SVEUČILIŠTE U ZAGREBU
PRIRODOSLOVNO-MATEMATIČKI FAKULTET
MATEMATIČKI ODSJEK

Algoritam za klasifikaciju slika baziran na tenzorima

Ante Ćubela i Mirna Lovrić

Zagreb
22. svibanj 2023.

Sadržaj

Uvod

Metode koje se baziraju na tenzorima postale su poprilično popularne u zadnje vrijeme te nalaze primjenu u područjima kao što je kvantna mehanika i računalna dinamika fluida (CFD). Primjerice, kanonska i tensor train dekompozicija koriste se u kvantnom strojnom učenju. Također, rekonstrukcija jednadžbi dinamičkog sustava jedan je od problema koji se rješava metodama koje se baziraju na tenzorima.

U ovom ćemo radu prezentirati algoritam za klasifikaciju višedimenzionalnih objekata koji se bazira na metodi korištenoj za problem rekonstrukcije jednadžbi dinamičkog sustava. Poslije ćemo ga konkretno primijeniti za klasifikaciju dvodimenzionalnih objekata - slika.

U nastavku rada definirat ćemo algoritam SINDy, njegovu tenzorsku verziju MANDy te kernel-based verziju MANDy algoritma, odnosno algoritam za klasifikaciju objekata.Zatim ćemo algoritam primijeniti na dva MNIST data seta i analizirati efikasnost algoritma.

1 SINDy

Algoritam SINDy (Sparse Identification of Non-linear Dynamics) je prvotno osmišljen za generiranje jednadžbi dinamičkog sustava iz danih podataka, no u ovom radu koristit ćemo ga za probleme klasifikacije.

Neka je dana ODJ

$$\dot{x} = f(x),$$

gdje je $f : \mathbb{R}^d \to \mathbb{R}^d$, te m mjerenja $x^{(j)}, j = 1, ..., m$ i odgovarajuće derivacije $y^{(j)} = \dot{x}^{(j)}$. Želimo rekonstruirati funkciju f na temelju danih mjerenja. Stavimo $X = \left[x^{(1)}... \ x^{(m)}\right] \in \mathbb{R}^{d\times m}$, $Y = \left[y^{(1)}... \ y^{(m)}\right] \in \mathbb{R}^{d\times m}$ te odaberimo *baznu* funkciju $\Psi : \mathbb{R}^d \to \mathbb{R}^n$, koja će predstavljati traženu funkciju f . Definiramo matricu $\Psi_X \in \mathbb{R}^{n\times m}$ kao

$$\Psi_X = \left[\Psi(x^{(1)}) \ ... \ \Psi(x^{(m)})\right]. \tag{1}$$

Odnosno, $\Psi_X(i, j) = \Psi(x^{(j)})^T e_i, \ \ i = 1, \ldots, n, j = 1, \ldots, m$, gdje e_i predstavlja $i-ti$ kanonski vektor. Sada se algoritam svodi na rješavanje minimizacijske zadaće

$$\min_{\Xi} ||Y - \Xi^T \Psi_X||_F. \tag{2}$$

Matricu $\Xi \in \mathbb{R}^{n\times d}$ zovemo matricom koeficijenata i njen proizvoljni stupac ξ_i predstavlja funkciju f_i, tj.

$$y_i^{(j)} \approx f_i(x^{(j)}) = \xi_i^T \Psi(x^{(j)}). \tag{3}$$

Drugim riječima, djelovanje funkcije f_i na vektor $x^{(j)}$ smo aproksimirali kao linearnu kombinaciju komponenti vektora $\Psi(x^{(j)})$. Rješenje gornje zadaće dano je izrazom

$$\Xi^T = Y\Psi_X^+, \tag{4}$$

gdje je Ψ_X^+ pseudoinverz matrice Ψ_X. Napomenimo kako je ovaj problem moguće promatrati i kada je $f : \mathbb{R}^d \to \mathbb{R}^{d'}$, gdje je $d' \neq d$, pa je tada $Y \in \mathbb{R}^{d'\times m}$.

