SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 1543

Duboki generativni modeli temeljeni na prostornom rasporedu dijelova slike

Matija Folnović

Zagreb, srpanj 2017.

Umjesto ove stranice umetnite izvornik Vašeg rada.
Da bi ste uklonili ovu stranicu obrišite naredbu `\izvornik`.

Zahvaljujem se mentoru, izv. prof. dr. sc. Siniši Šegviću na podršci i pomoći kroz preddiplomski i diplomski studij. Posebice mu se zahvaljujem na savjetima po pitanju odabira teme, implementacije, pisanje rada i odabiru vlastitog puta nakon završetka diplomskog studija.

Zahvaljujem se obitelji, prijateljima i kolegama u tvrtki CROZ d.o.o. na podršci kroz cijeli studij, i u dobrim i u lošim trenucima.

Sadržaj

1. Uvod

Računalni vid (engl. *Computer Vision*) je područje koje se bavi obradom, analizom i razumijevanjem slika. Problem kojim se bavimo u ovom radu je raspoznavanje ili klasifikacija slika. Sustav za raspoznavanje slika na ulazu dobiva sliku, a zadatak sustava je na izlazu postaviti razred ili klasu ulazne slike.

Za raspoznavanje slika, danas su popularne duboke neuronske mreže - konkretnije konvolucijske neuronske mreže. Problem takvih modela je u prenaučenosti. Jedan od načina na koji se taj problem rješava je uvođenjem regularizacije. Konkretan način regularizacije na koji se ovaj rad fokusira je učenje reverzibilne reprezentacije ulazne slike. Iz reprezentacije model mora naučiti rekonstruirati ulaznu sliku. Takav model naziva se autoenkoder i sastoji se od dva dijela: enkoder (kodira ulaznu sliku u reprezentaciju) i dekoder (rekonstruira ulaznu sliku iz reprezentacije).

Konvolucijski autoenkoder je strukturiran kao najobičniji autoenkoder koji koristi konvoluciju i sloj sažimanja maksimalnom vrijednošću u enkoderu i konvoluciju i naduzorkovanje (engl. *upsample*) u dekoderu. Takav model ne koristi lokacije maksimalnih vrijednosti (engl. $\arg\max$) iz sloja sažimanja. Jedan od fokusa ovog rada je istražiti možemo li koristiti lokacije maksimalnih vrijednosti za poboljšavanje preciznosti rekonstrukcije i efekt regularizacije nad točnosti klasifikacije. Time u reprezentaciji dobivamo i lokacije maksimalnih vrijednosti, tj. lokacije bitnih dijelova slike. To podsjeća na model kapsula[7]. Zbog toga će se u sklopu ovog rada evaluirati ponašanje modela ako ga učimo tako da je očekivana izlazna slika rekonstrukcije jednaka pomaknutoj ulaznoj slici. Time je model prisiljen naučiti translaciju. Opisani način je način na koji učimo kapsule. Dodatno, funkcija $\arg\max$ nije derivabilna, ali istražujemo drugačiji sloj sažimanja koji koristi derivabilnu aproksimaciju funkcije $\arg\max$.

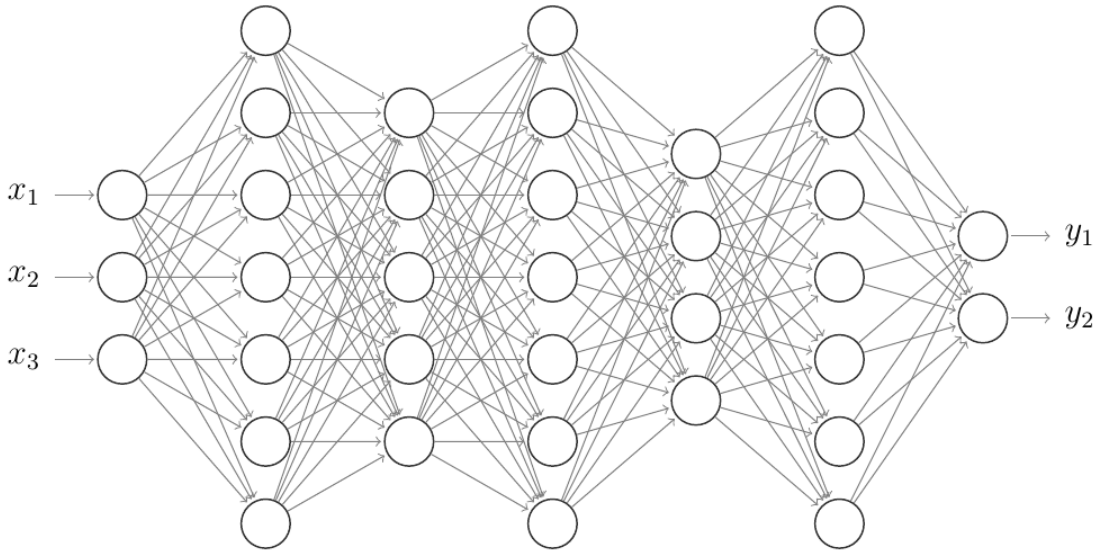
U drugom poglavlju će biti objašnjene osnove dubokih neuronskih mreža, konvolucijskih neuronskih mreža i autoenkodera. U trećem poglavlju bavit ćemo se modelom koji je u glavnom fokusu ovog rada: duboki što-gdje autoenkoder (engl. *Stacked What-Where Autoencoder*). Kroz četvrto poglavlje vidjet ćemo detalje programske izvedbe u razvojnom okviru *Tensorflow*, a u petom poglavlju rezultate provedenih eksperimenata.

2. Duboke neuronske mreže

U ovom poglavlju će biti objašnjeni temelji umjetnih neuronskih mreža, konvolucijskih neuronskih mreža i autoenkodera. Detaljnija objašnjenja nalaze se u literaturi.

2.1. Umjetna neuronska mreža

Umjetna neuronska mreža je skup neurona povezanih u nizu slojeva. Primjer neuronske mreže prikazan je na slici 2.1.



Slika 2.1: Primjer neuronske mreže

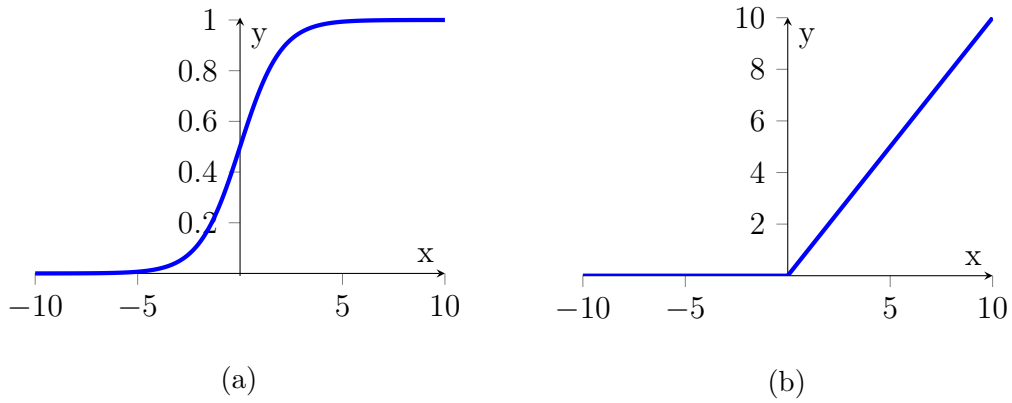
Prije objašnjavanja neuronske mreže, moramo objasniti što je neuron. Matematički rečeno, neuron je funkcija koja na ulazu prima vektor $\vec{x} = [x_1, x_2, ..., x_n]$, a na izlazu se nalazi skalarna vrijednost y. Funkcija je parametrizirana vektorom težina $\vec{w}$, skalarnom vrijednosti b koju nazivamo prag (engl. *bias*) i aktivacijskom funkcijom f (engl. *activation function*). Cijeli izraz za izlaz neurona prikazan je u nastavku.

$$y = f(\vec{x} \cdot \vec{w} + b) = f(\sum_{i=1}^{n} x_i \cdot w_i + b) \tag{2.1}$$

Aktivacijske funkcije koristimo kako bi postigli nelinearnost između slojeva. Bez nelinearne aktivacijske funkcije, neuronska mreža ne može naučiti bilo kakvu nelinearnost koja postoji u podacima. Osim toga, više slojeva bez aktivacijskih funkcija je ekvivalentno jednom takvom sloju. Koriste se dvije vrste aktivacijskih funkcija: logistička funkcija i ReLU, čiji su izrazi navedeni u nastavku, a grafovi prikazani na slici 2.2.

$$f(x) = \sigma(x) = \frac{1}{1 + e^{-x}} \tag{2.2}$$

$$f(x) = ReLU(x) = \max(0, x) \tag{2.3}$$



(a) (b)

Slika 2.2: Prijenosne funkcije: (a) logistička funkcija, (b) ReLU

Postoji i aktivacijska funkcija *softmax*[6], koja se koristi za normalizaciju izlaza $\vec{y} = [y_1, y_2, ..., y_N]$ tako da normalizirani izlaz zadovoljava dva svojstva:

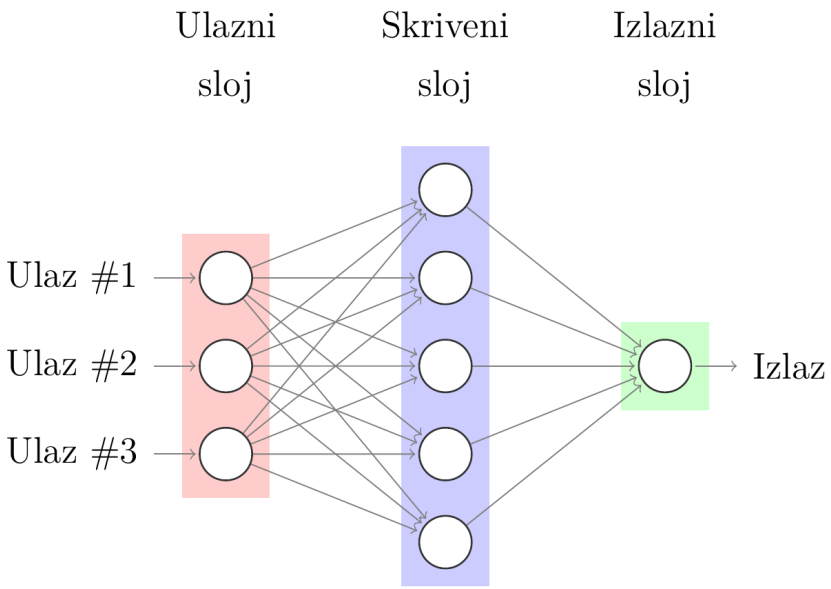
1. $\forall i \in [1, N] : y_i \in [0, 1]$
2. $\sum_{i=1}^{N} y_i = 1$

Izraz za aktivacijsku funkciju *softmax* prikazan je u nastavku.

$$f(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{N} e^{x_j}}$$

Bez smanjenja općenitosti, zamislimo neuronsku mrežu s dva sloja: prvi s $N = 5$, a drugi s $M = 1$ neurona. Na ulazu neuronske mreže nalazi se vektor

$\vec{x} = [x_1, x_2, x_3]$, a na izlazu se nalazi skalarna vrijednost y. Struktura mreže prikazana je na slici 2.3.



