Recognita történet

Kovács Emőke, Marosi István

Életünk az Elméleti Laborban

Kakukktojás voltam az Elméleti Laborban a sok prológos munkatárs között. Képfeldolgozással foglalkoztam, azazhogy csak foglalkoztam volna, mert semmilyen speciális képi eszközöm nem volt. Így csak a kockás papír felett ábrándoztam, többnyire arról, hogyan kellene felismerni a betűket. Nem sok hasznomat vették akkoriban, ennek ellenére Bálint és a laborbeli munkatársak szelíden, kedvesen, befogadtak, ezt ezúttal is köszönöm.

1984-ban vége szakadt az álmodozások korának, Náray Zsolt megbízott egy karakterfelismerő program fejlesztésével, és mindehhez egy TV kamerát is kaptam. A TV kamerák többtónusú képet adnak, teljesen alkalmatlanok a fekete-fehér kép leképzésére. Voltak már akkoriban szkennerek is a nagyvilágban, de a szigorú embargó miatt elérhetetlenek voltak számunkra.

Nekem a kamera is megfelelt, boldogan munkához láttam. Mindenekelőtt megpróbáltam előállítani egy viszonylag stabil bináris képet, de ez kamerával szinte lehetetlen. Bele kellett törődnöm, hogy ha bealkonyodik, a betűim meghíznak, ha kisüt a nap, elvékonyodnak. A nem megfelelő eszközből adódó hátrányok végül is előnnyé váltak, egész jól szimulálták a változatos betűvilágot, rögtön a munka kezdetén szembesítettek az OCR valódi nehézségeivel.

A szakirodalom a maszkillesztést ajánlotta a betűk felismerésére, de ezt elvetettem. Egyszerűen megvalósítható, de idő és memóriaigényes módszernek tűnt, féltem, hogy túlnő a PC-k kapacitásán, egyébként sem tetszett a vonalvastagságtól való függése miatt. Ehelyett a betűk kontúr vonalát analizáltam, felosztva azt konkáv és konvex ívszakaszokra. Ezeknek az egyszerű íveknek a helye, hossza, iránya, görbülete már egész jól leírta a betűt. Ez a módszer olyan rugalmas volt, hogy akár a kézzel írt blokkbetűket is felismerte. A betűket leíró tulajdonság vektor komponensei geometriai és topológiai sajátságokat számszerűsítettek, ezek többsége vizuálisan is ellenőrizhető volt, ennek a későbbi fejlesztés során nagy hasznát vettük.

Az első kerek-egész OCR programunk

1985-ben új ember került a laborba, a frissen végzett Marosi István, aki a SAS grafika írása közben nagy érdeklődéssel figyelte, hogy mit csinálok. Egyszer elkérte a Pascal nyelvű program forrásait, hazavitte, reggelig „elolvasta”, megfejtette az akkor már terjedelmes programot. Megtetszett neki a téma, kérte Bálintot, hogy ő is ezzel foglalkozhasson. Bálint rábólintott.
Ebben az időben egy külföldön élő magyar járt az intézetben, látva az OCR programot egy Microtek szkennert ajándékozott nekünk. Ezzel kezdetét vette az igazi munka. István egy-két nap alatt illesztette a szkennert, hat hét alatt átírtuk a Pascal programot assemblerre. Megtanítottunk néhány alapvető fontot. A program gyorsan és hibátlanul működött a jó minőségű nyomtatványokon, bizonyos betűtípusokra.

Bemutatásra igen alkalmas volt, mutogattuk is naphosszat, de valódi használatra még nem. Mi tele voltunk tervekkel, tudtuk, hogy sok-sok problémát kell még megoldani, de azt aligha gondoltuk, hogy életünk elkövetkezendő húsz évét ez a munka fogja betölteni.

Az OCR legnagyobb problémája: a szegmentálás

A betűket felismerni nem olyan nehéz dolog, de szegmentálni, azaz meghatározni a betűt alkotó képpontok összességét, néha nagyon nehéz. A halvány, vékony vonal sok részre szakad, a bold írás betűi összeolvadnak. Ezek a jelenségek nagyon durvák tudnak lenni, amint a mellékelt képeken is látszanak.

A Recognita korrekciós algoritmusok sokaságát tartalmazza, amelyek a szétszakadt vonalakat összekötik, az összeragadt betűket részekre vágják, majd újra megkísérlik felismerni a betűket. A korrekciós algoritmusok jósága, kidolgozottsági szintje meghatározó az OCR pontosságában. Terjedelmesebbek, munkaigényesebbek, mint maguk a felismerő alap algoritmusok.

