hyväksymispäivä arvosana

arvostelija

Koneoppiminen merkintunnistuksessa

Tuomo Salmenkivi

Helsinki 23.4.2018

HELSINGIN YLIOPISTO
Tietojenkäsittelytieteen osasto

HELSINGIN YLIOPISTO — HELSINGFORS UNIVERSITET — UNIVERSITY OF HELSINKI

Tiedekunta — Fakultet — Faculty	Laitos — Institution — Department
Matemaattis-luonnontieteellinen tiedekunta	Tietojenkäsittelytieteen osasto

Tekijä — Författare — Author

Tuomo Salmenkivi

Työn nimi — Arbetets titel — Title

Koneoppiminen merkintunnistuksessa

Oppiaine — Läroämne — Subject

Tietojenkäsittelytiede

Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidoantal — Number of pages
	23.4.2018	7 sivua + 0 liitesivua

Tiivistelmä — Referat — Abstract

Avainsanat — Nyckelord — Keywords

merkintunnistus, koneoppiminen

Säilytyspaikka — Förvaringsställe — Where deposited

Muita tietoja — övriga uppgifter — Additional information

Sisältö

1 Johdanto

Merkintunnistusta käytetään muuttamaan skannattujen ja usein tieto- tai kirjoituskoneella kirjoitettujen dokumenttien sisältö tietokoneen luettavaan merkistöön. Tämä mahdollistaa tekstin muokkaamisen, lukemisen ja säilömisen huomattavasti aiempaa tehokkaammalla tavalla. Tällaiselle toiminnalle on huomattavasti kysyntää niin kaupallisessa kuin epäkaupallisessakin toiminnassa. Staattisten ennalta määritetyssä muodossa olevien dokumenttien tunnistaminen on tutkimusalueena varsin pitkällä.

Hyvin suuri osa arkipäiväisesti informaatiosta ei kuitenkaan ilmene dokumenteissa vaan esimerkiksi liikkeiden julkisivuissa, tiekylteissä tai yleisesti ottaen tilanteissa, joita ei voida esittää ennalta määritetyssä muodossa. Useat tavallisimmat optiset merkintunnistusratkaisut eivät toimi tällaisissa luonnollisissa ympäristöissä hyvin, sillä nämä ratkaisut vaativat tyypillisesti toimiakseen mielellään ennalta määritellyn, mutta ainakin hyvin selkeän, tasaisen ja mahdollisimman häiriöttömän ympäristön.

Vaihtoehtoisia ratkaisuja perinteisemmille merkintunnistusratkaisulle on viime aikoina pyritty etsimään koneoppimisen kautta. Tällaisissa ratkaisuissa on tyypillisesti käytössä neuroverkkoja, jotka automaattisesti päivittävät tietokantaansa uusilla lisäyksillä dataa ja käyttää tätä arvioimaan merkkien arvoja.

Keskityn tässä tutkielmassa käsittelemään aluksi optisen tekstintunnistuksen käyttöä ja ongelmia yleisellä tasolla. Seuraavissa kappaleissa käsittelen optisen merkkitunnistuksen sovelluksia koneoppimisessa. Käyn myös sivuuttaen läpi myös sanantunnistusta sekä perinteemmän "tyhmän"järjestelmän osalta, että myös koneoppimista enemmän hyödyntävän älykkään tunnistamisen osalta.

2 Merkintunnistus yleisesti

2.1 Merkintunnistuksen historiaa

Varsinainen tekstintunnistustamisen tutkimus sellaisenaan kun se tänä päivänä tunnetaan alkoi 1950-luvulla tarpeesta tunnistaa tekstiä pankkisekeissä. Alustavassa sekkien merkintunnistuksessa käytettiin apuna rautaoksidia sisältävää mustetta, jolloin erityisen lukulaitteen lukupää kykeni tunnistamaan merkit niiden magneettisuuden perusteella. Tätä kutsittiin MICR-teknologiaksi (Magnetic Ink Character Recognition). MICR-teknologiassa käytettiin tyypillisesti kirjaisimia E-13B sekä CMC-7. Vakiokirjaisimien käyttäminen oli etenkin varhaisvaiheen merkintunnistuksessa äärimmäisen tärkeää, sillä kirjaisimen pitäminen standardina paransi tarkkuutta ja sitä kautta luotettavuutta sekä MICR- että myöhemmin myös OCR-teknologioissa.[1]

Varsinainen OCR-teknologia yleistyi vasta 1960-luvulla jota varten kehitettiin omat erityiset kirjaisimet, OCR-A sekä OCR-B.