Slika 2.3: Primjer dvoslojne neuronske mreže

U nastavku su prikazani izrazi neuronske mreže u matričnom obliku.

$$X = [x_1 x_2 x_3]^T \tag{2.4}$$

$$W^1 = \begin{bmatrix} W^1_{11} & W^1_{12} & W^1_{13} \\ W^1_{21} & W^1_{22} & W^1_{23} \\ W^1_{31} & W^1_{32} & W^1_{33} \\ W^1_{41} & W^1_{42} & W^1_{43} \\ W^1_{51} & W^1_{52} & W^1_{53} \end{bmatrix} \tag{2.5}$$

$$H = W^1 X \tag{2.6}$$

$$W^2 = \begin{bmatrix} W^2_{11} & W^2_{12} & W^2_{13} & W^2_{14} & W^2_{15} \end{bmatrix} \tag{2.7}$$

$$Y = W^2 H = [y] \tag{2.8}$$

2.2. Konvolucijske neuronske mreže

Konvolucijske neuronske mreže[5] su neuronske mreže kod kojih se na ulazu i izlazu nalaze dvodimenzionalni podaci, kao što su slike. Koristimo dvije različite vrste slojeva kod konvolucijskih neuronskih mreža: konvolucijski sloj (engl. *convolutional layer*) i sloj sažimanja (engl. *pooling layer*).

Konvolucijski sloj sadrži N neurona s jezgrom $K \in \mathbb{R}^M \times \mathbb{R}^M$. Ako je $X \in \mathbb{R}^N \times \mathbb{R}^N$ ulaz u neuron konvolucijskog sloja, izlaz je definiran kao:

$$Y = X * K$$

Gdje operator $*$ predstavlja 2D konvoluciju. Izlaz je matrica dimenzija $Y \in \mathbb{R}^{N-M+1} \times \mathbb{R}^{N-M+1}$.

U nastavku je prikazan primjer konvolucije nad matricama $X \in \mathbb{R}^5 \times \mathbb{R}^5$ i $K \in \mathbb{R}^2 \times \mathbb{R}^2$. Rezultat operacije konvolucije je matrica $Y \in \mathbb{R}^4 \times \mathbb{R}^4$.

$$Y_{00} = X_{00} \cdot K_{00} + X_{01} \cdot K_{01} + X_{10} \cdot K_{10} + X_{11} \cdot K_{11} \tag{2.9}$$

$$Y_{01} = X_{01} \cdot K_{00} + X_{02} \cdot K_{01} + X_{11} \cdot K_{10} + X_{12} \cdot K_{11} \tag{2.10}$$

$$\dots$$

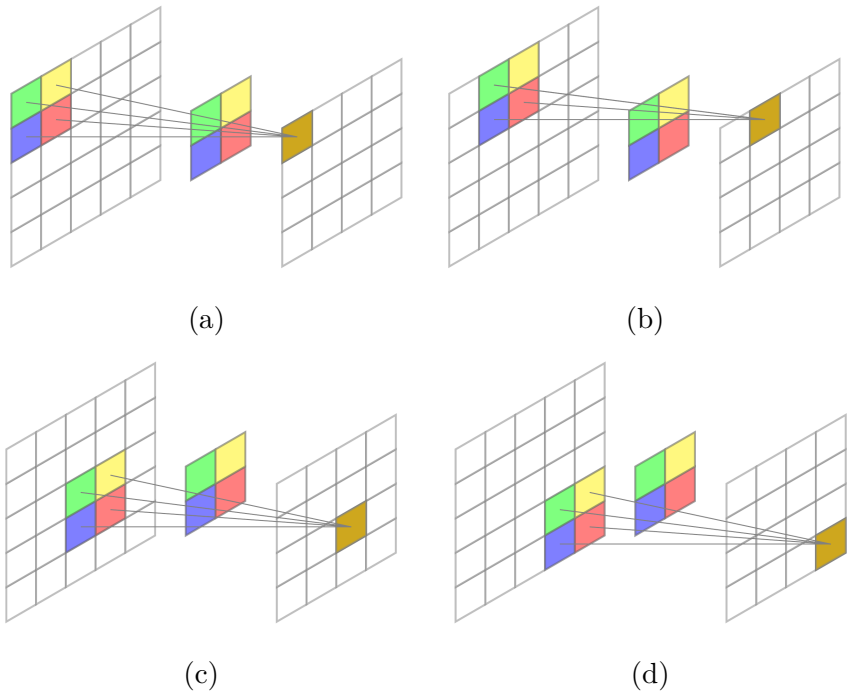
$$Y_{03} = X_{03} \cdot K_{00} + X_{04} \cdot K_{01} + X_{13} \cdot K_{10} + X_{14} \cdot K_{11} \tag{2.11}$$

$$Y_{10} = X_{10} \cdot K_{00} + X_{11} \cdot K_{01} + X_{20} \cdot K_{10} + X_{21} \cdot K_{11} \tag{2.12}$$

$$\dots$$

$$Y_{33} = X_{33} \cdot K_{00} + X_{34} \cdot K_{01} + X_{43} \cdot K_{10} + X_{44} \cdot K_{11} \tag{2.13}$$

$$\dots \tag{2.14}$$



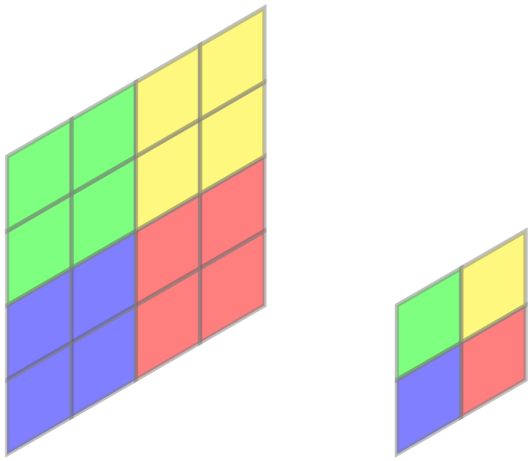
Slika 2.4: Ilustracija konvolucije: (a) za Y_{00}, (b) za Y_{10}, (c) za Y_{22} i (d) Y_{33}

Postoji više vrsta slojeva sažimanja: sažimanje maksimalnom vrijednošću (engl. *max pooling*) i sažimanje usrednjavanjem (engl. *mean pooling*).

Ulaznu sliku $X \in \mathbb{R}^N \times \mathbb{R}^N$ ćemo podijeliti na dijelove jednakih dimenzija $K \times K$, K je konstanta sažimanja. Kod sažimanja maksimalnom vrijednošću, u svakom djelu uzimamo maksimalnu vrijednost unutar njega dok kod sažimanja usrednjavanjem uzimamo srednju vrijednost. Na izlazu se nalazi slika $Y \in \mathbb{R}^{N/K} \times \mathbb{R}^{N/K}$.

U nastavku je prikazan primjer sloja sažimanja maksimalnom vrijednošću nad matricom $X \in \mathbb{R}^4 \times \mathbb{R}^4$, te konstantom sažimanja $K = 2$. Na izlazu se nalazi matrica $Y \in \mathbb{R}^2 \times \mathbb{R}^2$.

$$Y_{00} = max(X_{00}, X_{01}, X_{10}, X_{11}) \tag{2.15}$$
$$Y_{01} = max(X_{02}, X_{03}, X_{12}, X_{13}) \tag{2.16}$$
$$Y_{10} = max(X_{20}, X_{21}, X_{30}, X_{31}) \tag{2.17}$$
$$Y_{11} = max(X_{22}, X_{23}, X_{32}, X_{33}) \tag{2.18}$$

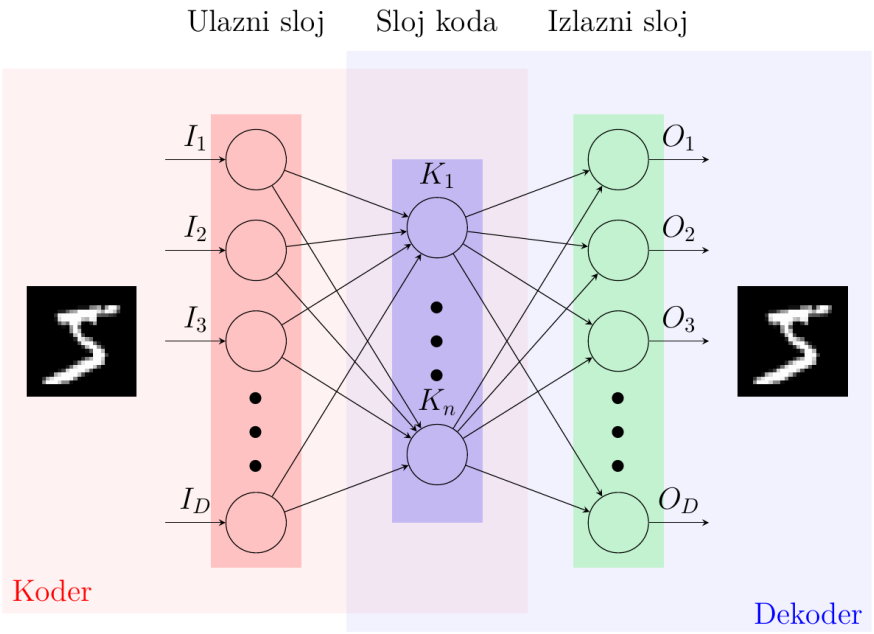


Slika 2.5: Ilustracija sloja sažimanja

2.3. Autoenkoder

Autoenkoder[4] je generativni model koji se sastoji od dva dijela: koder i dekoder. Na ulazu autoenkodera nalazi se $\vec{I} \in \mathbb{R}^D$. Zadatak kodera je transformacija ulaza $\vec{I}$ u kôd $\vec{K} \in \mathbb{R}^n$, gdje tipično vrijedi $n < D$. Zadatak dekodera je rekonstrukcija

ulaza $\vec{I}$ isključivo iz kôda $\vec{k}$ u izlaz $\vec{O} \in \mathbb{R}^D$.