Mégiscsak jó valamire a maszkillesztés

1989-ben kiváltunk az Elméleti Laborból, megalakult a Recognita Rt., ezzel véget is ért a békés életünk. A program már piacon volt, erős versenyben állt két amerikai és egy orosz fejlesztésű OCR programmal. A Recognita sebessége messze felülmúlta a versenytársakét, de pontosság tekintetében néha alulmaradt. Új módszert kellett találnunk a pontosság növelésére. Adaptív osztályozást, a maszkillesztés egy speciális alkalmazását dolgoztuk ki: Az elsődleges, kontúr analízisen alapuló felismerés során hibátlanul felismert betűk képét és kódját a program eltárolta, ezekből készültek el a maszkok. Ez után a lapot újra felismerve, most már a maszkok segítségével azonosította az első menetben bizonytalan vagy felismerhetetlen betűket. Az ily módon alkalmazott maszkillesztés elég biztonságosan működött, hiszen, a maszkok nem az egész betűvilágot írták le, csak az aktuális lapon találhatókat. Ahhoz, hogy sebességbeli előnyünket ne kelljen feláldozni, az automatikusan nyert, egymáshoz hasonlatos maszkokból betű osztályokat képeztünk egy speciális, némileg a neuron hálókra emlékeztető osztályozó program segítségével. A felismerés során az azonosítandó betűt tartalmazó osztályt kellett megtalálni, ami igen gyors művelet volt. A kétmenetes felismerés sokat javított a pontosságon.

Hatásos tanítás: megtanuljuk a hibáinkat is

A Recognita jó néhány tanító és tanuló algoritmust tartalmaz. A fejlesztő megtaníthatja a reprezentáns betűket, tetszőlegesen bővítheti ezt a betűtárat, az itt szereplő betűk kontúr analízissel nyert adataiból előállíthatja az elsődleges osztályozó fát. A program képes automatikusan megtanulni az olvasandó lapon lévő hibátlan betűket, az előzőkben leírtak szerint. Végül a felhasználó ki tudja javítani a hibásan felismert szöveget. A hibás betű kijavításakor egy bonyolult tanulási folyamat indul el. A program eltárolja a hibásnak talált betű képét, de megtanulja azt is, hogy miről mire tévedett, azaz eltárolja a betű valódi és hibásan felismert kódját is. Ez után végignézi az egész lapot, hasonló szituációkat keresve. Ha például a

felhasználó kijavított egy l betűt i betűre, akkor a program megvizsgálja az összes l betűt, illeszti a bitképét a javítotthoz, és ha hasonlónak találja, automatikusan kijavítja i-re. Az OCR tipikus, ismétlődő hibákat vét, ráadásul nehéz észrevenni, mert vizuálisan hasonló alakokat kever össze, nem úgy, mint a gépírónő, aki tipikusan felcseréli az egymás melletti betűket, vagy melléüt, amit könnyebb észrevenni. Úgy gondoltuk, hogy nagyon hasznos eszközt adunk a felhasználók kezébe. Sajnos azt tapasztaltuk, hogy csak kevesen használták, máig sem értjük, hogy miért.

Segít-e a szótár?

A kilencvenes évek közepétől már csaknem minden nyelvhez jó minőségű szótárak álltak rendelkezésre, kézenfekvő volt a gondolat, használjuk fel a szótár információt az OCR szöveg javítására. Sokan csodafegyvernek gondolták a szótár bevetését, de mi nem voltunk olyan optimisták. Az OCR hibák jó része csomósan jelentkezik a szegmentálás kudarcai miatt. Meg aztán a szótárak nem tartalmazzák a tulajdonneveket, meg a szakkifejezéseket, az újdonsült szavak is gyakran hiányoznak. Nehéz egy információt úgy használni, hogy annak csak bizonyos fokig hihetünk. Az első próbálkozásunk során a felismerési folyamat legvégén a kész eredményekre használtuk a szótárat, de ez nem volt igazán eredményes. Elvitte ugyan a hibák egynegyedét, de el is rontott valamivel többet. Végül is a felismerés teljes folyamatában felhasználtuk a szótár információit, gondosan egybevetve azt az alaki információkkal.

A mesterséges intelligencia programjai mind küszködnek egy problémával, nevezetesen, hogy felismernek, kitalálnak egy csomó mindent, de azt nem tudják jól jellemezni, hogy ez mennyire biztos, mennyire igaz. A felhasználóink szerették volna, ha egyértelműen kijelöltük volna számukra a hibás betűket, ehelyett vagy túl sokat, vagy túl keveset jelöltünk ki. Miért olyan nehéz számszerűsíteni a felismerés valószínű pontosságát? Nézzünk egy példát. Az ábrán látható egy c és egy e betű, az e betű vízszintes összekötő egyenese kiesett, ez tömegével előforduló jelenség a halvány képeken. A két betű között néhány képpont különbség van csak.

Az alakfelismerő programnak kutyakötelessége ezt a betűt hibátlan c betűnek látni. Csökkentheti ugyan a valószínűséget az, ha a szomszédos betűk mássalhangzók, de ez már nyelvfüggő feltétel, nem mindig igaz, például a németben gyakori az sch.

