2. feladat

Alakfelismerés macskákra

Közismert, hogy a macskás gif-ek milyen fontos szerepet játszanak/játszottak az Internet fejlődésében. Egy másik mostanában nagyon fontos kérdéskör pedig a számítógépes alak- vagy arcfelismerés. Ez a két megatrend inspirálta a következő feladatot.

Egy kedves ismerősünknek elszöktek a macskái. Ez gyakran előfordul, és hogy ilyenkor az összes környékbeli ház webkameraképén tudjunk keresni, automatizálni kell a keresést. Írjunk programot, ami egy ilyen képen megtalálja őket! Elsőként az egyik elveszett cicáról adunk egy képet:

lost-cat.gif

Mivel 7 egyforma macskáról van szó, egy cica képe elegendő lesz. Az egyik szomszéd verandáját mutató képen több olyan cicát is felfedezünk, amelyek nagyon hasonlítanak a mi elveszett macskáinkra:

cat-search.png

Azt kell kiderítenünk, hogy valóban a mi macskáink vannak-e ezen a képen. Az ilyen típusú feladatokat pl. neurális hálózatokon alapuló algoritmusokkal szokták megoldani, de mi most egy egyszerűbb megközelítést alkalmazunk.

A kereséshez és összehasonlításhoz az elveszett cica képét egy mátrixban tároljuk. Ezután kiszámoljuk ennek a képnek és a verandát mutató kép egyes részleteinek a korrelációját és feljegyezzük, hogy a nagy kép mely részein lesz a korreláció egy bizonyos határértéknél nagyobb.

A feladat megoldásához három dolgot kell megtanulni:

1) Mátrix tárolása egyindexes tömbben

Mátrixok tárolása C-ben a leghatékonyabb sorfolytonos alakban. Ez azt jelenti, hogy ha $cols$ a mátrix oszlopainak száma (azaz a mátrix szélessége), akkor a hagyományosan a_{ij} elemet $a[i * cols + j]$ módon tudjuk elérni (akár kiolvasás, akár értékadás céljából). Ehhez $rows * cols$ hosszúságú tömböt kell létrehozni (rows a sorok száma) . (Lásd az előadás anyagában).

E feladatban képeket fogunk mátrixban tárolni, amelyeket a 2-es beadandó feladathoz hasonló formátumban, .dat kiterjesztésű fájlokban is megadunk. A méretük (szélesség x magasság) kiolvasható a fájlnévből. Ezekben (a text fájlok és a python képkezelési hagyományát követve) a kezdő adat a kép bal felső sarkának felel meg. (Érdemes megjegyezni, hogy ezzel ellentétben pl. a bmp formátumok a bal alsó sarokkal kezdenek.)

2) Adatfájl fejlécének kezelése

Gyakran előfordul, hogy a bemenő adatfájlok (és az eredmény fájlok is) tartalmaznak egy fejlécet, amelyben az adatokkal kapcsolatos fontos információkat írunk le (pl ha az adatok több oszlopban vannak a fájlban, akkor az egyes oszlopokban található adatok jelentését). A fájlból való beolvasáskor a programnak figyelnie kell, hogy fejlécet olvassa-e vagy az adatokat tartalmazó részt, hiszen ez utóbbit kell feldolgoznia.

3) Két adatsor korrelációja

Korreláció számolásakor azt vizsgáljuk, hogy a két adatsor mennyire váltakozik együtt. Ezt úgy tudjuk megállapítani, hogy az átlagtól való eltéréseiket $(x - \overline{x})(y - \overline{y})$ alakban összeszorozzuk. (Itt $\overline{x}$ és $\overline{y}$ az átlagot jelöli.) Ezt a szorzatot azután átlagoljuk és viszonyítjuk ahhoz, hogy a két adatsor mennyire ingadozik (leosztjuk a szórásokkal):

$$corr(x,y) = \overline{(x - \overline{x})(y - \overline{y})} / (\sigma_x \sigma_y)$$

Így az x vagy y adatsort magával korreláltatva éppen 1-et kapunk.