2 MANDy

2.1 Tensor - Train dekompozicija

Neka je dan tenzor $\mathbf{T} \in \mathbb{R}^{I_1\times I_2\times ...\times I_N}$ reda N. Kažemo da tenzor $\mathbf{T}$ ima Tensor - Train (TT) dekompoziciju ako postoje brojevi $r_0, r_1, ..., r_N$, tenzori $\mathbf{T}_1, \mathbf{T}_2, ..., \mathbf{T}_N$, $\mathbf{T}_k \in \mathbb{R}^{r_{k-1}\times I_k\times r_k}$, za $k = 1, ..., N$ takvi da

$$\mathbf{T}(i_1, i_2, ..., i_N) = \mathbf{T}_1(:, i_1, :)\mathbf{T}_2(:, i_2, :) \cdots \mathbf{T}_N(:, i_N, :), \quad i_k \in I_k, \quad k = 1, 2, ..., N. \tag{5}$$

Brojeve r_k zovemo TT-rangove, a tenzore $\mathbf{T}_k$ TT-jezgre. Uočimo kako je $r_0 = r_N = 1$. Kako su TT-jezgre tenzori reda 3, (5) možemo zapisati kao $\mathbf{T}(i_1, i_1, ..., i_N) = T_1(i_1)T_2(i_2) \cdots T_N(i_N)$.

2.2 MANDy

Algoritam MANDy je tenzorska verzija SINDy algoritma. Konkretno, rješavamo istu zadaću kao u algoritmu SINDy, no sada umjesto jedne bazne funkcije $\Psi : \mathbb{R}^d \to \mathbb{R}^n$, uvodimo p baznih funkcija $\psi_\mu : \mathbb{R}^d \to \mathbb{R}^{n_\mu}, \mu = 1, 2, \ldots, p$. Nadalje, umjesto matrice $\Psi_X \in \mathbb{R}^{n \times m}$ definirane sa (1) sada imamo tenzor $\mathbf{\Psi}_X \in \mathbb{R}^{n_1 \times \cdots \times n_p \times m}$ takav da je $\mathbf{\Psi}_X(i_1, \ldots, i_p, j) = \psi_{1,i_1}(x^{(j)}) \ldots \psi_{p,i_p}(x^{(j)}), 1 \leq i_k \leq n_k, k = 1, \ldots, p, j = 1, \ldots, m$ te rješavamo minimizacijsku zadaću

$$\min_{\mathbf{\Xi}} ||Y - \mathbf{\Xi}^T \mathbf{\Psi}_X||_F, \tag{6}$$

gdje želimo dobiti tenzor koeficijenata $\mathbf{\Xi} \in \mathbb{R}^{n_1 \times \cdots \times n_p \times d}$ u TT formatu.

2.3 Kernel-based MANDy

Koristeći oznake kao i u prethodnom poglavlju, neka su dane matrice $X \in \mathbf{R}^{d \times m}, Y \in \mathbf{R}^{d' \times m}$, te bazne funkcije $\psi_\mu : \mathbb{R}^d \to \mathbb{R}^{n_\mu}, \mu = 1, 2, \ldots, p$. Želimo odrediti tenzor koeficijenata $\mathbf{\Xi} \in \mathbb{R}^{n_1 \times \cdots \times n_p \times d'}$ koji rješava optimizacijsku zadaću (6). Rješenje te zadaće dano je s $\mathbf{\Xi}^T = Y \mathbf{\Psi}_X^+$. Kako bismo izračunali pseudoinverz tenzora $\mathbf{\Psi}_X$ koristimo identitet $\mathbf{\Psi}_X^+ = (\mathbf{\Psi}_X^T \mathbf{\Psi}_X)^+ \mathbf{\Psi}_X^T$. Sada je

$$\mathbf{\Xi}^T = Y (\mathbf{\Psi}_X^T \mathbf{\Psi}_X)^+ \mathbf{\Psi}_X^T \tag{7}$$

