Määrittelydokumentti

4. marraskuuta 2022

Kurssi- ja tekijätiedot

Työn tekijä: Mikko Ahro
Opinto-ohjelma: Tietojenkäsittelytieteen kandidaatti
Kurssi: Tietorakenteet ja algoritmit harjoitustyö, periodi 2 / 2022
Käytetty ohjelmointikieli: Python
Muut ohjelmointikielet: ei muita kieliä kuin Python vertaisarviointiin riittävällä tasolla hallussa
Dokumentointikieli: suomi

MNIST numeroiden tunnistus k:n lähimmän naapurin (kNN) menetelmällä

Kyseessä on koneoppimisongelma, jossa käytetään kNN menetelmää käyttäen etäisyysmittana Hausdorffin etäisyyksiä.

Ratkaistava ongelma

Tavoitteena on tunnistaa käsinkirjoitettuja numeroita. Datana käytetään MNIST tietokantaa, jossa on yhteensä 70 000 käsinkirjoitettua numeroa 0-9, jaettuna harjoitusdataan 60 000 kpl sekä testidataan 10 000 kpl. kNN menetelmällä luodaan testidatasta malli käyttäen harjoitusdataa. Mallin toimivuus testataan testidalla, josta pyritään siis tunnistamaan numerot oikein. Kyseessä on siis luokitteluongelma.

Ratkaisun toimivuuden mittarina toimii ainakin virheellisesti tunnistettujen numeroiden määrä. Tässä vaiheessa on vaikea arvioida mikä on realistinen tavoite, mutta kirjallisuuden perusteella virhetunnistusprosentiksi tulisi realistisesti saada alle 10%.

Syötteet

Ohjelma saa syötteenä ainakin:

- MNIST tietokannan
- testattavan Hausdorffin etäisyyden $D_1 - D24$
- rajaavia parametreja mitä osuutta MNIST harjoitusdatasta ja testidatasta käytetään
- kNN algoritmille parametri k

MNIST tietokantaa käytetään ohjelman harjoitus- sekä testidatana. kNN-algotmi voidaan suorittaa eri vaihtoehtoisilla Hausdorffin etäisyyksillä, ja tämän vaihtoehdon määrittämiseen ohjelma saa syötteen D_n. Todennäköisesti ohjelman testaus- ja kehitysvaiheessa ei ole realistista ajaa koko MNIST tietokantaa vaan ajoja tulee rajata johonkin osajoukkoon, tähän käytetään rajaavia parametreja.

Käytetyt algoritmit ja tietorakenteet

Suunnitelma Python paketeista

Alustava suunnitelma käytettäville valmiille Python paketeille ja työkaluille:

- matplotlib tulosten visualisointiin
- numpy vektoridatan tallentamiseen ja vektoriaritmetiikkaan
- keras MNIST tietokannan lataamiseen. Binäärimuotoisen MNIST datan lukemiseen vaadittavan koodin toteutus ei ole oleellista tämän työn kannalta, ja koska MNIST data on valmiiksi saatavilla helposti luettavassa muodossa, ei binääridatan lukemista lähdetä toteuttamaan.
- pylint koodin laadun testaamiseen
- poetry sekä unittest yksikkötestaamiseen sekä testikattavuuden hallintaan

Todennäköisesti lopullinen toteutus tulee käyttmään myös muita Python paketteja.

Tietorakenteet

MNIST 28∗28∗255 (leveys x korkeus x harmaasävy) kuva konvertoidaan 28∗28∗1 muotoon (harmaasävy mustavalkoiseksi) kuvaksi, ja tullaan tallentamaan pistejoukkona numpy arrayksi. Kaikkia (tai käsiteltyä osaa tietokannasta) vastaava pistejoukko tullaan todennäköisesti tallentamaan listana, en näe tässä vaiheessa hyötyä käyttää tähän muita tietorakenteita. Tämä osuus saattaa tarkentua kun työ etenee.

kNN-algoritmia varten tulee todennäköisesti toteuttaa KD-puu.

