Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Távközlési és Médiainformatikai Tanszék

Passzív és aktív képosztályozás a gépi és emberi tanulás összehasonlításánál

TDK dolgozat - 2015

Papp Dávid

Konzulens: Dr. Szűcs Gábor

Absztrakt

A gépi tanulás és az emberi tanulás összehasonlítása izgalmas téma, melynél párhuzamba állítható a két folyamat. Ezek vizsgálatánál a tanult képek számának függvényében végeztem a képosztályozás pontosságának mérését. A gépi tanuláshoz nagy mennyiségű címkézett tartalomra van szükség, míg az emberi tanulásnál elég néhány, vagy akár egyetlen mintakép egy séma elsajátításához. A két folyamat közti alapvető és egyben legnagyobb különbséget a priori információk eltérő mértéke képezi. Korunk egyik alapvető célja, hogy a gép által elérhető eredményeket maximalizáljuk, így rengeteg emberi erőforrás takarítható meg.

Gépi tanulás esetén, kevés címkézett képnél nagy jelentősége van annak, hogy ezek a tanulás szempontjából mennyire hasznosak. Az aktív tanulásnál lehetőség van a címkézetlen képekből kiválasztani azokat, melyek címkéit a legjobban szeretnénk megkapni, így az a lekérdezés egy iteratív tanulási folyamatot eredményez. A címkézetlen képek lekérdezésére számos stratégia létezik, melyeket intenzív kutatómunka övez és fejlesztés alatt állnak a mai napig.

Munkámnak két fő célja volt, egyszer az emberi és gépi teljesítő képességek összemérése, képi tartalmak osztályozását illetően. Ennek elvégzéséhez egy precíz mérési tervet dolgoztam ki, amely lehetővé teszi az összehasonlítást egyenlő feltételek mellett, amennyiben eltekintünk az emberek tapasztalati tudásától. A másik cél egy aktív tanulórendszer elkészítése volt, melyben egy általam javasolt lekérdezési stratégiát használok. Az eljárás alapját a bizonytalansági mintavételezés képezi, melyet kiegészítek egy eloszlás alapú címkevizsgálattal, így két dimenzió mentén végzem a kiválasztást.

A dolgozatban bemutatom a kidolgozott mérési tervet, az elkészített osztályozó rendszert, melyet elsőként passzív tanulással teszteltem, majd integráltam bele az aktív tanuló modult. Az emberi és gépi tanulás összehasonlítására 5 különböző képhalmazt használtam, melyeket manuálisan gyűjtöttem össze és címkéztem fel. Az eredményeket kiértékeltem és összemértem. Végül a passzív és aktív osztályozó rendszereim szélesebb körű vizsgálata érdekében egy nagyméretű képhalmazon végzett tesztet és annak eredményeit mutatom be.

Tartalomjegyzék

1 Bevezető

Manapság, a digitális világ növekedésének köszönhetően rengeteg fénykép található különböző adathordozókon, szerte a világban. Az interneten fellelhető képi tartalmak száma pedig megbecsülhetetlen, így korunk egyik alapvető problémája ennek a hatalmas adatmennyiségnek a rendszerezése. Ez rendkívül fontos számtalan alkalmazás számára, amelyek képekkel foglalkoznak, gondoljunk csak egy képkereső programra, vagy képnézegetőre. Ezen alkalmazások többféle módszert használnak a felhasználók segítése érdekében, hogy minél egyszerűbben találjanak számukra minél relevánsabb fényképeket, valamint több típusú részfeladatot kell megoldaniuk, mint például a képek osztályozása, rangsorolása, csoportosítása.

1.1 Szemantikus, egycímkés képosztályozás

Munkám során képi tartalmak osztályozásával foglalkoztam, amely nem más, mint képek kategóriába, vagy kategóriákba sorolása úgy, hogy az egyes kategóriákat előre meghatározzuk. A képosztályozás önmagában egy átfogó, tágabb feladatkört határoz meg, hiszen több olyan fontos attribútuma van, melyek részben vagy egészében változtatják meg a megvalósítás mikéntjét.

Fontos szempont, hogy a képeket mi alapján soroljuk az egyes kategóriákba, hiszen felhasználhatjuk a kép vizuális tartalmát, illetve annak metaadatait, mint például a neve, keletkezés dátuma vagy egy hosszabb leírás a képre vonatkozóan. Mivel nekem nem álltak rendelkezésemre metaadatok a képekről, így kizárólag azok tartalmi, más néven szemantikus információit használtam besorolási alapul. Ennek előnye, hogy egy képet semmi nem jellemez jobban, mint az, ami rajta látható, hátránya viszont az, hogy ezeket az információkat elő kell állítanunk különböző matematikai módszerek segítségével, melyek bonyolultsága nem elhanyagolható. Tartalom alapján, emberi szemmel a képeket szinte tévesztés nélkül lehet kategóriákba sorolni, azaz osztályozni, viszont ez nagyon időigényes és (emberi) erőforrás igényes feladat. Ezzel szemben, a számítógép kontextusában egy kép pixeleket, színeket, alakokat, textúrákat tartalmaz, melyekből meghatározni azt, ami egy ember számára egyértelmű egy képre ránézve, közel sem triviális feladat. Nem ritka, hogy több száz kategória közül kell eldöntenie egy képosztályozó algoritmusnak, hogy melyik az az egy vagy néhány, amelyikbe az

adott kép beleillik. Alapvető probléma az is, hogy hasonló szemantikus információval bíró képek is lehetnek nagyon különbözőek. Erre példaként tekintsük az 1. ábra képeit, ahol mindkét kép azonosan a „madár” kategóriába tartozik, viszont mégis jelentősen eltérnek egymástól, például színvilágukat vagy elrendezésüket tekintve. Tehát a szemantikus képosztályozásnak két fontos és nehéz folyamata van, az első a képek olyan módú reprezentálása, amely lehetővé teszi azok összehasonlítását, a második pedig a képek osztályozása ezen kinyert információk alapján. A képosztályozási eljárások által elért eredmények általában gyengék, még bonyolult, nagy futási időt igénylő algoritmusok használata esetén is, mivel a fentebb említett okokból adódóan nagyon összetettek és rengeteg a hibalehetőség. Körülhatárolt területeken viszont jó eredménnyel alkalmazhatóak, mint például az orvosi diagnosztika, vagy a karakterfelismerés.

1. ábra: Madaras képek különbözőségének szemléltetése

A dolgozatban kitérek az emberi és gépi tanulás különbségeire is, illetve össze is hasonlítom azok teljesítő képességeit, több különböző teszt alapján (lásd 5. fejezet). Az emberi tanulás eredményeinek méréshez egy hallgató társammal dolgoztunk ki egy közös mérési tervet, és az általa megvalósított, böngészőben futó web-alkalmazást használtuk, melyben a felhasználók különböző képhalmazokat tudtak osztályozni. A gépi tanulást teljesen egyedül dolgoztam ki és valósítottam meg, melyhez (mint általában minden gépi tanuláson alapuló eljáráshoz) szükség van egy tanuló és egy teszt halmazra. Továbbá mindkét halmazhoz szükség van a bennük található képek valódi kategória besorolásaira, más néven osztálycímkéire. Ez azért fontos, hogy a tanítás során visszacsatolást nyerjünk, illetve a tesztelés során az eredményeket mérni tudjuk. A valós osztálycímkék halmazát az angol elnevezésnek megfelelően *ground truth*-nak nevezzük.

Egy képosztályozási feladat szempontjából nem mindegy, hogy milyen a képek és a kategóriák közti viszony. Abban az esetben, ha egy képet határozottan egyetlen osztályhoz rendelünk, akkor egycímkés, amennyiben egy képet egyszerre több osztályhoz is rendelhetünk, többcímkés osztályozásról beszélünk. A 2. ábra bal oldali képén az egycímkés osztályozásra láthatunk egy példát, tehát a képet egyedül az „autó” kategóriába tesszük. A jobb oldali kép esetében pedig egyszerre kettő kategóriát is hozzárendelünk a képhez, tehát itt többcímkés osztályozást láthatunk. Én az egycímkés módszert valósítottam meg a képosztályozó algoritmusban, azaz c darab kategória esetén, minden képhez tartozik egy c dimenziós r vektor:

$$r_i \in \{0,1\}, ahol\ i = 1,2, \dots, c \qquad (1)$$

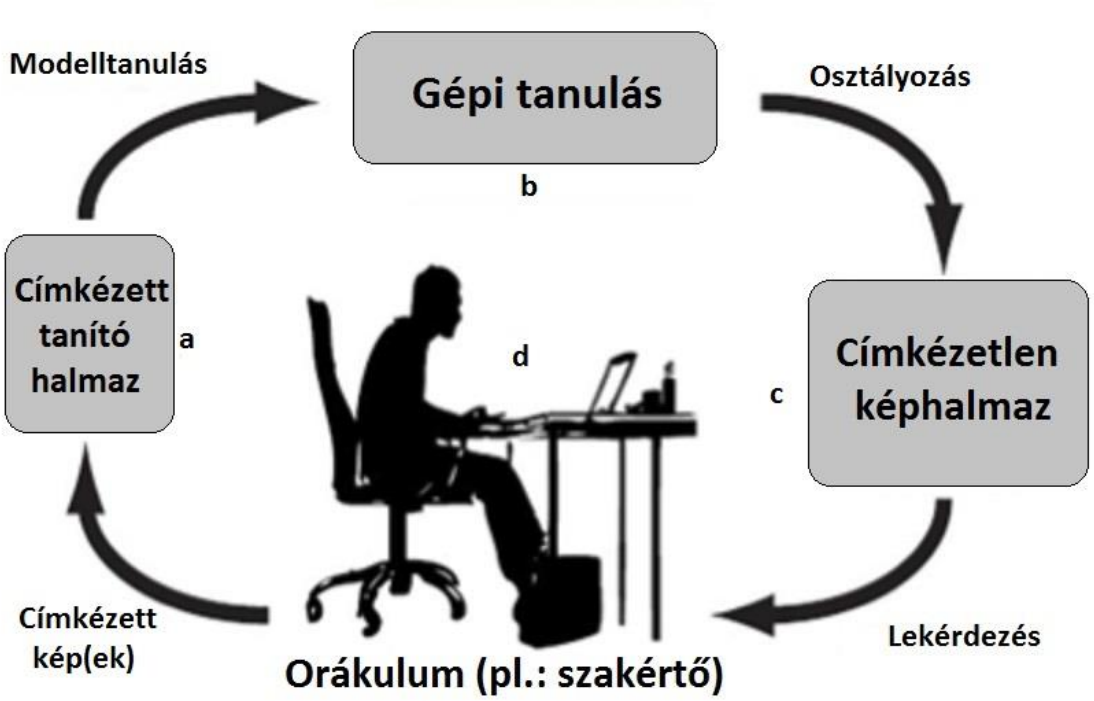
és r_i-nek csak egyetlen eleme lehet 1 értékű, ez jelöli, hogy melyik osztályba soroltuk az adott képet.

2. ábra: Bal oldalon az egycímkés, jobb oldalon a többcímkés osztályozás szemléltetése

1.2 Gépi tanulásos módszerek

A képosztályozáshoz szükségem van tehát egy tanuló és egy teszthalmazra, valamint azok ground turth információira. A 3. ábra segítségével bemutatom, hogyan valósítottam meg a gépi tanulást, amely lehetővé teszi a teszt képek osztályozását. Kiindulási állapotban, adott számomra egy címkézett tanító halmaz (a), valamint egy címkézetlen képhalmaz (c, teszthalmaz), továbbá rendelkezésemre áll egy gépi tanuló algoritmus (b) és egy lekérdezhető, úgynevezett *orákulum* (d). Első lépésben a tanító halmaz alapján egy modelltanulási folyamatot hajtok végre, a tanuló algoritmussal. Ezt követően, a címkézetlen képhalmazból egy képet kiválasztva, és azt az orákulumnak átadva, az megmondja a lekérdezett kép valós osztálycímkéjét, és kiegészíti azzal a

címkézett állományt, valamint törli a címkézetlen állományból. Így a következő lépésben egy frissített tanító képhalmaz áll rendelkezésemre, mely alapján újra tanítok. A lépéseket addig folytatom, míg minden képet fel nem címkéz az orákulum.



3. ábra: Képosztályozási folyamat körforgása

A fenti eljárás során, minden lépésben a tanulási folyamatot követően osztályozom azt a képet, amelyiket utána címkézésre elküldök, tehát az algoritmus futásának végére minden kép pontosan egyszer kerül tesztelésre. Minél „később” osztályozok egy képet, annál több tanító képből tanul a rendszer. Fontos kérdés, hogy a lépéseknél mikor melyik képet válasszam ki tesztelésre, majd címkézésre, és itt a hangsúly a címkézésen van. Megkülönböztetünk *passzív* és *aktív tanulási* módszereket, melyek a kiválasztás sorrendjének felállítási módjában különböznek. Passzív tanulás esetén egy véletlenszerű, statikus sorrendet határozunk meg, és minden lépésben e sorrend szerint haladva osztályozunk, majd címkézünk. Tehát tulajdonképpen ez az általános módszer, mikor a tanításhoz kiválasztott képek sorrendje számunkra érdektelen, szóval nem is végzünk „valós kiválasztást”.

Az *aktív tanulás* [6] egy speciális esete a félig felügyelt gépi tanulásnak, mely során a tanulórendszernek lehetősége van a tanuló adathalmaz egyes mintáit kiválasztani, vagyis a kiválasztást meghatározó sorrend fontos. Aktív tanulási módszereknél, az adatok elérhetősége szempontjából megkülönböztetünk lekérdezés

szintézist, szelektív mintavételt, valamint halmaz alapú mintavételt, melyek abban térnek el egymástól, hogy milyen adatok állnak rendelkezésre és a tanulórendszer azokhoz hogyan tud hozzáférni. Az első esetben a tanulórendszer állít elő egy mintát, amelyre címkézést kér, a második módszernél a címkézetlen adatok folyamatosan érkeznek a tanulórendszerhez, és azt kell eldöntenie, hogy ezek közül melyek címkéit kéri, illetve melyeket hagy figyelmen kívül. Ezek közül a halmaz alapú mintavétel az, amelyik osztályozási feladatokba hatékonyan beépíthető, így én is ezt választottam, ehhez szükség van a képhalmazok ground truth információira, és a tanulórendszer maga választhatja ki a képhalmazból, hogy melyik kép címkéjét kéri. Az elvárás az aktív tanulás használatával szemben az, hogy a tanulórendszer gyorsabban, vagyis kevesebb tanuló adat felhasználásával legyen képes megtanulni egy modellt, mint hagyományos véletlen mintavételezéssel, azaz passzív tanulással. Ezért rengeteg alkalmazási területtel is rendelkezik, mint például a dokumentumosztályozás, webes keresés, email szűrés, rákkutatás, betegségek szimptómáinak meghatározása.

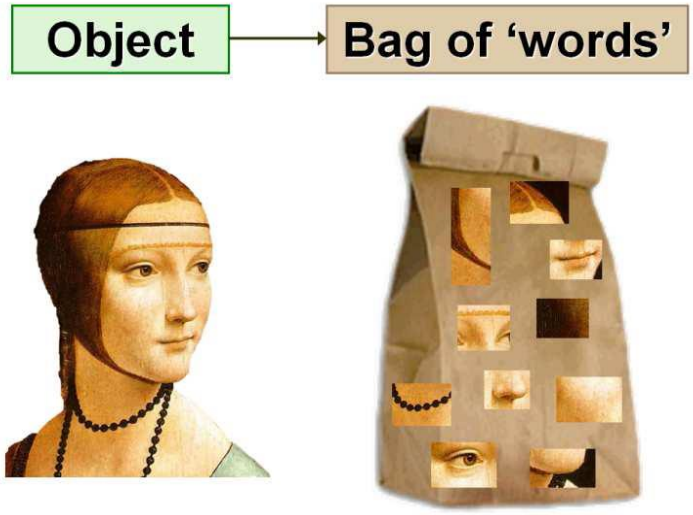
1.3 Dolgozat felépítése

A dolgozat célja, hogy összehasonlítsam az emberi és gépi tanulást a tartalom alapú képosztályozás témakörében. A második fejezetben bemutatom, hogyan történik a képek matematikai, tartalom alapú reprezentálása. A harmadik fejezetben a passzív tanulással történő osztályozást fejtem ki, melyhez a képekből kinyert információkat használom. Ezt követően az aktív tanulási módszereket tárgyalom, és az általam javasolt lekérdezési stratégiát fejtem ki. Az ötödik fejezetben az elkészített rendszerek teszteléséről, az emberi és gépi tanulás különbségeiről, valamint az elért eredményekről lesz szó. A hatodik fejezetben a passzív és aktív tanuló módszereket mérem össze a Caltech101 képhalmazon [25], valamint a PASCAL VOC2010 tanító és validáló képein [15], végül pedig összefoglalom munkámat.

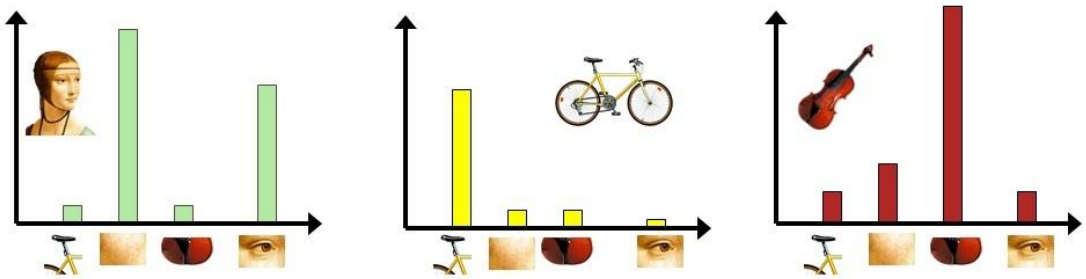
2 Képek reprezentálása

2.1 Vizuális kódszavak

Az eljárásom alapját egy általános, hasonló feladatokban gyakran használt módszer képezi, ami nem más, mint a szózsák modell (*Bag of Words*) képeken alkalmazott változata [26][32]. Az elképzelés lényege, hogy egy képet a rajta szereplő vizuális elemek, más néven *vizuális kódszavak* összességével reprezentálunk, és ezek térbeli elhelyezkedésétől eltekintünk (ezt szemlélteti a 4. ábra). Tehát egy képet csupán a vizuális kódszavak eloszlásával jellemzünk, és ebből próbálunk meg következtetni a szemantikus tartalmára (lásd 5. ábra).



4. ábra: BoW modell (forrás: [26])



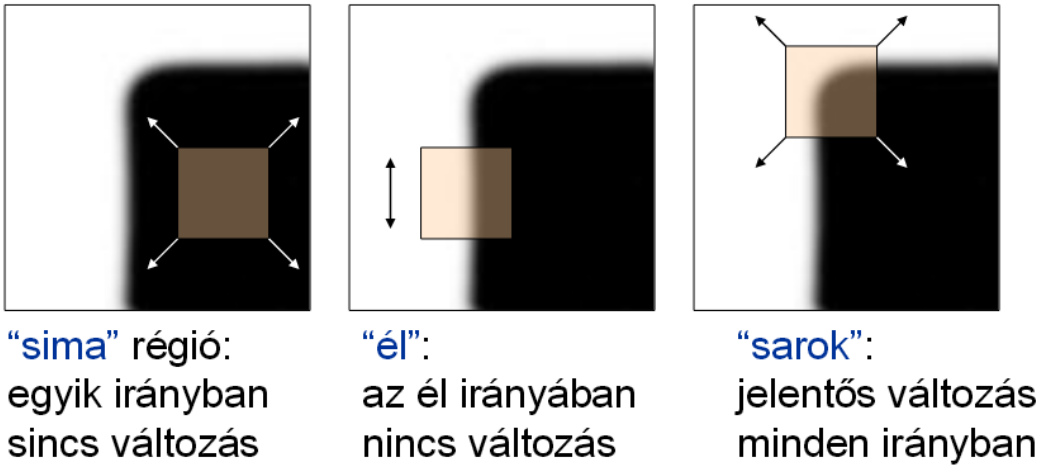
5. ábra: Vizuális kódszavak eloszlása (forrás: [26])

A vizuális kódszavak meghatározásához úgynevezett *alacsony szintű leíró*kra (röviden: leíró) van szükségünk, mely a kép egy adott pontjának környezetében előforduló lokális tulajdonságokat foglalja magában. Ezek lehetnek színek, textúrák,

alakok és még sok más. Az algoritmus első feladata tehát, hogy meghatározza azokat a pontokat, melyek érdemesek leírók számítására, más néven a *kulcspontok*at.

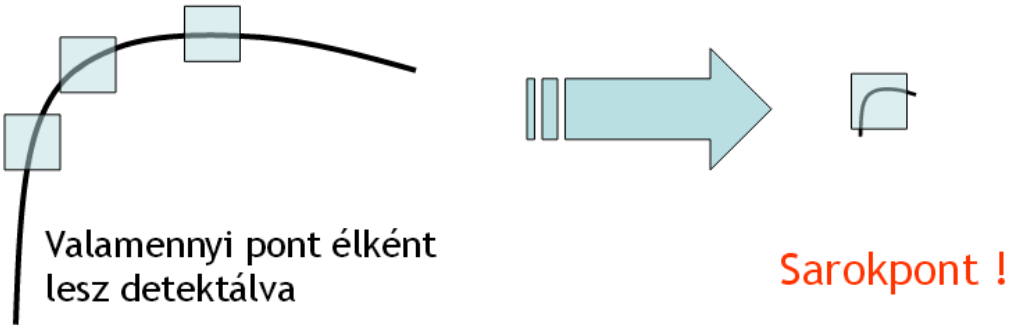
2.1.1 Kulcspontok meghatározása

A kulcspontok meghatározásához a Harris-Laplace detektort használtam, melynek alapja a Harris detektor, vagy más néven kombinált sarok és él detektálás [8][24]. Az eljárás lényege, hogy egy képen azonosítja az éleket, ahol a kép pontjainak intenzitásértékeiben nagy változás következik be. Két él találkozásánál (sarokpontnál) pedig az intenzitásváltozás, azaz a derivált, két irányban is nagy abszolút értékű lesz. Ennek meghatározására egy csúszó ablakot vizsgál, ezt szemlélteti a 6. ábra.



6. ábra: Sarokpont detektálás csúszó ablakot vizsgálva (forrás: [24])

Mivel minden számításhoz deriváltakat használ, ezért azok invariánsak az eltolásra és az intenzitás változására, viszont csak részben invariánsak a skálázásra. A 7. ábra bal oldalán látható, hogy egy vonal mentén éleket detektál az algoritmus, viszont egy másik skálázásban ugyanaz a vonal sarokként lesz detektálva, mivel a vizsgált ablakba belefér az egész.