Kontrakcijom $\mathbf{\Psi}_X$ i $\mathbf{\Psi}_X^T$ se dobije Gramova matrica G dimenzija $m \times m$. Tu Gramovu matricu računamo koristeći neku od *kernel* funkcija k na sljedeći način:

$$G_{i,j} = k\left(x^{(i)}, x^{(j)}\right) = \mathbf{\Psi}\left(x^{(j)}\right)^T \mathbf{\Psi}\left(x^{(i)}\right) \tag{8}$$

To bi zbog načina dobivanja tenzora $\mathbf{\Psi}_X$ značilo da vrijedi:

$$k\left(x^{(i)}, x^{(j)}\right) = \prod_{\mu=1}^{p} \psi_\mu\left(x^{(j)}\right)^T \psi_\mu\left(x^{(i)}\right). \tag{9}$$

Kako je *kernel* funkcija produkt, dobije se da se Gramova matrica G može izračunati kao Hadamardov produkt:

$$G = \Theta_1 \odot \Theta_2 \odot \cdots \odot \Theta_p,$$

gdje su $\Theta_1, \ldots, \Theta_p$ matrice dimenzija $m \times m$ dane s:

$$\Theta_\mu = \left[\psi_\mu(x^{(1)}), \ldots, \psi_\mu(x^{(m)})\right]^T \cdot \left[\psi_\mu(x^{(1)}), \ldots, \psi_\mu(x^{(m)})\right].$$

To daje efikasniju implementaciju, no mi ćemo za računanje koristiti (8).
Sada definiramo matricu $Z := YG^+ \in \mathbb{R}^{d' \times m}$, koja se može dobiti rješavanjem sustava $ZG = Y$ (ili ako je G singularna, sustav se rješava u smislu najmanjih kvadrata). Funkcija klasificiranja f se onda dobije kao:

$$f(x) = Z \mathbf{\Psi}_X^T \mathbf{\Psi}(x) = Z \begin{bmatrix} k\left(x^{(1)}, x\right) \\ \vdots \\ k\left(x^{(m)}, x\right) \end{bmatrix}. \tag{10}$$

Indeks komponente od $f(x)$ s najvećom apsolutnom vrijednosti će biti predviđena klasa. Cijeli algoritam je opisan ispod:

Algoritam 1 Kernel-based MANDy

Ulazni podaci: Trening set X, matrica oznaka za trening set Y, test set $\hat{X}$, bazne funkcije
Izlazni podaci: Matrica oznaka $\hat{Y}$

1: Računanje Gramove matrice G koristeći (8), (9).
2: Rješavanje sustava $ZG = Y$.
3: Definiranje funkcije klasificiranja f koristeći (10).
4: Primjena funkcije f na svaki vektor iz test seta i spremanje rezultata u matricu $\hat{Y}$.
5: Indeks komponente svakog stupca $\hat{y}^{(i)}$ od $\hat{Y}$ s najvećom apsolutnom vrijednosti će biti predviđena klasa za sliku $\hat{x}^{(i)}$ iz $\hat{X}$.

Pri računanju matrice G, koristimo sljedeću kernel funkciju:

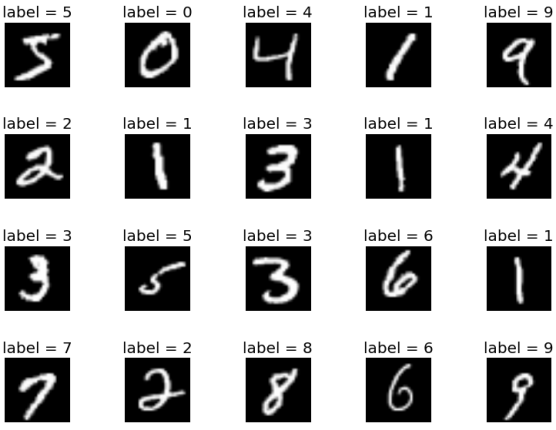
$$k\left(x, x'\right) = \prod_{i=1}^{d} \cos\left(\alpha(x_i - x_i^{'})\right) \tag{11}$$