Algoritmit

Hausdorffin etäisyydet

Hausdorffin etäisyyksien laskemiseksi tarvitaan seuraavia suunnattuja etäisyyksiä [3]:

$$d_1(\mathcal{A}, \mathcal{B}) = \min_{a \in \mathcal{A}} d(a, \mathcal{B}) \tag{1}$$

$$d_2(\mathcal{A}, \mathcal{B}) = {}^{50}K^{th}_{a \in \mathcal{A}} d(a, \mathcal{B}) \tag{2}$$

$$d_3(\mathcal{A}, \mathcal{B}) = {}^{75}K^{th}_{a \in \mathcal{A}} d(a, \mathcal{B}) \tag{3}$$

$$d_4(\mathcal{A}, \mathcal{B}) = {}^{90}K^{th}_{a \in \mathcal{A}} d(a, \mathcal{B}) \tag{4}$$

$$d_5(\mathcal{A}, \mathcal{B}) = \max_{a \in \mathcal{A}} d(a, \mathcal{B}) \tag{5}$$

$$d_6(\mathcal{A}, \mathcal{B}) = \sum_{a \in \mathcal{A}} d(a, \mathcal{B}) \tag{6}$$

sekä seuraavia suuntaamattomia etäisyyksiä:

$$f_1(d(\mathcal{A}, \mathcal{B}), d(\mathcal{B}, \mathcal{A})) = \min(d(\mathcal{A}, \mathcal{B}), d(\mathcal{B}, \mathcal{A})) \tag{7}$$

$$f_2(d(\mathcal{A}, \mathcal{B}), d(\mathcal{B}, \mathcal{A})) = \max(d(\mathcal{A}, \mathcal{B}), d(\mathcal{B}, \mathcal{A})) \tag{8}$$

$$f_3(d(\mathcal{A}, \mathcal{B}), d(\mathcal{B}, \mathcal{A})) = \frac{d(\mathcal{A}, \mathcal{B}) + d(\mathcal{B}, \mathcal{A})}{2} \tag{9}$$

$$f_4(d(\mathcal{A}, \mathcal{B}), d(\mathcal{B}, \mathcal{A})) = \frac{N_a d(\mathcal{A}, \mathcal{B}) + N_b d(\mathcal{B}, \mathcal{A})}{N_a + N_b} \tag{10}$$

Näistä suuntaamattomista ja suunnatuista etäisyyksistä saadaan Hausdorffin etäisyydet:

directed distance	function			
	f_1	f_2	f_3	f_4
d_1	D_1	D_2	D_3	D_4
d_2	D_5	D_6	D_7	D_8
d_3	D_9	D_{10}	D_{11}	D_{12}
d_4	D_{13}	D_{14}	D_{15}	D_{16}
d_5	D_{17}	D_{18}	D_{19}	D_{20}
d_6	D_{21}	D_{22}	D_{23}	D_{24}

kNN algoritmi

kNN algoritmi toteutetaan itse (ts. ei käytetä Pythonin scipy tms valmiita algoritmeja). kNN käyttää etäisyysmittana yllämääriteltyjä Hausdorffin etäisyyksiä.

Aika- ja tilavaativuudet

Kirjallisuuden [1] [2] mukaan kNN-algoritmin aika sekä tilavaativuuden pitäisi olla $O(dn^2)$ brute force ratkaisulla, missä d on piirteiden määrä. Kuitenkin KD-puuta käyttämällä pitäisi päästä $O(dn)$ aikavaativuuteen.

Tämä on kuitenkin vain kNN aika- ja tilavaativuus, Hausdorffin etäisyyksien laskennan aika ja tilavaativuutta en tässä vaiheessa osaa arvioida. Tarkennetaan myöhemmin.

Viitteet

[1] Giuseppe Bonaccorso. *Machine learning algorithms : popular algorithms for data science and machine learning*. eng. Second edition. Birmingham ; Packt Publishing, 2018. ISBN: 1-78934-548-0.

[2] Giuseppe Bonaccorso. *Mastering machine learning algorithms : expert techniques to implement popular machine learning algorithms and fine-tune your models*. eng. 1st edition. Birmingham ; Packt, 2018. ISBN: 1-78862-590-0.

[3] M. Dubuisson ja A. Jain. "A modified Hausdorff distance for object matching". Teoksessa: *Proceedings of 12th International Conference on Pattern Recognition*. Vol. 2. Los Alamitos, CA, USA: IEEE Computer Society, 1994, s. 566,567,568. DOI: 10.1109/ICPR.1994.576361. URL: https://doi.ieeecomputersociety.org/10.1109/ICPR.1994.576361.