Slika 2.6: Shema autoenkodera

Matematički izrazi za koder i dekoder su prikazani u nastavku.

$$K = f(W_{IK} \cdot I + b_{IK}) \tag{2.19}$$

$$O = f(W_{KO} \cdot K + b_{KO}) \tag{2.20}$$

U koderu, W_{IK} su težine, a b_{IK} prag između ulaznog sloja i sloja koda. U dekoderu, W_{KO} su težine, a b_{KO} prag između sloja koda i izlaznog sloja. Funkcija f je aktivacijska funkcija.

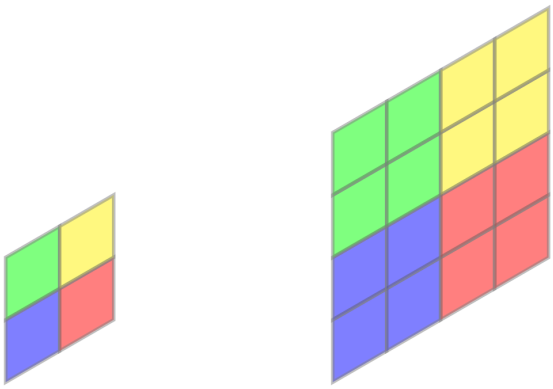
Ako se vrijednosti ulaznih i izlaznih vektora nalaze unutar intervala $[0, 1]$, težine kodera i dekodera možemo učiti minimizirajući funkciju pogreške koja se temelji na funkciji sličnosti između originalnog vektora (ulaz) i rekonstrukcije (izlaz), navedenu u nastavku.

$$L = -\sum_{k=1}^{N}(I_k \cdot logO_k + (1 - I_k) \cdot log(1 - O_k)) \tag{2.21}$$

N je broj ulaznih podataka, I_k je k-ti ulazni, a O_k k-ti izlaz iz dekodera.

U autoenkoderu možemo koristiti i konvolucijski sloj i sloj sažimanja. Problem nastaje u dekoderu - potrebna nam je obrnuta operacija od sloja sažimanja.

Operacija koju možemo koristiti kao obrnutu operaciju sloja sažimanja je naduzorkovanje (engl. *upsample*), koja se svodi na ponavljanje svake vrijednosti u oba smjera broj puta koliko je velika jezgra. Ilustracija naduzorkovanja prikazana je na slici 2.7.



Slika 2.7: Ilustracija naduzorkovanja

3. Duboki što-gdje autoenkoder

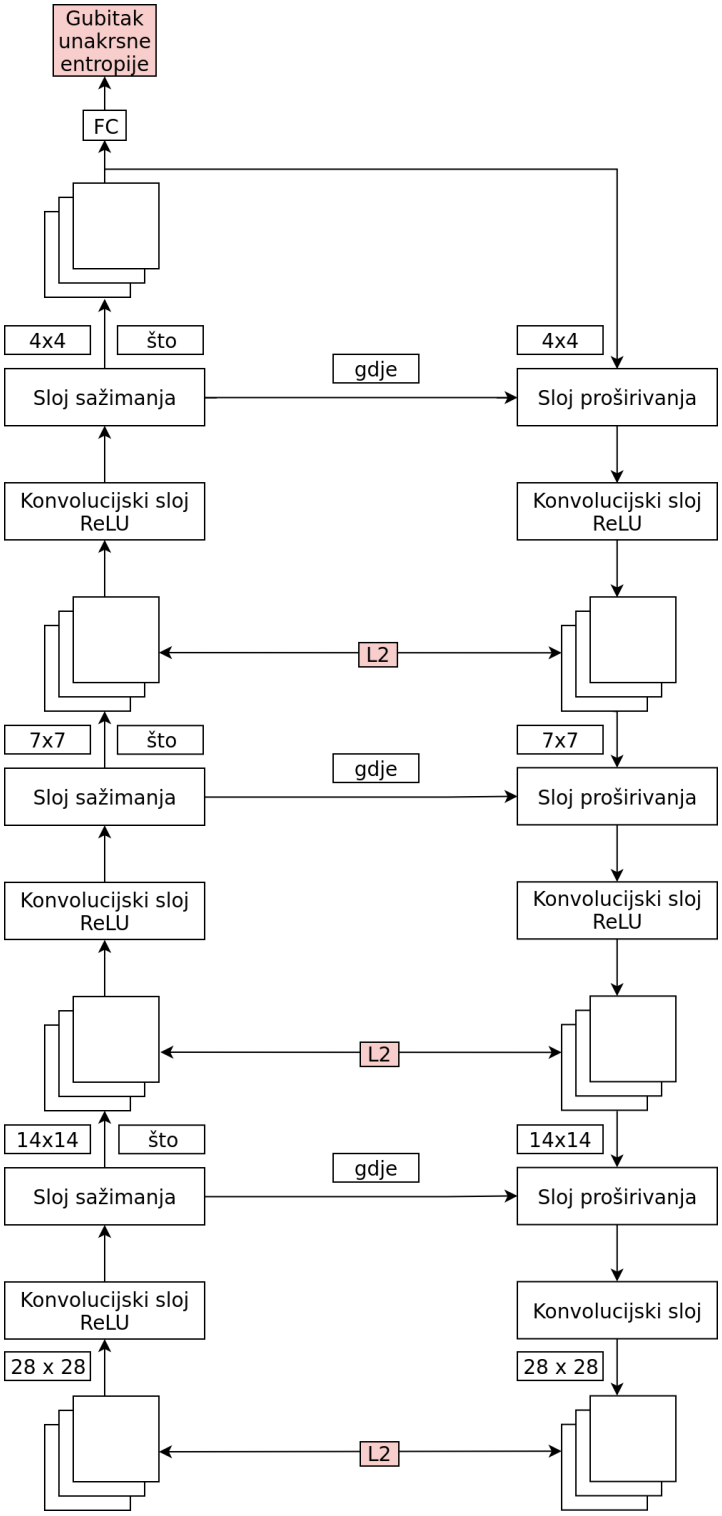
Duboki što-gdje autoenkoder (engl. *Stacked what-where autoencoder*) je konvolucijski autoenkoder koji istovremeno sadrži i diskriminativni i generativni dio. Time pruža mogućnost učenja modela na nadzirani, nenadzirani ili polunadzirani način bez mijenjanja samog modela i metode učenja.

Struktura modela prikazana je na slici 3.1. Svaki sloj kodera sastoji se od konvolucijskog sloja, aktivacijske funkcije *ReLU* i sloja sažimanja. Koder je prikazan na lijevoj strani slike. Na desnoj strani slike prikazan je dekoder, koji se sastoji od sloja proširivanja (engl. *unpool*), konvolucijskog sloja i aktivacijske funkcije ReLU. U odnosu na konvolucijski autoenkoder opisan u poglavlju 2.3, ovaj model se konceptualno razlikuje u tri stvari, koje će u nastavku poglavlja biti detaljnije objašnjene:

1. sloj sažimanja maksimalnom vrijednošću je prilagođen tako da na izlazu ima dvije vrijednosti: maksimalne vrijednosti u svakoj jezgri i lokacije maksimalnih vrijednosti u svakoj jezgri

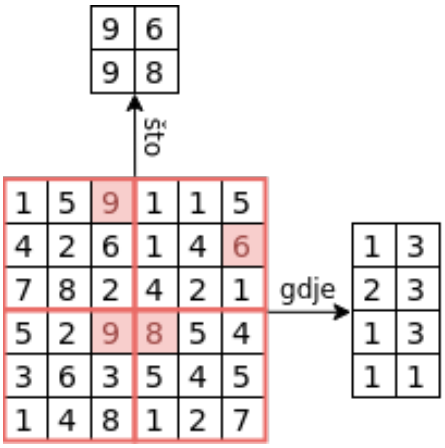
2. kako bi postigli derivabilnost sloja sažimanja po lokaciji, uvodimo meki sloj sažimanja maksimalnom vrijednošću koji aproksimira *max* i *argmax* derivabilnim izrazima

3. L2 gubitak između reprezentacije nakon svakog sloja sažimanja i izlaza nakon pripadajućeg naduzorkovanja



Slika 3.1: Struktura dubokog što-gdje autoenkodera. Svaki sloj kodera (lijevo) sastoji se od konvolucijskog sloja, aktivacijske funkcije $ReLU$ i sloja sažimanja. Svaki sloj dekodera (desno) sastoji se od sloja proširivanja, konvolucijskog sloja i aktivacijske funkcije ReLU. Sloj sažimanja propušta informaciju o lokacijama prema sloju proširivanja u dekoderu.

Sloj sažimanja maksimalnom vrijednošću smo proširili s lokacijama maksimalnih vrijednosti (engl. *arg max*) u svakoj jezgri uz postojeće maksimalne vrijednosti maksimalne vrijednosti (engl. *max*). Maksimalna vrijednost predstavlja *što*, dok lokacije maksimalnih vrijednosti predstavljaju *gdje* u *što-gdje autoenkoder*. To je na primjeru ilustrirano na slici 3.2.