3 Primjena

3.1 Opis datasetova

3.1.1 MNIST Handwritten Digits

Originalan MNIST dataset sadrži ukupno 70 000 grayscaleanih slika rukom pisanih znamenki od 0 do 9 dimenzija 28×28. Dakle svaka slika sadrži 784 piksela, te svaki piksel ima vrijednost u rasponu od 0 do 255. Svaku sliku ćemo normalizirati na interval $[0, 1]$.



Slika 5.1 Primjeri slika iz MNIST handwritten digit dataseta

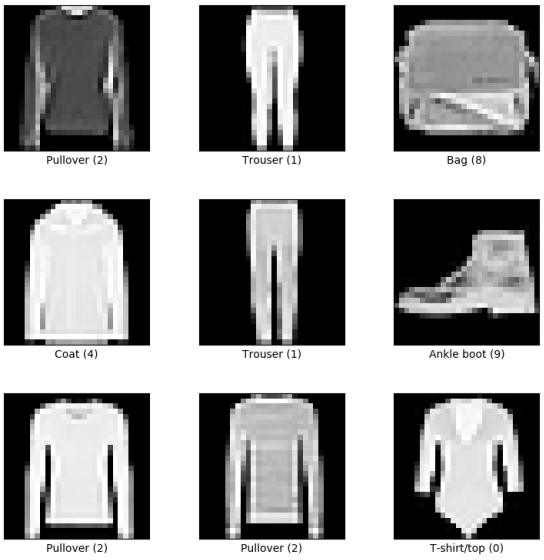
Prvih 60 000 slika smo iskoristili za treniranje, a preostalih 10 000 za testiranje algoritma. Također, svaka slika je vektorizirana, tj. matrica dimenzije 28×28 preoblikovana je u vektor veličine 784×1 na način da se stupci matrice pišu jedan ispod drugoga, krenuvši od prvog stupca. Koristeći oznake već definirane u radu, imamo $d = 784, d' = 10, x^{(j)} \in \mathbb{R}^d, j = 1, \ldots, d$ predstavlja $j - tu$ sliku iz dataseta, $y^{(j)} \in \mathbb{R}^{d'}$ je odgovarajuća oznaka te slike, odnosno vektor $y^{(j)}$ na poziciji i sadrži 1 ako slika $x^{(j)}$ predstavlja znamenku $i-1$, a inače je vrijednost 0.

3.1.2 MNIST Fashion

Slično kao i kod originalnog MNIST-a, MNIST fashion sadrži 70 000 grayscaleanih slika dimenzija 28×28 podijeljenih u 10 klasa :

1. Majica kratkih rukava/top
2. Hlače
3. Pulover
4. Haljina
5. Kaput
6. Sandale
7. Košulja
8. Tenisice
9. Torba
10. Čizme

Slika 5.2 Primjeri slika iz MNIST Fashion dataseta



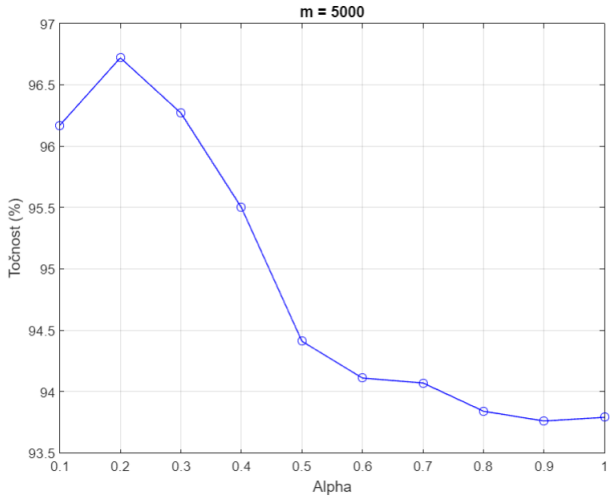
Slika 5.3 Primjeri slika iz MNIST Fashion dataseta

I te ćemo slike također normalizirati te podijeliti u jednakom omjeru kao za originalan MNIST na slike za treniranje i testiranje.