2.2 Merkintunnistuksen pääpiirteet

Merkintunnistuken tarkoituksena on siirtää dataa tietokoneen ulkopuolisesta maailmasta muotoon, jossa sitä voidaan tutkia, editoida ja säilöä tehokkaasti.Esimerkiksi hakusanojen käyttö tiedon etsimisessä suuresta määrästä dataa jota tietokone ei osaa käsitellä on ilman merkintunnistusta mahdotonta.

2.2.1 Tiedoston siirtäminen koneelle

Ensimmäinen vaihe tekstin saamisessa tietokoneen käsiteltävään merkistömuotoon on dokumentin siirtäminen tietokoneen muistiin. Nykyään digitaalisilla kameroilla kuvatiedosto saadaan kuvanottohetkellä automaattisesti digitaaliseen muotoon ja se voidaan myöhemmin ladata helposti kovalevylle tai flash-muistiin, mutta merkintunnistuksen alkuaikoina pakollinen ensimmäinen vaihe oli dokumentin skannaaminen. Ennen yleiskäytössä olevan skannerin yleistymistä, merkintunnistusta kehittävien tahojen ongelmana oli skannerilaitteiden kehitys, mutta yleisskannerien yleistyttyä ei tämä ollut enää suuri ongelma.

Ensimmäisen vaiheen jälkeen koneella on nyt käytössään digitaalisessa muodossa oleva kuvatiedosto. Kuvatiedosto ei vielä itsessään sisällä mitään tietoa merkeistä tai kuvassa olevasta kirjoituksesta vaan pelkästään pikseleistä, niiden väreistä ja sijainneista suhteessa kuvaan. Mikäli teksti on kirjoitus- tai tietokoneella ja samalla fontilla kirjoitettua on tässä vaiheessa tekstin tunnistaminen nykyteknologialla kohtalaisen helppoa. Vaikeudet alkavat siinä vaiheessa kun haluamme tutkia käsinkirjoitettua tai poikkeuksellisessa ympäristössä olevaa tekstiä. Ulkomaailmasta otetussa valokuvassa on niin paljon häiriötekijöitä, että tekstin löytäminen on vaikeaa ja käsinkirjoitetussa tekstissä merkkien tarkkuus suhteessa vertailuarvoihin heittää niin paljon, että tekstin tulkitseminen on haastavaa.