A felismerési valószínűség hitelesebb megközelítésében sokat segített a szótár. Ennek örült a felhasználó, de mi is nagy hasznát vettük, nem korrigáltunk feleslegesen, csak szótári szavakból vettük a mintabetűket, ezzel elkerültük azt, hogy egy hibásan felismert, automatikusan felvett mintabetűvel elfertőzzük az egész lapot. A felismerési hibastatisztikáinkat beépítettük a nyelvi analízis eljárásaiba. Ha egy szót nem találtunk szótárinak, akkor megkerestük a legrosszabbul felismert betűt vagy betűcsoportot, majd ezt helyettesítettük azokkal a betűkkel, amelyeket a hibastatisztikák szerint a leggyakrabban tévesztett el a program, majd az így generált szót ellenőriztük újra a szótárral.
Természetesen különös óvatossággal kellett kezelni a nagybetűvel kezdődő szavakat, a rövid vagy kötőjellel kapcsolt szavakat.

Minden óvatosság ellenére a nyelvi analízissel néha rontottunk is, mókás eredményeket szültünk. Egyszer egy Surján László által kiadott levelet olvastunk be. A levél alján volt a géppel írt aláírás, amit a kézjegy jócskán lefedett. A program kínlódott a lap alján, vágott, kötözött, kontúrt tisztított, szótárt használt, míg végre győzedelmesen kiírta az eredményt: Sírján Zászló.
A manapság gyakran használt multi szót is gyakran olvastuk muftinak, mert ugye a mufti egy rendes szó, ami benne van a szótárban, míg a multi csak egy mai zsargon, az l és f betűk pedig bizonyos fontokban nagyon hasonlítanak egymásra.

Három OCR program tudásának egyesítése

1996-ban a Recognita céget megvásárolta az amerikai Caere cég, amely ugyancsak OCR programot, a világon leginkább elterjedt OmniPage-et fejlesztette. Nem sokkal később a ScanSoft vásárolta fel a Caere-t. A Scansoft is OCR fejlesztő cég volt, a TextBridge gazdája. Így egy cég kezébe került három OCR program, már csak az orosz fejlesztés őrizte meg önállóságát. A három program körülbelül egyforma jónak volt mondható, váltakozó szerencsével szerepeltünk a különböző tesztekben. Mégis volt egy igen nagy különbség: az amerikai fejlesztő csapatokban már alig dolgozott eredeti szerző, míg a magyaroknál még mindenki megvolt. Részben ez a tény, részben a magyar fejlesztés olcsósága azt eredményezte, hogy fokozatosan a teljes OCR fejlesztés Magyarországra került.

Három OCR engine-nel rendelkeztünk, a gépek sebessége már megengedte, hogy akár mindhárom engine lefusson. A két amerikai program eredményeit felhasználtuk a Recognitában, súgtak nekünk a bizonytalan szituációkban. Általában magunknak hittünk, de amikor nem volt a döntéshez elegendő adatunk, akkor megkérdeztük a másik két engine véleményét is. A VOTE algoritmusok nagyon sokat javítottak az eredményeken, az OmniPage, mert ez a neve az egyesített programnak, verhetetlen a pontosság tekintetében. A sebességével már nem dicsekedhetünk, de ez a felhasználókat kevésbé zavarja.

A fejlesztőrendszer és tesztelés fontossága

Képfelismerő programok fejlesztésekor látni kell a képet a feldolgozás minden fázisában, lehetőséget kell adni a kép változtatására, együtt kell látni az alakzatokat a rájuk vonatkozó mennyiségekkel. A Recognita fejlesztőrendszerét a Macinthos mintájára készítettük (akkor Windows még nem volt). A képernyő-egér-esemény kezelés pontosan illeszkedett a mi problémáinkhoz. A fejlesztőrendszer együtt nőtt a felismerő programmal, néha többet dolgoztunk azért, hogy láthatóvá tegyünk egy-egy jelenséget, mint az azt követő algoritmus fejlesztésén. A végső konklúzió mindig az volt, hogy megérte.

A tesztelés fontosságát kezdettől fogva láttuk, de sehogy sem tudtunk odahatni, hogy reprezentatív, nagyméretű tesztanyag készüljön. Végül az amerikai kollégáktól kaptunk (vettünk) egy sok ezer oldalból álló anyagot, amelyet a Nevadai Egyetemen készítettek. Nagyon sokféle írást tartalmazott, elég pontos etalon szövegekkel. Ettől kezdve ugrásszerűen javult a felismerési pontosságunk. Olyan szövegillesztő programot írtunk, ami összesítette az előző állapothoz képest bekövetkező javulásokat és romlásokat, megmutatta azok helyét, és nagyon gyorsan működött. Így minden apró algoritmikus változtatás hatását azonnal ellenőrizhettük.

A Recognita fejlesztése során rengeteget debugoltunk. Egy-egy felismerési hiba pontos okát csak úgy láthattuk, ha a kritikus algoritmust utasításról utasításra követtük. Ezért volt a szobánk falán a jelmondat: *“Nature wasn't designed but debugged into perfection.”*