3.2 Rezultati

3.2.1 MNIST Handwritten Digits

Prvo smo uzeli 5000 slučajno odabranih podataka za treniranje i za te podatke varirali parametar α od 0.1 do 1.0 s korakom od 0.1, kako bismo pronašli koja nam vrijednost parametra α daje najbolje rezultate. Kao metriku uspješnosti smo koristili točnost (*accuracy*), te smo testirali algoritam na svih 10 000 podataka za testiranje. Koristili smo 5000 podataka za treniranje kako bi se algoritam izvršio u dovoljno kratko vrijeme. Ovisnost točnosti o parametru α je pokazana na slici ispod.

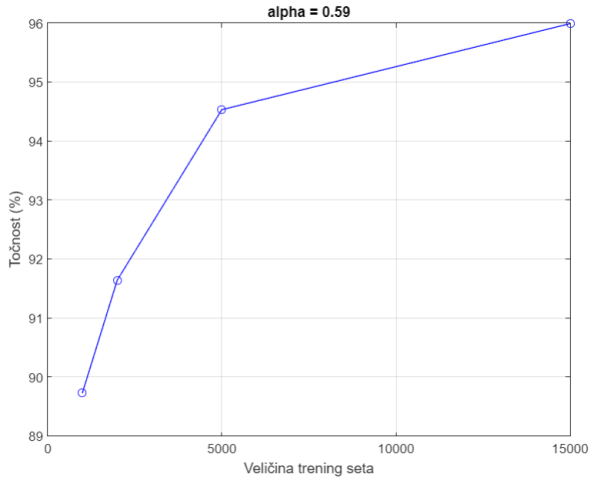


Slika 5.4 Ovisnost točnosti o parametru α

Za točniji pregled vrijednosti, rezultati su prikazani i u tablici ispod. Tako ćemo rezultate prikazivati i dalje u ovome poglavlju.

$\boldsymbol{\alpha}$	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
	96.17%	96.72%	96.27%	95.50%	94.41%	94.11%	94.07%	93.84%	93.76%	93.79%

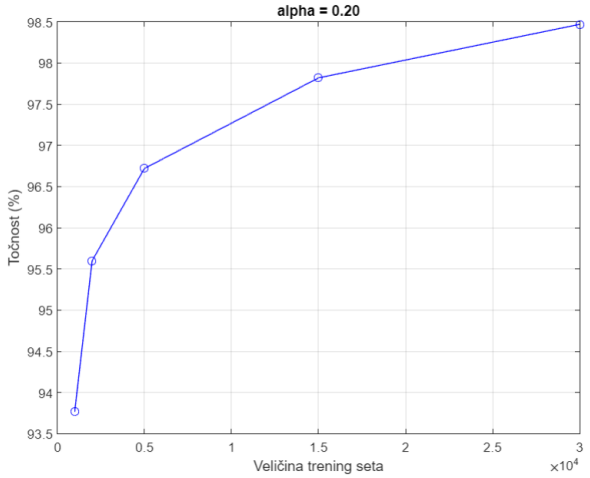
Zatim smo uzeli parametar $\alpha = 0.59$ kao što su autori rada [1]. Varirali smo veličinu dataseta za treniranje time što smo uzeli redom: 1000, 2000, 5000 i 15 000 slučajno odabranih podataka. Nismo mogli uzeti više od 30 000 podataka, jer kreiranje matrice G za taj broj podataka (dimenzije od G su tada $30\,000 \times 30\,000$) zahtjeva oko 7 Gb RAM memorije, što je otprilike maksimalno koliko smo imali na raspolaganju. Ovisnost točnosti o broju podataka uzetom za treniranje, uz vrijednost $\alpha = 0.59$, je pokazana na slici ispod ispod.