A mennyiségek indexes alakját felhasználva néhány átalakítással olyan alakhoz jutunk, amit technikailag könnyebb kiszámolni:

$$corr(x,y) = \frac{\overline{xy} - \overline{x} \cdot \overline{y}}{\sqrt{\left(\overline{x^2} - \overline{x}^2\right)} \sqrt{\left(\overline{y^2} - \overline{y}^2\right)}} \ ,$$

ahol az átlagokat egymáshoz hasonlóan az egyszerű összegekből kapjuk:

$\overline{x} = \left(\sum_{i=1}^{n} x_i\right) / n$, $\overline{y} = \left(\sum_{i=1}^{n} y_i\right) / n$, $\overline{x^2} = \left(\sum_{i=1}^{n} x_i^2\right) / n$, $\overline{y^2} = \left(\sum_{i=1}^{n} y_i^2\right) / n$, $\overline{xy} = \left(\sum_{i=1}^{n} x_i y_i\right) / n$,

és n az adatsorok hossza.

Ugyanez kétindexes adatkészletünk esetében úgy számolható, hogy mindkét indexben összegzünk, azaz $\sum_{i=1}^{n}$ helyett $\sum_{i=1}^{n} \sum_{j=1}^{m}$ írandó, ahol n és m a mátrix sorainak ill. oszlopainak száma, valamint a mennyiségek i indexe helyett ij indexpár használandó.

Ezek után

Már elvileg minden adott ahhoz, hogy egy macskák alakfelismerésére alkalmas programot írjunk.

Az alábbi a)--g) pontok szerinti lépéseket ajánlott követni, és a működő változatot main-a.c, main-b.c stb. néven elmenteni, mielőtt továbblépünk a következő pontra. Mivel az egyes pontok a korábbi bővítésével adódnak, csak az utolsó működő változatot kell beadni, azt is az előbbiek szerint nevezve el.

A program a következő parancssori argumentumokat olvassa be, pontosan ebben a sorrendben:

első_fájl_neve szélessége magassága második_fájl_neve szélessége magassága

Az adatfájlok megtalálhatóak a halnum.public/feladat2 könyvtárban.

A korrelációt számoló függvény tesztelésére használható kisméretű adatfájlok: smiley10x11.dat, sad10x11.dat, tree10x11.dat, smileybig24x19.dat.

Megtekintésre gif formátumban: smiley, sad, tree, smileybig

A feladathoz tartozó adatfájlok: Elveszett cica (lost-cat80x57-h.dat) , Hol van a mi macskánk? (cat-search640x360-h.dat).

a)

Írjunk függvényt, ami beolvas és az 1)-es pont szerint eltárol egy adatfájlt!

A program futtassa le a fentebb említett függvényt két megadott adatfájllal, és utána ellenőrzésül írassuk is ki az adatokat egy másik függvénnyel olymódon, hogy az adatokat sorokra vannak tördelve! De ez a kiírás csak akkor történjen meg, ha a kép méretei kisebbek egy határnál (pl. 20), hiszen a nagy mátrixok már amúgy is nehezen áttekinthetőek.

Tesztelésre használjuk ezeket a fájlokat: smiley10x11.dat, sad10x11.dat, tree10x11.dat, smiley10x11.dat

Megtekintésre mindegyik elérhető gif formátumban is. (A)

b)

Írjunk függvényt, ami kiszámolja a két megadott kép közötti korrelációt, megadva neki a két beolvasott tömböt, és azok méretét! Egyelőre feltételezhetjük, hogy a tömbök egyforma méretűek. A bevezetőben szereplő képleteket használva érdemes az összes összegzést egyetlen kettős ciklusban elvégezni. Írassuk is ki a korreláció értékét! De a kis képek esetére vonatkozó kiírás is maradjon meg!

Teszteljük programunkat a smiley—smiley, smiley—sad és smiley—tree párosításban! Ez úgy értendő, hogy a parancsargumentumokban e fájlok neveit és méreteit adjuk meg. Azt várjuk, hogy azonos képeknél 1, hasonló képeknél 1-hez közeli, a nagyon eltérőknél 0-hoz közeli értéket kapunk. (A)

c)

Bővítsük a korreláció függvényt úgy, hogy a két kép mérete különböző is lehessen! A **másodikként megadott kép** vízszintesen is és függőlegesen is nagyobb vagy egyenlő méretű lehet, mint az első. A korrelációszámolás a közös részen fusson le, a nagyobb kép kilógó részét figyelmen kívül hagyva! Gondoljuk meg, hogy ehhez a két kép méretei közül melyeket kell átadni a függvénynek! Ezt tesztelhetjük a smiley figurát tartalmazó megnövelt kép segítségével: a sad—smileybig és sad—smiley összehasonlításoknak azonos eredményt kell adniuk. (A)

d)

Bővítsük a programot olymódon, hogy kezelni tudjon egy egyszerű fejlécet az adatfájlban. Ez azt jelenti, hogy a beolvasás során figyelje, a beolvasott sor nem kezdődik-e '#' karakterrel vagy nem üres-e, mert ezek a fejléchez tartoznak. A '#' karakterrel kezdődő sorokat egyszerűen írja ki.