Slika 3.2: Prošireni sloj sažimanja maksimalnom vrijednošću

Kao što je prikazano na slici 3.1, *što* prosljeđujemo dalje u sljedeći sloj kodera, a *gdje* prosljeđujemo u pripadajući sloj proširivanja u dekoderu. Ovo možemo tumačiti i kao preskočnu vezu[3] (engl. *skip connection*). Preskočna veza omogućava gradijentu preskakanje slojeva, što u literaturi pomaže u učenju modela. U odnosu na naduzorkovanje, sloj proširivanja koristi *gdje* kako bi vrijednost postavila samo na lokaciju koja se nalazi u *gdje*.

U radu se istražuje modeli koji koriste jednu od dvije vrste sloja sažimanja maksimalnom vrijednošću: tvrdi (engl. *hard*) i meki (engl. *soft*). Tvrdi sloj sažimanja je onaj opisan u poglavlju 2.2. Njegov problem je taj što funkcija arg max pomoću koje se računa *gdje* nije derivabilna. Meki sloj sažimanja definiran je izrazima u nastavku.

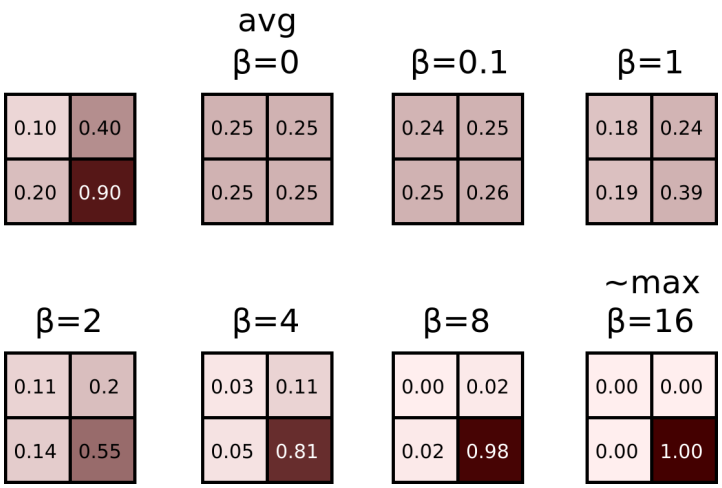
$$m_k = \sum_{N_k} z(x,y) \cdot \frac{e^{\beta \cdot z(x,y)}}{\sum_{N_k} e^{\beta \cdot z(x,y)}} \approx \max_{N_k} z(x,y) \tag{3.1}$$

$$p_k = \sum_{N_k} \begin{bmatrix} x \\ y \end{bmatrix} \cdot \frac{e^{\beta \cdot z(x,y)}}{\sum_{N_k} e^{\beta \cdot z(x,y)}} \approx \arg\max_{N_k} z(x,y) \tag{3.2}$$

U izrazima, $\sum_{N_k}$ predstavlja sumu po cijeloj jezgri, (x, y) predstavlja poziciju u ulaznoj matrici, $z(x, y)$ predstavlja vrijednost u ulaznoj matrici na toj poziciji,

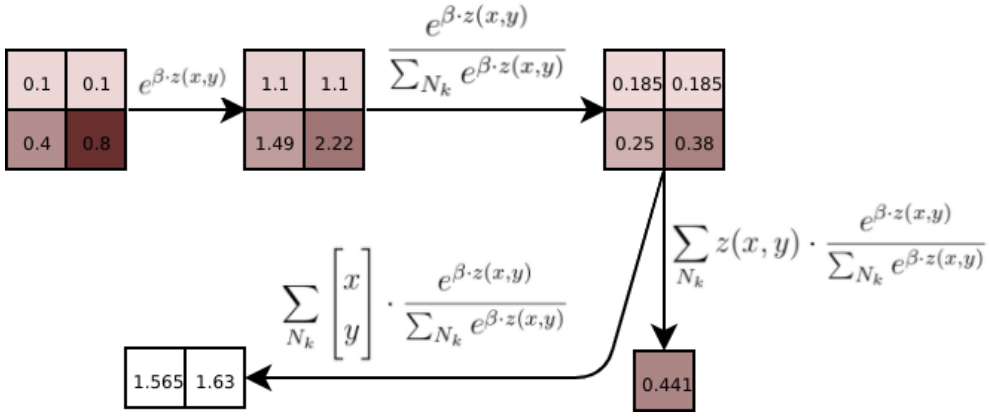
β predstavlja konstantu mekog sloja sažimanja, m_k predstavlja saženu vrijednost za jezgru k, a p_k predstavlja lokaciju sažete vrijednosti.

Izrazi su parametrizirani parametrom $\beta \geq 0$. Što je β veća, to se meki sloj sažimanja približava tvrdom sloju sažimanja maksimalnom vrijednošću. To se može vidjeti iz dijela izraza $e^{\beta \cdot z(x,y)}$, gdje će veća β povećati razliku između najveće vrijednosti i ostalih vrijednosti unutar jezgre, čime će u cijelom izrazu najveća vrijednost imati jaču vrijednost. S druge strane, što je β manja, to se meki sloj sažimanja približava tvrdom sloju sažimanja usrednjavanjem. Utjecaj parametra β prikazan je na slici u nastavku.



Slika 3.3: Utjecaj parametra β na izlaz mekog sloja sažimanja na primjeru jednog ulaza. Prva matrica (gore lijevo) je ulazna matrica sloja sažimanja, dok su ostale matrice izlazne matrice u ovisnosti o parametru β.

Koraci izračuna izraza 3.1 i 3.2 ilustrirani su na primjeru na slici u nastavku.



Slika 3.4: Ilustracija izraza za prošireni sloj sažimanja maksimalnom vrijednošću

Cijeli model učimo minimizirajući gubitak koji je naveden u nastavku.

$$L = \lambda_{CE} \cdot L_{CE} + \lambda_{L2_{rec}} \cdot L2_{rec} + \lambda_{L2_M} \cdot L2_M \tag{3.3}$$

$$L_{CE} = -\sum_{i=1}^{N} y_i \cdot log(y_i') \tag{3.4}$$

$$L2_* = \frac{1}{n} \cdot \left\| x - x' \right\|_2 \tag{3.5}$$

Koristimo gubitak unakrsne entropije (L_{CE}) za klasifikacijski gubitak, a L2 normu ($L2_*$) za rekonstrukcijski gubitak. $L2_{rec}$ je rekonstrukcijski gubitak između originalnog ulaza u model i izlaza iz dekodera, a $L2_M$ je rekonstrukcijski gubitak između unutarnjih reprezentacija. U izrazu L_{CE}, y_i je i-ta značajka ulazne slike, a y_i' je rekonstruirana i-ta značajka.

Model možemo učiti na sljedeća četiri načina samo mijenjajući hiperparametre λ_{CE}, $\lambda_{L2_{rec}}$ i λ_{L2_M}, bez mijenjanja strukture modela ni načina učenja:

- potpuno nadzirano (bez regularizacije): $\lambda_{CE} \neq 0$, $\lambda_{L2_{rec}} = 0$, $\lambda_{L2_M} = 0$
- potpuno nadzirano (s regularizacije): $\lambda_{CE} \neq 0$, $\lambda_{L2_{rec}} \neq 0$, $\lambda_{L2_M} \neq 0$
- nenadzirano: $\lambda_{CE} = 0$, $\lambda_{L2_{rec}} \neq 0$, $\lambda_{L2_M} \neq 0$
- polunadzirano: $\lambda_{CE} \neq 0$, $\lambda_{L2_{rec}} \neq 0$, $\lambda_{L2_M} \neq 0$

4. Programska izvedba

Za cijelu programsku izvedbu korišten je programski jezik *Python 3.6.0* i razvojni okviri *NumPy* i *Tensorflow*. U nastavku poglavlja opisani su detalji programske izvedbe.

4.1. Podatkovni skup - MNIST

Podatkovni skup MNIST sastoji se od 50000 slika znamenaka od 0 do 9 za učenje, 10000 slika za validaciju i 10000 slika za testiranje. Slike su dimenzija 28×28 s vrijednostima u intervalu $[0, 1]$. U radu je isprobana normalizacija slika na $\mu = 0, \sigma = 1$, što je rezultiralo gorim rezultatima zbog čega su svi eksperimenti pokrenuti bez normalizacije slika. Dio skupa prikazan je na slici 4.1.

Slika 4.1: 100 nasumično odabranih znamenki iz skupa MNIST

4.2. Meki sloj sažimanja

Kako bi izbjegli pisanje svoje implementacije operacija koje direktno implementiraju izraze 3.1 i 3.2, odlučili smo se implementirati izraze koristeći samo postojeće *Tensorflow* operacije. Također, kako bi pojednostavili implementaciju sloja proširivanja u dekoderu, *gdje* ne kodiramo s dvije vrijednosti (x, y).

Srž implementacije je u tome što zajednički desni dio u izrazima 3.1 i 3.2 predstavlja matricu "koliko je svaka vrijednost zastupljena u odnosu na ostale vrijednosti u jezgri", uz svojstvo da je suma vrijednosti jednaka 1 (softmax). Taj dio ćemo prozvati našim *gdje*. Za naš *što* ćemo koristiti hadamardov produkt matrica *gdje* i ulazne matrice z, provučen kroz tvrdi sloja sažimanja usrednjavanjem. Zbog ovoga je i dekoder pojednostavljen, koji je sada samo umnožak *gdje* i rekonstruirani *što* nakon sloja naduzorkovanja. Shema implementacije prikazana je na slici 4.2.