Slika 5.5 Ovisnost točnosti o broju trening podataka za $\alpha = 0.59$

Ponovili smo postupak, uključivši i veći trening dataset s 30 000 podataka, za parametar $\alpha = 0.2$, koji se u kratkoj analizi pokazao kao ponajbolji izbor. Konačni rezultati su prikazani na slici ispod.

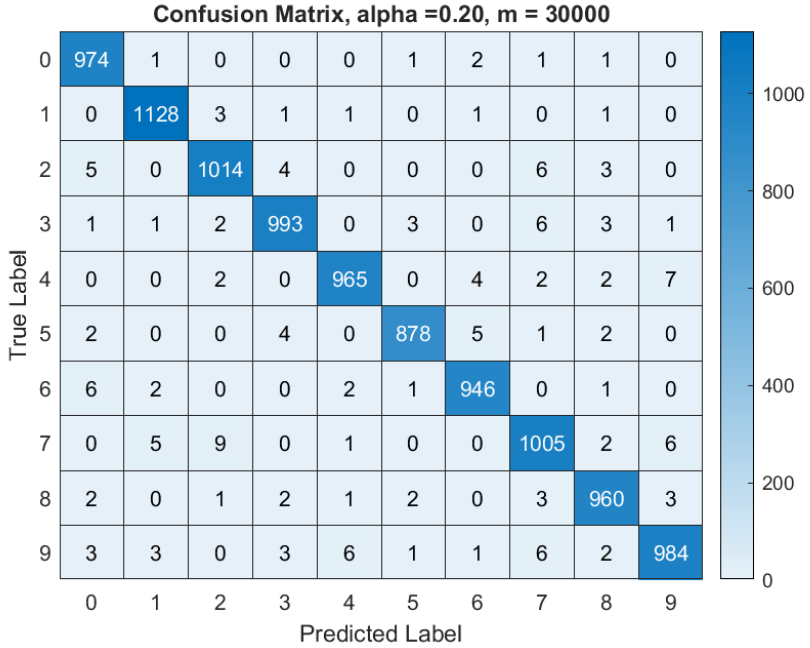
#TS	1000	2000	5000	15 000
	89.73%	91.64%	94.53%	95.99%



Slika 5.6 Ovisnost točnosti o broju trening podataka za $\alpha = 0.2$

#TS	1000	2000	5000	15 000	30 000
	93.77%	95.60%	96.72%	97.82%	98.47%

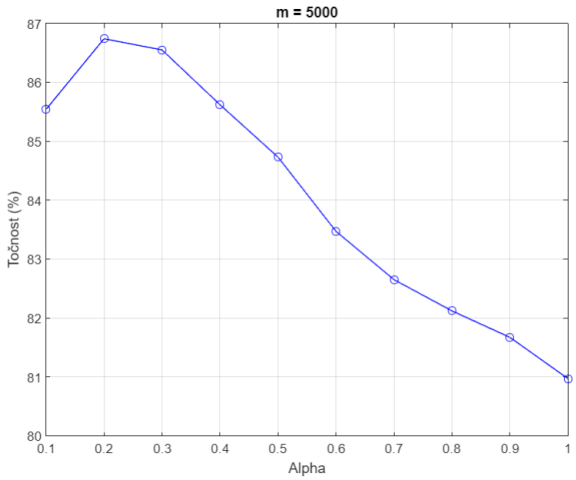
Ispod je prikazana i matrica zabune za model koji je postigao točnost od 98.47% na test setu.