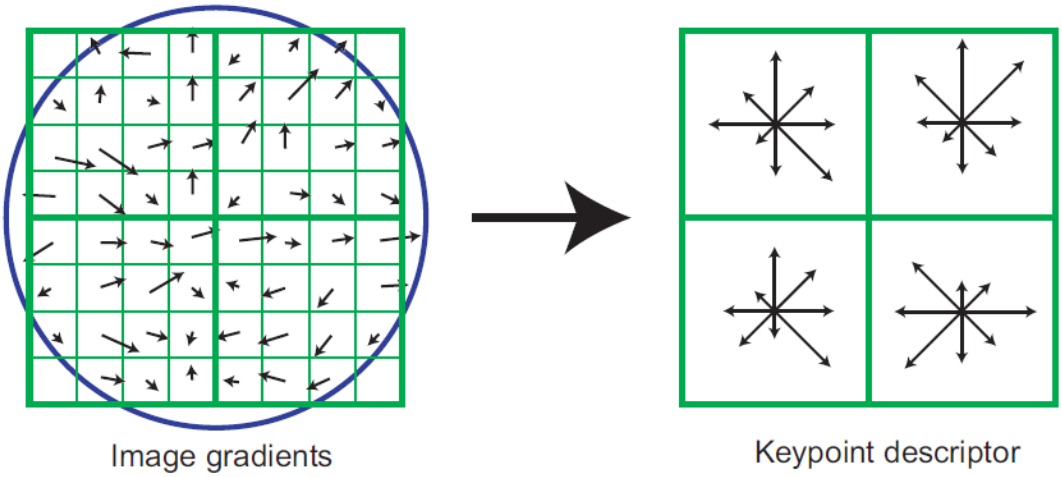


7. ábra: Harris sarokdetektor nem invariáns a skálázásra (forrás: [24])

A Harris detektort K. Mikolajczyk és C. Schmid fejlesztette tovább úgy, hogy az invariáns legyen a skálázásra is, ezt nevezik Harris-Laplace detektornak [28]. Minden képre lefuttatom a detektort, majd a kapott kulcspontokból minden kép mellé elkészül egy lista, amely tartalmazza azokat a fontos pontokat, ahol leíró vektort akarok számítani.

2.1.2 A SIFT leíró

A második főbb lépés, hogy minden kulcspontot SIFT (Scale Invariant Feature Transform) leíróval jellemzek, melyet David G. Lowe fejlesztett ki [27]. Ez tulajdonképpen egy 128 dimenziós vektor, amely egy pont lokális környezetében lévő gradiensek orientációjából számolható. A 8. ábra alapján részletesen bemutatom a SIFT működését, amelyet a kulcspontok jellemzésére használtam.



8. ábra: SIFT leíró működése (forrás: [27])

Elsőnek a gradiens irányát és nagyságát számoljuk ki minden egyes mintavételi pontban. A mintavételi pontok meghatározásához $n \times n$ cellát jelölünk ki egy fontos pont körül. Minden egyes cellát további $k \times k$ részre osztunk. Ez alkotja a mintavételi pont körüli vizsgált környezetet, ahogy az az ábra bal oldalán is látható. A számított gradiensek nagyságát egy Gauss-függvény szerint súlyozzuk, majd a súlyozott gradiensekből cellánként készítünk egy-egy orientáció szöghisztogramot, d irányra. Ennek eredménye látható az ábra jobb oldalán. Az ábrán $n = 2, k = 4, d = 8$. A SIFT standard paraméterezése a következő: $n = 4, k = 4, d = 8$. Végül a hisztogramok egymásutánjából kapunk egy $n^2 \times d$ dimenziós vektort. Ez az alapbeállítások mellett 128 dimenziót jelent, ahogy én is használtam.

A módszer invariáns a fényviszonyokra, amit a vektor normalizálásával érhetünk el, mivel ha egy kép pixeleit szorozzuk egy konstanssal, a gradiensek nagyságában fellépő változást a normalizálás semlegesíti. A fényerő változások esetén egy konstanst adunk a pixelekhez, viszont ez nem befolyásolja a gradienseket, mivel azt a pixelek különbségeiből számítjuk. A nem-lineáris megvilágítási változásokkal már nagyobb probléma van, mivel nem lehet egységesen kezelni a gradiensek változását. Ezek nagy különbségeket tudnak okozni a gradiensek nagyságában, viszont az irányukat nem változtatják. Tehát ennek a befolyását úgy tudjuk csökkenteni, ha meghatározunk egy küszöb értéket, amellyel korrigáljuk a gradiensek nagyságát, így az nem kap akkora szerepet, mint az orientációja. Ehhez a leíró vektor minden 0,2-nél nagyobb elemét 0,2-re cseréljük, majd újra normalizálunk. Ezt a 0,2-es értéket Lowe javasolta [27], miután kikísérletezte azt ugyanazon képek megvilágításbeli változtatgatásával.

A fent ismertetett eljárással tehát minden kép minden kulcspontjában kiszámítom a SIFT leírókat. Az egymáshoz hasonló leírók jelentenek egy vizuális kódszót. Ezek meghatározásához klaszterezem az összes képből kinyert SIFT leírót a GMM (Gaussian Mixture Model) módszer segítségével [30].

2.1.3 Gaussian mixture model

A GMM egy generatív módszer az alacsony leírók klaszterezésére, tehát az egyes klaszterekhez egy-egy valószínűségi modellt rendel [30]. Ebben az esetben egy pont tartozhat több klaszterbe is, bizonyos valószínűséggel. A pontok valószínűségi sűrűségfüggvényét K darab Gauss-féle függvény súlyozott összegével írja le:

$$p(X|\lambda) = \sum_{j=1}^{K} \omega_j g\left(X|\mu_j, \sigma_j\right) \quad (2)$$

Itt az $X = \{x_1, x_2, \dots, x_N\}$ jelöli az M dimenziós adatvektorokat (SIFT leírók), λ a keresendő paramétert, $g(X|\mu_j, \sigma_j)$ a Gauss-féle függvényeket (ahol μ_j a várható értéket, σ_j a szórást jelöli), valamint ω_j pedig a súlyozó együtthatókat, amelyekre a következő megkötésnek kell teljesülnie:

$$\sum_{j=1}^{K} \omega_j = 1 \quad (3)$$

A GMM λ paraméterét ML (Maximum Likelihood) becsléssel határozzuk meg. Ennek célja egy olyan λ paraméter jóslása, amely maximalizálja a valószínűségét annak, hogy a megadott ponthalmaz keletkezett. Az x_i adatvektorok közt függetlenséget feltételezve a 2. egyenlet a következő módon alakul:

$$p(X|\lambda) = \prod_{i=1}^{N} p(x_i|\lambda) \quad (4)$$

Ez a kifejezés nem-lineáris a λ paraméterre nézve, tehát a direkt maximalizálása nem lehetséges. Erre egy speciális iteratív módszert szokás használni, az EM (Expectation Maximization) algoritmust [9][14]. Ennek lényege, hogy induljunk ki egy kezdeti λ paraméterből, majd ebből becsüljünk minden lépésben egy újat (λ'), amire $p(X|\lambda') \geq p(X|\lambda)$ teljesül. Ez után az új paraméter lesz a következő iteráció kezdeti paramétere, és így tovább. Egészen addig folytatjuk az iterációt, amíg konvergenciát nem tapasztalunk vagy elértük a maximálisan megengedett iterációk számát. A kezdeti paraméter meghatározásához pedig K-means klaszterezést használtam [34].

A GMM eredményeként megkapom a vizuális kódszavakat (az egyes Gauss függvények), melyekre tekinthetünk úgy, mint a képhalmaz tömör reprezentálására. A célom az, hogy minden képet jellemezni tudjak a tartalma alapján, méghozzá olyan módon, hogy azok összehasonlítása, megkülönböztetése, osztályozása lehetővé váljon. Így mindenképpen szükségem van egy olyan magasabb szintű képi leíró megalkotására, amely nem csak egy kulcspontot jellemez, hanem egy teljes képet. Erre a feladatra a Fisher-vektor használom.

2.2 Fisher-vektor

A Fisher-vektort [20] a képfeldolgozási témakörök közül a képosztályozásban a legelterjedtebb [16], így az én algoritmusomba tökéletesen beleillik. Ez egy rendkívül komplex leíró, amely arra használható, hogy egy teljes képet jellemezzünk vele egyetlen

vektor formájában. Kiszámításához szükség van a SIFT leírókra, illetve a vizuális kódszavakra, tehát a GMM-re. Valójában azt fogja ez megmondani, hogy egy kép milyen eloszlásban tartalmaz vizuális elemeket, ahol ezek a vizuális elemek a képhalmaz egészéből kerülnek ki. Ez tehát egy egységes és egyértelmű reprezentációja lesz az képeknek. A következőkben pontosan megadom, hogyan épül fel egy ilyen Fisher-vektor.

A GMM tárgyalásánál bevezetett jelöléseknek megfelelően legyen $p(X|\lambda)$ a valószínűségi sűrűségfüggvény, amelynek λ a paramétere ($\lambda = \{\omega_j, \mu_j, \sigma_j | j = 1 \dots K\}$), legyenek $X = \{x_1, x_2, \dots, x_N\}$ az M dimenziós adatvektorok, melyek itt nem az összes SIFT leírót jelentik, csak azokat, melyek egy adott képre vonatkoznak. A sűrűségfüggvény gradiense, azaz a logaritmusának deriváltja megadja, hogyan írja le a legjobban a modell az adatvektorokat, tehát a képet, így tulajdonképpen ez a mennyiség a Fisher-vektor:

$$\nabla_\lambda \log p(X|\lambda) \tag{5}$$

A kiszámítását a következő néhány lépésben mutatom be. A továbbiakban vezessük be az $L(X|\lambda) = \log p(X|\lambda)$ jelölést. Amennyiben feltesszük, hogy az adatvektorok egymástól kölcsönösen függetlenek, akkor $L(X|\lambda)$ felírható a következőképpen:

$$L(X|\lambda) = \sum_{i=1}^{N} \log p(x_i|\lambda) \tag{6}$$

Ez azért lesz igaz, mert a logaritmusok összege a szorzatok logaritmusával egyenlő, tehát az 4. egyenletben használt produktum felcserélhető summára. Annak a valószínűsége, hogy az x_i adatvektort a GMM generálta a következő:

$$p(x_i|\lambda) = \sum_{j=1}^{K} \omega_j g(x_i|\mu_j, \sigma_j) \tag{7}$$

Továbbá jelölje $\gamma_i(j)$ annak a valószínűségét, hogy az x_i adatvektort a j-edik Gauss-függvény generálta:

$$\gamma_i(j) = p(j|x_i, \lambda) = \frac{\omega_j p_j(x_i|\lambda)}{\sum_{jj=1}^{K} \omega_{jj} p_{jj}(x_i|\lambda)} \tag{8}$$

A súlyozási együtthatókra pedig most is érvényes a 3. egyenlet által leírt megkötés. Ezzel felbontottuk az $L(X|\lambda)$-et és kifejeztük a Gauss-függvények súlya, várható értéke és szórása alapján, melyek értékét tudjuk a GMM-ből. Az utolsó lépés, hogy deriváljuk az $L(X|\lambda)$ függvényt, méghozzá rendre a λ paraméter elemei szerint

egyenként. A következő egyenletekben megadom, hogyan írhatóak fel az egyes deriváltak:

$$\frac{\partial L(X|\lambda)}{\partial \omega_j} = \sum_{i=1}^{N} \left[\frac{\gamma_i(j)}{\omega_j} - \frac{\gamma_i(1)}{\omega_1} \right] \quad (9)$$

$$\frac{\partial L(X|\lambda)}{\partial \mu_j^m} = \sum_{i=1}^{N} \gamma_i(j) \left[\frac{x_i^m - \mu_j^m}{(\sigma_j^m)^2} \right] \quad (10)$$

$$\frac{\partial L(X|\lambda)}{\partial \sigma_j^m} = \sum_{i=1}^{N} \gamma_i(j) \left[\frac{(x_i^m - \mu_j^m)^2}{(\sigma_j^m)^3} - \frac{1}{\sigma_j^m} \right] \quad (11)$$

Itt $j = 1, 2, \dots, K$ jelöli a megfelelő Gauss-függvényt, és $m = 1, 2, \dots, M$ jelöli az adott dimenziót. Ezek után a gradiens vektorok dimenziónkénti eltérő nagysága miatt még egy további normalizálás [16][23] is szükséges, melynek menetét nem részletezem. A Fisher-vektor tehát a következők szerint alakul ki:

- Az $L(X|\lambda)$-et deriváljuk egyenként a K darab Gauss-függvény súlyai szerint. Mivel a 3. egyenlet miatt, például ω_1 kiszámítható a többi súlyozási együttható ismeretében, ezért csak $K - 1$ szabad paraméter van. Tehát ez $K - 1$ elemet ad a Fisher-vektorba.
- Az $L(X|\lambda)$-et deriváljuk egyenként a K darab Gauss-függvény várható értékei szerint. Mivel ezek M dimenziósak, ezért ez $K \times M$ elemet fog adni a Fisher-vektorba.
- Az $L(X|\lambda)$-et deriváljuk egyenként a K darab Gauss-függvény szórásai szerint. Ez szintén $K \times M$ elem.

Ebből következik, hogy a Fisher-vektor összesen $K(2M + 1) - 1$ dimenziós. Az én implementációmban $K = 256$, mivel ennyi Gauss-függvényt definiálok (tesztek után ez bizonyult átlagosan a legjobbnak), $M = 128$, mivel a SIFT leírók 128 dimenziósak. Ez azt jelenti, hogy $K(2M + 1) - 1 = 65791$ az én esetemben, tehát egy Fisher-vektor 65791 dimenziós. Ez tehát egyértelműen meghatároz egy adott képet, így a továbbiakban ezeket a rendkívül magas dimenziójú vektorokat fogom felhasználni az osztályozáshoz.

3 Képosztályozás passzív tanulással

Ahogy azt a bevezetésben már említettem, gépi tanulásos módszerrel dolgozok. Ebben a fejezetben a gépi tanuláson belül a passzív tanulás menetét fogom bemutatni, pontosabban, a képosztályozás folyamatát, mely során passzívan tanul az algoritmus. Jelölje T a teljes képhalmazt, amelyet osztályozni szeretnék, és jelölje S azt a kiindulási képhalmazt, mely címkéi a tanulórendszer számára már ismertek. A célunk, hogy T minden elemére adjunk egy jóslatot a kategóriájára vonatkozóan, ehhez dinamikusan fogom kezelni mind a T, mind az S halmazokat. Kezdetben a tanulóhalmazunk S, és a teszthalmazunk T, majd a T-ből kiválasztunk és ki is törlünk egy képet és azzal bővítjük az S halmazt, így a rendszer tudása egyre növekszik, és várhatóan pontosabb becslést tud majd adni a következő kép(ek)re. Fontos megjegyezni, hogy ilyenkor a képpel együtt annak valódi osztálycímkéjét is átadom a tanulórendszernek. Passzív tanulás esetén, az S bővítése nem szabályozott, hiszen egy véletlenszerűen generált, statikus sorrend alapján kerülnek át az egykori tesztképek a tanulóhalmazba.

Kifejtettem, hogyan lehet passzív tanulás használatával egy adott képhalmazt több lépésben osztályozni, viszont arról még nem beszéltem, hogy maga az osztályozás egy lépésen belül, hogyan történik. A következő alfejezetekben ennek menetét részletesen áttekintem.

3.1 Bináris lineáris osztályozás

Adottak a mintahalmaz képein kiszámolt magas szintű leírók (Fisher-vektorok) $\vartheta_1, \vartheta_2, \dots, \vartheta_N$, valamint képenként a c dimenziós r vektor (lásd 1.1 rész). Egy teszt képet a magas szintű leírója (ϑ) alapján osztályozok úgy, hogy minden c kategóriához meghatározok egy $f(\vartheta)$ értéket, amely azt fogja megadni, hogy az adott kategóriájú objektum mekkora bizonyossággal szerepel a képen. Ezt úgy teszem meg, hogy létrehozok c darab egymástól független bináris osztályozót, melyekkel külön-külön elvégzem a tanítást és az osztályozást. Minden tanításnál a kiválasztott l sorszámú osztályba tartozó képek a pozitív minták, az összes többi pedig negatív minta. Bináris osztályozónként meghatározok minden tanító képhez egy y_i változót a következő szerint:

$$y_i = \begin{cases} 1, & ha\ r_l = 1 \\ -1, & ha\ r_l = 0 \end{cases} \tag{12}$$

A létrehozott y_i változó adja meg, hogy az adott kép pozitív vagy negatív minta a tanítóhalmazban. Ezt „one-against-all" módszernek nevezzük, és egyszerűsége ellenére rendkívül jól használható, ezért én is ezt használom. Bináris osztályozóként az SVM (*Support Vector Machine*) egyik változatát használom amelyet C-SVC osztályozónak neveznek. Mielőtt rögtön rátérnék ennek az eljárásnak az ismertetésére, néhány fontos dolgot tisztázni kell, hogy érthető és áttekinthető legyen az általam használt módszer is.

Bináris osztályozásról akkor beszélünk, ha a választható kategóriák száma kettő, ezen felül az osztályozó lineáris, ha egyetlen hipersíkkal el tudja különíteni a két osztály pontjait. Bináris lineáris esetben tehát a tanítópontokat két részre szeparálja egy hipersík. Formálisan, az $n \times \vartheta + B = 0$ egyenletű hipersík lineárisan szeparálja a tanítópontokat, ha:

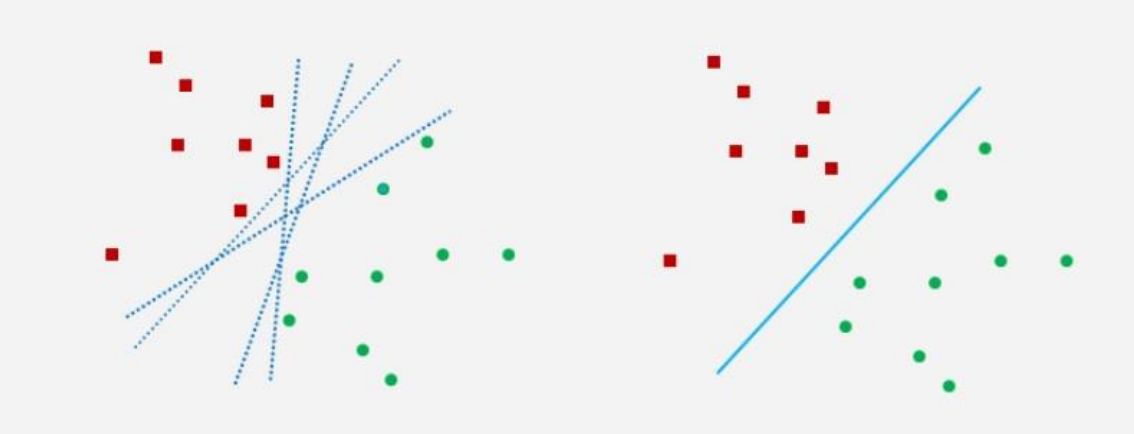
$$\begin{matrix} n \times \vartheta_i + B \geq a > 0,\ ha\ y_i = 1 \\ n \times \vartheta_i + B \leq -a < 0,\ ha\ y_i = -1 \end{matrix} \qquad i = 1 \dots N \tag{13}$$

Egy új pont a térben (teszt kép) a hipersíktól vett előjeles távolsága alapján sorolható az egyik vagy a másik osztályba. Ezt a mennyiséget a pont és a hipersík normálvektorának skaláris szorzata adja, tehát egy ϑ vektorra adott $f(\vartheta)$ jóslat a következőképpen írható fel:

$$f(\vartheta) = g(n * \vartheta) = g\left(\sum_{m=1}^{M} n_m \vartheta_m\right) \tag{14}$$

Itt M jelöli a dimenziók számát, a g függvény pedig a megfelelő értékkészletbe képezi az osztályozó kimenetét. Mivel az n vektorral tulajdonképpen súlyozzuk a ϑ vektort, ezért ezt súlyvektornak is nevezzük.

3.2 Szupport vektor gépek

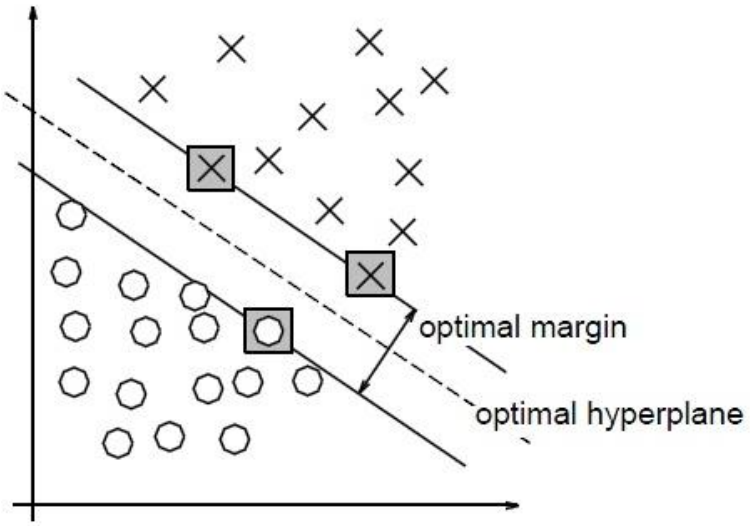


9. ábra: Egy lineárisan szeparálható feladat különböző megoldásai (forrás: [2])

A B. Boser, I. Guyon és V. Vapnik által javasolt SVM [5] alapváltozata lineáris szeparálásra képes, de egy kiterjesztett változata nemlineáris szeparálásra is lehetőséget nyújt. Lineárisan szeparálható esetben végtelen sok lineárisan szeparáló hipersík létezik, amelyek mindegyike hibátlanul szétválasztja a tanítópontokat. A szeparálás minősége azonban az egyes megoldások esetén jelentősen eltérő lehet (lásd 9. ábra). Az az elválasztás eredményez jobb általánosító képességet, amely a két osztályba tartozó tanítópontok között, a tanítópontoktól a lehető legnagyobb távolságra helyezkedik el. A lineáris bináris osztályozási feladat megoldását adó SVM ezt az „optimális” szeparátor síkot határozza meg. A pontok közt középre elhelyezett szeparáló felületet a tanítópontoktól egy margó, azaz egy biztonsági sáv választja el.

3.2.1 Maximális margójú szeparálás

Ez a módszer (a standard SVM) a lineárisan szeparálható feladatok lineáris megoldását adja, amihez a szeparáló hipersíkok közül a maximális margójút választja, ahogy a 10. ábra is szemlélteti, így biztosítja a fentebb leírt tulajdonságot.