Dodatno, gornji izrazi su promijenjeni na način u nastavku, kako bi se poboljšala numerička stabilnost. Ovo se ispostavilo kao ključan detalj da bi implementacija radila za $\beta > 4$.

$$m_k = \sum_{N_k} z(x,y) \cdot \frac{e^{\beta \cdot z(x,y) - \max_{N_k}(\beta \cdot z(x,y))}}{\sum_{N_k} e^{\beta \cdot z(x,y) - \max_{N_k}(\beta \cdot z(x,y))}} \approx \max_{N_k} z(x,y) \tag{4.1}$$

$$p_k = \sum_{N_k} \begin{bmatrix} x \\ y \end{bmatrix} \cdot \frac{e^{\beta \cdot z(x,y) - \max_{N_k}(\beta \cdot z(x,y))}}{\sum_{N_k} e^{\beta \cdot z(x,y) - \max_{N_k}(\beta \cdot z(x,y))}} \approx \arg\max_{N_k} z(x,y) \tag{4.2}$$

4.3. Tvrdi sloj sažimanja

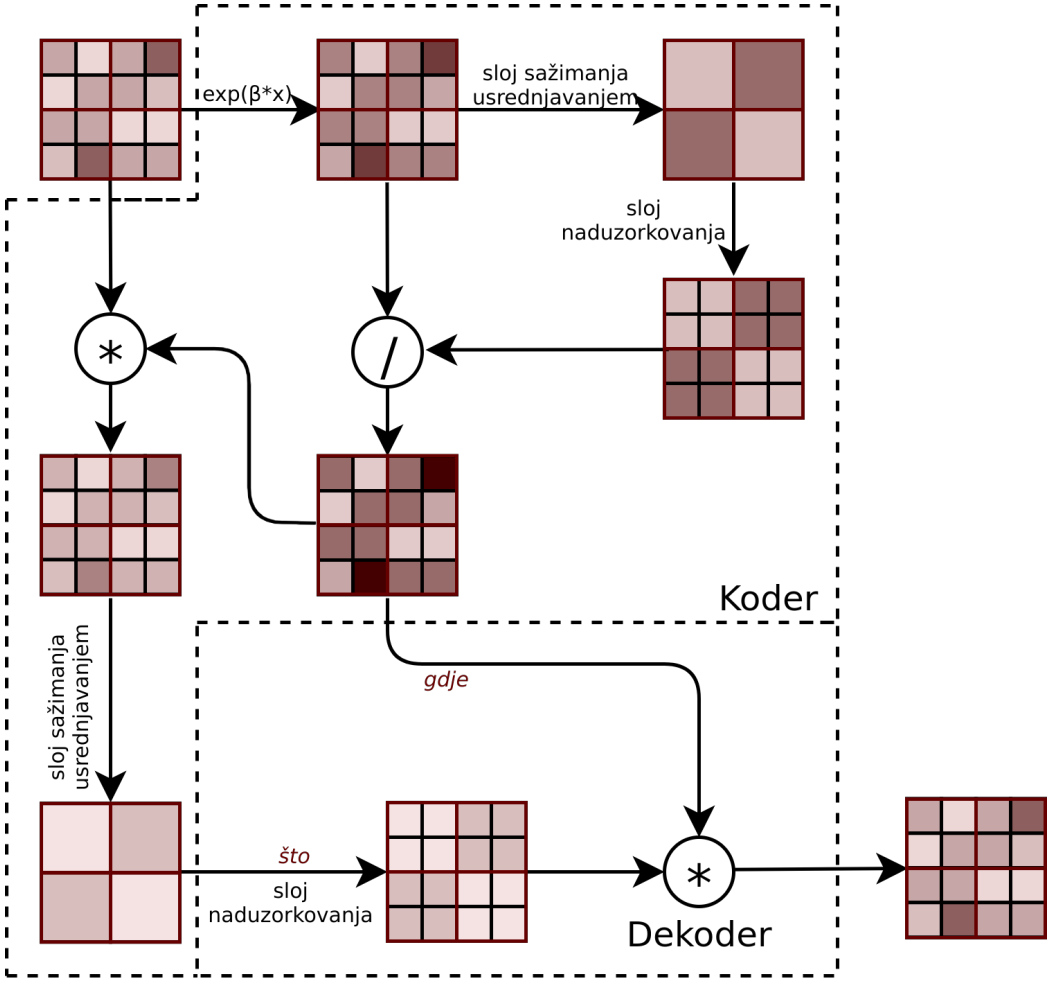
Tensorflow sadrži implementaciju[1] sloja sažimanja maksimalnom vrijednošću koja na izlazu također ima i lokaciju maksimalne vrijednosti. S druge strane, ne sadrži sloj proširivanja koji bi radio tako da ulazne vrijednosti ne kopira na sve lokacije (naduzorkovanje), nego samo u odgovarajuću lokaciju koja se nalazi u *gdje*. Postoji otvoren problem[2] na Githubu i zaobilazno rješenje [3] koje koristimo u ovom radu. Zaobilazno rješenje se svodi na preoblikovanje matrice *gdje* ($N \times 2$)

[1] `https://www.tensorflow.org/api_docs/python/tf/nn/max_pool_with_argmax`

[2] `https://github.com/tensorflow/tensorflow/issues/2169`

[3] `https://github.com/tensorflow/tensorflow/issues/2169#issuecomment-291238088`

na dva vektora dimenzije N i korištenjem operacije *scatter_nd* čiji su parametri vrijednosti i lokacije na koje operacija postavlja te vrijednosti u izlaznoj matrici.



Slika 4.2: Skica implementacije mekog sloja sažimanja

4.4. Ostali detalji

Sve modele učimo koristeći algoritam minimizacije s adaptivnom stopom učenja - Adam sa stopom učenja $\alpha = 0.001$ i veličinom grupe 256. Koristimo tehniku ranog zaustavljanja (engl. *early stopping*) - zaustavljamo učenje ako nakon 100 epoha nema poboljšanja na skupu za validaciju. Modeli koji koriste meki sloj sažimanja se uče 3-4 puta sporije od modela s tvrdim slojem sažimanja, zbog čega kod takvih modela koristimo normalizaciju nad grupom (engl. *batch norm*).

5. Rezultati

U nastavku će biti prikazani rezultati obavljenih eksperimenata. U eksperimentima se koristi notacija `(16)5c-(32)3c-2p-10fc`, gdje `(16)5c` predstavlja konvolucijski sloj sa 16 filtara i jezgrom veličine 5×5, `2p` predstavlja sloj sažimanja s jezgrom veličine 2×2, a `10fc` predstavlja potpuno-povezani sloj s 10 neurona. Hiperparametar λ_{L2} predstavlja koeficijent uz oba L2 gubitka.

5.1. Nadzirana klasifikacija

U ovom eksperimentu evaluiramo točnost modela u kontekstu nadzirane klasifikacije. Nadzirana klasifikacija znači da tijekom učenja koristimo oznake svih 50000 slika. Koristimo arhitekturu `(64)5c-2p-(64)3c-2p-(64)3c-2p-10fc`. Uspoređujemo dobivene rezultate kada koristimo tvrdi sloj sažimanja u odnosu na meki sloj sažimanja. Gubitak koji minimiziramo naveden je u nastavku.

$$L = L_{CE} + \lambda_{L2} \cdot L2_{rec} + \lambda_{L2} \cdot L2_M$$

Model evaluiramo u ovisnosti o parametru λ_{L2}. Za $\lambda_{L2} = 0$ vrijedi da je cijeli model jednak klasičnoj klasifikacijskoj konvolucijskog mreži, bez rekonstrukcije.

Rezultati su navedeni u tablicama 5.1 (tvrdi sloj sažimanja) i 5.2 (meki sloj sažimanja). Kod mekog sažimanja koristimo $\beta = 1$.

λ_{L2}	Točnost učenja	Validacijska točnost	Točnost testiranja
0.0	100.00	99.56	99.45
0.5	100.00	**99.60**	**99.42**
1.0	100.00	99.50	99.45
1.5	99.96	99.44	99.41
2.0	99.97	99.44	99.36
2.5	100.00	99.46	99.24
3.0	99.97	99.50	99.39

Tablica 5.1: Dobivena točnost nakon potpuno nadziranog učenja modela s tvrdim sažimanjem maksimalnom vrijednošću

λ_{L2}	Točnost učenja	Validacijska točnost	Točnost testiranja
0.0	99.99	99.56	99.55
0.5	100.00	99.50	99.44
1.0	100.00	**99.64**	**99.37**
1.5	99.76	99.40	99.33
2.0	99.92	99.42	99.26
2.5	99.87	99.34	99.19
3.0	99.82	99.28	99.11

Tablica 5.2: Dobivena točnost nakon potpuno nadziranog učenja modela s mekim sažimanjem maksimalnom vrijednošću

U literaturi [8] je za istu konfiguraciju modela uz korištenje tvrdog sažimanja maksimalnom vrijednošću dobivena točnost testiranja 99.29% za $\lambda_{L2} > 0$ u odnosu na 99.24% za $\lambda_{L2} = 0$. Hiperparametar λ_{L2} ćemo odabrati tako da odaberemo onu vrijednost za koju model postiže najbolju validacijsku točnost. Vidljivo je da je dobivena bolja točnost testiranja korištenjem tvrdog sloja sažimanja (tablica 5.1) u odnosu na meki sloj sažimanja (tablica 5.2). Dobiveni rezultati u tablici 5.1 pokazuju bolje rezultate u odnosu na literaturu[8].

5.2. Polunadzirana klasifikacija

U poglavlju 5.1, arhitekturu `(64)5c-2p-(64)3c-2p-(64)3c-2p-10fc` smo nadzirano evaluirali uz korištenje tvrdog i mekog sloja sažimanja. U ovom eksperimentu želimo evaluirati regularizacijski efekt koji postiže takva arhitektura, uz korištenje rekonstrukcijskog i klasifikacijskog gubitak i korištenje lokacija maksimalnih vrijednosti. Evaluaciju regularizacijskog efekta postižemo tako da tijekom učenja modelu dajemo manji broj označenih slika.
i

Konkretnije, trenutni algoritam za provođenje polunadziranog učenja je:

- u neparnim epohama provodi se nenadzirano učenje: prolazi se kroz cijeli skup (50000) označenih slika i minimizira se samo rekonstrukcijski gubitak
- u parnim epohama provodi se nadzirano učenje: prolazi se kroz manji skup (N) označenih slika i minimizira se istovremeno rekonstrukcijski i klasifikacijski gubitak

Manji skup stvaramo tako da, za svaki od 10 razreda nasumično biramo $\frac{N}{10}$ slika iz cijelog skupa označenih slika. U takvom skupu dodatno pomiješamo primjere i dobivamo manji skup veličine N, balansiran po razredima.

