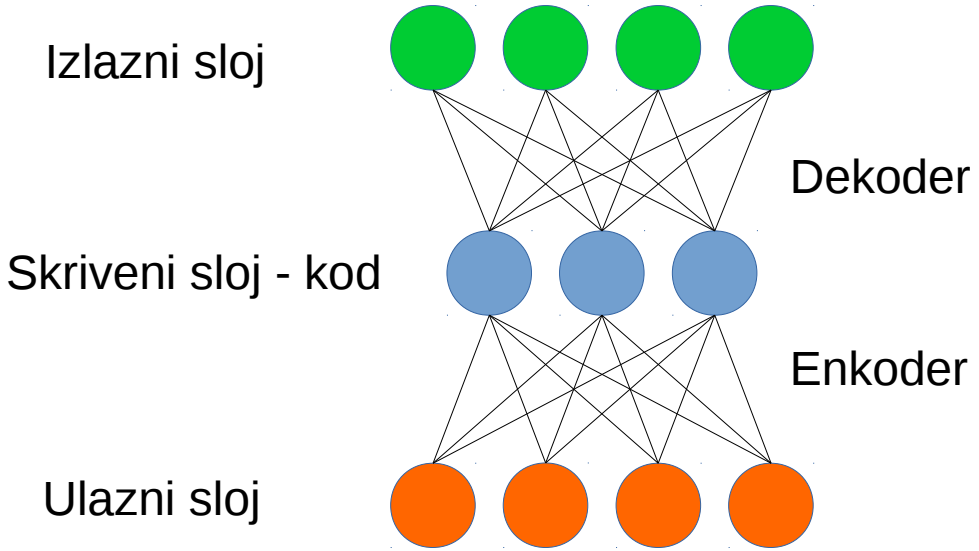
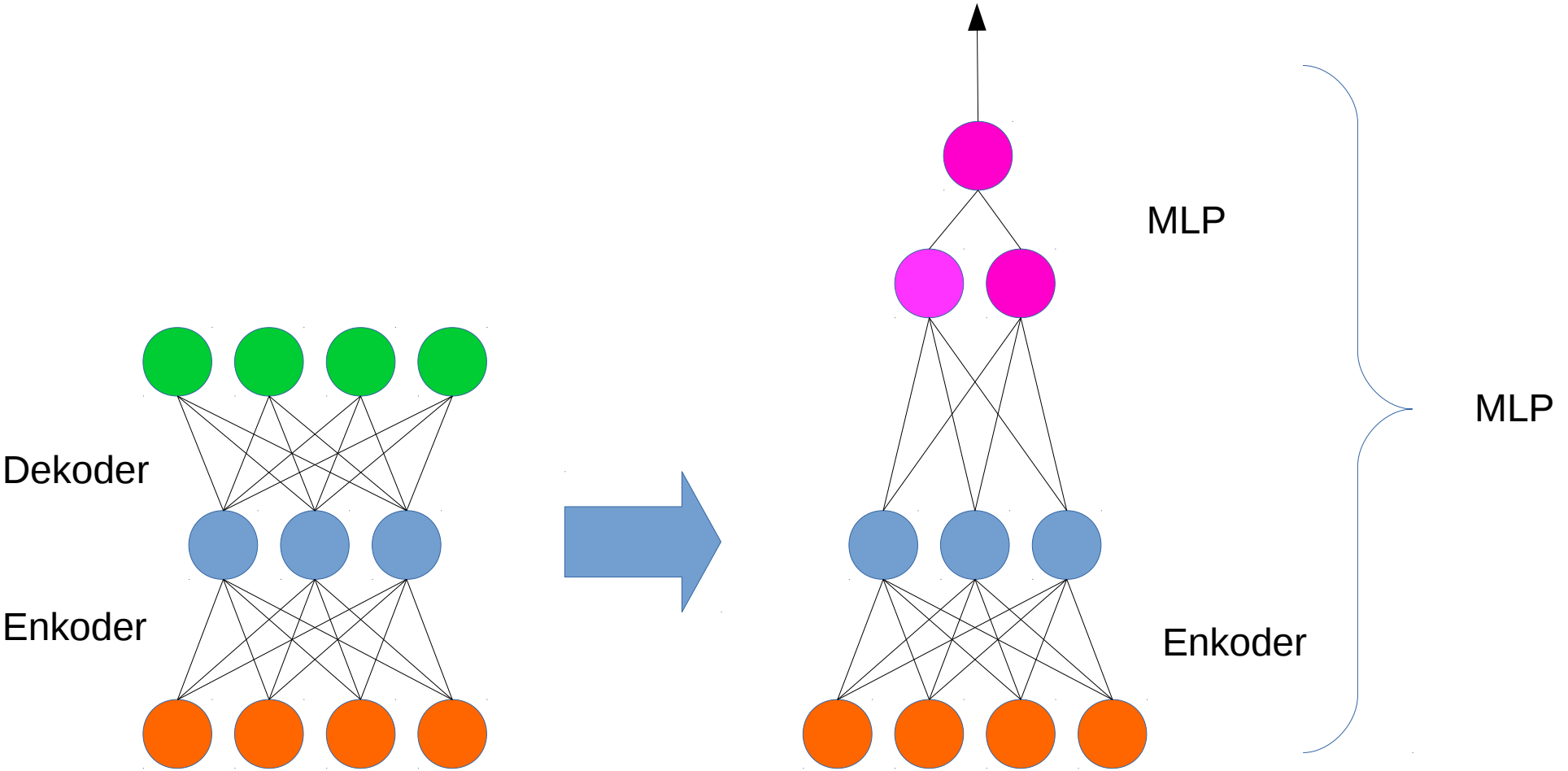
Generativni modeli
autoenkoderi

Autoenkoderi - ukratko

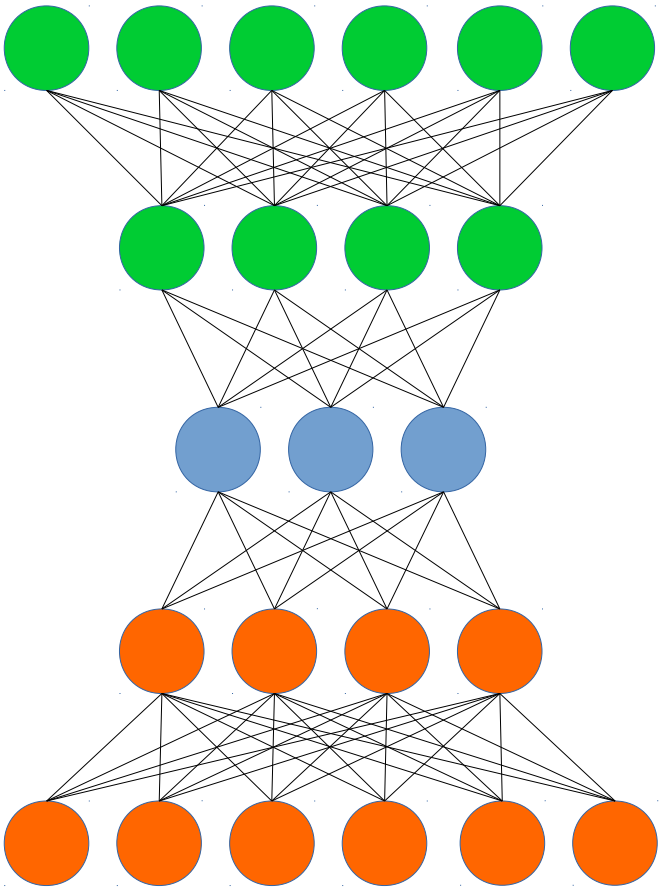
- Cilj je naučiti mrežu da što vjernije kopira ulazni vektor
- Sama funkcija kopiranja ne donosi ništa novo pa nas ni ne zanima previše
  - Cilj je izbjeći direktno kopiranje
- Zanima nas što se dešava u centralnom skrivenom sloju