10. ábra: Az optimális hipersík margója (forrás: [5])

A hipersík paramétereinek skálázhatósága lehetővé teszi, hogy felírjuk a margón elhelyezkedő tanítópontok és a hipersík közti távolságra a következőket:

$$\begin{aligned} n \times \vartheta_i + B \geq 1,\ ha\ y_i = 1 \\ n \times \vartheta_i + B \leq -1,\ ha\ y_i = -1 \end{aligned} \tag{15}$$

Ekkor a szeparáló hipersíkhoz legközelebb eső, különböző osztályba tartozó bemeneti vektorok közötti, a síkra merőlegesen mért távolság a következő:

$$\frac{1}{\|n\|} - \frac{-1}{\|n\|} = \frac{2}{\|n\|} \tag{16}$$

ahol a $||\ ||$ operátor az euklideszi normázás műveletét jelöli. Az optimális hipersík által biztosított margó tehát: $\frac{1}{\|n\|}$. Ebből az következik, hogy az optimális hipersík az, amelyikre $\|n\|$ minimális, az alábbi feltétel mellett:

$$y_i(n \times \vartheta_i + B) \geq 1 \qquad i = 1 \ldots N \tag{17}$$

A fentiek alapján tehát egy (n^{opt}, B^{opt}) párost keresünk, amire teljesül minden kritérium. Ez egy feltételes szélsőérték-keresési probléma, amelynek megoldását egy Lagrange kritérium megoldásával kereshetjük. Az ebből következő duális probléma egy kvadratikus programozási feladathoz vezet [2][10]. Az ide vágó levezetésektől eltekintek, egyből a probléma megoldása utáni részre ugrok.

Az optimumhoz tartozó Lagrange multiplikátorok (α^*-k) nagy része általában 0, így mind a súlyvektor kifejezésében, mind az SVM válaszában a tanítópontoknak csak egy része (N helyett N_S) vesz részt. Azokat a tanítópontokat, amelyek részt vesznek a

megoldás kialakításában, szupport vektoroknak nevezzük (innen ered az elnevezés is). Ezért az SVM-ben a tér tényleges dimenziója nem a tanítópontok számával, hanem a szupport vektorok számával egyezik meg. Ez jelentős egyszerűsítést jelent a válasz számításában, különösen abban az esetben, ha $N_S \ll N$. Az SVM-nek ezt a tulajdonságát ritkasági tényezőnek nevezzük. Továbbá az is igaz, hogy ahol $\alpha_i^* > 0$, ott $y_i(n^{opt} \times \vartheta_i + B^{opt}) = 1$, vagyis a szupport vektorok a hipersíkhoz legközelebbi tanítópontok, amelyek pontosan a margó határán helyezkednek el.

Az n^{opt} tehát a szupport vektorok és a Lagrange együtthatók által meghatározható:

$$n^{opt} = \sum_{i \in N_S} \alpha_i^* \, y_i \vartheta_i \tag{18}$$

Az optimális eltolás értéke (bias) B^{opt} az $y_i(n^{opt} \times \vartheta_i + B^{opt}) = 1$ egyenletből számolható, ahol ϑ_i egy szupport vektor. Végül, az SVM válasza a következő formában írható fel:

$$f(\vartheta) = \sum_{i \in N_S} \alpha_i^* y_i \vartheta_i \times \vartheta + B^{opt} \tag{19}$$

3.2.2 A szoft margójú szeparálás

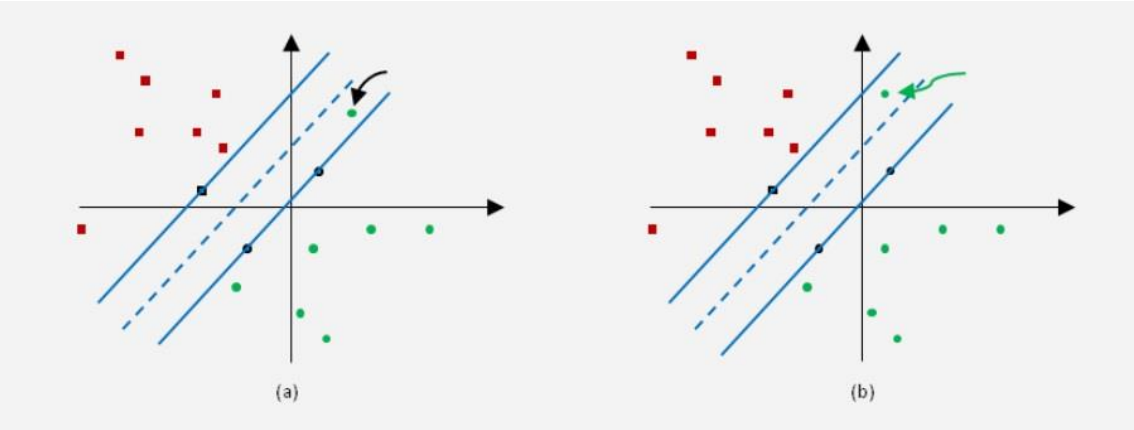
A gyakorlatban ritkán adódik olyan feladat, amely lineárisan szeparálható úgy, hogy még osztályozási tartalék is biztosítható. Általában a hibamentes lineáris szeparálás nem lehetséges, azonban néhány mintapont esetén megengedhető, hogy a margón belülre kerüljenek. Ezt lágy vagy szoft margójú megoldásnak nevezzük. A módszer lineárisan nem szeparálható feladatok lineáris megoldását teszi lehetővé.

A 17. egyenlőtlenségben leírt feltétel ebben az esetben nem állja meg a helyét minden i-re, így egy új feltétel szükséges, ahol úgynevezett gyengítő változókat használunk, hogy a 17. egyenlőtlenségnek nem megfelelő pontokat „büntessük”:

$$y_i(n \times \vartheta_i + B) \geq 1 - \xi_i \qquad i = 1 \ldots N \tag{20}$$

A fenti egyenlőtlenségben ξ_i a gyengítő változó, amelyre igazak a következő megállapítások:

- $\xi_i = 0$ esetén visszakapjuk a 17. egyenlőtlenséget
- $0 < \xi_i \leq 1$ esetben tekintsük a 11. ábra (a) részét
- $\xi_i > 1$ esetre pedig a 11. ábra (b) részén láthatunk példát



11. ábra: (a): mintapont osztályozása helyes, de a mintapont a margóra, vagy a margó és a hipersík közé esik. (b): a mintapont osztályozása helytelen (forrás: [2])

Az optimális hipersíkot a maximális margójú esethez hasonlóan határozzuk meg, annyi módosítással, hogy a hibás osztályozások száma minimális legyen, miközben továbbra is törekszünk a lehető legnagyobb margó elérésére. Ennek megfelelően az $\|n\|$ helyett a következő kifejezést kell minimalizálni:

$$J(n) = \frac{1}{2}n^2 + C(\sum_{i=1}^{N} \xi_i) \qquad (21)$$

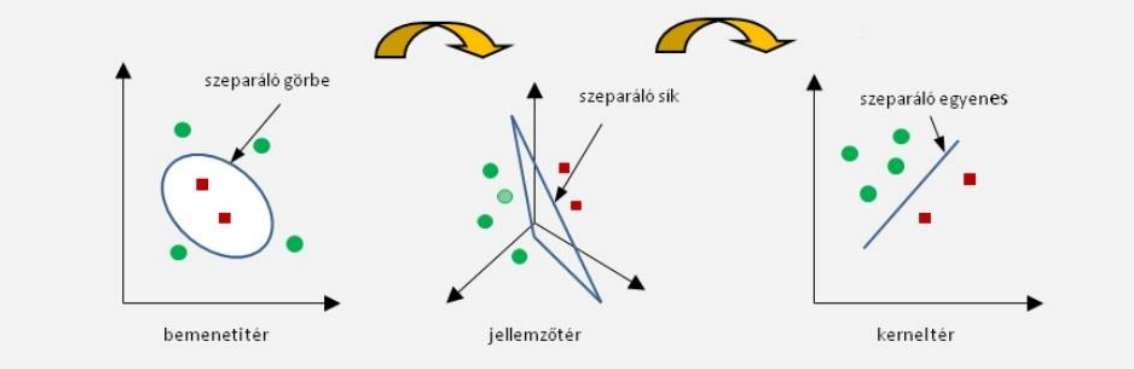
Itt a C paraméter a két tag közti kompromisszumot állítja be. A C-t hiperparaméternek is szokás nevezni, és általában kereszt-validációs módszerrel határozhatjuk meg.

A megoldást most is a megfelelő Lagrange kritérium felírása adja. A gyengítő változók bevezetése az optimalizálási feladat duális alakját csak a Lagrange multiplikátorokra vonatkozó feltételekben módosítja. Az α^*-k egy része most is 0, tehát ugyanúgy lesznek szupport vektorok, és egyéb, a megoldást nem befolyásoló mintapontok. Az osztályozó válasza megegyezik a 19. egyenletben felírtakkal, viszont ebben az esetben a B^{opt} az $y_i(n^{opt} \times \vartheta_i + B^{opt}) - 1 + \xi_i = 0$ egyenletből számolható.

3.2.3 Nemlineáris szeparálás

Az osztályozási feladatok túlnyomó részének megoldását lineáris szeparálással nem lehet kivitelezni. Ilyen esetben célszerű az úgynevezett *kernel-trükk* használata, mely segítségével egy (elképzelt) jellemző térben már megoldható a lineáris szeparálás. A 12. ábra bal képén látható, hogy a piros illetve zöld $\{\vartheta_i\}$ mintapontok nem szeparálhatóak lineárisan, viszont egy $\vartheta \to \varphi(\vartheta)$ nemlineáris dimenziónövelő

transzformációval olyan jellemző teret (12. ábra középső képe) kapunk, melyben már a feladat megoldható lineárisan. A jellemző térben való számítások nem triviálisak, ezért használjuk a kernel teret. A kettő közti kapcsolatot a skaláris szorzás teremti meg: $Ker(\vartheta_i, \vartheta_j) = \varphi(\vartheta_i) \times \varphi(\vartheta_j)$, ahol Ker egy alkalmasan választott kernel függvény. A kernel térben (12. ábra jobb képe) a mintapontok lineárisan szeparálhatóakká válnak, miközben a jellemzőtérbeli számításokat kiküszöböltük, mivel a kernel függvény argumentumai a bementi tér mintapontjai (ezt hívjuk kernel-trükknek). Fontos megjegyezni, hogy itt a bemeneti teret a Fisher-vektorok alkotják, a jellemző tér pedig egy implicit módon definiált tér, melyet a kernel-trükk használatával átugrunk.



12. ábra: Transzformáció a bemeneti tértől a kernel térig (forrás: [2])

A fenti módszerhez tehát szükség van egy úgynevezett kernel függvényre, melyet belső szorzatból származtatunk, és a következők igazak rá:

- Mindig két argumentuma van, és ezekre nézve szimmetrikus
- Az alábbi tulajdonságokat általában kielégítik:
  - Értéke pozitív és radiálisan szimmetrikus
  - Azonos argumentumok esetén az értéke maximális
  - Két argumentum távolságának monoton csökkenő függvénye

Az 1. táblázat láthatók a leggyakrabban előforduló kernel függvények.

Lineáris	$\boldsymbol{Ker(\vartheta_i, \vartheta_j) = \vartheta_i \times \vartheta_j}$
Polinomiális (d fokszámú)	$Ker(\vartheta_i, \vartheta_j) = (\vartheta_i \times \vartheta_j + 1)^d$
Gauss-féle (radiális bázisfüggvények, RBF)	$Ker(\vartheta_i, \vartheta_j) = e^{-\frac{\|\vartheta_i - \vartheta_j\|^2}{2\sigma^2}}$
Tangens hiperbolikusz (k és θ konstansok)	$Ker(\vartheta_i, \vartheta_j) = tanh(k(\vartheta_i \times \vartheta_j) + \theta)$

1. táblázat: A leggyakoribb kernel függvények

Nemlineáris megoldás a lineáris eset megoldásából egyszerűen a $\vartheta \rightarrow \varphi(\vartheta)$ helyettesítéssel érhető el. Ilyenkor az optimális hipersíkot az $n \times \varphi(\vartheta) + B = 0$ alakban keressük. A megoldáshoz most is a Lagrange multiplikátorok meghatározása szükséges. A különbség annyi, hogy a duális feladatnál a $\vartheta \rightarrow \varphi(\vartheta)$ helyettesítést alkalmazzuk. Amennyiben a jellemzőtérben a maximális margójú megoldást kapjuk, akkor a multiplikátorokra szabott feltétel $0 \leq \alpha_i$, ha csak a szoft margójú változat megoldható, akkor a feltétel $0 \leq \alpha_i \leq C$.

A fentiek alapján az SVM válasza ebben az esetben a következő:

$$f(\vartheta) = \sum_{i \in N_S} \alpha_i{}^* y_i \varphi(\vartheta_i) \times \varphi(\vartheta_j) + B^{opt} \tag{22}$$

A kernel trükköt felhasználva, a $\varphi(\vartheta_i) \times \varphi(\vartheta)$ szorzatot felírhatjuk $Ker(\vartheta, \vartheta_i)$ formában, egy magfüggvényként. Ezt bevezetve a nemlineáris osztályozó válasza:

$$f(\vartheta) = \sum_{i \in N_S} \alpha_i{}^* y_i Ker\left(\vartheta_i, \vartheta_j\right) + B^{opt} \tag{23}$$

Külön nem említettem, viszont a 22. és 23. egyenletek alapján látszik, hogy nemlineáris esetben is lesznek szupport vektorok, tehát az SVM ilyenkor is tud „ritka” megoldást adni. Az SVM megoldásának előnye a többi hasonló osztályozóval szemben abból származik, hogy a $\varphi(\vartheta)$ nemlineáris transzformációt nem közvetlenül, hanem a kernel függvényen keresztül közvetve definiáljuk. Így akár végtelen dimenziójú térbe is képezhetjük az adatokat, ha a kernel számolható (például RBF magfüggvényt használva).

Az általam használt C-SVC osztályozó kernel térben oldja meg a feladatot (ehhez RBF kernel függvényt használtam), így a nem lineáris problémát szoft margójúra redukálja. Ehhez egy C paramétert kell megadni az algoritmusnak, ahogy a 21. egyenletben látható. A fejezet elején említettem, hogy minden iterációban végrehajtom egy kép osztályozását, ezzel a C-SVC osztályozóval, így végül minden teszthalmazbeli kép egyszer tesztelésre kerül. Az osztályozás eredményét az 5. fejezetben ismertetem.

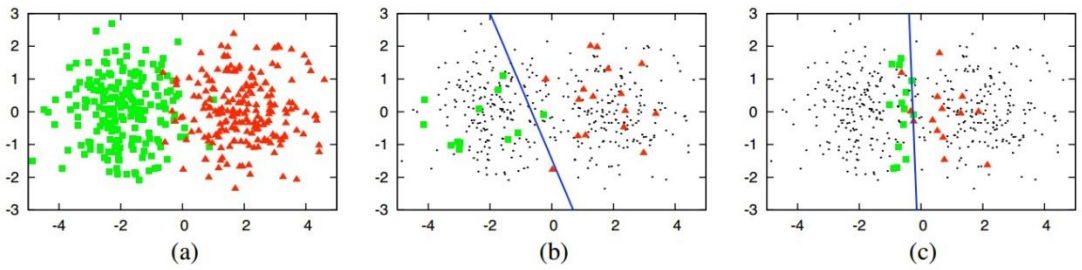
4 Képosztályozás aktív tanulással

Aktív tanulás esetén a tanulórendszer szekvenciálisan választ ki olyan mintákat, melyek még nem címkézettek, és megpróbálja ezt olyan módon megtenni, hogy a frissen megismert címkével a lehető legnagyobb mennyiségű információra tegyen szert. Jelöljük továbbra is T-vel a teljes képhalmazt, melyet osztályozni szeretnénk, és tegyük fel, hogy már vannak címkézett mintáink S (a kiindulási tanítóhalmaz). Az eljárás lényege hasonló a passzív tanulásnál már leírtakhoz, viszont az s sorrend kialakítása nem véletlenszerűen történik, illetve nem is határozható meg előre a teljes T-re. Elképzelhető, hogy minden iteráció után változik az, hogy melyik kép címkéjének megtanulása bír a legtöbb információval, hiszen a korábbi lépésben kapott friss tudással újra kell ezeket kalkulálni minden képre. Ezért dinamikusan alakul ki a sorrend, és a végső sorrendet csak az utolsó iteráció után tudjuk biztosan megmondani.

Az algoritmus kritikus pontja a potenciális címkék által hordozott információmennyiség meghatározása, hiszen ez befolyásolja a tanulás „gyorsaságát”, ami az aktív tanulás használatának előnye. Éppen ezért, erre több stratégia típus létezik, melyek a mai napig fejlesztés, kutatás alatt állnak.

4.1 Lekérdezési stratégiák

4.1.1 Bizonytalansági mintavételezés



13. ábra: A halmaz alapú mintavételezés és bizonytalansági mintavételezés hatékonysága (forrás: [6])

A legegyszerűbb és leggyakrabban használt stratégia, melynek lényege, hogy a tanulórendszer olyan képet választ ki címkézésre, amelynek a címkéjében valamilyen

mérték alapján a legkevésbé biztos. Egyszerűsége ellenére nagyon jól használható, példaként tekintsük a 13. ábra összehasonlítását. Az (a) képen egy 400 elemű adathalmaz látható, kettő osztályból, a (b) képen logisztikus regresszióval osztályoztuk a pontokat 30 címkézett adat alapján, melyeket véletlenszerűen választottunk (70%-os pontosságot eredményezett). A (c) képen hasonlóan osztályoztuk a képeket, viszont a 30 címkézett adatot bizonytalansági mintavételezés alapján választottuk, így 90%-os pontossággal sikerült a pontokat szétválasztani. Legyen $T = \{t_1, t_2, \dots, t_n\}$ azon képek halmaza, melyeket a tanulórendszer címkézésre küldhet az orákulumnak, és $L = \{l_1, l_2, \dots, l_m\}$ a lehetséges címkék halmaza. Több elterjedt változata is van, hogy milyen mérték alapján határozzuk meg a bizonytalanságot. Bináris osztályozás esetén legegyszerűbb azt a képet kiválasztani címkézésre, melyhez tartozó címkék feltételes valószínűségei 0,5 közeliek, így egy bizonytalan minta valódi címkéjét tudhatjuk meg [13][12]. Amennyiben kettőnél több a lehetséges címkék száma $m > 2$, válasszuk azt a képet, melynek címkéjében a legbizonytalanabbak vagyunk:

$$t^* = argmax_t(1 - P(l^*|t)) \tag{24}$$

ahol l^* jelöli a modell szerint legvalószínűbb címkét az adott képhez, valamint t^* jelöli a kiválasztott képet (ezeket a továbbiakban is így fogom jelölni). Ez a megoldási módszer gyakran használt, például szöveg keresése és kiemelése a képből (dokumentumszegmentálás) [3]. A probléma az előző módszerrel az, hogy kizárólag a legvalószínűbb címkét veszi figyelembe, így tulajdonképpen eldobjuk a többi címkéből adódó információt. Ezt részben kiküszöböli a legkisebb margó meghatározása [36], a képekhez tartozó legvalószínűbb és a második legvalószínűbb címkék között:

$$t^* = argmin_t(P(l_1^*|t) - P(l_2^*|t)) \tag{25}$$

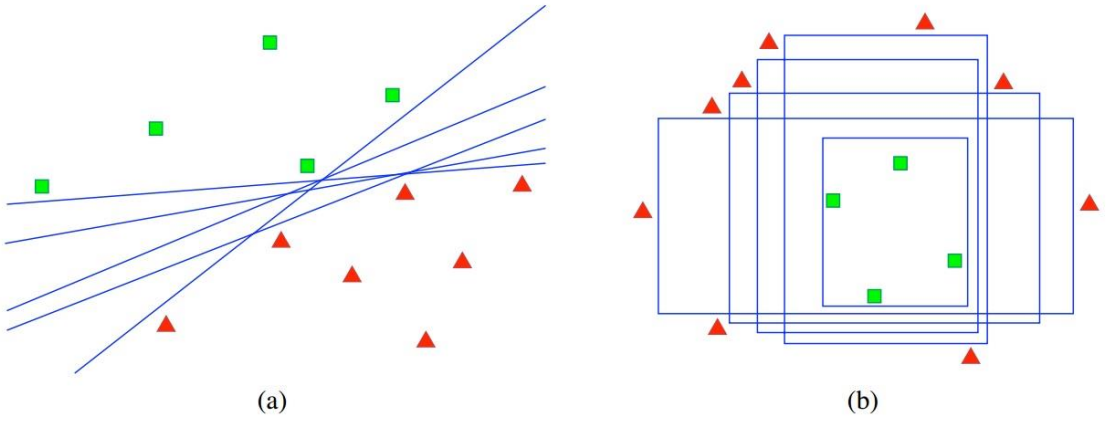
A módszer alapötlete, hogy olyan képek címkéjének becslése könnyű feladat, melyek két legvalószínűbb címkéi távoliak, viszont egy olyan kép címkézése nehéz, melynél ez a távolság kicsi, hiszen ez „kétértelmű”, így ennek a képnek a lekérdezése megold egy nehéz feladatot az osztályozó számára. Fontos megjegyezni, hogy nagyméretű címkehalmaz esetén ez a módszer is túl sok információt hagy figyelmen kívül. A következő módszer minden lehetséges címkézést figyelembe vesz, éppen ezért ez a leggyakrabban használt [21][22] a felsoroltak közül. A lényege, hogy az összes lehetséges kép közül a maximális entrópiájút választja:

$$t^* = argmax_t - \sum_{i=1}^{m} P(l_i|t) \times \log P(l_i|t) \tag{26}$$

Bizonytalansági mintavételezést nem csak valószínűségi osztályozók esetén lehet használni, hiszen Lewis és Cattlet [12] döntési fákat használt a stratégiával felfedezéséhez, valamint Tong és Koller [33] szupport vektor gépek használatával kísérletezett. A szeparáló hipersíkhoz legközelebbi minta lekérdezése olyan megközelítést eredményez, amely analóg a valószínűségi bináris osztályozó, logisztikus regresszió, valamint a naiv Bayes osztályozó bizonytalansági mintavételezésével. Ez azért is fontos, mivel a fent említett módszerek bináris osztályozó esetén ekvivalensek.

4.1.2 Verziótér csökkentés

A lehetséges megoldások azon halmazát, melyek konzisztensek az eddig címkézett képhalmazzal, *verziótér*nek [35] nevezzük. Más szavakkal, ugyanazon osztályozási modell, eltérő paraméterezés mellett különböző megoldási variánsokat ad, így egyfajta „mozgásteret” alakítanak ki a lehetséges megoldások között (lásd 14. ábra). Ezen modellek a már címkézett képek megoldásaival egyetértenek (hiszen azok a megoldások maguk a címkézések), a címkézetlen képek lehetséges címkézéseiben pedig nem értenek egyet, ezek a különvélemények alkotják a verzió teret. A verziótér csökkentés módszer lényege, hogy az eltérő vélemények halmazát minimalizáljuk lépésről lépésre.