Veličina skupa	λ_{L2}	Točnost učenja	Validacijska točnost	Točnost testiranja	Validacijska točnost (literatura)
3000	0.0	99.93	98.42	98.17	97.00
	0.01	100.00	**98.78**	98.53	
	0.1	100.00	98.56	98.35	
	0.5	100.00	98.70	98.55	97.50
	1.0	99.97	98.62	98.60	97.50
	1.5	100.00	98.24	98.29	97.50
1000	0.0	99.90	97.14	97.15	95.00
	0.01	100.00	**97.48**	97.12	
	0.1	100.00	97.16	97.13	
	0.5	100.00	96.94	97.11	96.00
	1.0	99.80	96.82	96.42	96.00
	1.5	100.00	96.80	96.55	96.00
600	0.0	99.83	**96.50**	96.40	93.00
	0.01	100.00	96.20	96.27	
	0.1	100.00	96.28	96.40	
	0.5	99.83	95.66	95.63	94.50
	1.0	99.83	95.76	95.61	94.50
	1.5	99.83	95.62	95.41	94.50
100	0.0	99.00	87.20	88.62	83.00
	0.01	100.00	**89.30**	90.25	
	0.1	100.00	88.70	88.96	
	0.5	99.00	86.72	88.41	87.50
	1.0	100.00	85.70	86.35	88.00
	1.5	100.00	86.90	87.99	88.25

Tablica 5.3: Dobivena točnost nakon polunadziranog učenja modela s tvrdim sažimanjem maksimalnom vrijednošću

Veličina skupa	λ_{L2}	Točnost učenja	Validacijska točnost	Točnost testiranja
3000	0.0	100.00	98.20	98.30
	0.01	100.00	**98.76**	98.62
	0.1	100.00	98.34	98.33
	0.5	99.93	97.86	97.62
	1.0	99.97	97.74	97.58
	1.5	99.97	97.36	97.46
1000	0.0	100.00	97.20	96.90
	0.01	100.00	**97.62**	97.37
	0.1	100.00	97.06	97.06
	0.5	99.90	96.24	95.81
	1.0	99.90	95.74	95.40
	1.5	99.90	95.50	94.81
600	0.0	99.83	95.54	95.67
	0.01	100.00	**96.50**	96.33
	0.1	100.00	95.82	95.53
	0.5	99.83	94.72	94.36
	1.0	99.83	94.50	94.17
	1.5	99.83	93.90	93.59
100	0.0	100.00	87.04	87.41
	0.01	100.00	**89.04**	89.66
	0.1	100.00	86.80	87.69
	0.5	100.00	85.34	85.49
	1.0	90.00	74.72	73.93
	1.5	90.00	74.44	74.00

Tablica 5.4: Dobivena točnost nakon polunadziranog učenja modela s mekim sažimanjem maksimalnom vrijednošću

Veličina skupa	tip sloja sažimanja	λ_{L2}	Validacijska točnost	Točnost testiranja	Točnost testiranja ($\lambda_{L2} = 0.0$)	Validacijska točnost (literatura)
3000	tvrdi	0.01	98.78	98.53	98.17	97.50
	meki	0.01	98.76	**98.62**	98.30	
1000	tvrdi	0.01	97.48	97.12	97.15	96.00
	meki	0.01	97.62	**97.37**	96.90	
600	tvrdi	0.00	96.50	**96.40**	96.40	94.50
	meki	0.01	96.50	96.33	95.67	
100	tvrdi	0.01	89.30	**90.25**	88.62	88.25
	meki	0.01	89.04	89.66	87.41	

Tablica 5.5: Pregled dobivenih točnosti za λ_{L2} koji postiže najveću validacijsku točnost

Autori[8] su evaluirali model za $\lambda_{L2} \in \{0.0, 0.5, 1.0, 1.5, ..., 3.0\}$, dok smo mi evaluirali za $\lambda_{L2} \in \{0.0, 0.01, 0.1, 0.5, 1.0, 1.5\}$. U tablicama 5.1 i 5.2 vidljivi su dobiveni rezultati za tvrdi i meki sloj sažimanja. U njima je vidljivo da se za svaki $\lambda_{L2} \geq 0.5$ (s regularizacijom) dobivaju gori rezultati u odnosu na $\lambda_{L2} = 0.0$ (bez regularizacije). Zbog toga smo proveli eksperimente i s $\lambda_{L2} \in \{0.01, 0.1\}$, koji u većini slučajeva pokazuju bolje rezultate u odnosu na $\lambda_{L2} = 0.0$.

Za lakšu usporedbu, proveli smo optimizaciju hiperparametra λ_{L2}: za svaki N, odabrali smo hiperparametar λ_{L2} koji postiže najbolju validacijsku točnost. Pregled rezultata prikazan je u tablici 5.5. U pregledu, vidljivo je da se za $N \in \{1000, 3000\}$ dobivaju bolji rezultati s mekim slojem sažimanja, a za $N \in \{100, 600\}$ se dobivaju bolji rezultati s tvrdim slojem sažimanja.

Autori nisu evaluirali model u kontekstu polunadzirane klasifikacije uz korištenje mekog sloja sažimanja niti su proveli opisanu optimizaciju hiperparametara niti objavili točnosti testiranja. Zbog toga postoji usporedba s literaturom samo po validacijskog točnosti s modelom koji koristi tvrdi sloj sažimanja. U tablici je vidljivo da dobivamo bolju validacijsku točnost od one u literaturi.

5.3. Ovisnost rezultata o hiperparametru β

U poglavlju 5.2, arhitekturu `(64)5c-2p-(64)3c-2p-(64)3c-2p-10fc` smo polunadzirano evaluirali uz korištenje tvrdog i mekog sloja sažimanja. U eksperimentima s mekim slojem sažimanja, vrijedilo je $\beta = 1$.

U ovom eksperimentu želimo evaluirati točnost navedenog modela po parametrima β (uz parametar λ_{L2}). Zbog velike količine eksperimenata, evaluiramo samo za $N = 100$ i $N = 600$, gdje se zbog malog broja označenih slika očekuje najveća razlika u točnosti. U tablici 5.6 možemo vidjeti dobivene rezultate za $N = 600$, a u tablici 5.7 za $N = 100$. Vidljivo je da su za $N = 600$ najbolji hiperparametri $\beta = 0.1$ i $\lambda_{L2} = 0.01$, dok su za $N = 100$ najbolji hiperparametri $\beta = 0.0$ i $\lambda_{L2} = 0.01$.

Veličina skupa	β	λ_{L2}	Točnost učenja	Validacijska točnost	Točnost testiranja
600	0.0	0.0	100.00	96.90	97.07
		0.01	100.00	97.00	97.23
		0.1	100.00	96.52	96.55
	0.1	0.0	100.00	96.96	97.04
		0.01	100.00	**97.06**	**97.04**
		0.1	100.00	96.40	96.12
	1	0.0	99.83	95.54	95.67
		0.01	100.00	96.50	96.33
		0.1	100.00	95.82	95.53
	25	0.0	100.00	95.98	95.98
		0.01	100.00	96.24	95.86
		0.1	100.00	95.42	95.02
	50	0.0	100.00	95.84	95.93
		0.01	100.00	96.16	95.73
		0.1	100.00	95.04	95.18
	100	0.0	99.83	95.52	95.92
		0.01	100.00	96.12	95.84
		0.1	100.00	95.18	94.87

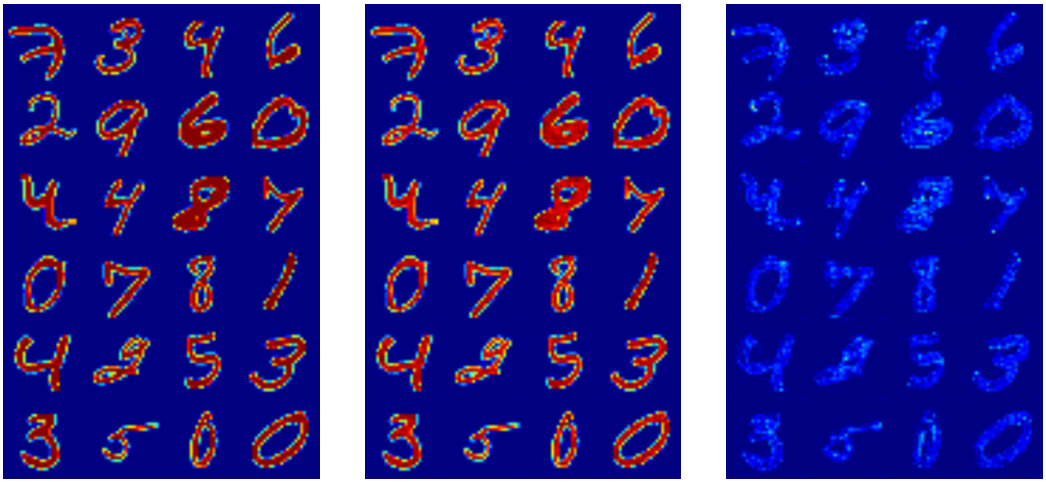
Tablica 5.6: Dobivena točnost za $N = 600$ nakon evaluacije modela s mekim sažimanjem maksimalnom vrijednošću po hiperparametru β

Veličina skupa	β	λ_{L2}	Točnost učenja	Validacijska točnost	Točnost testiranja
100	0.0	0.0	100.00	89.04	89.78
		0.01	100.00	**90.78**	**91.45**
		0.1	100.00	89.10	89.42
	0.1	0.0	100.00	89.44	90.19
		0.01	100.00	90.24	90.87
		0.1	100.00	87.20	86.44
	1	0.0	100.00	87.04	87.41
		0.01	100.00	89.04	89.66
		0.1	100.00	86.80	87.69
	25	0.0	100.00	85.58	86.07
		0.01	100.00	89.82	89.64
		0.1	100.00	88.52	88.69
	50	0.0	99.00	84.84	86.13
		0.01	100.00	88.42	88.51
		0.1	100.00	86.88	87.17
	100	0.0	99.00	85.16	86.13
		0.01	100.00	89.04	89.07
		0.1	100.00	87.72	86.47

Tablica 5.7: Dobivena točnost za $N = 100$ nakon evaluacije modela s mekim sažimanjem maksimalnom vrijednošću po hiperparametru β

5.4. Nenadzirano učenje

Ovaj eksperiment evaluira mogućnost modela da nauči rekonstrukciju bez poznavanja razreda slika iz skupa za učenje. To evaluiramo nad arhitekturom sličnom onom iz poglavlja 5.1: `(64)5c-2p-(64)3c-2p-(64)3c-2p`. Prikazani su rezultati nakon samo par epoha, za meki sloj sažimanja maksimalnom vrijednošću. Slični rezultati dobiju se i s tvrdim slojem sažimanja. Rezultati pokazuju da model s lakoćom nenadzirano nauči rekonstrukciju.