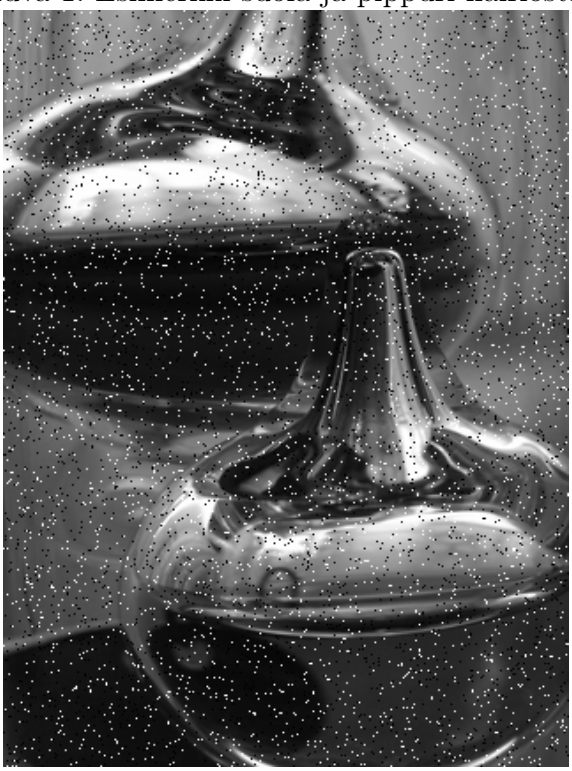
2.2.2 Tiedoston esiprosessointi

Nykyisissä merkintunnistusjärjestelmissä on usein erilaisia esiprosessointivaiheita, joiden tarkoituksena on parantaa merkintunnistuksen tarkkuutta. Tällaisia prosesseja ovat esimerkiksi skannatun dokumentin suoruuden korjaaminen, kuvassa esiintyvän häiriön poistaminen tai kuvasuuhteen ja skaalan normalisointi. Myös eräs tyypillinen vaihe merkintunnistuksessa on kuvan siirtäminen harmaaväriskaala- tai värikuvasta mustavalkoiseksi, jolloin valkoinen tausta voidaan erottaa mustasta tekstistä. Tällöin ohjelmiston tarvitsee analysoida pelkästään mustaa osaa ja prosessi helpottuu. Kuvan muuttaminen binäärikuvaksi, jossa on siis vain kaksi väriä, onkin usealle merkintunnistusjärjestelmälle välttämätön vaihe, sillä suuri osa kaupallisista merkintunnistusjärjestelmistä toimii vain binäärikuvilla. [2]

Häiriöiden poistaminen on esiprosessoinnin vaihe, jossa kuvasta pyritään poistamaan häiriötä kuvasta. Häiriö voi olla joko sattumanvaraista epäjohdonmukaista häiriötä tai se voi olla kuvauslaitteesta tai muokkausalgoritmeista johtuvaa aiheuttamaa joh-

donmukaista häiriötä. Eräs tyypillinen häiriö mitä kuvista löytyy on niin kutsuttu suola ja pippuri- häiriö.

Kuva 1: Esimerkki suola ja pippuri-häiriöstä.



Tällä tarkoitetaan häiriöitä, jotka ilmenevät täysin ympäristöstään riippumattomina valkoisina tai mustina pikseleinä. Tyypillisesti tällaista häiriötä on vain pienessä määrässä kuvan pikseleitä. Toinen useasti esiintyvä häiriön tyyppi on Gaussian-häiriö. Gaussian-häiriössä kaikki kuvan pikselit ovat muuttuneet oikeasta väristään. Yleensä Gaussian-häiriössä muutos on huomattavan pientä.

Molempien esimerkkinä käyttämieni häiriöiden hoitamisessa pitää punnita hyötyjä ja haittoja. Ensimmäinen olennainen tekijä ongelmien hoidossa on käytettävissä oleva prosessointiteho. Digitaalikameralla on käytössä huomattavan pieni prosessori verrattuna esimerkiksi tietokoneella tehtyyn kuva-analyysiin. On pohdittava tapauskohtaisesti, milloin mikäkin korjaustoimenpide on kannattava ja milloin ei. Toinen olennainen tekijä on päättäminen siitä, voidaanko kuvan yksityskohtien tarkkuutta uhrata, jotta saadaan suurempi määrä häiriötä kuvasta pois.

Eräs toinen esiprosessoinnin toimi on tekstin jakaminen alueisiin. Tällä tarkoitetaan eksplisiittistä tekstin jakamista esimerkiksi kappaleisiin tai lainauksiin. Tämän avulla erilaisia osia tekstistä voidaan analysoida tehokkaammin, sillä alueilla saatetaan tietää olevan tietynlaisia ominaisuuksia, jotka vaikeuttaisivat muuten merkkien tai sanojen tunnistusta alueen sisällä. Alueisiin jaon ensimmäinen osa on häiriöiden prosessointi pois kuvasta[4]. Tässä vaiheessa tulee olla tarkkana, sillä pilkut ja pisteet saatetaan tulkita häiriöiksi. Tämän estämiseksi vaihe on tehtävä varoen. Seuraa-

va vaihe on kuvan muuttaminen binäärimuotoon. Tämä tarkoittaa siis sitä, että jokainen pikseli on joko täysin valkoinen tai täysin musta. Seuraavaksi kuvasta etsitään toisiinsa kiinnityneet mustat pikselit. Nämä ovat kuvan "symbolit". Jokaiselle symbolille lasketaan pinta-ala ja keskipiste. Seuraavaksi etsitään symbolin n lähintä naapurisymbolia. O'Gorman[4] ehdottaa, että n olisi vähintään neljä, sillä tyypillisesti neljänneksi kauimpana oleva symboli on eri rivillä tutkittavasta symbolista. Lopuksi tutkitaan vierekkäisten symboliparien keskipisteiden etäisyyttä, jonka perusteella voidaan päätellä dokumentin vinous. Mikäli dokumentti huomataan vinoksi, korjataan sen kulmaa ja palataan symbolien luontivaiheeseen. Mikäli dokumentti vastaavasti oli hyväksyttävän suorassa, voidaan päätellä mitkä symbolit ovat kiinnittyneitä toisiinsa samoissa linjoissa ja mitkä pystysuunnassa. Näin dokumentista voidaan havaita rivit, kappaleet tai otsikot.