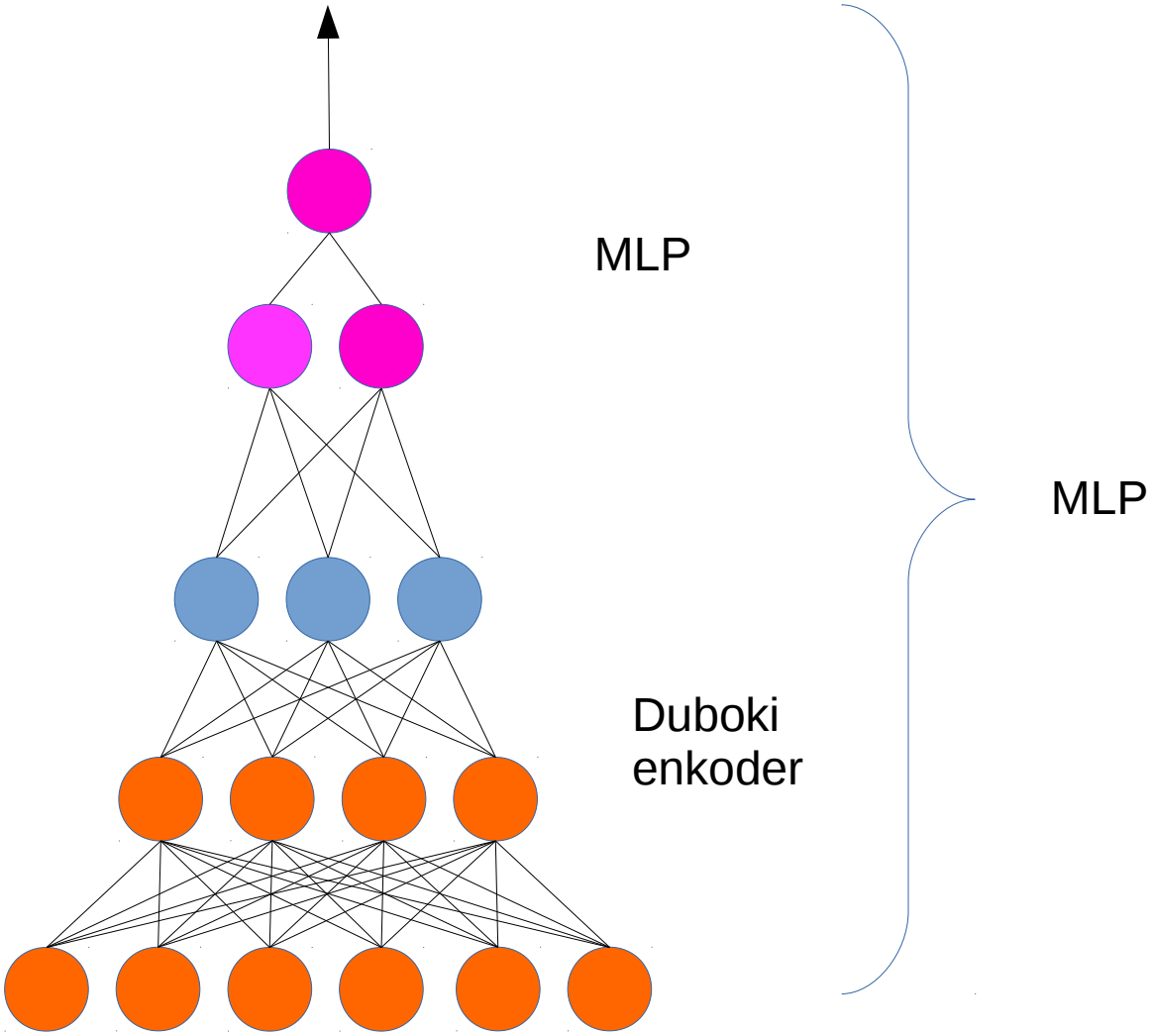
Autoenkoderi - primjena



Duboki autoenkoder

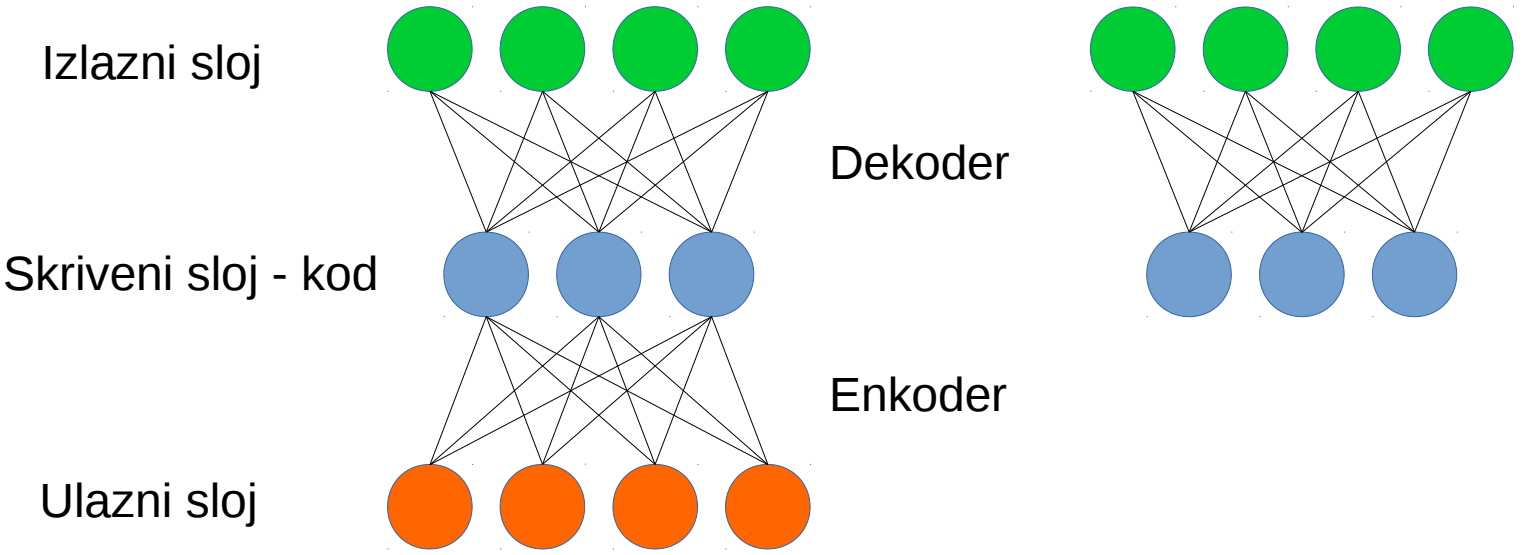


Duboki autoenkoder - primjena



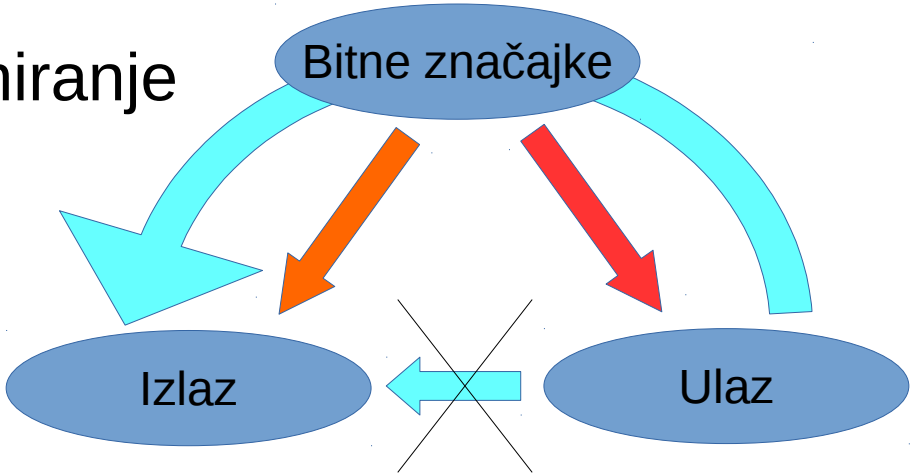
Autoenkoderi

- Mogu biti deterministički ili stohastički
- Svrha (prividna) im je generiranje ulaznog vektora iz bitnih značajki ulaznog vektora
  - Generativni model



