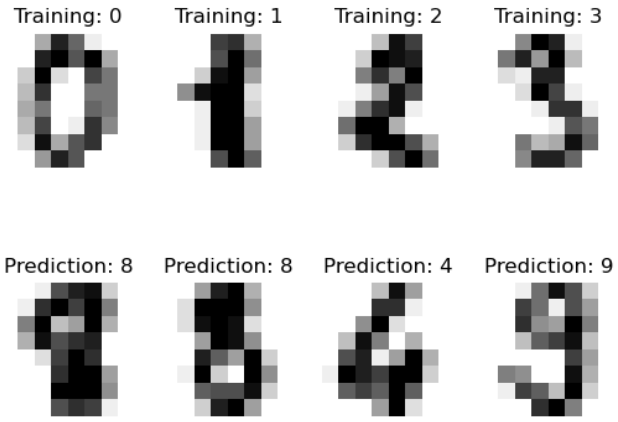
„A Python egy általános célú, nagyon magas szintű programozási nyelv, melyet Guido van Rossum holland programozó kezdett el fejleszteni 1989 végén, majd hozott nyilvánosságra 1991-ben. A nyelv tervezési filozófiája az olvashatóságot és a programozói munka megkönnyítését helyezi előtérbe a futási sebességgel szemben.

A Python többek között a funkcionális, az objektumorientált, az imperatív és a procedurális programozási paradigmákat támogatja. Dinamikus típusokat és automatikus memóriakezelést használ, ilyen szempontból hasonlít a Scheme, Perl és Ruby nyelvekhez, emellett szigorú típusrendszerrel rendelkezik.

A Python úgynevezett interpreteres nyelv, ami azt jelenti, hogy nincs különválasztva a forrás- és tárgykód, a megírt program máris futtatható, ha rendelkezünk a Python értelmezővel. A Python értelmezőt számos géptípusra és operációs rendszerre elkészítették, továbbá számtalan kiegészítő könyvtár készült hozzá, így rendkívül széles körben használhatóvá vált.” [2]

Scikit-Learn

Az SK Learn digits adatacsomagja 180 mintát 8x8 pixel felbontásban tartalmaz minden egyes számjegyre (0-9). A minták előállításához 43 ember kézírását használták fel.
A példákban fekete háttérrel, fehérrel írt számok találhatóak, ezért ugyanilyen tulajdonságú képekre működik a legjobban az algoritmus.



Programfejlesztés Lépései

- Importálandó könyvtárak importálása
- A tanuló algoritmus betanítása.
- A teszt kép beolvasása, átméretezése.
- Teszt kép normalizálása (intenzitás csökkentése)
- Predikció
- Eredmény kiíratása
- Tesztelés
- Hibakezelés

PROGRAM FUTTATÁSÁHOZ SZÜKSÉGES ESZKÖZÖK:

- ❑ Java program futtatására alkalmas fordító
- ❑ A program futtatásához szükséges csomagok telepítése
- ❑ Tesztelendő kép megléte
- ❑ config.py és main.py forrásfájlok

PROGRAM FUTTATÁSÁHOZ SZÜKSÉGES CSOMAGOK:

- ➢ scikit-image
- ➢ scikit-learn

PROGRAM FUTTATÁSÁNAK LÉPÉSEI:

- ➔ **A tesztelendő képnek az elérési útját és nevét** meg kell adni a config.py-ban. Az IMAGE_DIR = kódsornál idézőjelek között kell megadni a mappa elérési útját. Továbbá a TEST_IMAGE = kódsornál idézőjelek között kell megadni a fájl nevét és kiterjesztését.
- ➔ config.py futtatása (F5)
- ➔ main.py futtatása(F5)

Alapértelmezetten a program a forrásfájl elérési útján belül az images nevű almappába várja a tesztképeket

Tesztelés eredményei

Az alábbi táblázatban látható, hogy a programhoz a mellékelt teszt képekre, milyen kimenetet ad az algoritmus.

A program a 27 teszt esetből 8x adott rossz eredményt.
Leginkább a 2-es, 3-as karakterek felismerése jelentette a problémát.
A tesztesetek 70,37%-ban működött rosszul.
Fontos megjegyezni, hogy az algoritmus működésének hatékonyságára pár példa alapján nem érdemes megállapítani. Több ezer inputtal lehetne mélyre menő következtetéseket levonni.
A mintául szolgáló adatcsomag tesztelésekor a Scikit Learn 97%-os pontosságot állapított meg.

Sorszám	Beolvasott Kép	Képen lévő karakter	Felismert karakter	OK
1	0.jpg	0	0	IGAZ
2	0_b.jpg	0	0	IGAZ
3	1.jpg	1	1	IGAZ
4	1_b.jpg	1	1	IGAZ
5	2.jpg	2	4	HAMIS
6	2_b.jpg	2	7	HAMIS
7	2_c.jpg	2	2	IGAZ
8	2_d.jpg	2	1	HAMIS
9	3.jpg	3	1	HAMIS
10	3_b.jpg	3	1	HAMIS
11	3_c.jpg	3	7	HAMIS
12	3_d.jpg	3	3	IGAZ

13	4_a.jpg	4	4	IGAZ
14	4_b.jpg	4	4	IGAZ
15	4_c.jpg	4	4	IGAZ
16	5.jpg	5	5	IGAZ
17	5_b.jpg	5	5	IGAZ
18	5_c.jpg	5	5	IGAZ
19	6.jpg	6	4	HAMIS
20	7.jpg	7	7	IGAZ
21	7_b.jpg	7	7	IGAZ
22	7_c.jpg	7	7	IGAZ
23	8.jpg	8	8	IGAZ
24	8_b.jpg	8	8	IGAZ
25	8_c.jpg	8	7	HAMIS
26	9.jpg	9	9	IGAZ
27	9_b.jpg	9	9	IGAZ

IGAZ	19	70,37%
HAMIS	8	29,63%

Felhasznált irodalom

[1] Brown, Erik W. Applying Neural Networks to CharacterRecognition,1992

[2] https://hu.wikipedia.org/wiki/Python_(programoz%C3%A1si_nyelv)

[3]https://scikitlearn.org/stable/auto_examples/classification/plot_digits_classification.html