14. ábra: Két különböző bináris osztályozási modell eredményei eltérő paraméterezés mellett. A címkézett mintákban minden megoldási változat egyetért, viszont mégis eltérőek, ezért a potenciális teszt minták osztályaiban nem feltétlenül értenek egyet, így verzió teret alakítanak ki (forrás: [6])

A legelterjedtebb algoritmus a hasonló feladatok megoldására, az úgynevezett *query-by-committee* (QBC, [18]), mely számon tartja a verziótér elemeit, és a tanulórendszer minden iterációban azt a képet választja, melynek lehetséges

címkézésben az eltérés maximális. Ahhoz, hogy ezt meg tudjuk tenni, két feltételnek kell teljesülnie:

1. számon kell tartani minden lépésben a verziótér elemeit
2. definiálni kell egy mértéket, mely megadja a verziótér elemeinek bizonytalanságát az egyes képekhez (tehát, hogy a lehetséges címkézés mennyire eltérő)

Generatív modellek esetén az 1. pont könnyen teljesíthető, amennyiben a verziótér elemeit a $P(M|T_L)$ eloszlás szerint mintavételezzük [1], ahol M a modellt, T_L pedig a már címkézett képeket jelöli. Diszkriminatív vagy nem valószínűségi modelleknél két algoritmus terjedt el, a *query-by-boosting* és a *query-by-bagging* [29]. Az előbbi esetben minden iterációban újraszámítjuk az eloszlást úgy, hogy a félrecímkézett képek nagyobb valószínűséggel rendelkezzenek, majd ez alapján választunk képet. A második eljárás lényege, hogy minden lépésben egyre kisebb képhalmazt generálunk, melybe minden képet azonos valószínűséggel választunk be.

A második feltételben említett bizonytalansági szint (*level of disagreement*) meghatározásához többféle módszert használhatunk, melyek közül a két legfontosabbat emelem ki. Az első eljárás az entrópia alapú kiválasztás [19], melyre tekinthetünk úgy, mint a bizonytalansági mintavételezésnél megismert maximum entrópia módszerének QBC alapú általánosítására:

$$t^* = argmax_t - \sum_{i=1}^{m} \frac{V(l_i)}{C} \times \log \frac{V(l_i)}{C} \tag{27}$$

ahol C a verziótér kialakításánál felhasznált modellek száma, $V(l_i)$ pedig a t képhez l_i címkét rendelő modellek száma. A második elterjedt módszer a Kullback-Leibler (KL) divergencia [1] alkalmazása:

$$t^* = argmax_t \frac{1}{C} \sum_{c=1}^{C} D(P_{M^{(c)}} || P_{\{C\}}) \tag{28}$$

ahol $M^{(c)}$ egy konkrét modellt jelöl, $P_{M^{(c)}}$ pedig ennek a modellnek a valószínűségét, valamit $\{C\}$-vel jelöljük a modellek együttesét, és $P_{\{C\}}$-vel a modellek által meghatározott „átlagos valószínűséget”. A $D(\cdot \,||\, \cdot)$ jelölés pedig a valószínűségek különbségét, eltérését méri [31], vagyis a KL divergenciát:

$$D(P_{M^{(c)}} || P_{\{C\}}) = \sum_{i=1}^{m} P_{M^{(c)}}(l_i|t) \times log \frac{P_{M^{(c)}}(l_i|t)}{P_{\{C\}}(l_i|t)} \tag{29}$$

A QBC mellett léteznek más próbálkozások is a verziótér csökkentésére, mint a szelektív mintavétel két neurális hálózat, a legspecifikusabb és a legáltalánosabb által adott megoldások alapján [11]. Bizonyos osztályozási modellek a verziótér csökkentéséhez szükséges számításokat implicit támogatják, ilyen például az SVM, ahol a verzióteret a lehetséges súlyozások adják [33].

4.1.3 Egyéb lekérdezési stratégiák

Ebben az alfejezetben néhány további módszert említek meg a címkézendő kép meghatározására, melyeket nem tervezek megvalósítani a képosztályozó rendszerben, ezért működésüket nagy vonalakban ismertetem.

Általános eljárás, hogy a tanulórendszer azt a képet küldi az orákulumnak, melynek valódi címkéjét megtudva, a *tanult modell várható változása* maximális. Diszkriminatív valószínűségi osztályozók esetén használhatjuk a *gradiens várható hosszát* (*expected gradient length*, EGL [4][3]), mint mértéket a modell változásának mérésére, hiszen hasonló osztályozási modellek tanításra gradiens alapú optimalizálást alkalmaznak. Ehhez vezessük be a $\nabla_M(T_L)$ jelölést a T_L tanult címkék esetén az M modell gradiensére, valamint a $\nabla_M(T_L \cup \langle t, l \rangle)$ jelölést az új gradiensre, melyet a $\langle t, l \rangle$ kép-címke páros hozzáadása és tanítása után kapunk. Mivel a lekérdező algoritmus nem tudja a t képhez tartozó valódi l címkét, ezért a gradiens hosszát becsüljük az összes lehetséges címkézés fölött:

$$t^* = argmax_t \sum_{i=1}^{m} P(l_i|t) \times ||\nabla_M(T_L \cup \langle t, l_i \rangle)|| \qquad (30)$$

Hasonló módszer a tanult modell segítségével történő osztályozási *hiba várható értékének minimalizálása*. Az ötlet lényege, hogy becsüljük meg a $T_L \cup \langle t, l \rangle$ címkézett képhalmazon tanított modell hibájának mértékét a T_U címkézetlen képhalmazon, és válasszuk a minimális „jövőbeli” hibát eredményező t képet. Erre használhatjuk a 0/1 veszteségfüggvény minimalizálását:

$$t^* = argmin_t \sum_{i=1}^{m} P(l_i|t) \times \sum_{j=1}^{u} 1 - P'\left(l^* \middle| t^{\{U\}}\right) \qquad (31)$$

ahol P' a $T_L \cup \langle t, l_i \rangle$ képhalmazon tanult modell által adott feltételes valószínűséget, u a címkézetlen képek számát, $t^{\{U\}}$ pedig egy konkrét címkézetlen képet jelöl. Az eddig felsorolt összes eljárás közül ennek van a legmagasabb

számításigénye, hiszen meg kell becsülni a várható hibát U felett, valamint minden egyes új címke esetén újra kell tanítani a modellt.

4.2 A javasolt módszer

Az eljárás alapját a bizonytalansági mintavételezés képezi, melyet egy címkeeloszlás vizsgálattal egészítettem ki. Ahogy fentebb említettem, szupport vektor gépeket használok az osztályozáshoz, mely esetén az osztályozó hipersíkhoz legközelebb elhelyezkedő minta kiválasztásával megkapom azt a képet melynek címkéjében az osztályozó a legbizonytalanabb [33][7][17]. Ez az általános módszer, viszont helyette én kiszámítottam a címkézetlen képek entrópiáját az összes lehetséges címke felett:

$$H_j = -\sum_{i=1}^{m} P(l_i|t_j) \times \log P(l_i|t_j) \tag{32}$$

ahol H_j jelöli a j-edik kép entrópiáját, m a kategóriák számát, $P(l_i|t_j)$ pedig az abban való bizonyosságot, hogy a t_j képnek l_i a valós osztálycímkéje. Továbbá, minden képhez hozzárendelek egy úgynevezett „büntetőpontot”, melyet az alapján számolok, hogy az adott kép valós osztályából mikor választottam utoljára:

$$Penalty_j = CTR_{L_j} \times \frac{1}{m} \tag{33}$$

$$i^*(t_j) = \operatorname{argmax}_i P(l_i|t_j) \tag{34}$$

$$L_j = i^*(t_j) \tag{35}$$

ahol $Penalty_j$ jelöli a j-edik kép büntetőpontját, L_j a j-edik képnek becsült osztálycímkét, CTR_{L_j} pedig azt számolja, hogy mennyi lépés óta nem tanult a rendszer L_j címkét, az m pedig a kategóriák száma. Ezeket minden képre összefésülöm egy β súlyozó paraméter segítségével:

$$HP_j = (1 - \beta) \times H_j + \beta \times Penalty_j \tag{36}$$

Végül egy sorrendet alakítok ki, ahol az első helyen a legnagyobb HP értékű kép áll, és attól függően, hogy egy lépésben mennyi új címkét kérdez le a tanulórendszer, a sorrend első néhány képét választja. Az eljárás előnye, hogy nem engedi aránytalanul sok vagy aránytalanul kevés kép lekérdezését egyetlen kategóriából, így főleg kevés kép tanulása esetén magasabb pontosságot érhetünk el. Ez akkor mutatkozik meg leginkább, mikor sok, akár több száz kategóriánk van (lásd 5. fejezet).

5 Tesztelés: emberi és gépi tanulás

5.1 Felhasznált képhalmazok

A teszteléshez öt különböző képhalmazt használtam, melyek mindegyike manuálisan kigyűjtött és címkézett képekből áll. Mivel célom az emberi és gépi tanulás összehasonlítása, ezért viszonylag kisméretűek a teszthalmazok, hogy az emberi tanulás kiértékelésében részt vevő személyek számára ne legyen túl megterhelő a rengeteg kép osztályozása. Az alább táblázatban összefoglaltam ezek méretét, a definiált kategóriák számát, illetve a tesztelő személyek számát.

	Méret	Kategóriák száma	Tesztelő személyek száma
Képhalmaz 1	100	7	12
Képhalmaz 2	100	8	9
Képhalmaz 3	116	6	12
Képhalmaz 4	105	5	6
Képhalmaz 5	106	6	7

2. táblázat: Teszteléshez használt képhalmazok adatai

A kiindulási címkézett halmazhoz minden osztályból választottam egy-egy képet, ez alapján kezdték mind a tesztelő személyek és az algoritmus a tanulást. A passzív tanulással történő osztályozáshoz megadtam a megfelelő méretű statikus szekvenciákat, melyek a címkézetlen képek érkezési sorrendjét határozzák meg, ez szintén egységes volt a webes felület (lásd 1.1 alfejezet) és a képosztályozó rendszer esetén.

5.2 Kiértékelés menete

Az osztályozások eredményeit a pontosság (*accuracy*) szempontjából értékeltem ki, melyhez szükség volt a tesztelt képek során született döntések eredményeire, hogy sikeres volt-e a kategóriába sorolás, vagy sem. A tesztképekhez tartozó statikus szekvencia mellé egy 0/1 döntési eredményt rendeltem, így minden képről eltároltam,

hogy annak osztályozását az adott tesztelő entitás (mely lehet gépi vagy emberi) miként végezte. Ez által, a pontosság meghatározása a következő képlet alapján történik:

$$accuracy = \frac{döntési\ vektor\ 1{-}eseinek\ száma}{statikus\ szekvencia\ mérete} \tag{37}$$

A folyamatosan bővülő tanulóhalmaz egyre több információt ad a tesztelőnek, így elméletben, a fennmaradó címkézetlen képeket nagyobb pontossággal képes osztályozni. Ennek mérése érdekében egy csúszó ablakot mozgatok a szekvencia elejétől a végéig, és mindig az ablak alatti szekvencia által kijelölt képek osztályozási pontosságát határozom meg:

$$accuracy_W = \frac{ablak\ alatti\ 1{-}esek\ száma}{ablak\ mérete} \tag{38}$$

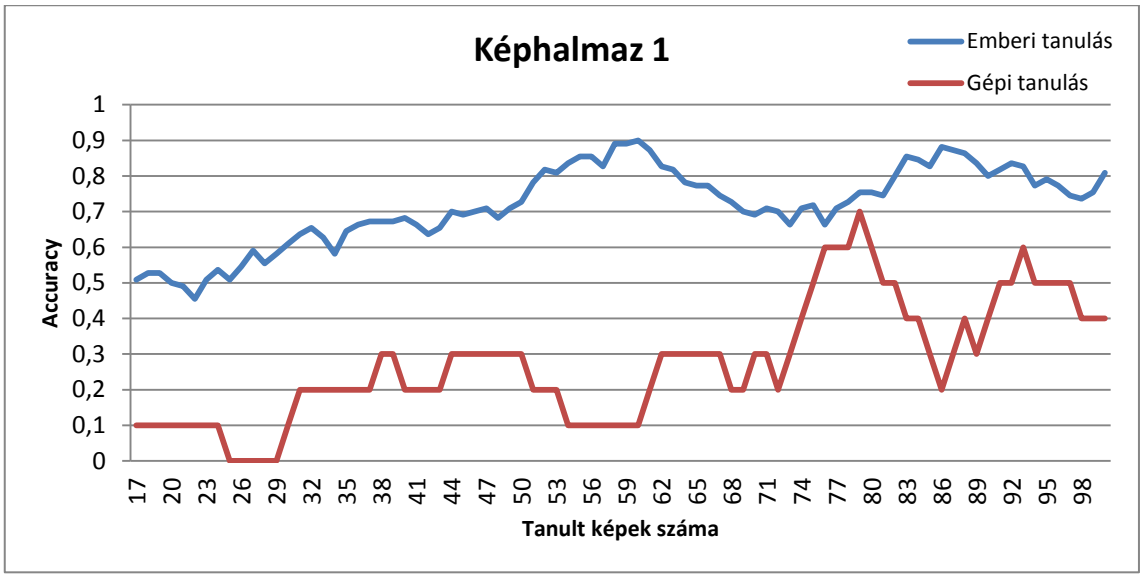
Ahogy a 2. Táblázatban látszik, minden képhalmazt több személy tesztelt, így a kapott pontossági értékeket átlagoltam. A következő alfejezetben a kapott eredményeket mutatom be, és hasonlítom össze. A kapott átlagos ablak alatti pontosságokat vonal diagramok segítségével ábrázolom, a tanított képek számának függvényében (tehát a szekvencián előre haladva). Az ablak méretét 10-re választottam, így a kiértékelés elején, a 10. beérkezett kép után kapom meg az első pontossági értéket, és az ábrázolás onnantól kezdődik az utolsó beérkező képig.

5.3 Eredmények

Az első képhalmaz különböző film poszterekből állt, a definiált kategóriák pedig az egyes filmek műfajai voltak (pl.: animációs, akció, vígjáték, horror, stb.). Ez azt jelenti, hogy az egyes osztályokba tartozó képek nem kizárólag vizuális hasonlóság alapján kapcsolódnak (lásd 15. ábra). Éppen ezért a képosztályozó algoritmusnak nehéz dolga volt ezzel a teszthalmazzal, hiszen az egyedül a tartalmi egyezéseket veszi figyelembe.

15. ábra: Példa képek az első képhalmazból, (a)-(g)-ig oszloponként azonos osztályú képekkel

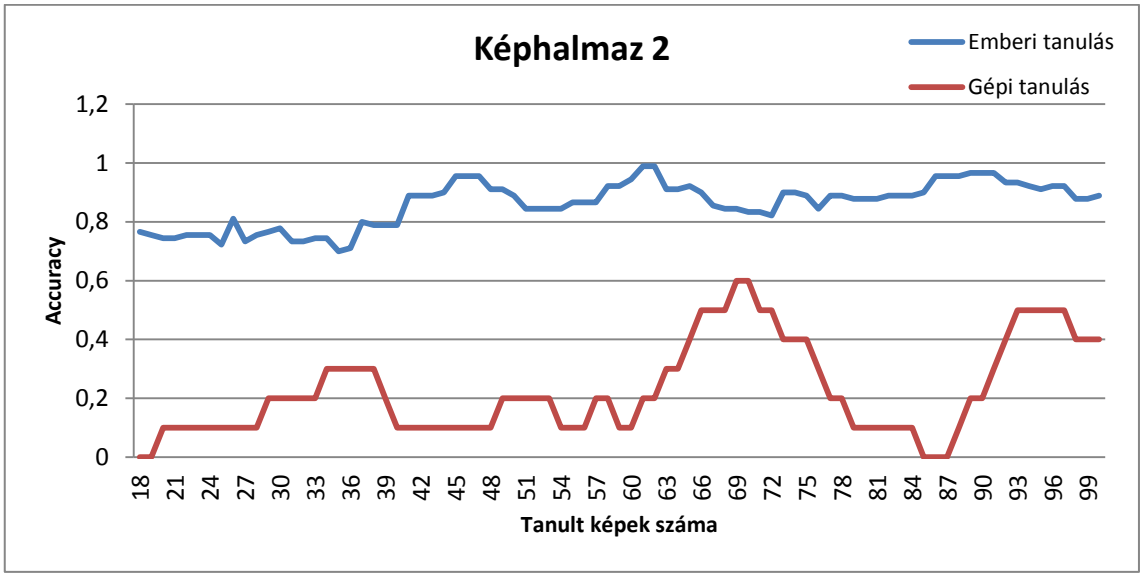
Ahogy a fenti ábrán látszik, az egyes kategóriánként megjelenített képek közt nem lehet egyértelmű vizuális kapcsolatot meghatározni. A kapott eredmények a **Hiba! A hivatkozási forrás nem található.** diagramján láthatók.



16. ábra: Az első képhalmaz tesztelése során kapott átlagos ablak alatti pontosságok a tanított képek számának függvényében, emberi és gépi tanulás esetén

Jól látszik, hogy a gépi tanulás, sőt, az emberi tanulás esetén is a tanított képek számának növekedésével a pontosság is növekszik (kevés kivétellel). Az algoritmus a szekvencia elején érkező képeket nagy hibával osztályozta, viszont 70 kép után sikerült javítani a pontosságon. Az emberi tanulás láthatóan az első képtől kezdve jobban teljesít, viszont ez várható is volt, hiszen a gépnek nincs priori ismerete, emberek számára viszont sok segítséget nyújthatnak tapasztalati ismeretek, például ebben az esetben, ha a tesztelő látta már azt az adott filmet, melyet a poszter ábrázol. Ezért előfordult a 0,9-es ablak alatti pontosság is, míg gépi tanulás esetén ez a legjobb esetben 0,7 volt.

A második képhalmaz a számítógép számára az elsőnél még nehezebb teszthalmaz volt, melyben a definiált kategóriák csekély mennyiségű vizuális hasonlóságot követeltek meg. A kialakított osztályok absztrakciós szintje magas volt, illetve előfordult, hogy csupán a képek „jelentése” alapján lehetett hasonlóságot meghatározni. Ilyen például az ételek osztály, melyben minden egyes kép, különböző ételt jelenít meg, vagy a műtárgyak kategória, melynek biztosan nincs tartalmilag hasonló képe (kivéve, ha ugyanazt a műtárgyat ábrázolja a kép, viszont ilyen nem volt). A gépi és emberi tanulás eredményeit a 17. ábra mutatja.

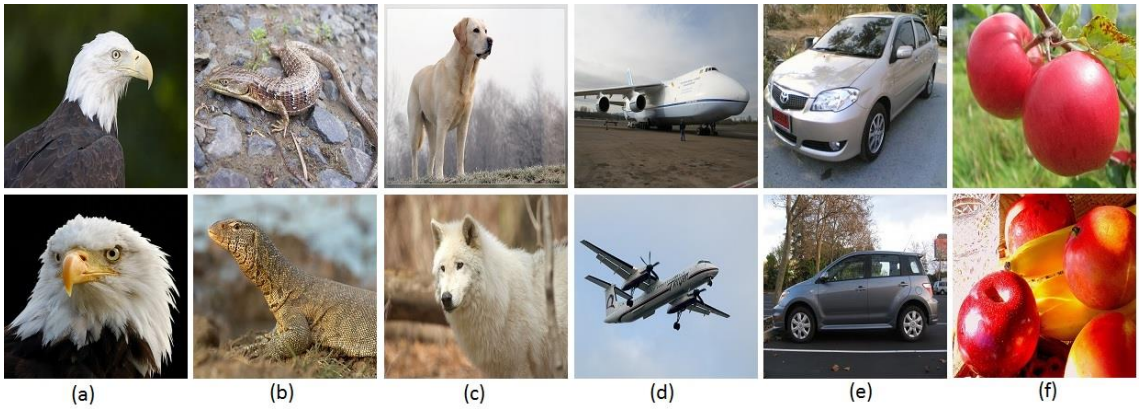


17. ábra: A második képhalmaz tesztelése során kapott átlagos ablak alatti pontosságok a tanított képek számának függvényében, emberi és gépi tanulás esetén

A fenti ábrán látható, hogy valójában nem sikerült megtanulnia a rendszernek az egyes kategóriák jellemzőit, mivel a görbe végén még 0 értéket is mérhetünk, ami azt jelenti, hogy az abban a pontban végződő 10 hosszú ablak alatt egyetlen helyes döntést sem hozott az algoritmus. Ezzel szemben az emberi tanulást bemutató ábrán szinte végig 0,8-0,9 között mozgott a pontossági érték. Ebben az esetben szintén elmondhatóak az első teszthalmaznál tett megállapítások, mi szerint az emberek tapasztalati tudása nagyban befolyásolja a tesztelést. A képhalmaz tipikus példa arra, hogy emberi szemmel tanulási folyamat nélkül is könnyen felismerhető két kép közt a hasonlóság.