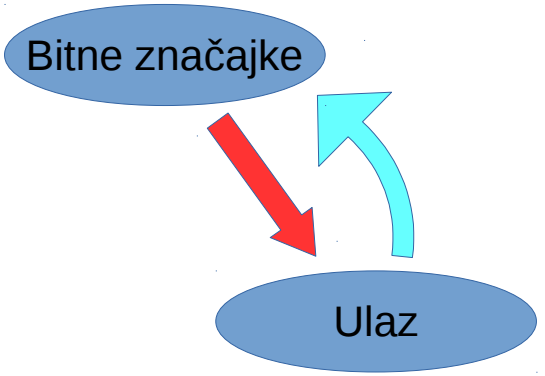
Autoenkoderi

- Izlaz autoenkodera nije osobito interesantan
  - Izlaz je što vjernija kopija ulaza
- Interesantan je skriveni sloj koji izvlači bitne značajke ulaznih vektora
  - Za primjene gdje takve značajke ne znamo ručno odrediti
  - Dobivene značajke ne moraju biti jednostavne za interpretaciju
  - Prilagođene su skupu za treniranje



Bitne značajke

- Bitne u kontekstu rekonstrukcije uzoraka iz skupa za treniranje
- Bitne značajke moraju dobro opisati sve bitne varijacije u skupu za treniranje koje su nužne za uspješnu rekonstrukciju
  - Bitne razlike između uzoraka za treniranje uzrokuju promjene u bitnim značajkama, a ne u manje bitnim značajkama

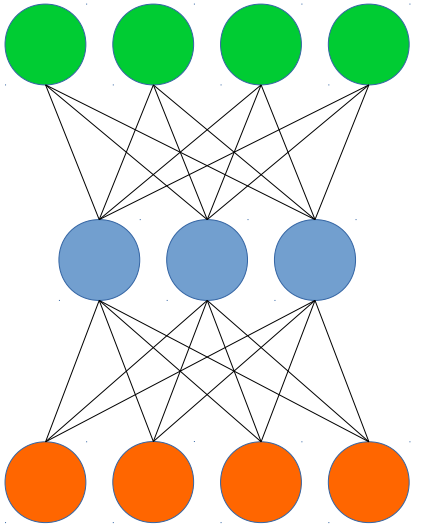


Kopiranje ulaz izlaz

- Minimalna greška na skupu za učenje postiže se direktnim kopiranjem ulaz-izlaz
- Promatrajući grešku rekonstrukcije stanje je savršeno
- Pretreniranja nema jer je greška blizu nule i na bilo kojem skupu za testirajne
- Ali zapravo situacija je gora od pretreniranja!!
  - Nije nužno da točnost bude 100%, ali može se postići
- Rješenje je: regularizacija

Regularizacija

- Jedan oblik regularizacije je usko grlo skrivenog sloja
  - Manje neurona nego ulazni sloj



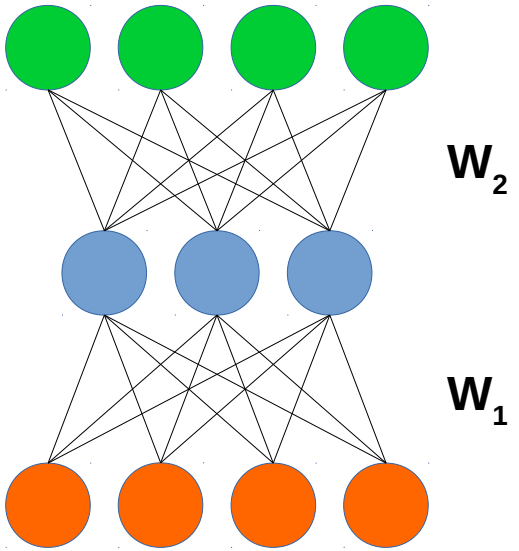
- Postoje i drugi mehanizmi koji se mogu uključiti u funkcioniranje mreže
  - Sparsitiy, robusnost na šum, nedostajući ulazi, ograničenja na derivacije
  - Veličina sloja je tada manje bitna

Tipovi autenkodera

- Klasični autoenkoder
- Denoising autoenkoder
- Contractive autoenkoder
- Sparse Autoenkoder
- Varijacijski autoenkoder

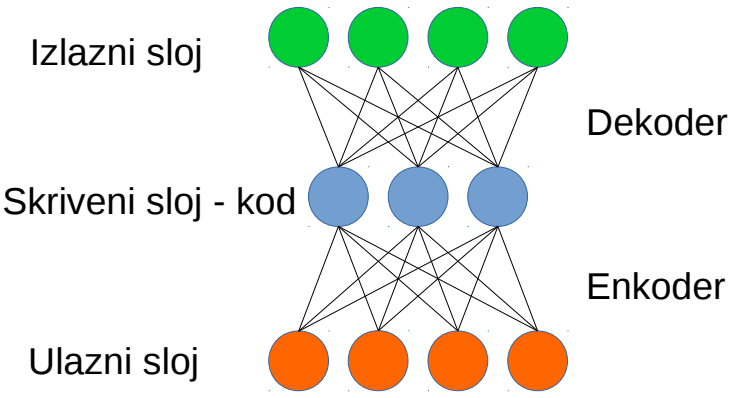
Klasični autoenkoder

- Linearni neuroni
  - Ekvivalent PCA
- Nelinearni neuroni
  - Generalizirana PCA
- Usko grlo centralnog skrivenog sloja sprječava kopiranje ulaz – izlaz



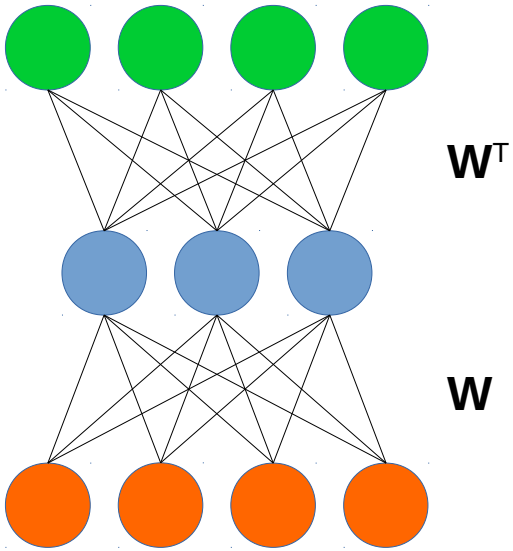
Klasični autoenkoder

- Usko grlo skrivenog sloja prisiljava mrežu na:
  - učenje bitnih značajki skupa za treniranje
  - zanemarivanje nebitnih značajki
  - redukcija dimenzionalnosti
  - izbjegavanje direktnog kopiranja
- Da bi to funkcioniralo mora postojati neka strukturiranost u ulaznim podacima – korelacija nekih elemenata
  - Princip ne bi dobro funkcionirao kada bi ulazni podaci bili slučajni - šum