Slika 5.7 Matrica zabune za model treniran na 30 000 podataka uz $\alpha = 0.2$

3.2.2 MNIST Fashion

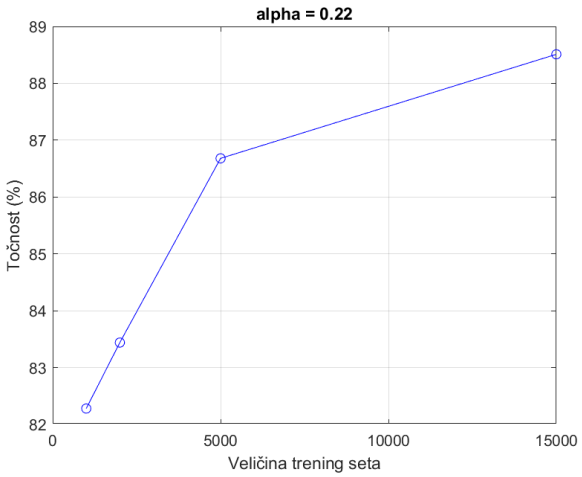
Broj podataka i dimenzija slika za ovaj dataset je jednaka kao i za MNIST Handwritten Digits dataset, pa je i naša analiza rezultata ista. Na slici ispod je prikazana ovisnost točnosti algoritma o parametru α za 5000 slučajno odabranih podataka za treniranje.



Slika 5.8 Ovisnost točnosti o parametru α

$\boldsymbol{\alpha}$	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
	85.54%	86.74%	86.55%	85.62%	84.73%	83.47%	82.65%	82.12%	81.67%	80.97%

Odlučili samo još istražiti točnost za vrijednosti α između 0.20 i 0.29 s korakom 0.01, te se na kraju odlučili koristiti parametar $\alpha = 0.22$. Na slici ispod su prikazane točnosti algoritma za različite veličine slučajno odabranog trening seta.