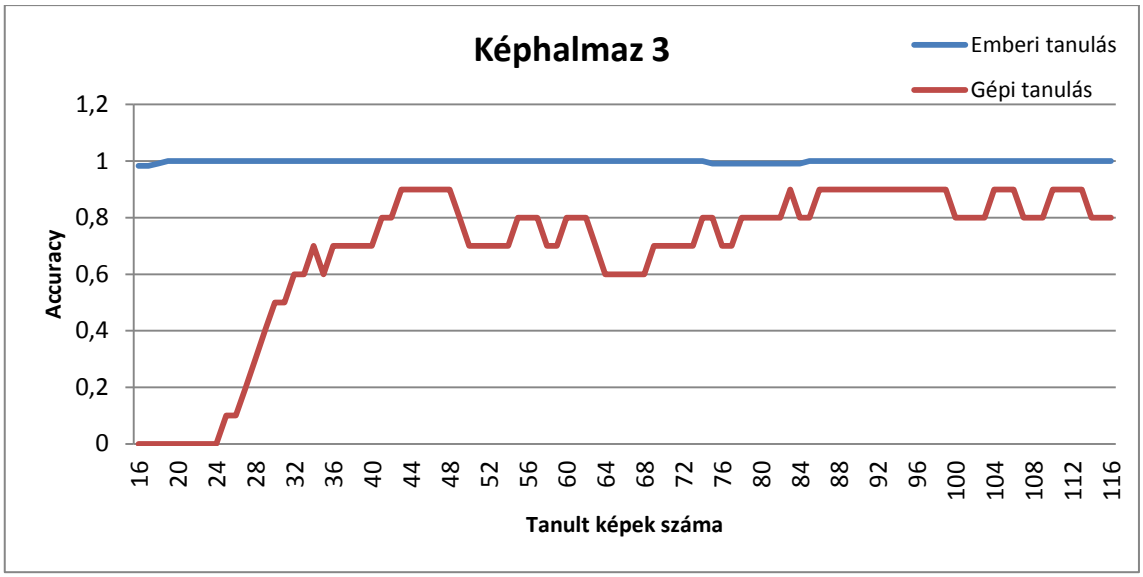
A harmadik képhalmaz a leghasonlóbb a képosztályozási feladatoknál megszokott teszthalmazhoz. Az itt definiált osztályokban (sas, kutyafélék, hüllők, gyümölcsök, szárazföldi négykerekű járművek, repülőgépek) jelentős vizuális

hasonlóság van jelen (lásd 18. ábra), így gépi tanulással hatékonyan kategorizálhatóak a képek. A következő ábrán minden osztályból 2-2 képet jelenítettem, hogy szemléltessem a különbséget az első és harmadik teszthalmaz között. Jól látható, hogy a 15. ábra kategóriáival ellentétben most tényleg tartalmi hasonlóság adja a kategorizálás alapját, és tulajdonképpen az osztályozó rendszert ilyen képhalmaz tesztelésére készítettem fel, ami az eredményeken is látszik.



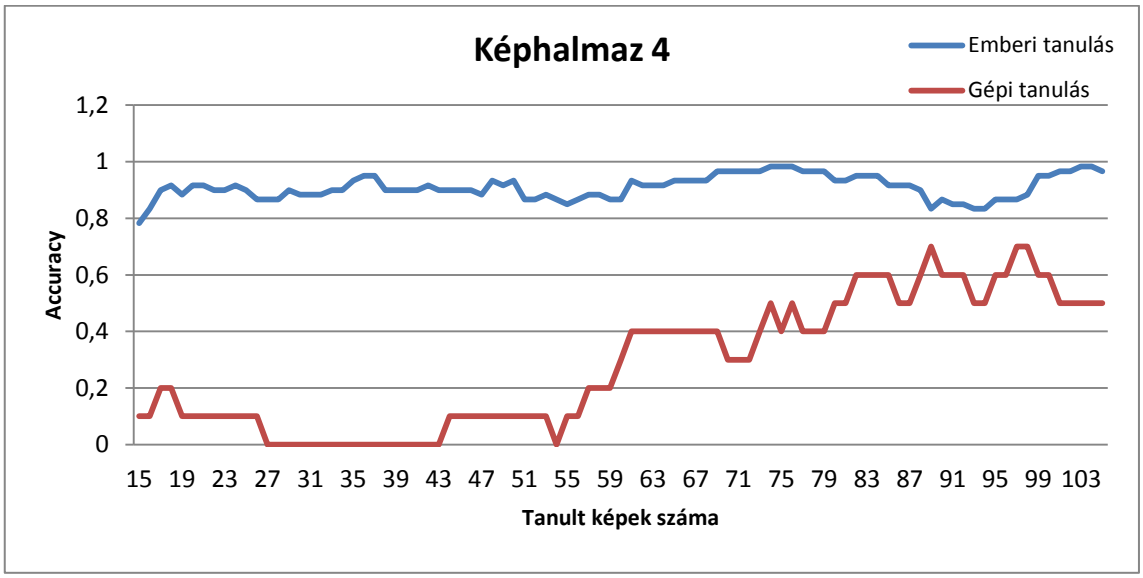
18. ábra: Példa képek a harmadik képhalmazból, (a)-(f)-ig oszloponként azonos osztályú képekkel

A kategóriák meghatározásnál annyi nehezítést alkalmaztam, hogy néhány esetben több különböző objektum is beletartozik ugyanabba az osztályba, tehát úgynevezett *superclass*-okat definiáltam. Például a gyümölcsök kategóriát a banán, narancs, alma, körte, barack, málna és áfonya alkotja, illetve a kutyafélék osztályt a farkas, róka és kutya objektumok alkotják, valamint ugyanez igaz a szárazföldi négykerekű járművek esetén is. Látni fogjuk, hogy ilyen esetben a felhasználók szinte tévesztés nélkül képesek megmondani az egyes képek valódi osztályát. Ez nem meglepő, hiszen például egy rókát egy narancstól előzetes tanítási folyamat nélkül is 100%-os biztonsággal és pontossággal vagyunk képesek megkülönböztetni. A 19. ábra szemlélteti a gépi, valamint az emberi tanulás eredményeit.



19. ábra: A harmadik képhalmaz tesztelése során kapott átlagos ablak alatti pontosságok a tanított képek számának függvényében, emberi és gépi tanulás esetén

A gépi tanulás eredményeiből is jól látható, hogy ez egy olyan teszthalmaz, melyre az algoritmus fel van készítve, így a korábban jósolt javulás a pontosságban egyértelműen észrevehető, a tanított képek számának növekedésével. Az első néhány képet hibásan osztályozza a rendszer, viszont nagyjából 30 tanító kép után stabilan 0,7-0,9 közé áll be az ablak alatti átlagos pontosság.

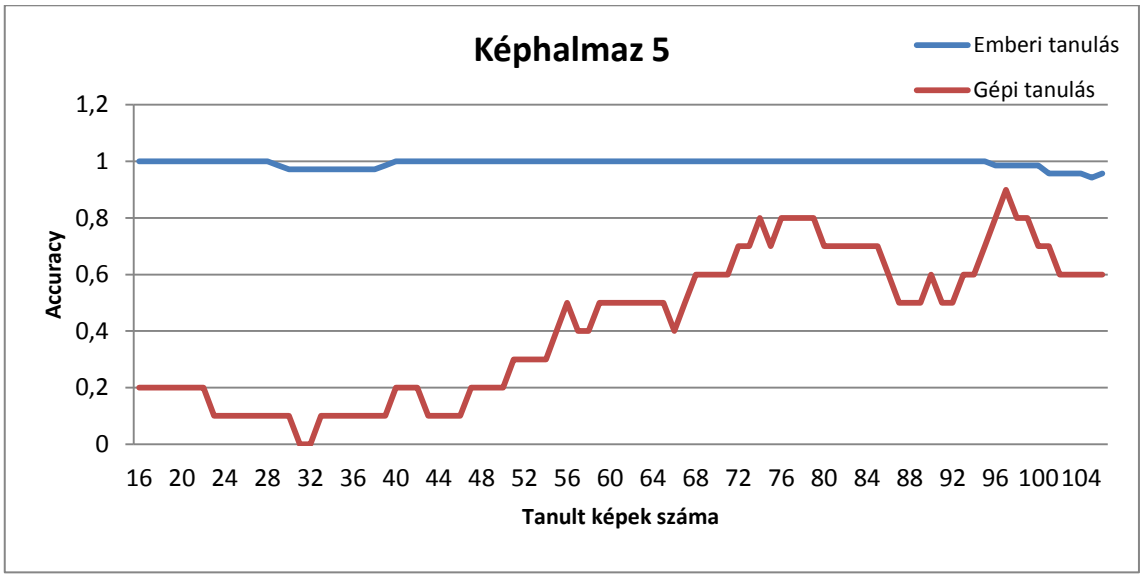


20. ábra: A negyedik képhalmaz tesztelése során kapott átlagos ablak alatti pontosságok a tanított képek számának függvényében, emberi és gépi tanulás esetén

Ahogy a fenti ábrán látható, az itt definiált kategóriák (vizes palack, kerék, cipő, nyitott nyílászáró, póló) megtanulásához több tanító képre volt szüksége a rendszernek.

Az előző teszthez hasonlóan itt is a tartalmi egyezés volt az osztálypéldányok közti hasonlóság, ezért egy tényleges tanulási görbe itt is megfigyelhető, szemben az első két képhalmazzal, ahol még jelentős letörések voltak megfigyelhetőek a görbéken. Nagyjából 50-60 tanító kép után figyelhető meg javulás az osztályozásban, és az ablak alatti pontosság 0,6-0,7 között áll be a teszthalmaz végére (lásd 20. ábra). Fontos megjegyezni, hogy a képek közt több olyan is előfordult, melyek egynél több kategóriába is beleillettek az itt definiáltak közül, így ehhez többcímkés osztályozás jobban illeszkedett volna. Ezt a tesztet kísérletként végeztem, hogy kiderüljön, ilyen adottságú képhalmazt az egycímkés algoritmussal mekkora tévesztéssel tudok osztályozni. Az emberi tanulásnál ismét megfigyelhető az, hogy a legelső kép tesztelésétől kezdve 0,9-1 közötti a pontosság, ami a sokszor említett priori tudás következménye.

Az ötödik képhalmazban olyan osztályok kerültek definiálásra, melyek valamivel kevésbé (vizuálisan) egyező osztálypéldányokból álltak, mint a harmadik teszt esetén, viszont itt is a tartalmi egyezés a szempont. A gépi és emberi tanulás eredményét a 21. ábra jeleníti meg. Az algoritmus 40 tanító kép után kezd egyre pontosabban osztályozni, viszont a tanulási folyamat tovább tart, hiszen a 40. képtől a 75. képig alacsonyabb a görbe meredeksége. Az emberi tanulással kapott eredmények a harmadik tesztnél látottakhoz hasonlítanak, hiszen itt is (emberi szemmel) triviális tartalmi egyezések alapján kellett a képeket kategorizálni.



21. ábra: Az ötödik képhalmaz tesztelése során kapott átlagos ablak alatti pontosságok a tanított képek számának függvényében, gépi és emberi tanulás esetén

A fenti tesztek jól jellemzik az emberi és gépi tanulás közti különbségeket. A legfontosabb különbség, hogy az emberek által birtokolt előzetes információ egy jó alapot nyújt az osztályok közti hasonlóság gyors, akár azonnali felismeréséhez. Igazán jól összemérni a kettőt akkor lehetne, ha olyan képeket tudnánk előállítani, melyek a tesztelő személyek semmilyen eddig tapasztalatához, ismereteihez nem köthetőek, így számukra is elengedhetetlen lenne az előzetes tanulás, ahogy a gépi tanulás esetén láttuk. Erre használhatnánk például absztrakt, gép által véletlen generált képeket.

A csúszó ablak segítségével jól mérhetővé vált az, hogy a pontosság hogyan változik az osztályozás során (főleg gépi esetben), viszont a teljes teszthalmazon számított pontossági érték (lásd 35. egyenlet) meghatározása nem képes a javulás megmutatására. Ennek oka, hogy a teljes döntési vektort használjuk a kiszámításhoz, figyelmen kívül hagyva azt, hogy a döntési vektorban (elemszámában) előre haladva az osztályozáshoz felhasznált tanulóanyag mérete növekszik. Ennek ellenére ezeket is kiértékeltem, és a 3. táblázat összefoglaltam, ahol W, G és E rendre az ablak alatti, gépi és emberi pontosságtípusokat jelölik. Például $max(accuracy_{W.G})$ a 36. egyenlet alapján számolt ablak alatti pontosságok közül a maximálist jelenti, az adott teszthalmazra, gépi tanulás esetén, míg $átlag()$-al a 35. egyenletben felírt képlet alapján kapott pontosságot jelölöm.

	Teszt 1	Teszt 2	Teszt 3	Teszt 4	Teszt 5
$\mathbf{max}(accuracy_{W.G})$	0,7	0,6	0,9	0,7	0,9
$átlag(accuracy_{G})$	0,269	0,217	0,664	0,290	0,420
$max(accuracy_{W,E})$	0,9	0,989	1	0,983	1
$átlag(accuracy_{E})$	0,716	0,859	0,998	0,901	0,991

3. táblázat: Teszthalmazonkénti pontosság értékek összesítése

6 Tesztelés: passzív és aktív tanulás

6.1 Felhasznált képhalmazok

A teszteléshez két, képfeldolgozási témakörben gyakran használt képhalmazt választottam, a *PASCAL VOC2010* és *Caltech101* gyűjteményeket [15][25]. Az előbbi 20 kategóriából és 10103 képből áll, melyből egy 1000 méretű részhalmazt képeztem úgy, hogy kategóriánként véletlen valószínűséggel választottam 50-50 képet. Ebből a részhalmazból 2 darab 5 kategóriájú, 2 darab 10 kategóriájú további részhalmazokat képeztem, melyeket szintén véletlenszerűen választottam. Ezeken, illetve a teljes 20 kategóriából álló részhalmazon is több alkalommal futtattam az osztályozást (lásd 4. táblázat), a különböző tanulási stratégiákkal, melyek a következők voltak:

1. Véletlen kiválasztás: a szokásos passzív tanulás, a következő tanító mintát véletlenszerűen választja a címkézetlen képek közül
2. Entrópia alapú bizonytalansági mintavételezés: a címkézetlen képek közül a maximális entrópiájút választja
3. Entrópia alapú bizonytalansági mintavételezés kiegészítve egy címkeeloszlás vizsgálattal: a 4.2 alfejezetben már ismertetett eljárás

	Kategóriák száma	Futtatások száma	Képek száma
PASCAL 1	5	5	250
PASCAL 2	5	5	250
PASCAL 3	10	3	500
PASCAL 4	10	3	500
PASCAL 5	20	2	1000

4. táblázat: PASCAL VOC2010 tanító és validáló képeiből előállított teszthalmazok

A Caltech101 képhalmaz összesen 8677 képből és 102 kategóriából áll, melyek közül 1 úgynevezett „zaj” kategória. Ezt elhagyva, egy 3030 képből álló részhalmazát képeztem a teljes gyűjteménynek úgy, hogy minden osztályból véletlen valószínűséggel

kiválasztottam 30-30 képet. Ebből a részhalmazból 3 darab 5 kategóriájú, 3 darab 10 kategóriájú és 2 darab 20 kategóriájú további részhalmazokat képeztem, melyeket szintén véletlenszerűen választottam (lásd 5. táblázat). Ezeken is több alkalommal futtattam az osztályozást, a fentebb felsorolt tanulási eljárásokkal külön-külön, valamint egyszer a teljes 101 kategóriából álló részhalmazt is teszteltem.

	Kategóriák száma	Futtatások száma	Képek száma
CALTECH101 1	5	5	150
CALTECH101 2	5	5	150
CALTECH101 3	10	3	300
CALTECH101 4	10	3	300
CALTECH101 5	20	2	600
CALTECH101 6	20	2	600
CALTECH101 7	101	1	3030

5. táblázat: Caltech101 képeiből előállított teszthalmazok

Azért választottam a fenti két kiindulási képhalmazt, mivel a PASCAL VOC2010 tipikusan egy nehezebb, komplexebb képekből álló gyűjtemény, míg a Caltech101 képei ehhez képest könnyebben osztályozhatóak. Így kipróbálhattam az elkészített rendszert mind könnyű, mind pedig nehéz teszthalmazokon.

6.2 Kiértékelés menete

Az ismertetett teszthalmazok tesztelése és kiértékelése eltérően történt, mint ahogy azt az emberi és gépi tanulás összehasonlításánál bemutattam. Lépésenként egy (lekérdezett) kép osztályozása helyett a teljes fennmaradó címkézetlen képhalmazt osztályoztam. A lépések addig tartanak, míg a teszthalmaz felét le nem kérdezi a tanulórendszer, tehát az utolsó iterációban pontosan 50-50 százalék a tanító és a teszt képek aránya. Az egyes teszthalmazokon végrehajtott többszöri tesztek eredményeit átlagoltam, majd kiértékeltem és elkészítettem a konfúziós mátrixokat, külön-külön minden egyes kategóriára (lásd 6. táblázat).

		Valódi osztálycímke alapján	
		Beletartozik	*Nem tartozik bele*
Becsült osztálycímke alapján	*Beletartozik*	TP	FP
	Nem tartozik bele	FN	TN

6. táblázat: Konfúziós mátrix egy adott kategóriára vonatkozóan

A fenti táblázatban a TP (True Positive) jelöli azoknak a képeknek a számát, melyeket helyesen a valódi kategóriájába sorolt a rendszer, FP (False Positive) pedig azon képek száma, melyeket helytelenül sorolt ebbe az osztályba. Az FN (False Negative) jelöli tévesen más osztályba sorolt képek számát, végül a TN (True Negative) pedig a helyesen más kategóriába sorolt képekét. Ezek alapján többféle mutatót számoltam ki:

$$Fedés\ (recall):\ Rec = \frac{TP}{TP+FN} \tag{39}$$

$$Pontosság\ (precision):\ Prec = \frac{TP}{TP+FP} \tag{40}$$

$$Osztályozási\ pontosság\ (accuracy):\ Acc = \frac{TP}{TP+FP+FN+TN} \tag{41}$$

$$Average\ Precision\ (AP):\ AP = \sum_{i=1}^{|S|} Prec(i) \times \Delta Rec(i) \tag{42}$$

A fenti négy mutató az adott kategóriára vetítve minősíti az osztályozó rendszer teljesítőképességét. Az AP képletében az $|S|$ jelöli az aktuális címkézetlen képállomány méretét, $\Delta Rec(i)$ pedig a fedés mutató változását adja meg az $i-1$. és i. lépések között. Fontos megemlíteni, hogy itt a tesztképeken az adott kategóriába sorolás bizonyossága alapján egy sorrendet állítunk fel, és a lista elejéről lépkedünk a végére a képlet szerint. A következő két mutató már a teljes rendszert jellemzi, alapjuk, hogy a kategóriánként kapott értékeket átlagolják:

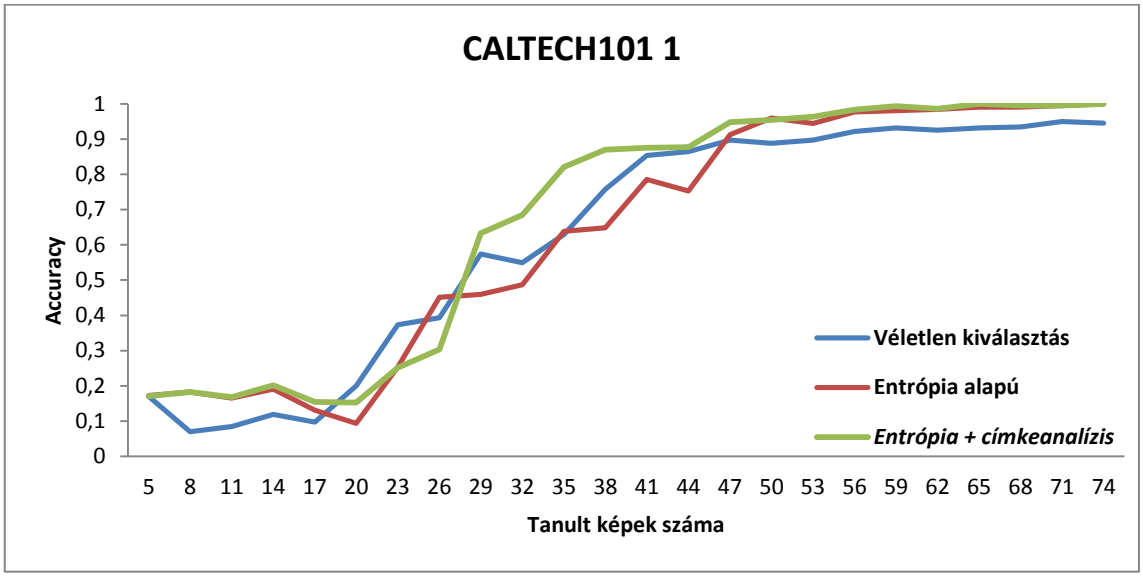
$$Mean\ Average\ Precision\ (MAP):\ MAP = \sum_{i=1}^{|C|} \frac{AP_i}{|C|} \tag{43}$$

$$Átlagos\ osztályozási\ pontosság:\ Accuracy = \sum_{i=1}^{|C|} \frac{Acc_i}{|C|} \tag{44}$$

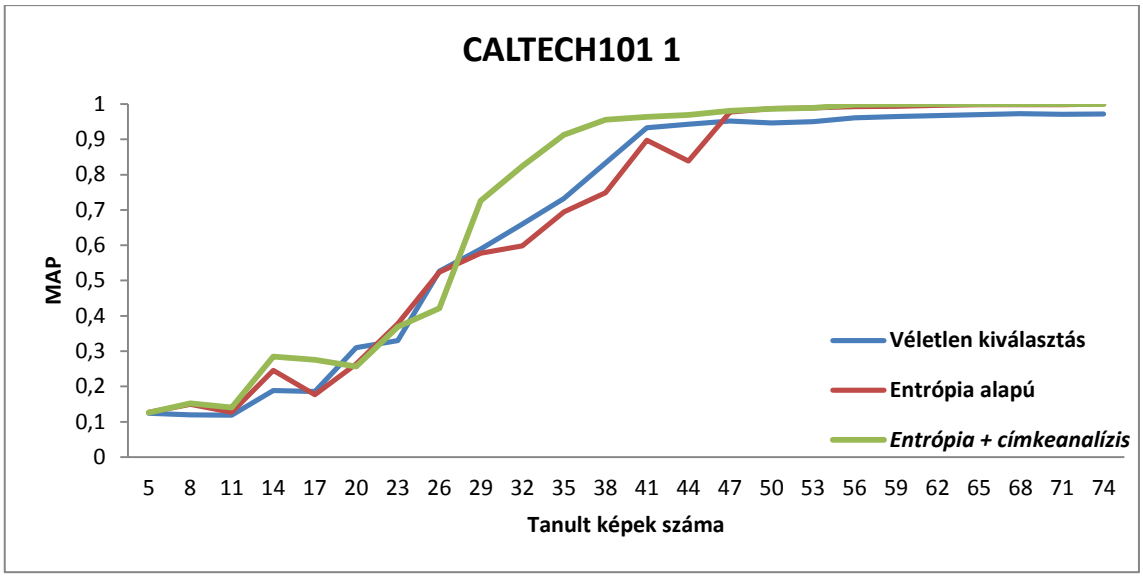
Itt a $|C|$ jelöli a kategóriák számát, i indexel pedig az adott kategória esetén kapott értékeket jelölöm. A következő részben a tesztek eredményeinek kiértékelésével kapott MAP és Accuracy mutatókat fogom bemutatni.