Vezanje težina

- Vezanje težina ulaznog i izlaznog sloja
  - Sprječava "linearizaciju" nelinearnih neurona
    - Male težine u enkoderu kod tanh() aktivacije
    - Sličan efekt postiže se normalizacijom težina
- Sličan upotreba i kod Contractive autoenkodera
- Gradijent za pojedinu težinu tada je suma gradijenata iz oba sloja



Denoising autoenkoder

- Dodaje se šum u ulazne podatke
- Autoenkoder mora naučiti uklanjati šum uz enkodiranje
  - Dodavanje šuma je korisno i u drugim primjenama
    - Može se dodavati ulaznim podacima, težinama,...
  - Slično kao MLP kojeg se trenira za uklanjanje šuma
  - Efektivno povećanje skupa za treniranje
- Zašto kvariti sve šumom?

Denoising autoenkoder

- Efikasno uklanjanje šuma može se postići tako da AE nauči bitne karakteristike ulaznih podataka
  - Npr. distribuciju ulaznih podataka p(**x**)
- Kakav šum koristiti?
  - Slučajno postavljanje ulaznih elemenata na nulu
  - Salt & pepper šum
  - Gaussov šum
  - Svaki model šuma donosi nove hiperparametre!

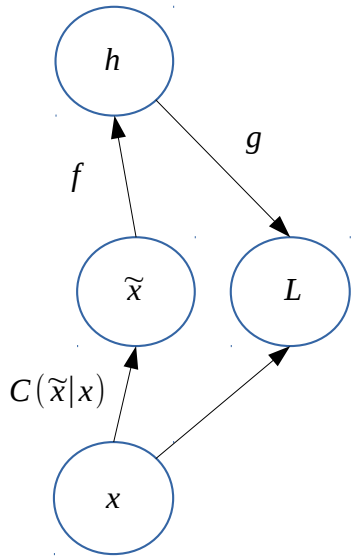
Denoising autoenkoder

- Treniranje se može provoditi na način da se maksimizira vjerojatnost točne rekonstrukcije iz zašumljenog uzorka

$$p_{reconstr}(x|\tilde{x}) = p_{decoder}(x|h=f(\tilde{x}))$$

  - Provodi se stohastic gradient descent
- Funkcija cijene može biti

$$L = -\log p_{decoder}(x|h=f(\tilde{x}))$$



Denoising autoenkoder

- Treniranje
  1) Odabir uzorka x iz skupa za treniranje
  2) Kreiranje zašumljenih verzija $\tilde{x}$
     - uzorkovanje
  3) Procjena vjerojatnosti rekonstrukcije na temelju x i $\tilde{x}$
  4) Minimizacija funkcije cijene L

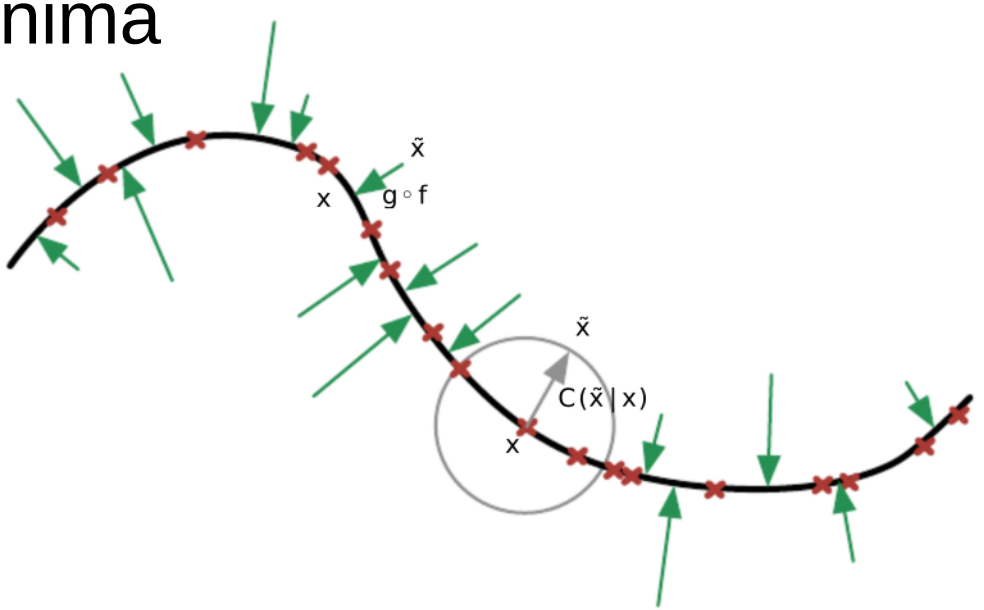
Denoising autoenkoder

- Alternativa maksimizaciji vjerojatnosti je minimizacija srednje kvadratne pogreške

$$\|g(f(\tilde{x}))-x\|^2$$

- Naučiti će polje vektora koji ispravljaju greške u zašumljenim uzorcima
  - Ne pretjerano zašumljenima
- Uči se polje gradijenta

$$g(f(\tilde{x}))-x$$



Denoising autoenkoder

- Možemo ga gledati i kao MLP koji se trenira za uklanjanje šuma
  - DAE uz to uči i pronaći bitne značajke
- Jednostavni su za implementaciju
  - Samo se dodaje šum
- Treniranje koje podrazumijeva uzorkovanje, traje duže

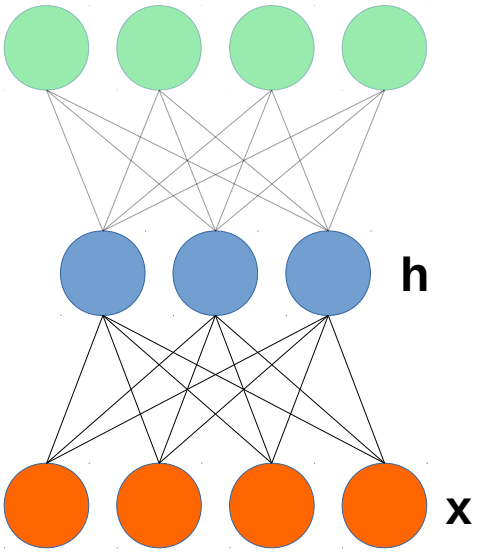
Contractive autoenkoder

- Drugačija regularizacija – rezultat je sličan kao kod DAE
- Cilj je da se vrijednosti skrivenog sloja ne mijenjaju ako se ulaz (malo) promijeni
  - Derivacije od *h=f(x)* s obzirom na *x* trebaju biti nula
- Regularizacijski član funkcije cijene koji bi idealno trebao biti 0

$$L(\boldsymbol{x}, g(f(\boldsymbol{x}))) + \Omega(\boldsymbol{h})$$

$$\Omega(\boldsymbol{h}) = \lambda \left\| \frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}} \right\|_F^2 = \lambda \sum_j \sum_k \left(\frac{\partial f(\boldsymbol{x})_j}{\partial x_k} \right)^2$$

  - Suma kvadrata Jakobijeve matrice enkoderske funkcije *f*
- Ideja je slična kao i kod DAE za umjereni Gaussov šum
  - Ideja da se izlaz ne bi puno trebao mijenjati za male promjene ulaza