A d) feladatrésztől kezdve a programnak nem kell tudnia kezelni a fejléc *nélküli* adatfájlokat. A fájlok fejléces változataival ugyanúgy kell működnie, mint az c) részben kért számolásnak és a korábbi kiíratásoknak. Tehát ellenőrzésként, ha a kép méretei kisebbek egy határnál (pl. 20), akkor írassuk is ki a beolvasott adatokat! Teszteljük a programot pl. a smiley10x11-h.dat, sad10x11-h.dat fájlokkal! (T)

e)

Az egyik keresett cica képe a lost-cat80x57-h.dat fájlban található. Keressük meg legalább az egyik ilyen cicát a cat-search640x360-h.dat képen! (E képek megnézhetőek gif formátumban.)

Megnyitva az adatfájlokat, észrevehetjük, hogy van egy fejlécük, továbbá, hogy a számértékek nem pontosan olyan elrendezésben szerepelnek bennük, mint a korábban használt smiley, sad, stb. fájlokban. Gondoljuk át, hogy ez miatt át kell-e írni a programban a beolvasó függvényt?

A keresés előtt hagyjuk lefutni az eddigi programrészt, ami kiírja a képadatokat (kis kép esetén) és a korrelációt! Így nem kell külön beadni a d) résznek megfelelő programot.

A keresést úgy tehetjük meg, hogy a nagyobb képre képzeletben rárajzolunk a kicsivel azonos méretű téglalapot, és ezt a téglalapot a nagy képben néhány (javasolt:4-5) pixel lépésekben vízszintesen és függőlegesen is végigléptetjük. Minden pozícióban hasonlítsuk össze a kijelölt képrészletet a kis képpel! Ezt kétféleképp próbálhatjuk megvalósítani:

- Módosítjuk az c) pontban írt függvényt úgy, hogy megadhassunk neki egy eltolást is, és azt a számolásban figyelembe vesszük.
- Vagy pedig a függvény módosítása nélkül, a függvény meghívásakor a kiválaszott részlet pozíciójának megfelelően módosítjuk a függvénynek átadott pointert.

Semmiképp se pakoljuk át a kiválasztott rész értékeit egy másik tömbbe az összehasonlítás kedvéért, mert az időigényes!

Vigyázzunk, hogy csak olyan eltolást adjunk meg, hogy a függvény érvényes indexeket címezzen meg a két mátrixban!

Amennyiben a két adattömb közötti korreláció értéke egy határ feletti (javasolt határ: 0.4), akkor írjuk ki a téglalap pozícióját és a korreláció értékét! Reméljük, hogy ha az egyik cicát sikerül megtalálni a nagy képen és azután hazavisszük, a többi úgyis jön vele! (T)

f)

Pontosítsuk a keresést! Ehhez a d) pont szerinti léptetés során az 1. feladat minimumkereséséhez hasonlóan tároljuk el a korreláció maximumának pozícióját és értékét, a végén pedig írjuk ki azt! Utána az így talált maximum környékén már 1 pixeles lépéseket végezve keressük meg a maximumot pontosabban, és írjuk ki! (T)

g) Szorgalmi:

A e) pontban egy cicát találtunk, haza vittük, de mégse jött vele a többi. Azokat is meg kell keresni! A cicák a nagy képen lehetnek messzebb, mint a portrén, így kisebbnek látszhatnak. Ezért terjesszük ki a keresést olyan módon, hogy a korreláció függvényünkből csinálunk egy összehasonlító függvényt, aminek kicsinyítési arányt is meg lehet adni. A kicsinyítési arányt lépésenként (pl. 10%) csökkentve kereshetjük a többi cicát. Építsük be a függőleges tengelyre való tükrözés lehetőségét is!