(a) (b) (c)

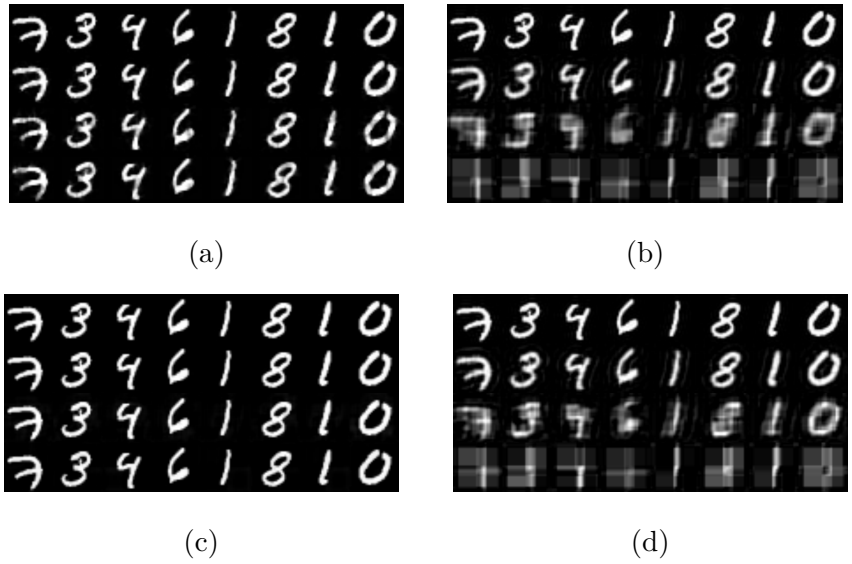
Slika 5.1: Rezultati nakon nenadziranog učenja, boja piksela je jednaka interpoliranoj vrijednosti između 0 (plava) i 1 (crvena): (a) ulaz, (b) rekonstrukcija, (c) apsolutna razlika ulaza i rekonstrukcije

5.5. Bitnost parametra *gdje*

S ovim eksperimentom želimo ilustrirati zašto je bitan *gdje*. To ilustriramo nenadziranim učenjem modela `(16)5c-(32)3c-Xp`, gdje je X parametar modela - veličina sloja sažimanja. Evaluiramo rekonstrukciju s tvrdim i mekim sažimanjem, koristeći dva načina naduzorkovanja:

1. bez *gdje* - kopiranje vrijednosti na sve pozicije u jezgri
2. s *gdje*:
   (a) tvrdo sažimanje: kopiranje vrijednosti na poziciju u jezgri koja se nalazi u *gdje*, dok se na ostale pozicije postavlja vrijednost 0
   (b) meko sažimanje: interpoliranje vrijednosti ulaza na način na koji je prikazano na slici 4.2

Rezultati su prikazani na slici 5.2. Možemo primijetiti da je po rezultatima informacija *gdje* ključna kako bi se uspješno rekonstruirala ulazna slika, i s tvrdim i s mekim sažimanjem.



(a) (b)

(c) (d)

Slika 5.2: Ilustracija rekonstrukcije, odozgora prema dolje su dobiveni rezultati s veličinom sloja sažimanja (X) 2, 4, 8, 16: (a) tvrdo sažimanje s *gdje*, (b) tvrdo sažimanje bez *gdje*, (c) meko sažimanje s *gdje* i (d) meko sažimanje bez *gdje*

5.6. Usporedba s kapsulama

Koncept kapsula[7] predstavlja autoenkoder čiji se kod sastoji od dvije informacije: prisutnost (vjerojatnost) i lokacija vizualnog entiteta (x, y). Za primjer uzmimo rukom pisane znamenke. U modelu, zadatak jedne kapsule bi mogao biti prepoznavanje gornjeg kružića u znamenkama 8 i 9, druga kapsula bi mogla prepoznavati donji kružić u znamenkama 6 i 8, treća kapsula bi mogla prepoznavati donji potez u znamenci 9 itd. Možemo zaključiti da je zadatak sustava kapsula dekompozicija ulaza na dijelove.

Autori[8] su proveli eksperiment u kojem su usporedili ponašanje *što-gdje autoenkodera* u kontekstu modela kapsula. Konkretnije, podesili su parametre modela na takav način da se na izlazu zadnjeg sloja sažimanja nalazi samo jedna vrijednost (po svakom filtru). Time simuliraju kapsule, gdje svaki filter i pripadajući *gdje* u zadnjem sloju predstavlja izlaz kapsule.

Autori su za ovaj eksperiment koristili arhitekturu s dva sloja sažimanja: `(32)5c-(32)3c-2p-(32)3c-16p`. Primijetimo da je veličina filtara prije zadnjeg sloja sažimanja 14×14. U ovom radu smo promijenili arhitekturu tako da je jezgra zadnjeg sloja sažimanja dimenzije 14×14, kako bi zadovoljili svojstvo

softmaxa:

$$\sum_{N_k} \frac{e^{\beta \cdot z(x,y)}}{\sum_{N_k} e^{\beta \cdot z(x,y)}} = 1 \tag{5.1}$$

koje neće vrijediti ako je veličina jezgre zadnjeg sloja sažimanja jednaka 16×16.

Model učimo nenadzirano na dva načina:

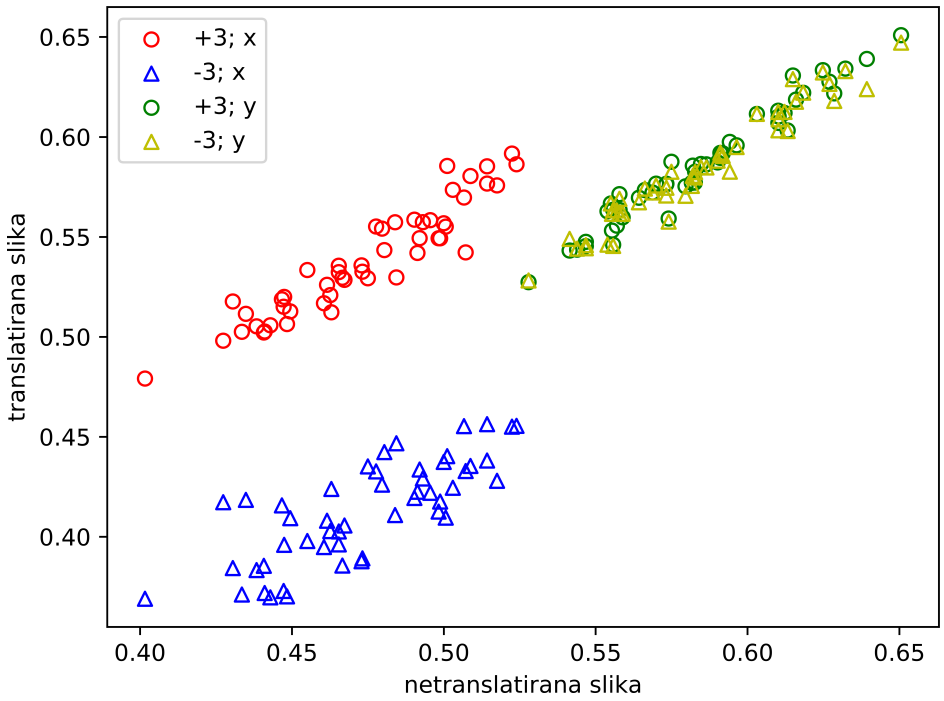
1. ulazna i očekivana izlazna slika su transformirana originalna slika
2. ulazna slika je transformirana originalna slika, a očekivana izlazna slika je originalna (netransformirana) slika

Prvi način je način na koji su autori učili[8], dok je drugi način bliži načinu učenja originalnih kapsula[7]. U nastavku su prikazani rezultati samo prvog načina, dok su za drugi način dobiveni slični rezultati. Na slici 5.3 prikazani su rezultati rekonstrukcije, koji pokazuju da je ovakav model moguće naučiti.

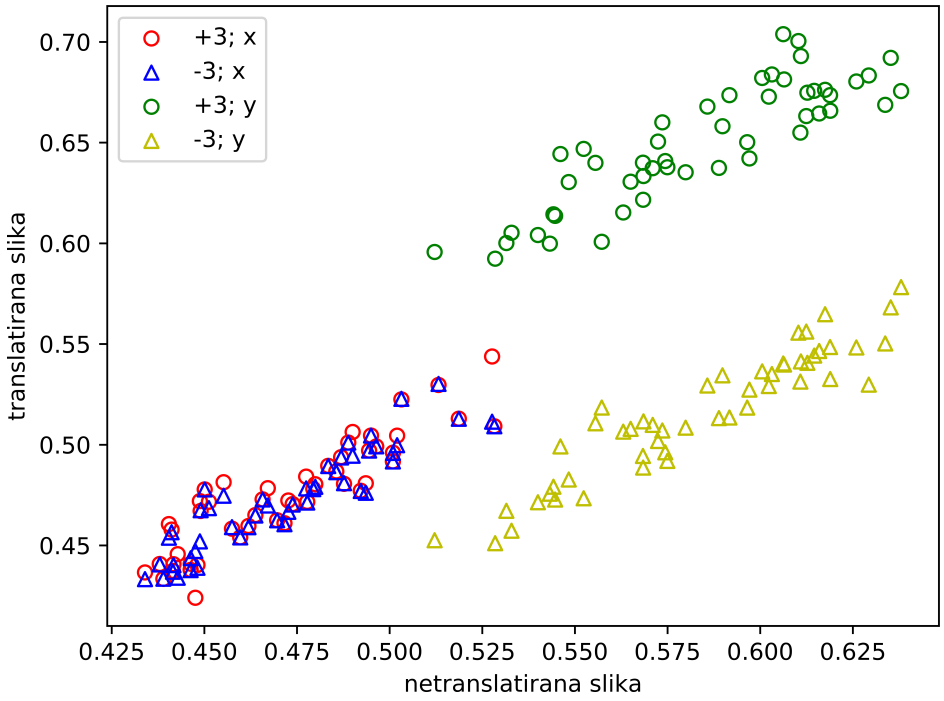
Slika 5.3: Rezultati rekonstrukcije u eksperimentu s kapsulama, po recima: (prvi) ulazna slika, (drugi) očekivani izlaz, (treći) rekonstruirani izlaz, (četvrti) ulazna slika translatirana u lijevo za 3 piksela, (peti) rekonstruirana slika za ulaz u četvrtom retku, (šesti) ulazna slika translatirana u desno za 3 piksela, (sedmi) rekonstruirana slika za ulaz u šestom retku

Na slikama 5.4 i 5.5 su prikazane *što* i *gdje* vrijednosti u prvom filtru (od 32) zadnjeg sloja enkodera, tako da uspoređujemo vrijednosti kada se na ulazu nalazi originalna slika u odnosu na translatiranu originalnu sliku. Na slici 5.4 moguće je vidjeti da model nauči pravilan odnos vrijednosti lokacije između originalne slike

i translatirane slike. Na slici 5.5 očekujemo iste vrijednosti *gdje* za +3 i -3, kao što je dobiveno u [8]. Kao što je vidljivo na slici, to se ne događa.