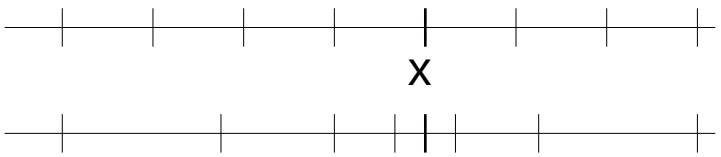


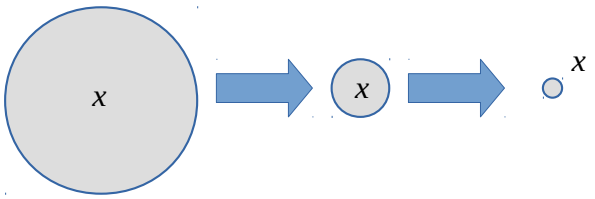
Contractive autoenkoder

- Regularizacijski član se primjenjuje na enkoderski dio f(x), a ne na dekoderski dio g(f(x)) – slično kao DAE
- Cilj je pronaći značajke koje objašnjavaju značajne (bitne) varijacije u skupu za treniranje
- Ostale varijacije u ulaznim uzorcima ne bi trebale utjecati na skriveni sloj

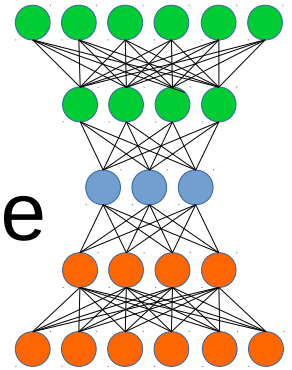
Contractive autoenkoder

- Contracting input space to output space
  - Sažimanje prostora je samo lokalno oko uzoraka za učenje
  - Susjedstvo oko ulaznog uzorka se sažima u manje susjedstvo oko odgovarajućeg izlaza
  - U prostoru između ulaznih uzoraka može se desiti suprotno – širenje prostora
- Regularizacijski član uz minimizaciju greške postiže
  - Veći dio značajki (neurona skrivenog sloja *h*) slabo ovisi o ulazu
  - Manji dio koji dobro opisuje varijacije u ulaznim uzorcima mora ovisiti o ulazu – na to će ih prisiliti minimiziranje pogreške rekonstrukcije





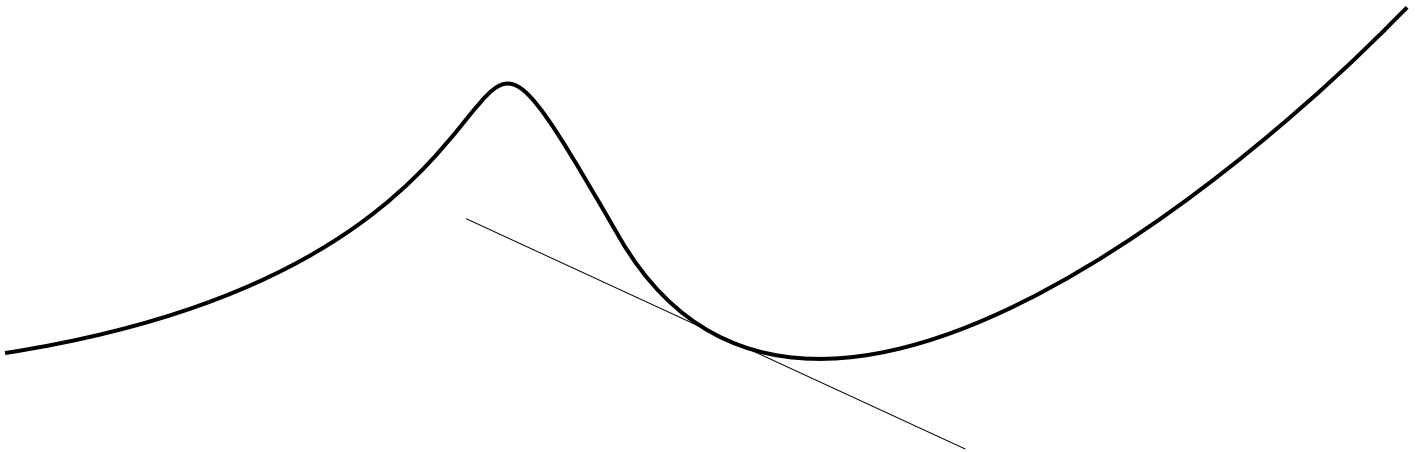
Contractive autoenkoder



- Kod dubokih CAE radi se pohlepno treniranje pojedinih slojeva kao plitkih autoenkodera
  - Konačni duboki autoenkoder će isto sažimati prostor
- Konfiguracija u kojoj su težine enkodera male, a dekodera velike, omogućuje dobru rekonstrukciju, ali regularizacijski član čini neefikasnim
  - Male vrijednosti f(x) smanjuju i derivacije
  - Vezanje težina enkodera i dekodera sprječava takve probleme

Contractive autoenkoder

- Estimiraju tangencijalne ravnine višedimenzionalne plohe na kojoj leže uzorci za treniranje



Contractive autoenkoder

- Prednosti
  - Deterministički gradijent – nema uzorkovanja kao kod DAE
- Nedostaci
  - Prilikom treniranja treba određivati gradijente za svaki neuron skrivenog sloja

Sparse autoenkoderi

- Efektivno se smanjuje kapacitet skrivenog sloja – dobro za generalizaciju
  - Broj neurona u skrivenom sloju može biti veći nego u ulaznom
- Uvjet rijetkosti se nameće u skrivenom sloju
  - Teži se tome da samo manji broj neurona skrivenog sloja bude aktivan za bilo kakav ulaz
  - Vrijednosti aktivacija za aktivno i neaktivno stanje ovise o aktivacijskim funkcijama
    - 1 i 0 za sigmoidnu aktivacijsku funkciju
    - 1 i -1 za tanh

Sparse autoenkoderi – varijanta 1

- Rijetkost se nameće kroz uvjet za očekivanu vrijednost aktivacija svih skrivenih neurona

$$E(h_i) = \rho$$

- Parametar rijetkosti ρ postavlja se na vrijednost malo veću od aktivacije koja odgovara neaktivnom stanju
  - Npr. -0.9 ako je neaktivno stanje -1

Sparse autoenkoderi – varijanta 1

- Treniranje se obavlja u dva koraka
  - Prvo se koristi gradient descent (backpropagation)
  - Zatim se mreža modificira da se postigne rijetkost
- Estimacija očekivane aktivacije *h* svih skrivenih neurona ρ se osvježava u svakoj iteraciji

$$\hat{\rho}_i(n+1) = \lambda\,\hat{\rho}_i(n) + (1-\lambda)\,h_i(n)$$

- Parametar λ obično se postavlja na vrijednost neznatno manju od 1
- Kako bi se postiglo željena očekivanja, može se iskoristit koeficijente pomaka b_i svakog skrivenog neurona

$$h_i = f\left(\sum_{j=1}^{n} w_{ij}\,x_j + b_i\right)$$

Sparse autoenkoderi – varijanta 1

- U slučaju da se vrijednost ρ_i udaljili od zadane vrijednosti ρ, želimo je ponovo približiti ρ
  - Ako je odstupanje ρ_i od ρ veće, želimo jaču korekciju
- Za odabir nove vrijednosti koeficijenta pomaka b_i koristimo izraz

$$b_i(n+1) = b_i(n) - \beta(\hat{\rho}_i(n) - \rho)$$

- β je novi hiperparametar

Sparse autoenkoderi – varijanta 2

- Alternativni način za rijetkost u skrivenom sloju estimira očekivanje kao srednju vrijednost aktivacija za sve ulazne uzorke

$$\hat{\rho}_j = \frac{1}{m} \sum_{i=1}^{m} a_j(x_i)$$

- Dodatna komponeneta funkcije cijene koja se koristi u backpropagation algoritmu može biti:

$$\sum_{j=1}^{n} \rho \log \frac{\rho}{\hat{\rho}_j} + (1-\rho) \log \frac{1-\rho}{1-\hat{\rho}_j} = \sum_{j=1}^{n} KL(\rho \| \hat{\rho}_j)$$