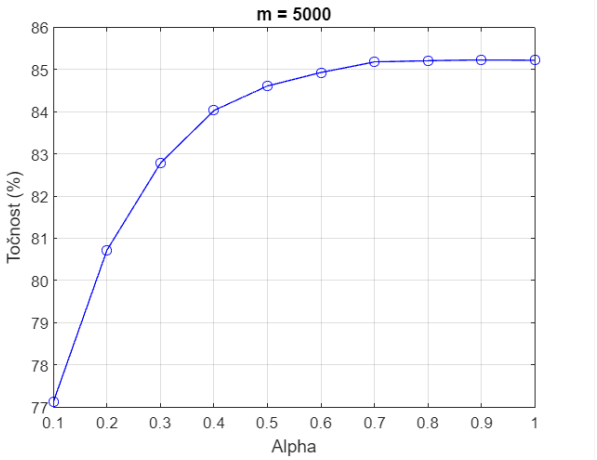


Slika 5.9 Ovisnost točnosti o parametru α

#TS	1000	2000	5000	15 000
	82.28%	83.44%	86.68%	88.51%

3.2.3 Smanjenje dimenzija

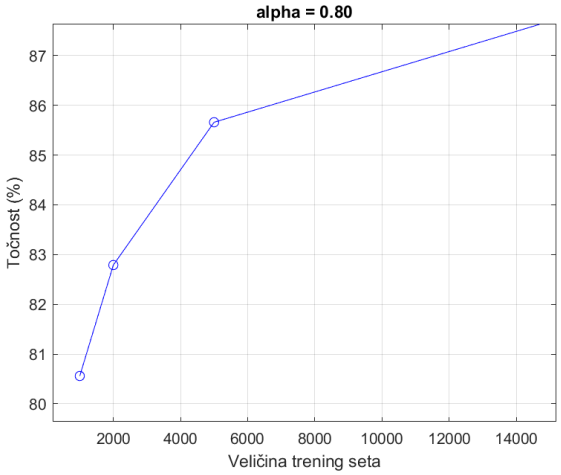
Isprobati ćemo još jedan pristup koji su iznijeli autori rada [1]. Smanjiti ćemo dimenzije slika na sljedeći način. Grupirati ćemo piksele slika u kvadrate dimenzija 2×2 bez presjeka i uzeti prosjek ta 4 piksela. Ta dobivena vrijednost će biti vrijednost piksela u novoj slici. Kada to učinimo za cijelu sliku, dobijemo novu sliku dimenzija 14×14. Prednost takvog pristupa je brže vrijeme izvođenja, te korištenje manje memorije kroz algoritam. Memorijska ograničenja za nas i dalje ostaju ista, jer je za m trening slika i dalje potrebno alocirati i izračunati matricu dimenzija $m \times m$ što za 60 000 slika zahtjeva oko 26.8 Gb RAM memorije u MATLAB-u.
Iskoristiti ćemo takav pristup na MNIST Fashion datasetu. Postupak treniranja algoritma će biti analogan kao i prije. Prvo ćemo uzeti 5000 nasumično odabranih slika iz skupa za treniranje i varirati vrijednosti parametra α od 0.1 do 1.0 s korakom 0.1. Rezultati su prikazani na slici ispod.



Slika 5.10 Ovisnost točnosti o parametru α

α	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
	77.13%	80.72%	82.78%	84.03%	84.61%	84.93%	85.18%	85.21%	85.23%	85.22%

Odlučili smo se za vrijednost $\alpha = 0.8$ i nastavili s treniranjem analogno kao i prije. Rezultati su prikazani na slici ispod.



Slika 5.11 Ovisnost točnosti o parametru α

#TS	1000	2000	5000	15 000
	80.56%	82.79%	85.66%	87.69%

3.2.4 Kratki zaključak

Možemo sada napraviti kratku usporedbu s rezultatima autora rada. Zanimljivo je da su autori rada za oba dataseta dobili bolje rezultate na slikama smanjenih dimenzija sa oba algoritma koje su iznijeli u radu. Koristeći *kernel-based* MANDy algoritam na originalnim slikama su postigli točnosti od 97.24% za MNIST Handwritten Digits dataset i 88.37% za MNIST Fashion dataset. Koristeći mnogo manje trening podataka zbog računalnih ograničenja (autori su imali 128 Gb RAM memorije na raspolaganju), postigli smo rezultate od 98.47% te 88.51% respektivno na tim datasetovima. Za MNIST Handwritten Digits smo koristili 30 000, tj. 50% od ukupnih podataka za treniranje, dok smo za MNIST Fashion dataset koristisli 15 000, odnosno 25% od ukupne količine podataka. Sva mjerenja do sada su pokazala poboljšanje performansi algoritma sa značajnim povećanjem trening dataseta, pa se da naslutiti da ta uspješnost može biti i mnogo bolja.
Koristeći slike smanjenih dimenzija, autori su postigli točnosti od 98.75% te 88.82% na navedenim datasetovima. Mi smo iskušali tu metodu samo na MNIST Fashion datasetu i postigli točnost od 87.69% koristeći 15 000 podataka za treniranje, tj. 25% ukupnih danih podataka. Razlike u uspješnosti bi se možda mogle prepisati odabiru parametra α. Parametar α kojeg su autori koristili u radu ($\alpha = 0.59$) se znatno razliku od optimalnih izbora za α iz naših mjerenja.

Literatura

[1] S. Klus, P. Gelß: *Tensor-based algorithms for image classification*, Department of Mathematics and Computer Science, Freie Universität Berlin, 14195 Berlin, Germany 2019.

[2] Z. Drmač: *Napredne Linearne i Nelinearne Numeričke Metode u Analizi Podataka*. PMF-MO Zagreb, 2023.

[3] MNIST dataset: https://www.kaggle.com/datasets/zalando-research/fashionmnist https://www.kaggle.com/datasets/hojjatk/mnist-dataset

[4] MNIST Fashion dataset: https://www.kaggle.com/datasets/zalando-research/fashionmnist