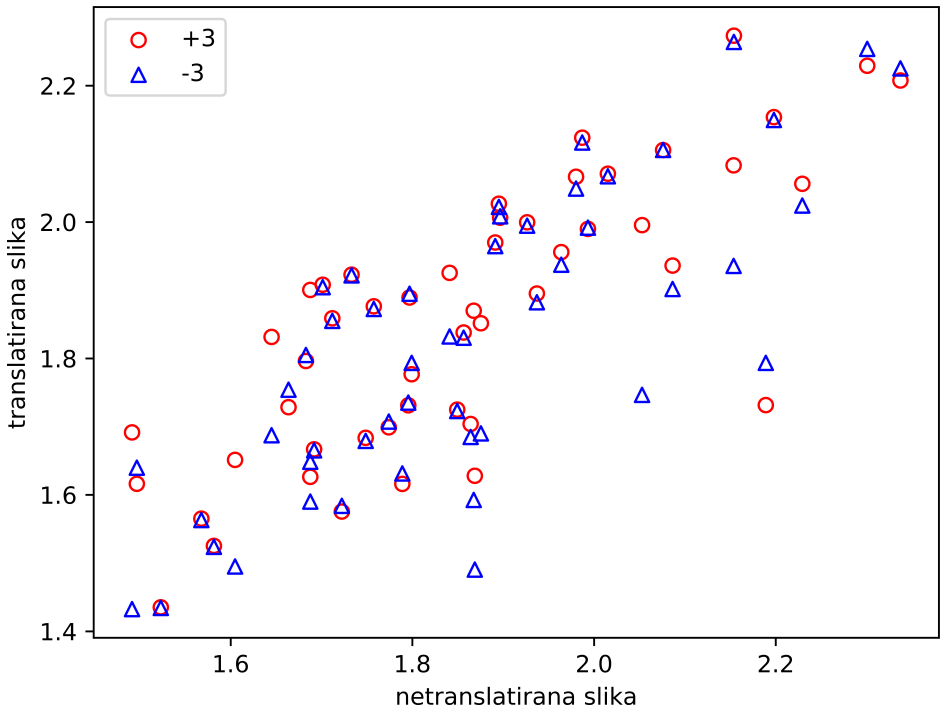


(a)

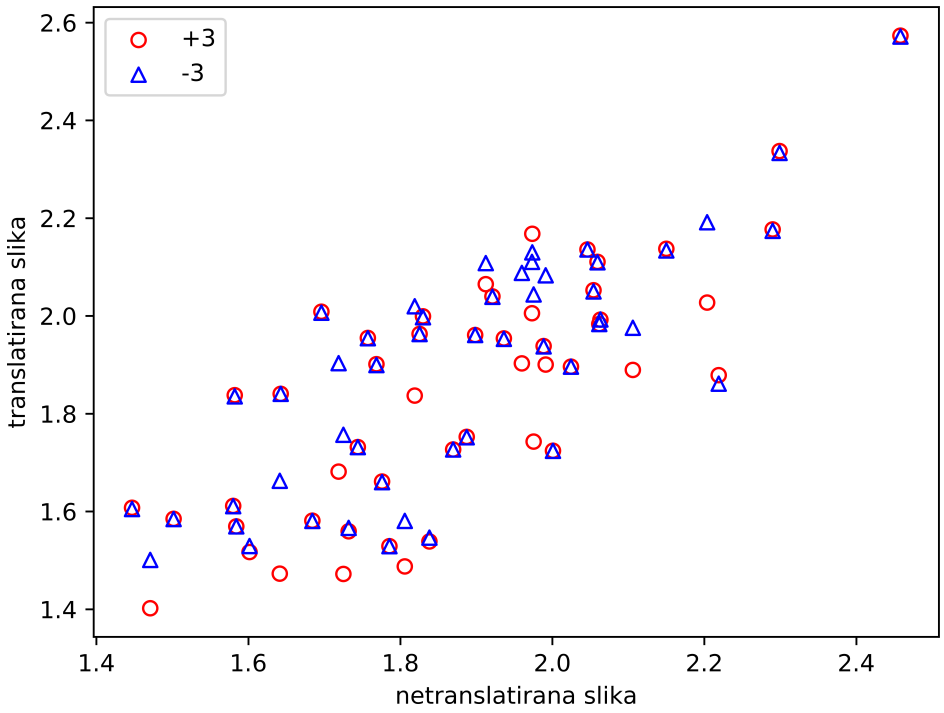


(b)

Slika 5.4: Vrijednosti *gdje* parametara, x-os su netranslatirane a y-os su translatirane slike: (a) horizontalno translatirane slike, (b) vertikalno translatirane slike



(a)



(b)

Slika 5.5: Vrijednosti *što* parametara, x-os su netranslatirane a y-os su translatirane slike: (a) horizontalno translatirane slike, (b) vertikalno translatirane slike

6. Zaključak

U ovom radu evaluiran je duboki što-gdje autoenkoder i uspoređeni su rezultati s onima u literaturi [8]. Dobiveni su rezultati bolji od onih u literaturi. U radu smo proveli i eksperimente koji se ne nalaze u literaturi. Eksperimenti u kojem evaluiramo i meki sloj sažimanja u kontekstu nadzirane ili polunadzirane klasifikacije pokazuju rezultate u kojima je tvrdi sloj i bolji i gori od mekog sloja, ovisno o veličini skupa označenih slika. Eksperiment u kojem evaluiramo model po vrijednostima hiperparametra β pokazuje da model radi bolje s malom vrijednosti parametra β. Evaluirali smo model i nad podatkovnim skupom STL-10[1], ali nismo uspjeli rezultate bliske literaturi[8].

U budućem radu je moguće istražiti mogućnost primjene što-gdje autoenkodera u modelu kapsula. Osim toga, u budućem radu bi mogli istražiti zašto trenutna implementacija ne radi nad podatkovnim skupom STL-10. Također, svi eksperimenti su provedeni samo jednom - idealno bi htjeli pokrenuti eksperimente više puta, te napisati srednju vrijednost točnosti.

Literatura

[1] Adam Coates, Andrew Ng, i Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. svezak 15 od *Proceedings of Machine Learning Research*, stranice 215–223, Fort Lauderdale, FL, USA, 11–13 Apr 2011. PMLR. URL `http://proceedings.mlr.press/v15/coates11a.html`.

[2] Matija Folnović. *Duboke neuronske arhitekture za lokalizaciju objekata*, 2015. URL `http://www.zemris.fer.hr/~ssegvic/project/pubs/folnovic15bs.pdf`.

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, i Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. URL `http://arxiv.org/abs/1512.03385`.

[4] G. E. Hinton i R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. Srpanj 2006. URL `https://www.cs.toronto.edu/~hinton/science.pdf`.

[5] Alex Krizhevsky, Ilya Sutskever, i Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. U *Advances in Neural Information Processing Systems 25*.

[6] H. W. Lin, M. Tegmark, i D. Rolnick. Why does deep and cheap learning work so well? *ArXiv e-prints*, Kolovoz 2016.

[7] Sida Wang. *Learning to Extract Parameterized Features by Predicting Transformed Images*, 2011. URL `http://nlp.stanford.edu/~sidaw/home/_media/papers:thesis.pdf`.

[8] J. Zhao, M. Mathieu, R. Goroshin, i Y. LeCun. Stacked What-Where Auto-encoders. *ArXiv e-prints*, Lipanj 2015.

Duboki generativni modeli temeljeni na prostornom rasporedu dijelova slike

Sažetak

Iako moderni duboki modeli raspoznaju slike bolje od ljudi, problem klasifikacije slika još ne možemo smatrati riješenim. Poznato je da su duboki modeli skloni precijeniti pouzdanost predikcije, što može dovesti do vrlo neugodnih pogrešaka. Taj problem može se umanjiti primjenom reprezentacija koje sadrže prostorni raspored dijelova slike. Konkretno, prenaučenost se može umanjiti korištenjem regularizatora koji prisiljava model da nauči rekonstruirati ulaznu sliku. Ova ideja je proučavana kroz objavljen model koji se zove Duboki Što-Gdje Autoenkoder (engl. *Stacked What-Where Autoencoder*). Istražujemo prednosti i mane ovog pristupa u kontekstu polunadzirane klasifikacije jednostavnih slika i prikazujemo eksperimentalne rezultate na skupu MNIST.

Ključne riječi: duboki generativni modeli, učenje reprezentacije, polunadzirano učenje

Deep generative models based on spatial layout of image parts

Abstract

Although modern deep models classify images better than humans, image classification cannot be regarded as a solved problem yet. It's known that deep models tend to overestimate prediction accuracy, which can lead to unpleasent errors. This problem can be reduced by exploiting representations containing spatial layout of image parts. In particular, overfitting can be reduced by supplying a regularizer which forces the model to be able to reconstruct the input image. This idea has been studied in the previously published model known as Stacked What-Where Autoencoder. We research advantages and disadvantages of this approach in regards to semi-supervised classification of simple images and present experimental results on the MNIST dataset.

Keywords: deep generative models, representation learning, semi-supervised learning