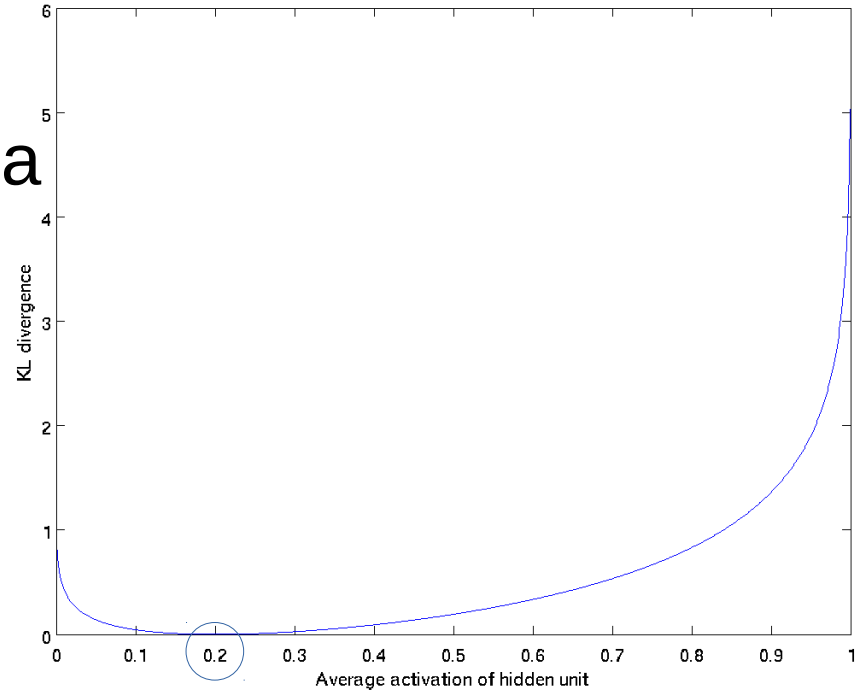
  - Aktivacije trebaju biti u rasponu [0 1]
  - Cijena je jednaka nuli kada estimacija poprimi željenu vrijednost ρ

Sparse autoenkoderi

- Backpropagation algoritam se modificira na način da se izračun lokalnog gradijenta za skriveni sloj proširi za dodatni član

$$\delta_i = \left(\left(\sum_j w_{ji} \delta_j \right) + \beta \left(-\frac{\rho}{\hat{\rho}_j} + \frac{1-\rho}{1-\hat{\rho}_j} \right) \right) f'(h_i)$$

- Elegantno rješenje, ali...
- Nedostatak je što za svaki korak backpropagatioin algoritma treba biti poznata prosjek aktivacije za sve uzorke iz skupa za treniranje



Duboki autoenkoderi

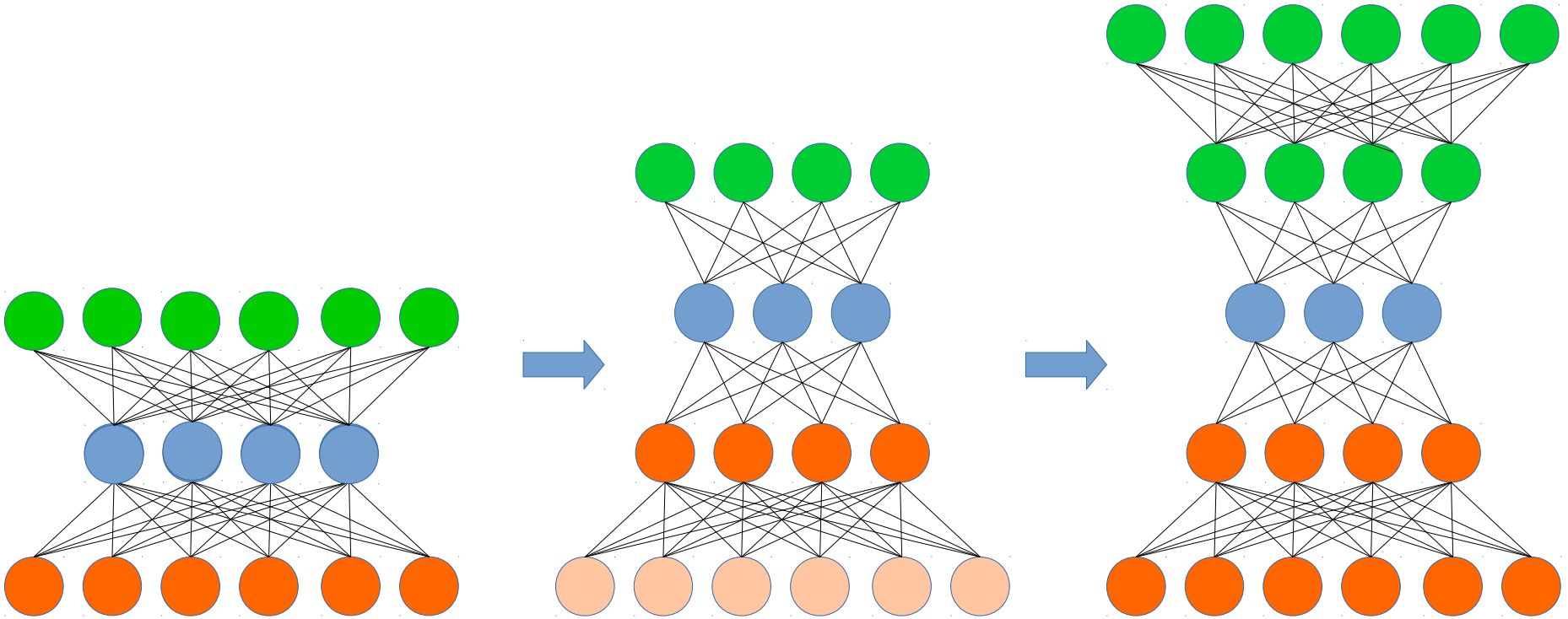
- Postepeno smanjivanje dimenzionalnosti
- Dubina donosi neke prednosti
  - Kompleksnija mreža može rješavati kompleksnije probleme
  - Eksperimentalno potvrđeno
- Neuroni moraju biti nelinearni
  - Linearne aktivacijske funkcije nemaju smisla
- Treniranje duboke mreže donosi probleme
  - Koristi se backpropagation – wanishing gradients
  - Problem inicijalizacije
- Jedno rješenje je "pohlepno" učenje bez nadzora, sloj po sloj

Duboki autoenkoderi

- Autoenkoder s više skrivenih slojeva
- Enkoder postepeno smanjuje broj elemenata u svojim slojevima
- Zadnji sloj enkodera predstavlja ključne značajke
  - Arhitektura omogućuje hijerarhijsko grupiraje značajki
- Zadnji sloj enkodera se može koristiti za klasifikaciju
- Za vizualizaciju je praktično svesti broj elemenata u centralnom sloju na 1, 2 ili 3