6.3 Eredmények

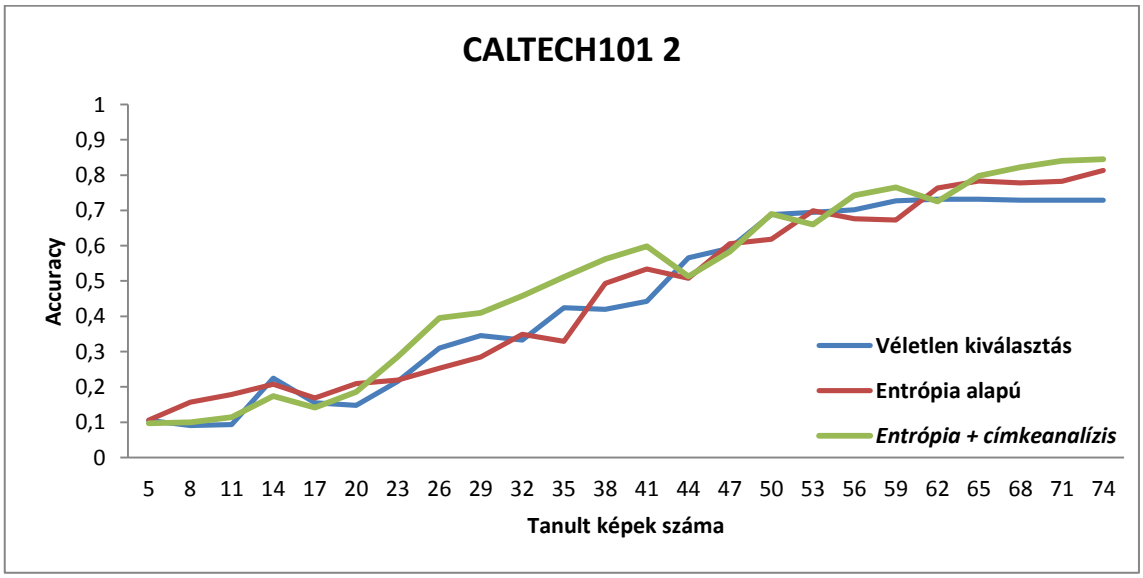
Ebben a fejezetben ismertetem az eredményeket, minden egyes teszthalmazhoz két diagram tartozik, egyiken az Accuracy másikon a MAP mutató került megjelenítésre, külön-külön az egyes lekérdezési stratégiák szerint. A kísérleti tapasztalataim azt mutatták, hogy a 36. egyenletben felírt β paramétert 0,1-nek választva elegendő nagyságú súlyt tudok a büntetőpontokhoz rendelni ahhoz, hogy az általam javasolt új eljárás kellően eltérjen az alapeljárástól. A Caltech101 5 kategóriájú részhalmazai közül az elsőnek eredményeit a 22. ábra és a 23. ábra mutatja, valamint a második ilyen részhalmaz tesztelésével kapott eredmények kiértékelését Accuracy alapján a 24. ábra, illetve MAP alapján 25. ábra jeleníti meg. A továbbiakban is hasonló sorrendben jelenítem meg az eredményeket, tehát azonos kategóriaszámú részhalmazokat egymást követően, elsőként Accuracy majd MAP szempontjából kiértékelve. Látható, hogy vegyesen egyik majd másik stratégia ad jobb eredményt, viszont mire a képek feléhez érünk, az utolsó mérési pontban a címkevizsgálattal kiegészített bizonytalansági mintavétel adja a legmagasabb mutatókat. Megfigyelhető, hogy akár 0,2-es eltérés is előfordul a két különböző részhalmaz tesztelése során, ebből is látszik, hogy a véletlennek nagy szerepe van, hiszen előfordulhat, hogy pont az 5 „legnehezebb” kategóriát sorsoltuk ki a 101-ből.



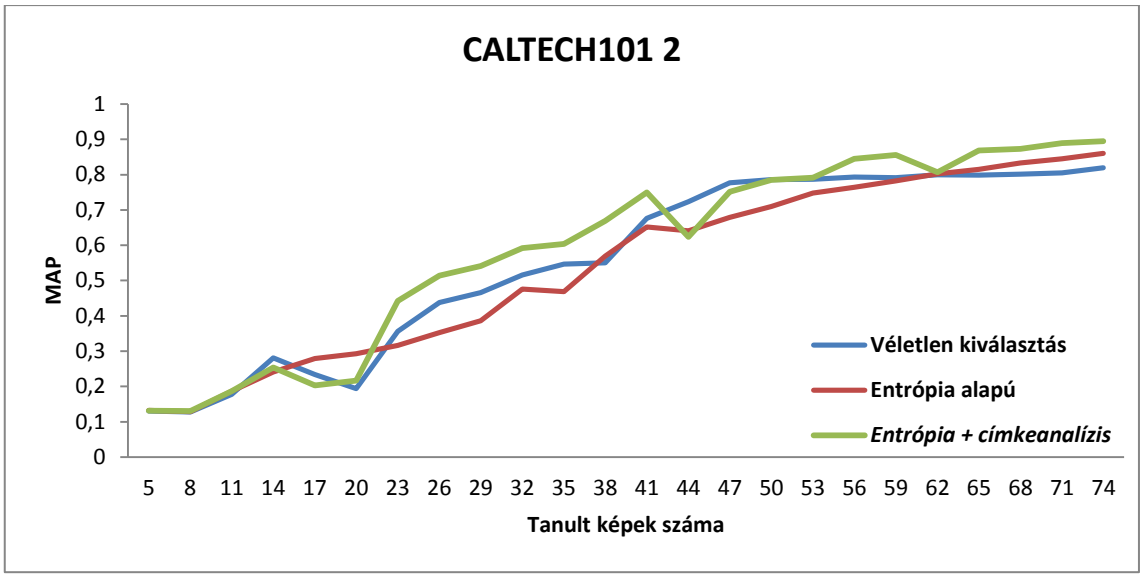
22. ábra: A Caltech101 képhalmazból véletlenszerűen választott 5 kategóriából álló részhalmazok közül az első tesztelése során kapott eredmények kiértékelése az Accuracy mutató szempontjából



23. ábra: A Caltech101 képhalmazból véletlenszerűen választott 5 kategóriából álló részhalmazok közül az első tesztelése során kapott eredmények kiértékelése a MAP mutató szempontjából

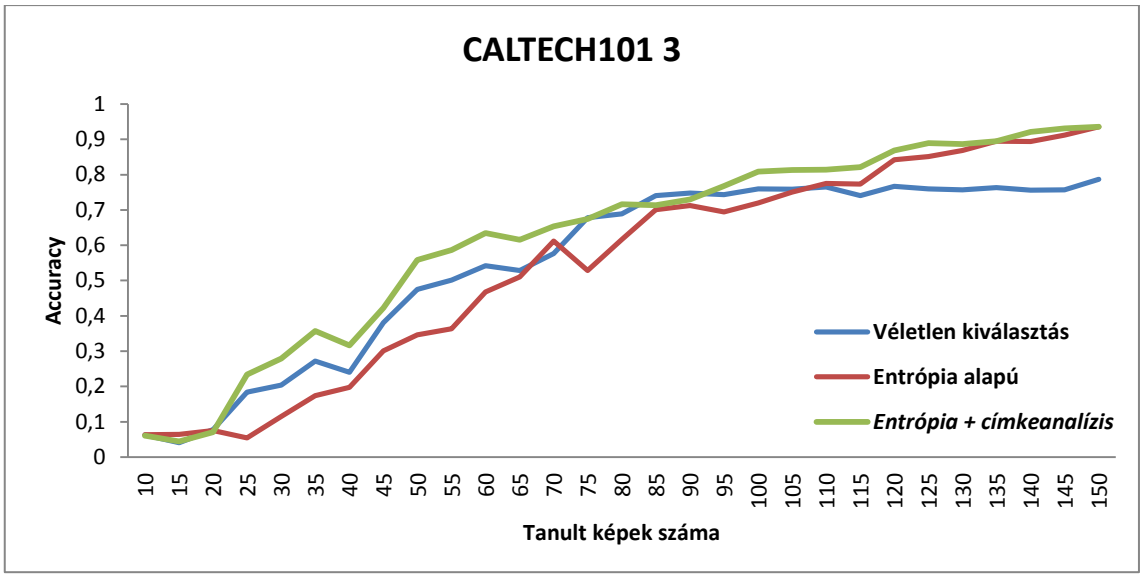


24. ábra: A Caltech101 képhalmazból véletlenszerűen választott 5 kategóriából álló részhalmazok közül a második tesztelése során kapott eredmények kiértékelése az Accuracy mutató szempontjából

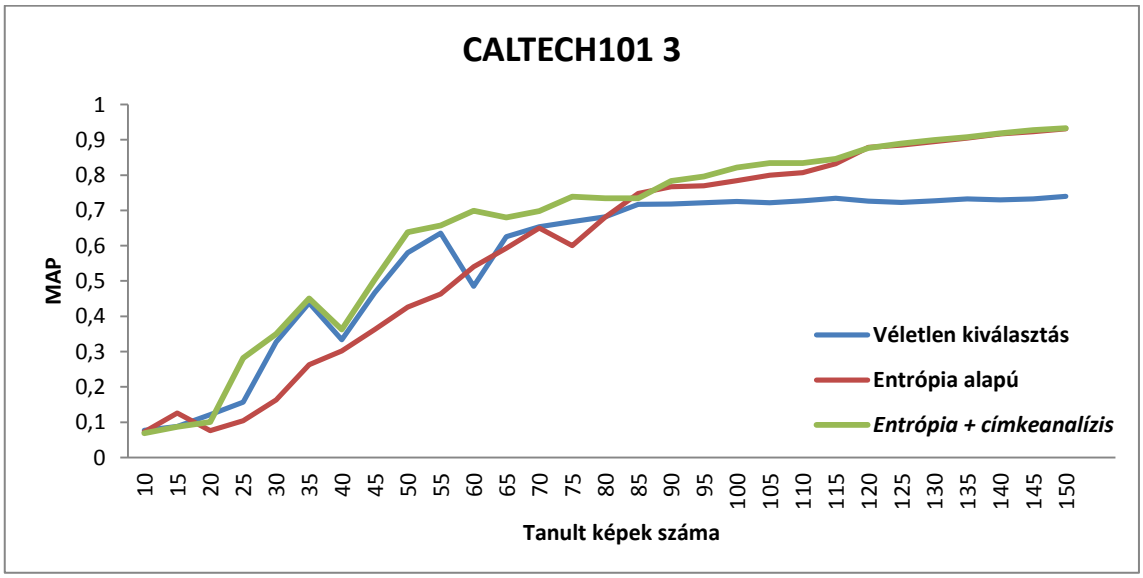


25. ábra: A Caltech101 képhalmazból véletlenszerűen választott 5 kategóriából álló részhalmazok közül a második tesztelése során kapott eredmények kiértékelése a MAP mutató szempontjából

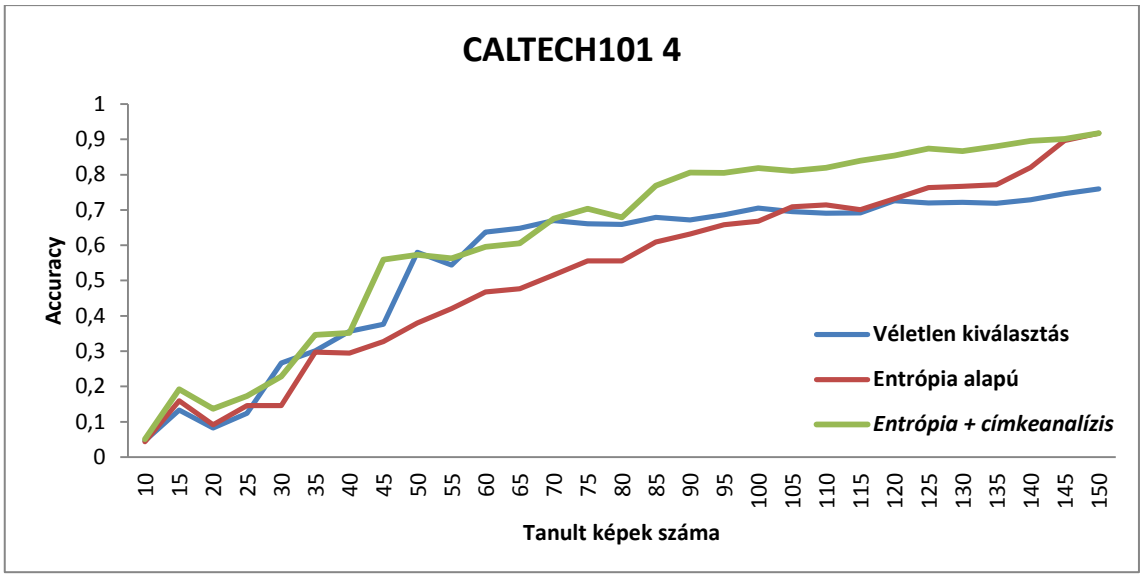
A 10 kategóriájú részhalmazok eredményét a 26. ábra, 27. ábra, 28. ábra és 29. ábra jeleníti meg. Itt már döntő részben a javasolt módszer adja a legjobb eredményt, illetve a képhalmaz felét megtanítva szintén mindig ez a legeredményesebb. Az is megfigyelhető, hogy az egyszerű entrópia alapú kiválasztás jóval elmarad a véletlen mintavételtől az első néhány lépésben, miközben a címkeanalízissel súlyozott változat követni, illetve túlteljesíteni is tudja azt. Itt egyértelműen megmutatkozik az az előny, ami az eloszlás vizsgálatából származik, mivel itt az történik, hogy a 2-es módszer sorra ugyanabból a kategóriából kérdezi le a képeket, az 1-es változatosan teszi ezt, mivel véletlenszerű, így jó eséllyel több különböző kategóriából válogat, végül a 3-as ötvözi a fenti kettőt és elosztottan választ magas információtartalmú címkéket.



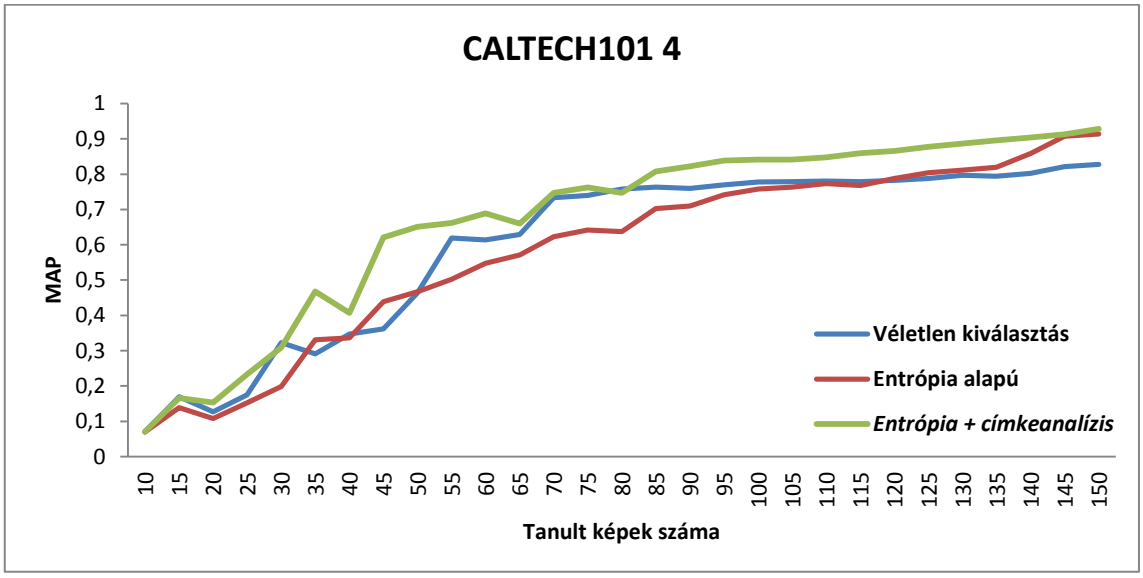
26. ábra: A Caltech101 képhalmazból véletlenszerűen választott 10 kategóriából álló részhalmazok közül az első tesztelése során kapott eredmények kiértékelése az Accuracy mutató szempontjából



27. ábra: A Caltech101 képhalmazból véletlenszerűen választott 10 kategóriából álló részhalmazok közül az első tesztelése során kapott eredmények kiértékelése a MAP mutató szempontjából

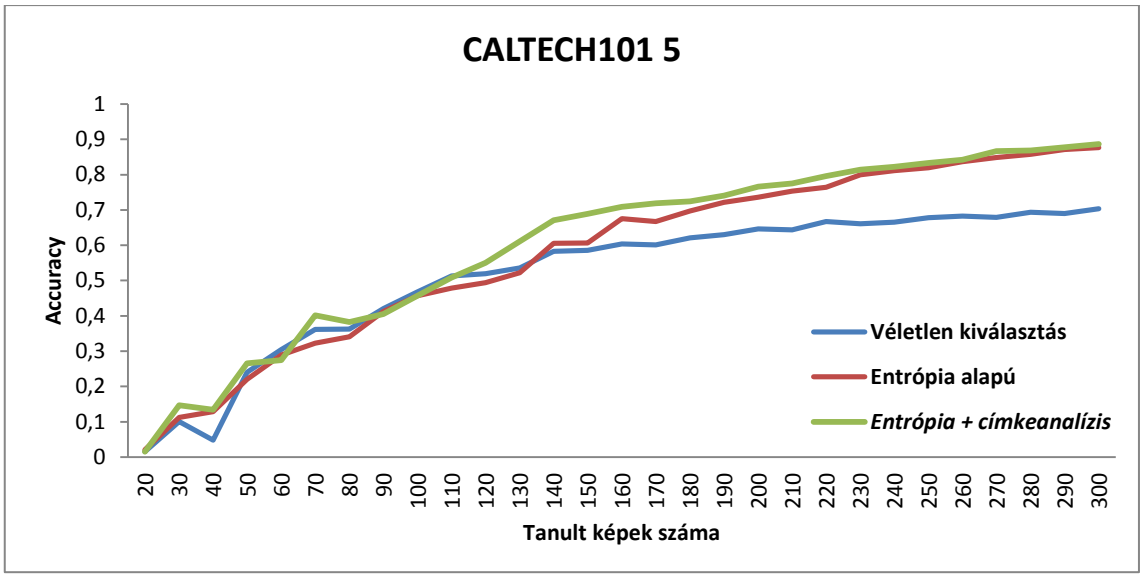


28. ábra: A Caltech101 képhalmazból véletlenszerűen választott 10 kategóriából álló részhalmazok közül a második tesztelése során kapott eredmények kiértékelése az Accuracy mutató szempontjából

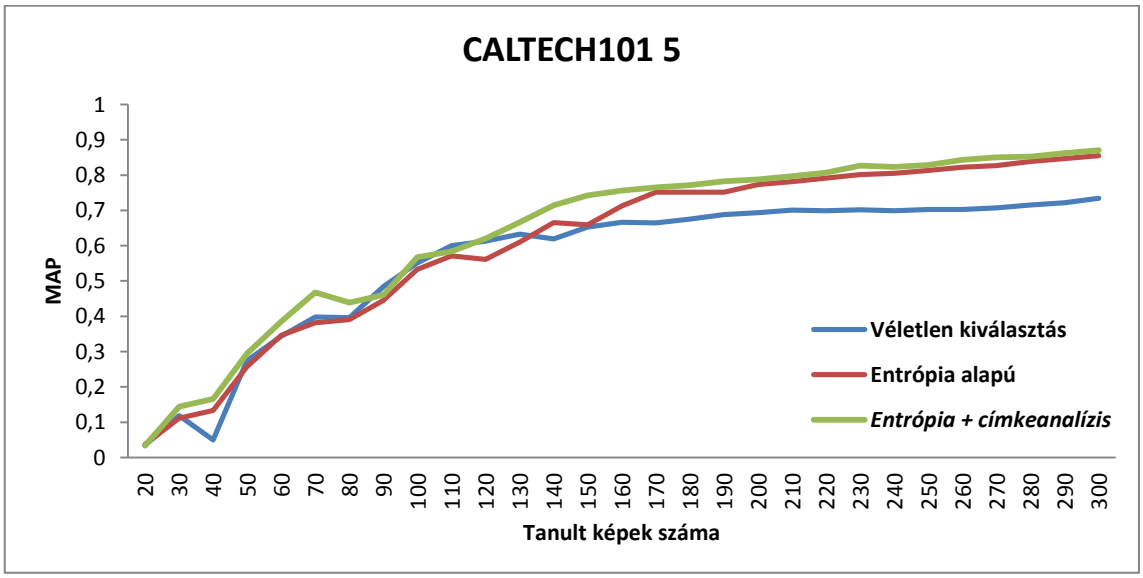


29. ábra: A Caltech101 képhalmazból véletlenszerűen választott 10 kategóriából álló részhalmazok közül a második tesztelése során kapott eredmények kiértékelése a MAP mutató szempontjából

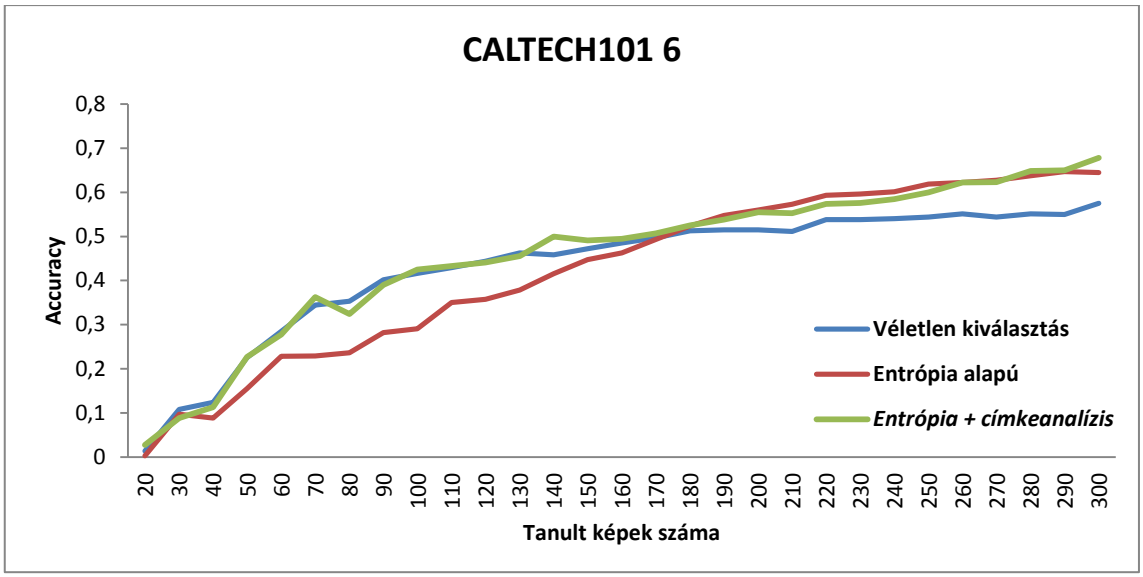
A következő négy diagram a 20 osztályból álló részhalmazokat, pontosabban az azok tesztelése során kapott eredmények kiértékelését mutatja. Az első (CALTECH101 5, lásd 30. ábra és 31. ábra) teszthalmaz esetén látható, hogy sokáig szinte együtt mozog a 3 módszer, viszont a véletlen kiválasztás hamarabb beáll egy „fix” szintre, a bizonytalansági mintavétel alapú módszerek pedig tovább emelkednek, és a kettő közül az eloszlás vizsgálattal kiegészített ad magasabb értékű mutatókat. Itt is igaz, hogy az utolsó mért pontban minden esetben a javasolt eljárás a legeredményesebb.