2.2.3 Hahmotunnistus

Seuraava vaihe vaihtelee riippuen merkintunnistusjärjestelmän implementaatiosta. Kun tarkastellaan dokumentteja, joissa kirjaisin on etukäteen tiedossa, voidaan yksinkertaisesti verrata jokaista merkkiä etukäteen tiedossa oleviin saman kirjaisimen merkkeihin. Tämä toimii siten, että verrataan tarkasteltavan merkin pikselien sijainteja jo tiedossa olevien merkkien pikseleiden sijainteihin. Kun saadaan osuma yhteensopivuudesta, tiedetään merkin olevan sama. Tämä vaihe ei kuitenkaan ole aina oikea tapa edetä. Esimerkiksi tilanteessa, jossa ei etukäteen tiedetä kirjaisinta, jota dokumentissa käytetään, saattaa tunnistusvaiheessa tulla ongelmia, sillä kirjaimet saattavat poiketa huomattavasti toisistaan eri kirjaisimissa.

2.2.4 Ominaisuuksien tunnistaminen

Tällaiseen tilanteeseen eräs kehitetty ratkaisu on ominaisuuksien tunnistaminen. Kun tunnistetaan merkin ominaisuuksia koko merkin sijaan ei merkin kirjaisimella ole välttämättä merkitystä. Esimkeriksi ison a-kirjaimen ominaisuuksina on käytännössä poikkeuksetta kaksi pystysuuntaista viivaa, jotka alkavat omista alakulmistaan ja kohtaavat keskellä sekä horisontaalinen viiva noin keskellä merkkiä joka yhdistää pystysuuntaiset viivat. Nyt kun etsimme tekstistä merkkejä, jotka täyttävät nämä ominaisuudet voimme suurella todennäköisyydellä löytää isot a-kirjaimet.

Kuva 2: Yksinkertaistettu esimerkki ison a-kirjaimen ominaisuuksista.

[3]

2.2.5 Luokittelu

Ominaisuuksien tunnistaminen hajottaa merkit pienemmiksi osiksi, kuten esimerkiksi viivoiksi, viivojen suunniksitai niiden kohtauspisteiksi. Merkkien jakaminen tällaisiin pienempiin osiin parantaa laskentatehoa sillä se pienentää merkkien dimensioita, joita tulee verrata. Näitä ominaisuuksia verrataan jo olemassa olevaan vektorirepresentaatioon merkistä, joka sisältää merkin yksinkertaistetut ominaisuudet.

2.3 Sanantunnistus

Usein erillisenä osana tunnistusohjelmistoissa on sanantunnistus. Sanantunnistuksen tarkoituksena on vahvistaa tiettyjen kirjainten ja kirjainyhdistelmien todennäköisyyttä sen perusteella, muodostuuko kirjaimista sana. Eräs menetelmä on myös tutkia sanojen kontekstia ja sitä kautta todentaa sanojen, ja merkkien, todennäköisyys.

2.3.1 Sanantunnistus 1

TODO

2.3.2 Sanantunnistus 2

TODO

3 Älykäs merkin- ja sanantunnistus

Tyypillisesti tietojenkäsittelytieteessä älykkäillä metodeilla tarkoitetaan ohjelmia, jotka kykenevät kehittämään itseään. Tyypillisesti koneoppimisessa tämä tapahtuu opetusdatan kautta, joka sisältää sekä esimerkin koneelle syötettävästä datasta, että vastauksen siihen mitä tämä esimerkki kuvastaa. Merkintunnistuksessa opetusvaihetta voidaan miettiä siten, että ensiksi syötetään ohjelmalle kuva yhdestä merkistä ja kerrotaan mikä tämän merkin arvo on. Esimerkkinä voidaan käyttää esimerkiksi kuvaa isosta A-kirjaimesta. Syötteenä on tällöin kuva ja vastauksena suuri A-kirjain.