Duboki autoenkoderi

- Pohlepno predtreniranje

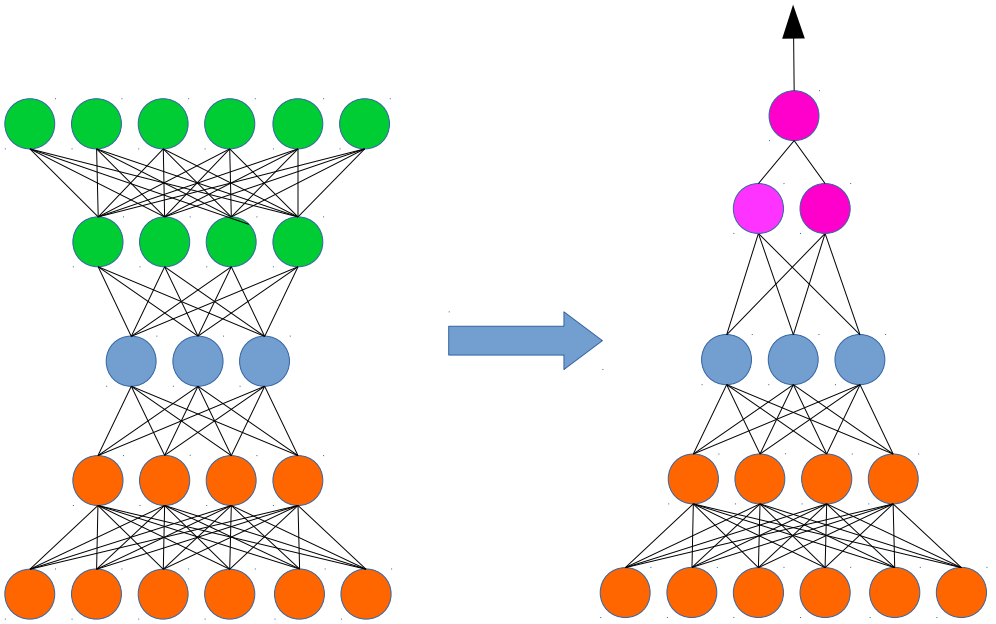


Duboki autoenkoderi

- Nakon pohlepnog predtreniranja slijedi fine-tuning
- Cilj je iskoristiti naučene težine skrivenih slojeva kao dobru inicijalizaciju za bilo koju svrhu
  - Pohlepno predtreniranje je trebalo omogućiti ekstrakciju bitnih značajki koje bi trebale biti univerzalno korisne za dani skup podataka

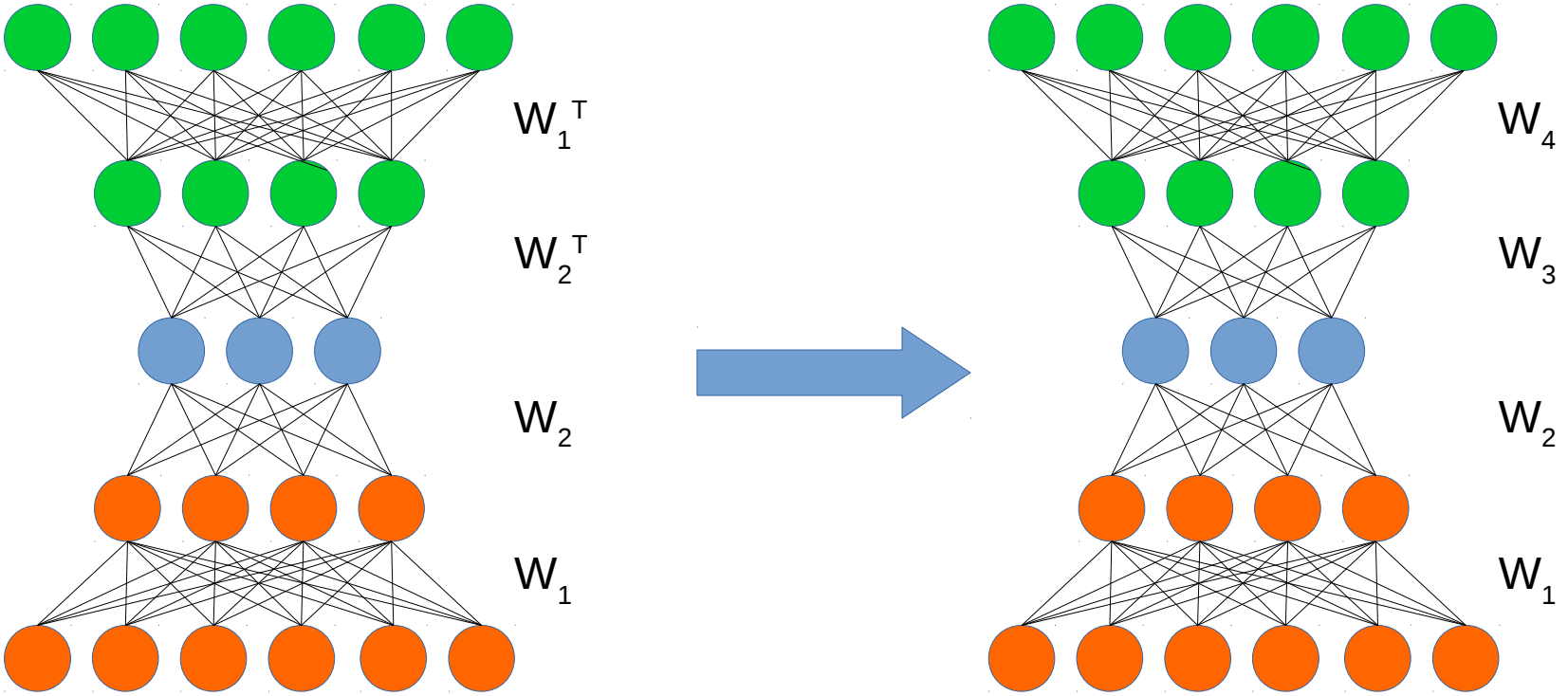
Fino podešavanje

- Fino podešavanje provodi se backpropagation algoritmom na konačnoj dubokoj mreži
- Ako je cilj klasifikacija dodaje se završni klasifikacjski sloj na centralni skriveni sloj (slučajno inicijaliziran)
- Praktično kada je označen samo manji dio skupa za treniranje



Fino podešavanje

- Ako je cilj generiranje uzoraka, podešava se čitavi duboki autoenkoder (bez vezanih težina enkoderskog i dekoderskog sloja)
  - Cilj je bolja rekonstrukcija

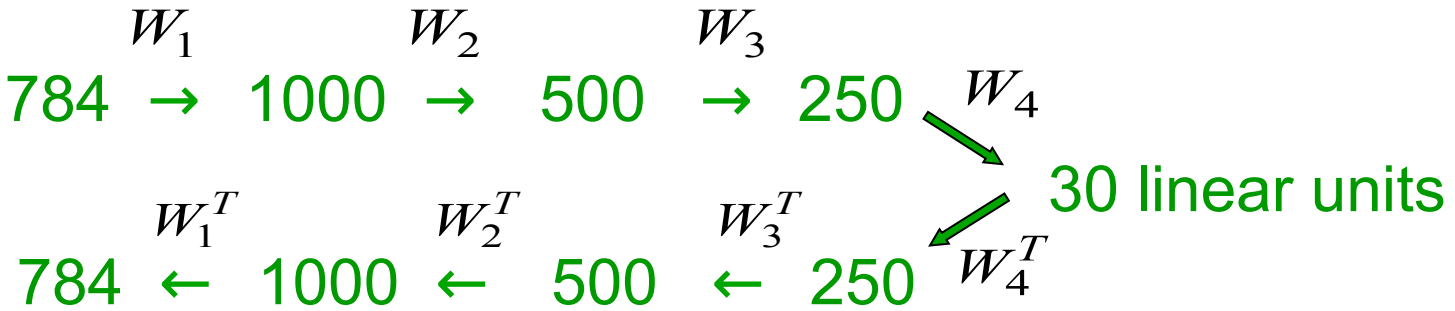


Duboki autoenkoderi

- Pohlepno predtreniranje rješava problem dubokih mreža kroz kvalitetnu inicijalizaciju
  - Mogu biti i RBM autoenkoderi
- Fino podešavanje obično ne dovodi do značajnih promjena težina
  - Uz gornju pretpostavku to nije niti poželjno