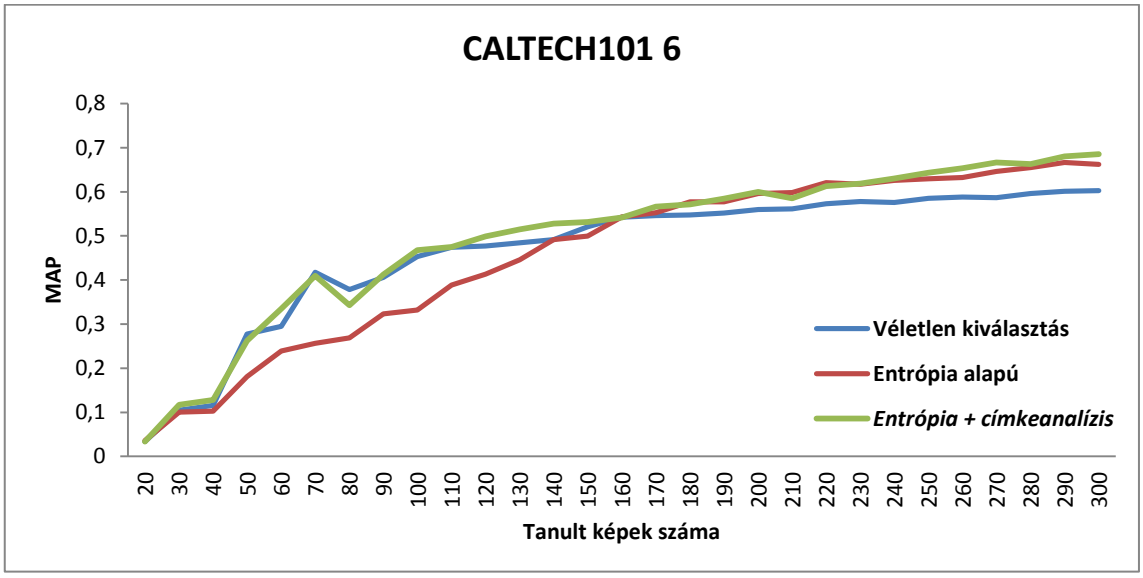
30. ábra: A Caltech101 képhalmazból véletlenszerűen választott 20 kategóriából álló részhalmazok közül az első tesztelése során kapott eredmények kiértékelése az Accuracy mutató szempontjából



31. ábra: A Caltech101 képhalmazból véletlenszerűen választott 20 kategóriából álló részhalmazok közül az első tesztelése során kapott eredmények kiértékelése a MAP mutató szempontjából

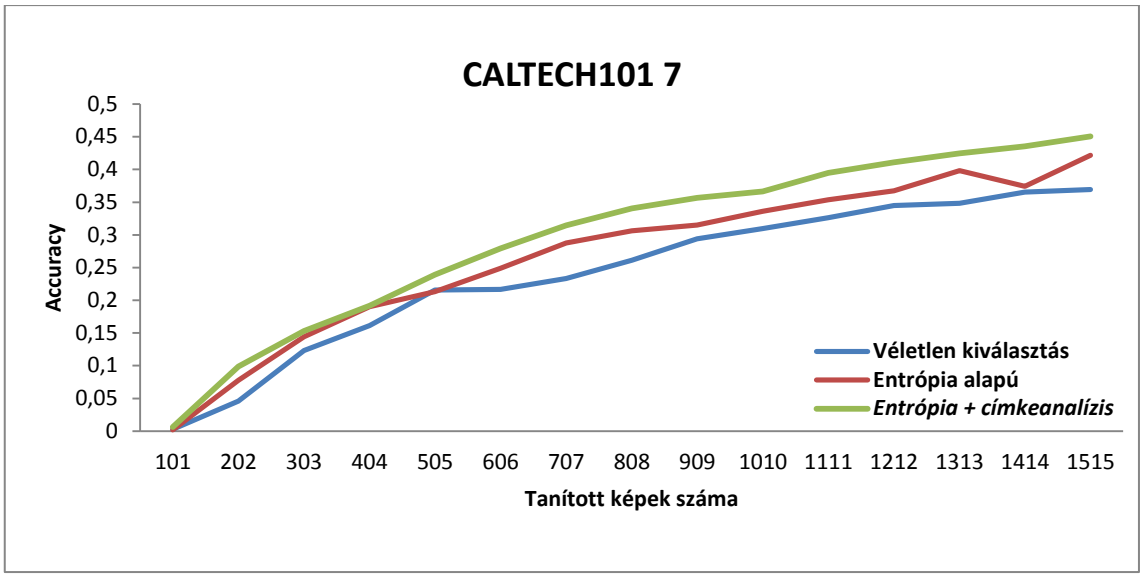


32. ábra: A Caltech101 képhalmazból véletlenszerűen választott 20 kategóriából álló részhalmazok közül a második tesztelése során kapott eredmények kiértékelése az Accuracy mutató szempontjából

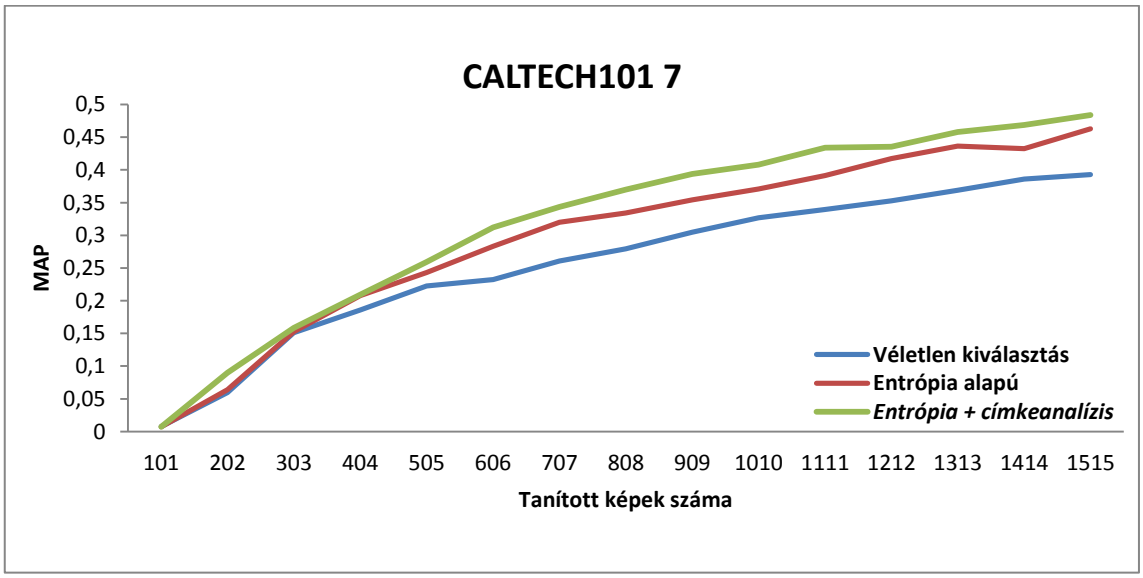


33. ábra: A Caltech101 képhalmazból véletlenszerűen választott 20 kategóriából álló részhalmazok közül a második tesztelése során kapott eredmények kiértékelése a MAP mutató szempontjából

Ez után a Caltech101 gyűjteményből minden kategóriát (a háttér kivételével) tartalmazó teszthalmaz osztályozása került sorra. A kapott eredményeket Accuracy alapján kiértékelve a 34. ábra, MAP alapján kiértékelve a 35. ábra foglalja össze. Jól látszik, hogy a 3-as módszer minden mért ponton megelőzi a további két módszert, úgy az Accuracy mint a MAP mutató szerint. Ahogy visszatekintünk ez eddigi diagramokra, elmondható, hogy a kategóriák számának növelésével, egyre nő a címkeanalízis használatának előnye.



34. ábra: A Caltech101 képhalmaz összes kategóriáját tartalmazó és osztályonként 30 elemű részhalmaz tesztelése során kapott eredmények kiértékelése az Accuracy mutató szempontjából



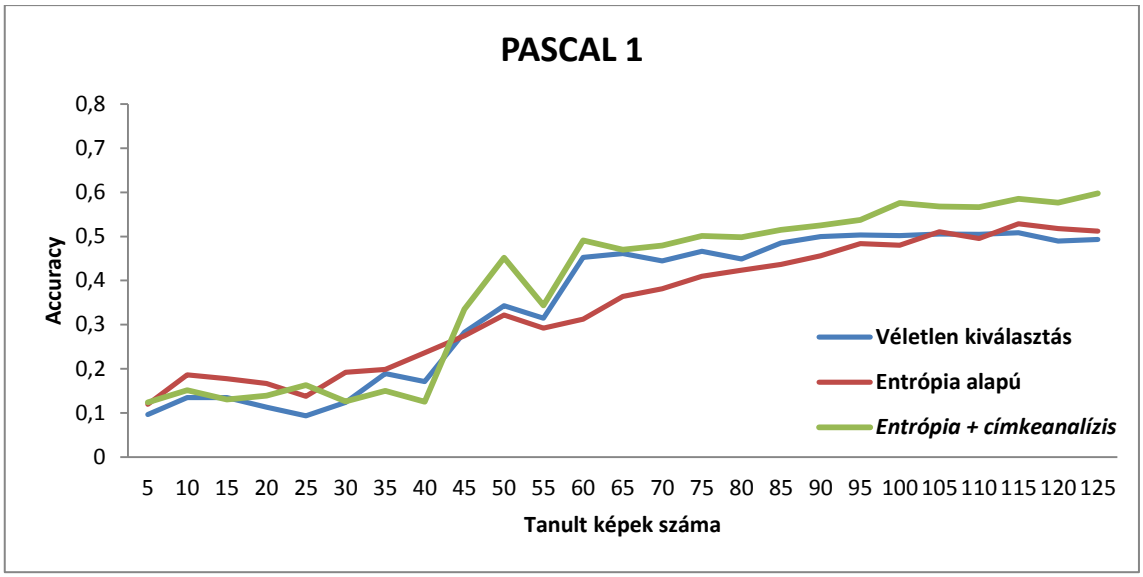
35. ábra: A Caltech101 képhalmaz összes kategóriáját tartalmazó és osztályonként 30 elemű részhalmaz tesztelése során kapott eredmények kiértékelése a MAP mutató szempontjából

A Caltech101 képekből készített teszthalmazok összesített eredményét a 7. táblázat mutatja. Itt az utolsó mért pontban vett mutatókat jelenítem meg, ahol 50-50 százalék a tanító és teszt képek aránya. Látható, hogy mindig a továbbfejlesztett eljárás adja a legjobb eredményt, néhány esetben fordul elő, hogy az eredménye pont megegyezik az egyszerű bizonytalansági mintavételezéssel. Továbbá, a kapott eredmények alapján elmondható, hogy a 3-as módszer nem csak megelőzi két „vetélytársát", de hatékonyan el is végzi az osztályzást.

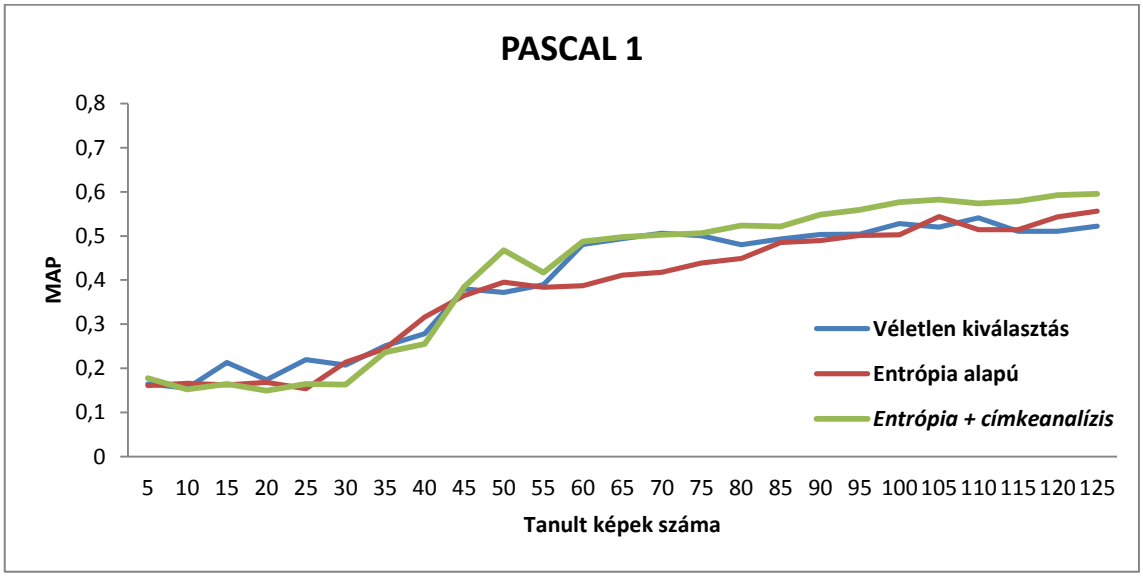
	Véletlen kiválasztás		Entrópia alapú		Entrópia + címkeanalízis	
	Accuracy	MAP	Accuracy	MAP	Accuracy	MAP
CALTECH101 1	0,945	0,972	1	0,99	***1***	***1***
CALTECH101 2	0,729	0,819	0,813	0,860	***0,848***	***0,895***
CALTECH101 3	0,787	0,739	0,935	0,931	***0,936***	***0,933***
CALTECH101 4	0,760	0,829	0,918	0,914	***0,918***	***0,928***
CALTECH101 5	0,703	0,735	0,877	0,855	***0,887***	***0,870***
CALTECH101 6	0,575	0,602	0,645	0,662	***0,678***	***0,686***
CALTECH101 7	0,369	0,393	0,421	0,462	***0,450***	***0,484***

7. táblázat: Caltech101 képeiből előállított teszthalmazok eredményei az utolsó mért pontban (50% tanító és 50% teszt kép)

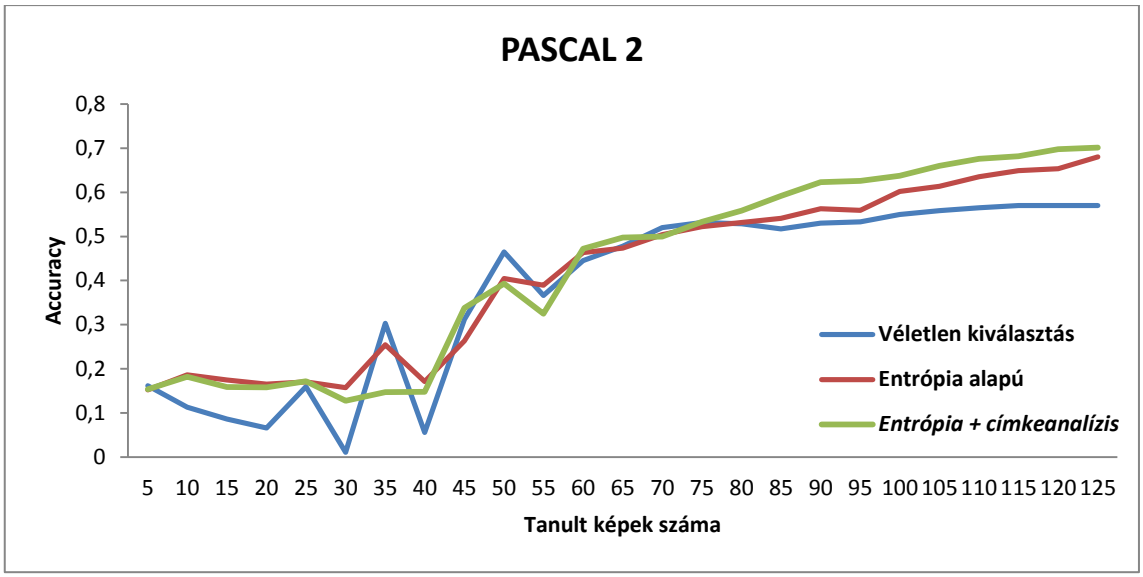
Az alábbi ábrák mutatják a PASCAL VOC2010 képekből alkotott 5 kategóriájú részhalmazok tesztelésének eredményeit (Accuracy: 36. ábra és 37. ábra, MAP: 38. ábra és 39. ábra). Jól látszik, hogy itt nehezebben osztályozható képekről van szó, mivel a mutatók jóval elmaradnak a Caltech101 5 osztályú részhalmazainál látottaktól. A három eljárás görbéje láthatóan sokszor keresztezi egymást, viszont 125 tanult képnél mindig a címke vizsgálattal kiegészített bizonytalansági mintavételezés van a legmagasabban.



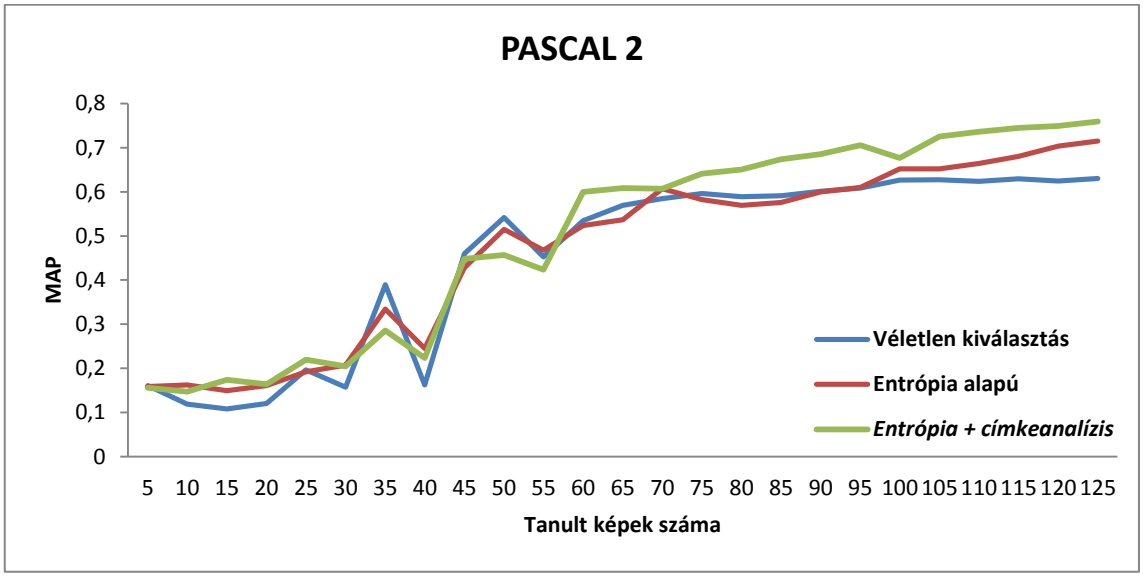
36. ábra: A PASCAL VOC2010 képhalmazból véletlenszerűen választott 5 kategóriából álló részhalmazok közül az első tesztelése során kapott eredmények kiértékelése az Accuracy mutató szempontjából



37. ábra: A PASCAL VOC2010 képhalmazból véletlenszerűen választott 5 kategóriából álló részhalmazok közül az első tesztelése során kapott eredmények kiértékelése a MAP mutató szempontjából



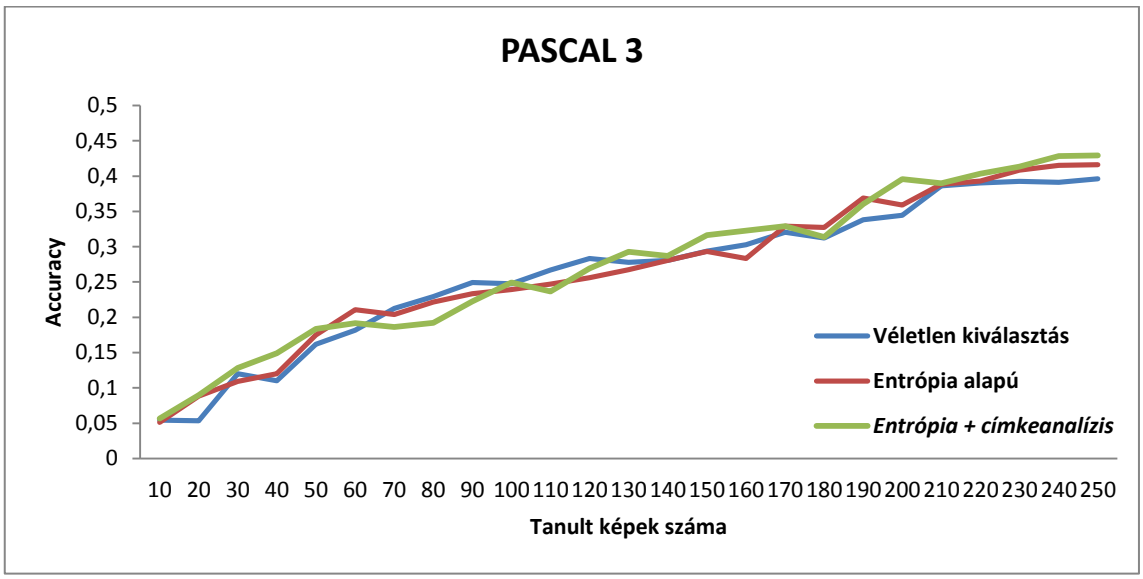
38. ábra: A PASCAL VOC2010 képhalmazból véletlenszerűen választott 5 kategóriából álló részhalmazok közül a második tesztelése során kapott eredmények kiértékelése az Accuracy mutató szempontjából