Viime vuosina suureen suosioon erilaisissa älykkäissä menetelmissä tietojenkäsittelytieteessä ovat nousseet neuroverkot. Neuroverkot ovat hyviä vaihtoehtoja ongelmille, joille ei ole yksiselitteistä algoritmistä ratkaisutapaa tai joiden algoritmisen ratkaisun löytäminen olisi liiallisen työlästä[5]. Neuroverkot ovat erityisen hyviä saamaan yleistettäviä vastauksia sekavasta ja monimutkaisesta datasta. Erityisesti kuvat, jotka sisältävät huomattavan määrän häiriötä ja virheitä, ovat täten hyvä ongelma neuroverkon ratkaistavaksi.

3.1 Neuroverkot

Neuroverkot toimivat jäljittelemällä ihmisen aivojen toimintaa. Neuroverkot koostuvat keinotekoisista neuroneista, jotka taas vastaavasti koostuvat synapseista, summaajasta sekä aktivaatiofunktiosta. Neuroneiden välillä oleville synapseille määritellään paino, jonka perusteella neuroverkko tekee oletuksia datasta joka sille on syötetty.

3.1.1 Neuroverkot fonttien opettelussa bittikartoilla

Isot kirjaimet, 26 kirjainta.

16*16 pikseliä, 256 bittiä = 256koko vectori, kun värillinenbitti = 1, jos ei bitti = 0.

Tyypillisesti käytetään mielummin 1 = 0.5, 0 = -0.5

Tulos = 26(1 jokaiselle kirjaimelle) vectori, jossa -0.5 jos ei oikea tulos ja 0.5 jos oikea.

Seuraavaksi harjoitetaan verkkoa. For the above task we can use one layer of neural network, which will have 256 inputs corresponding to the size of input vector and 26 neurons in the layer corresponding to the size of the output vector.

At each learning epoch(KIERROS?), all samples from the training set are presented to the network and the summary squared error is calculated.

When the error becomes less than the specified error limit, then the training is done and the network can be used for recognition.

3.1.2 Neuroverkot ominaisuuksien tunnistamisessa

Yllä oleva toimii, mutta se on rajoittunut erityisesti ongelmatilanteiden ja poikkeuksien suhteen, joita tekstintunnistuksessa usein tapahtuu. Esim fontit, tekstin skaala, häiriöt ymsyms.

Giving an NN OCR system bitmaps as input is somewhat problematic since humans don?t see characters at the pixel level, nor is the ?essence? of a character font conveyed by this pixelized representation.

When there are considerable bitmap variations in the definition of each font character, a better set of inputs to represent the data would be a set of classifiers, computable from the bitmap images, such that these classifiers are invariant to changes in font and point size.

Such classifiers might include topological characteristics, such as Euler number, compactness, and geometric properties, e.g., concave up. Of course, these features now need to be computed from the input images and given as input to the neural network OCR system. In addition, the system is invariant to changes in font and point size, so it cannot classify beyond labeling an input bitmap as say an ?e?, when we may want additional information such as the font and point size, e.g.,?e?, point size:

12, font: Times Roman. The point is that features typically provide some level of invariance, but at the same time, limit the degree of recognition.

In this case, since there is wide variation in font definitions, we could first have an NN-based OCR system that is invariant to font and scale to recognize the character. Once we know it?s an ?e?, we can match it against all ?e? font definitions in our font database to establish the exact font and point size

Lähteet

1 S. Mori, C. Y. Suen and K. Yamamoto. *"Historical review of OCR research and development"*, in Proceedings of the IEEE, vol. 80, no. 7, pp. 1029-1058, Jul 1992.

2 M. Sezgin and B. Sankur. *"Survey over image thresholding techniques and quantitative performance evaluation"*, Journal of Electronic Imaging 13(1), 146?165 (January 2004)

3 Optical character recognition (OCR)
`http://www.explainthatstuff.com/how-ocr-works.html`

4 L. O'Gorman, *"The document spectrum for page layout analysis,"* in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 15, no. 11, pp. 1162-1173, Nov 1993.

5 OCR, Neural Networks and other Machine Learning Techniques
`http://www.cvisiontech.com/resources/ocr-primer/ocr-neural-networks-and-other-`