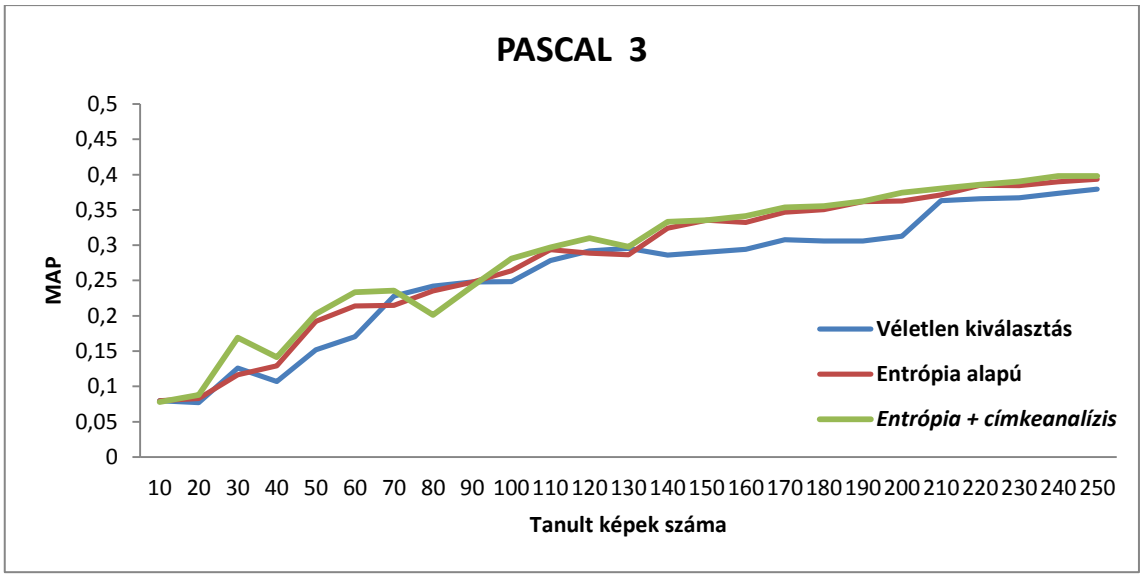
39. ábra: A PASCAL VOC2010 képhalmazból véletlenszerűen választott 5 kategóriából álló részhalmazok közül a második tesztelése során kapott eredmények kiértékelése a MAP mutató szempontjából

A fentebb írtak megfigyelhetők a 10 kategóriájú részhalmazoknál is, főként az első PASCAL 3 teszthalmaz esetén (lásd 40. ábra és 41. ábra). A PASCAL 4 teszthalmaznál pedig nagyjából 100 tanult kép után már jelentős javulást mutat a 3-as eljárás, ahogy a 42. ábra és 43. ábra diagramjai szemléltetik. Fontos megjegyezni, hogy ezen képek egy része többcímkés osztályozáshoz készült, melyre az én képosztályozó

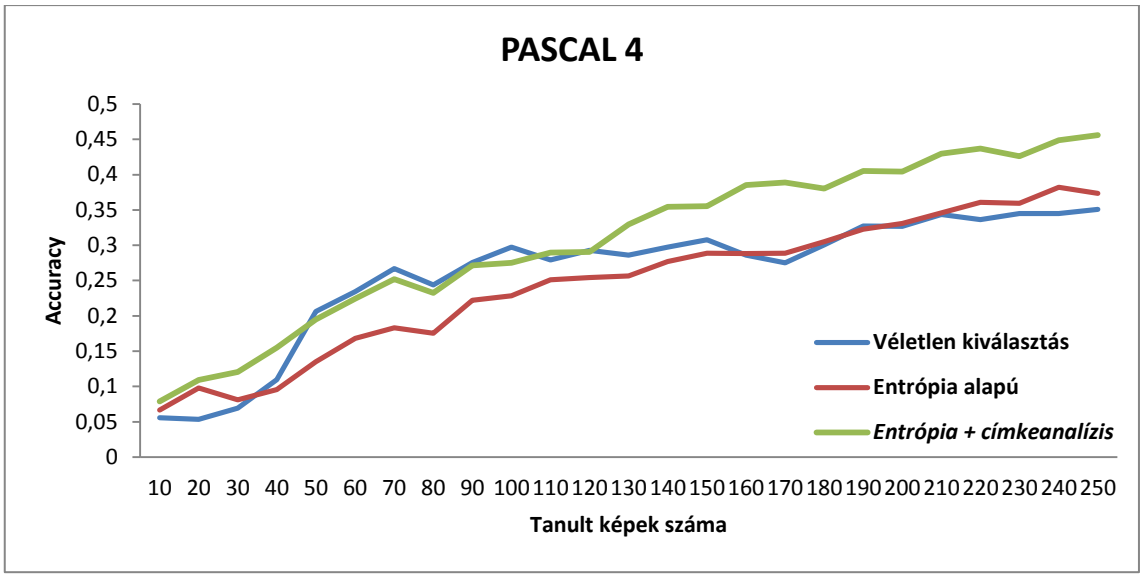
rendszerem nincs felkészítve. Ezért nem meglepő, hogy a mutatók elmaradnak a Caltech101 azonos kategóriaszámú részhalmazainál bemutatott eredményektől.



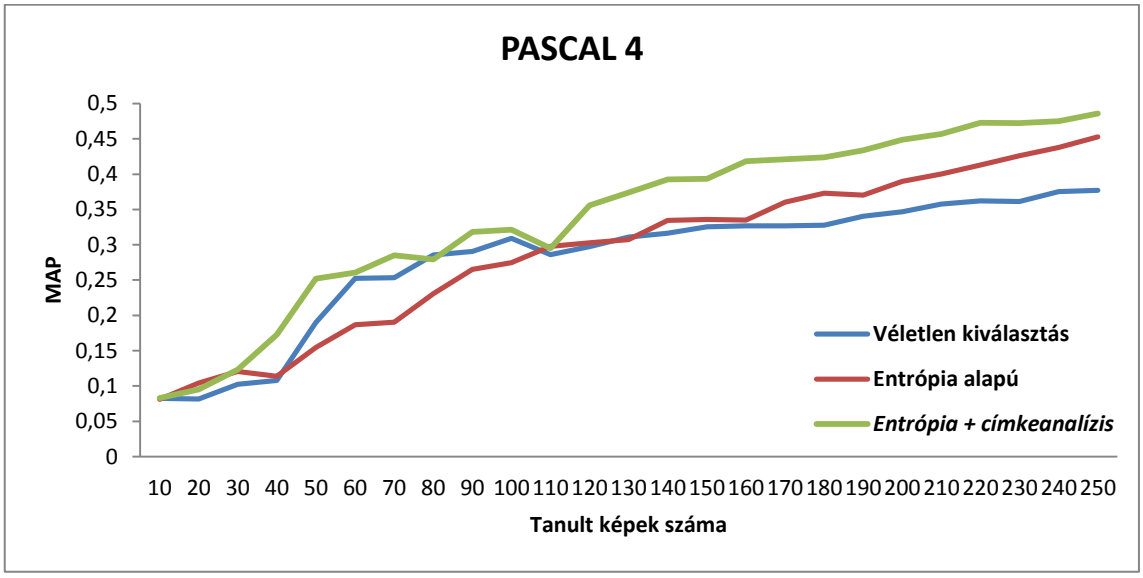
40. ábra: A PASCAL VOC2010 képhalmazból véletlenszerűen választott 10 kategóriából álló részhalmazok közül az első tesztelése során kapott eredmények kiértékelése az Accuracy mutató szempontjából



41. ábra: A PASCAL VOC2010 képhalmazból véletlenszerűen választott 10 kategóriából álló részhalmazok közül az első tesztelése során kapott eredmények kiértékelése a MAP mutató szempontjából

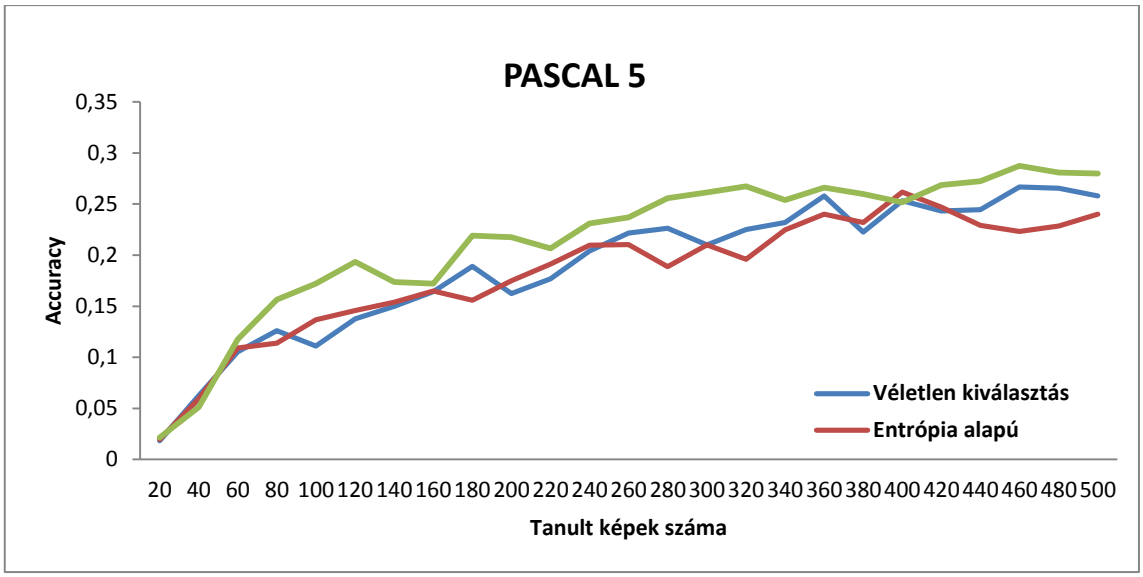


42. ábra: A PASCAL VOC2010 képhalmazból véletlenszerűen választott 10 kategóriából álló részhalmazok közül a második tesztelése során kapott eredmények kiértékelése az Accuracy mutató szempontjából

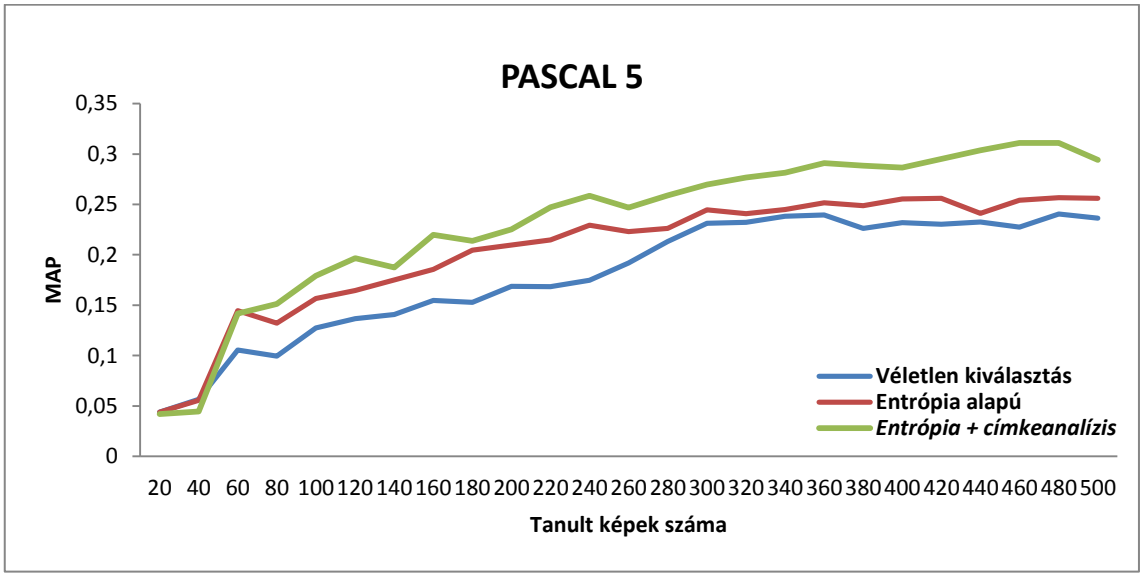


43. ábra: A PASCAL VOC2010 képhalmazból véletlenszerűen választott 10 kategóriából álló részhalmazok közül a második tesztelése során kapott eredmények kiértékelése a MAP mutató szempontjából

A PASCAL VOC2010 tanító és validáló képei estén is végeztem tesztet az összes kategóriával. Ennek eredményét a 44. ábra (Accuracy alapján) és a 45. ábra (MAP alapján) mutatja. Ahogy fentebb már említettem, a kategóriák számának növelésével a javulás mértéke is emelkedik. Az Accuracy érékeknél néhány kivételtől eltekintve, végig mindenhol az eloszlás vizsgálattal kiegészített entrópia számítás adja a legjobb eredményt.



44. ábra: A PASCAL VOC2010 képhalmaz összes kategóriáját tartalmazó és osztályonként 50 elemű részhalmaz tesztelése során kapott eredmények kiértékelése az Accuracy mutató szempontjából



45. ábra: A PASCAL VOC2010 képhalmaz összes kategóriáját tartalmazó és osztályonként 50 elemű részhalmaz tesztelése során kapott eredmények kiértékelése a MAP mutató szempontjából

Az utolsó mért pontban számított mutatókat a 8. táblázat foglalja össze. Látható, hogy a PASCAL teszthalmazok esetén is a javasolt módszer eredményezi a legmagasabb mutatókat.

	Véletlen kiválasztás		Entrópia alapú		Entrópia + címkeanalízis	
	Accuracy	MAP	Accuracy	MAP	Accuracy	MAP
PASCAL 1	0,493	0,522	0,512	0,556	***0,593***	***0,596***
PASCAL 2	0,570	0,630	0,680	0,715	***0,701***	***0,759***
PASCAL 3	0,396	0,379	0,416	0,394	***0,429***	***0,398***
PASCAL 4	0,351	0,377	0,373	0,453	***0,456***	***0,486***
PASCAL 5	0,258	0,236	0,240	0,256	***0,280***	***0,294***

8. táblázat: PASCAL VOC2010 tanító és validáló képeiből előállított teszthalmazok eredményei az utolsó mért pontban (50% tanító és 50% teszt kép)

7 Összefoglalás

A dolgozatban részletesen bemutattam az általam készített intelligens képfeldolgozó rendszert, amely passzív és aktív tanuló algoritmusok segítségével képes iteratívan osztályozni tetszőleges képhalmazt (vizuális hasonlóságok alapján). A munkám egyik fő célja az emberi és gépi tanulás összehasonlítása volt, melynek emberi részéhez egy hallgató társammal közösen dolgoztunk ki egy mérési tervet, és az általa megvalósított webes felületen keresztül mértük a tesztelő személyek által elért eredményeket. Az eredményeket kiértékeltem, és egyező beállítások mellett gépi tanulást használva is leosztályoztam összesen 5 különböző teszthalmazt, majd ezek eredményeinek kiértékelése következett. Az eredményeket összehasonlítottam osztályozási pontosság szempontjából, illetve egy csúszó ablak segítségével a tényleges tanulási folyamat „gyorsaságát” is lemértem (főként a gépi tanulásnál látszott ez).

A másik cél a passzív és aktív tanulás összehasonlítása volt, valamint egy meglévő lekérdezési stratégia továbbfejlesztése, és a javulás mértékének megfigyelése. Ennek elvégzése érdekében több teszthalmazt is készítettem, a PASCAL VOC2010 tanító és validáló képeiből, valamint a Caltech101 gyűjteményből. Az egyes teszthalmazokat több alkalommal is osztályoztam, majd az eredményeket átlagoltam. Az eredményeket osztályozási pontosság (accuracy) és átlagos pontosság (mean average preicison) szempontjából értékeltem ki, és a kiegészített eljárással sikerült minden tesztnél legmagasabb eredményt elérni.

A jövőben további aktív tanulási stratégiák továbbfejlesztésének tervét és implementációját szeretném elkészíteni. Továbbá, absztrakt (gép által készített) képeken is kipróbálom a tanulási folyamatokat. Itt az emberi tanulás és gépi tanulás összehasonlítása lesz izgalmas, hiszen az ilyen típusú képek nagy valószínűséggel nem köthetők korábbi emberi tapasztalathoz, így az összehasonlítás sokkal reálisabb képet mutat majd.

Irodalomjegyzék

[1] A. McCallum and K. Nigam. Employing EM in pool-based active learning for text classification. In Proceedings of the International Conference on Machine Learning (ICML), pages 359–367. Morgan Kaufmann, 1998.

[2] Altrichter Márta, Horváth Gábor, Pataki Béla, Strausz György, Takács Gábor, Valyon József: *Neurális hálózatok*, PANEM, 2006, http://mialmanach.mit.bme.hu/neuralis/ch06s03, (letöltés dátuma: 2015.10.26.)

[3] B. Settles and M. Craven. An analysis of active learning strategies for sequence labeling tasks. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 1069–1078. ACL Press, 2008.

[4] B. Settles, M. Craven, and S. Ray. Multiple-instance active learning. In Advances in Neural Information Processing Systems (NIPS), volume 20, pages 1289– 1296. MIT Press, 2008.

[5] Boser, B. - Guyon, I. - Vapnik, V.: *A Training Algorithm for Optimal Margin Classifier*, Proc. of the 5th Annual ACM Workshop on Computational Learning Theory, 1992, pp. 144-152.

[6] Burr Settles, Active Learning Literature Survey, Computer Siences Technical Report 1648, University of Wisconsin – Madison, 2009.

[7] C. Campbell, N. Cristianini, and A. Smola. Query learning with large margin classifiers. In ICML, 2000.

[8] C. Harris, M. Stephens: *A combined corner and edge detector*. In C. J. Taylor, editors, Proceedings of the Alvey Vision Conference, pages 23.1-23.6. Alvey Vision Club, September 1988. doi:10.5244/C.2.23.

[9] Carlo Tomasi: *Estimating Gaussian Mixture Densities with EM – A Tutorial*, Duke University. http://www.cs.duke.edu/courses/spring04/cps196.1/handouts/EM/tomasiEM.pdf (letöltés dátuma: 2015.10.26.)

[10] Corinna Cortes, Vladimir Vapnik: *Support-vector networks*, Machine Learning, Volume 20, Number 3, 1995, pp. 273-297.

[11] D. Cohn, L. Atlas, R. Ladner, M. El-Sharkawi, R. Marks II, M. Aggoune, and D. Park. Training connectionist networks with queries and selective sampling. In Advances in Neural Information Processing Systems (NIPS). Morgan Kaufmann, 1990.

[12] D. Lewis and J. Catlett. Heterogeneous uncertainty sampling for supervised learning. In Proceedings of the International Conference on Machine Learning (ICML), pages 148–156. Morgan Kaufmann, 1994.

[13] D. Lewis and W. Gale. A sequential algorithm for training text classifiers. In Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval, pages 3–12. ACM/Springer, 1994.

[14] Dempster, A., Laird, N., Rubin, D.: *Maximum Likelihood from Incomplete Data via the EM Algorithm*, Journal of the Royal Statistical Society 39(1), 1977, pp. 1–38.

[15] Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J. and Zisserman, A. *The PASCAL Visual Object Classes (VOC) Challenge*, International Journal of Computer Vision, 88(2), 303 338, 2010.

[16] Florent Perronnin, Chris Dance: *Fisher kernel on visual vocabularies for image categorization*, CVPR, Computer Vision and Pattern Recognition, 2007.

[17] G. Schohn and D. Cohn. Less is more: Active learning with support vector machines. In ICML, 2000.

[18] H.S. Seung, M. Opper, and H. Sompolinsky. Query by committee. In Proceedings of the ACM Workshop on Computational Learning Theory, pages 287–294, 1992.

[19] I. Dagan and S. Engelson. Committee-based sampling for training probabilistic classifiers. In Proceedings of the International Conference on Machine Learning (ICML), pages 150–157. Morgan Kaufmann, 1995.

[20] Jaakkola TS, Haussler D.: *Exploiting generative models in discriminative classifiers*. Advances in Neural Information Processing Systems (NIPS), Vol. 11, 1998, pp. 487-493.

[21] Jingbo Zhu, Huizhen Wang, Benjamin K. Tsou, Matthew Ma. 2010. Active Learning with Sampling by Uncertainty and Density for Data Annotations. IEEE Transactions on Audio, Speech and Language Processing (TASL), 18(6):1323-1331.

[22] Jingbo Zhu, Matthew Ma. 2012. Uncertainty-based Active Learning with Instability Estimation for Text Classification. ACM Transactions on Speech and Language Processing (TSLP), 8(4), article 5:1-21.

[23] Jorge Sánchez, Florent Perronnin, Thomas Mensink: *Improved Fisher Vector for Large Scale Image Classification*, COMPUTER VISION – ECCV 2010. Lecture Notes in Computer Science, 2010, Volume 6314/2010, pp. 143-156.

[24] Kató Zoltán: *Sarokpontok detektálása*, Képfeldolgozás és Számítógépes Grafika tanszék SZTE. http://www.inf.u-szeged.hu/~kato/teaching/DigitalisKepfeldolgozasTG/07-CornerDetection.pdf (letöltés dátuma: 2015.10.26.)

[25] L. Fei-Fei, R. Fergus and P. Perona. Learning generative visual models from few training examples: an incremental Bayesian approach tested on 101 object categories. IEEE. CVPR 2004, Workshop on Generative-Model Based Vision. 2004.

[26] L. Fei-Fei, R. Fergus, and A. Torralba: *Recognizing and Learning Object Categories*, Part1 – Single object classes: *Bag of Words models, Part-based models, and Discriminative models,* 2009, pp. 2-16.

[27] Lowe, D. G.: *„Distinctive Image Features from Scale-Invariant Keypoints”*, International Journal of Computer Vision, 60, 2, pp. 91-110, 2004.

[28] Mikolajczyk, K., Schmid, C.: *Scale & affine invariant interest point detectors,* International Journal on Computer Vision 60(1), 2004, pp. 63-86.

[29] N. Abe and H. Mamitsuka. Query learning strategies using boosting and bagging. In Proceedings of the International Conference on Machine Learning (ICML), pages 1–9. Morgan Kaufmann, 1998.

[30] Reynolds, D. A.: *Gaussian Mixture Models*, Encyclopedia of Biometric Recognition, Springer, Journal Article, February 2008. http://www.ll.mit.edu/mission/communications/ist/publications/0802_Reynolds_Biometrics-GMM.pdf (letöltés dátuma: 2014.10.07.)

[31] S. Kullback and R.A. Leibler. On information and sufficiency. Annals of Mathematical Statistics, 22:79–86, 1951.

[32] S. Lazebnik, A. Torralba, L. Fei-Fei, D. Lowe, C. Szurka: *Bag-of-Words models*, Lecture 9, 2012, pp. 1-32.

[33] S. Tong and D. Koller. Support vector machine active learning with applications to text classification. In Proceedings of the International Conference on Machine Learning (ICML), pages 999–1006. Morgan Kaufmann, 2000

[34] Szegedi Tudományegyetem - Informatikai Tanszékcsoport, Mesterséges Intelligencia II: *Felügyelet nélküli tanulás – A K-means klaszterező*, http://www.inf.u-szeged.hu/~ormandi/ai2/03-k-means.pdf (letöltés dátuma: 2015.10.26.)

[35] T. Mitchell. Generalization as search. Artificial Intelligence, 18:203–226, 1982.

[36] T. Scheffer, C. Decomain, and S. Wrobel. Active hidden Markov models for information extraction. In Proceedings of the International Conference on Advances in Intelligent Data Analysis (CAIDA), pages 309–318. Springer-Verlag, 2001.