Duboki autoenkoderi

- Prvi uspješni duboki autoenkoder (2006)



Generiranje uzoraka

- Dodatna primjena je u dohvaćanju sličnih uzoraka
  - Slični uzorci trebaju biti blizu u prostoru niže dimenzionalnosti
- Autoenkoderi mogu naučiti distribuciju ulaznih podataka
- Mogu se koristiti za uzorkovanje iz te distribucije
- MC algoritam uzorkovanja pomoću DAE

  1)Uzmi uzorak $\tilde{x}$ dodavanjem šuma na prethodni uzorak *x*

  2)Enkodiraj zašumljeni $\tilde{x}$ u *h*

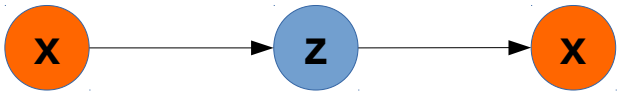
  3)Dekodiraj *h* u slijedeći x

Pravi generativni modeli?

- Varijanta AE koje smo razmotrili do sada nisu pravi generativni modeli
  - Osim DAE
- Enkoder generira vektor skrivenog sloja za zadani ulazni uzorak
- Dekoder će za vektor skrivenog sloja generirati odgovarajući uzorak na izlazu
  - On će biti sličan ulaznom uzorku
  - Bit će uvijek isti...
- Ne može se generirati nove tipične uzorke

Varijacijski autoenkoderi

- Tražimo skrivene varijable
- Odgovarajuća kombinacija skrivenih varijabli (možda uz manje varijacije) generira izlaz sličan ulazu koji je uzrokovao skrivene varijable
- Ako je skrivena varijabla slučajna, želimo generirati izlazne vektore koji odgovaraju skupu za treniranje

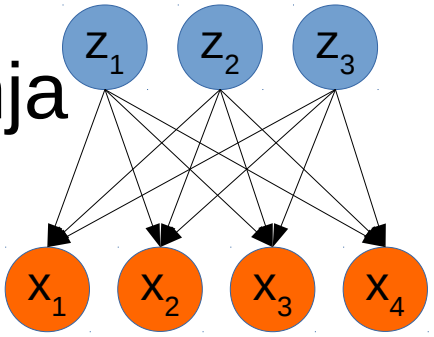


Varijacijski autoenkoderi

- Ideja je maksimizirati izlaznu vjerojatnost svakog uzorka iz skupa za treniranje p(**x**)
  - Model će tada moći generirati nove (izglednije) uzorke slične uzorcima za učenje
  - Tražimo i skrivene varijable z koje omogućuju to

$$p(\boldsymbol{x}) = \int p(\boldsymbol{x}|\boldsymbol{z};\theta)\, p(\boldsymbol{z})\, d\boldsymbol{z}$$

- Nepoznate distribucije ćemo aproksimirati normalnim distribucijama
- Želimo postići cilj sa što manje uzorkovanja

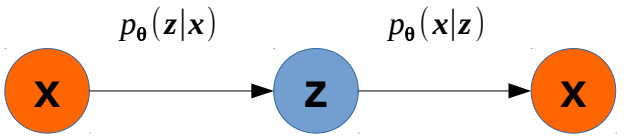


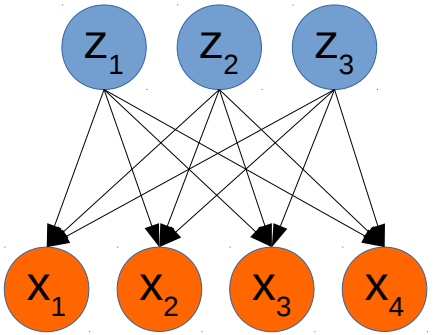
Varijacijski autoenkoderi

- Nema direktne veze s klasičnim autoenkoderima
- Ideja je maksimizacija vjerojatnosti svih ulaznih vektora $\mathbf{x}^{(i)}$
  - Rezultirajući model liči na klasične autoenkodere
  - Nema tuning hiperparametra za regularizacijski član
- Ekvivalent je maksimizacija logaritma vjerojatnosti

$$\log p_\theta(x^{(1)},\ldots,x^{(N)}) = \sum_{i=1}^{N} \log p_\theta(x^{(i)})$$

  - Θ su parametri distribucije *p*



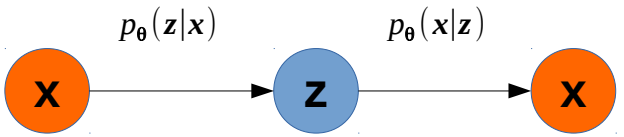


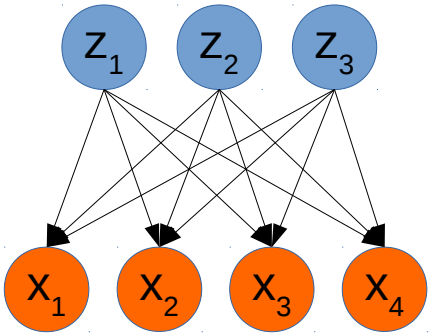
VAE skrivene varijable

- Pitanje je kako odrediti skrivene varijable **z**
  - Ne želimo ih ručno određivati
- Kod VAE zadajemo njihvu distribuciju

$$p(z) = N(0,1) \qquad p(\boldsymbol{x}) = \int p(\boldsymbol{x}|\boldsymbol{z};\theta)\, p(\boldsymbol{z})\, \boldsymbol{dz}$$

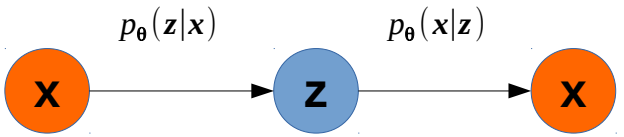
  - Ukoliko je veza **z** i **x** kompleksna (duboka) onda ćemo moći proizvesti bilo kakvu distribuciju p(**x**)
  - Tražimo tu vez / funkciju

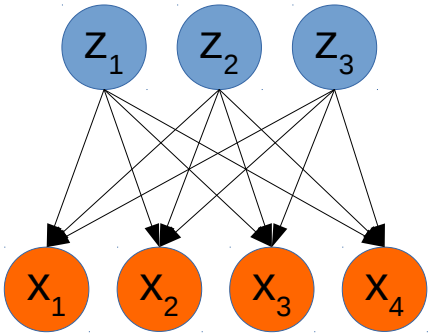




VAE dekoder

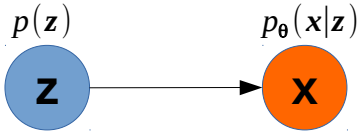
- Vezu između **z** i **x** ćemo realizirati kao MLP
  - Slojevi dekodera transformiraju **z** u "stvarno korisne" skrivene varijable
  - Dekoderski dio MLP-a transformira "stvarno korisne" skrivene varijable u željene izlaze





VAE dekoder

- Uzmimo samo dekoderski dio

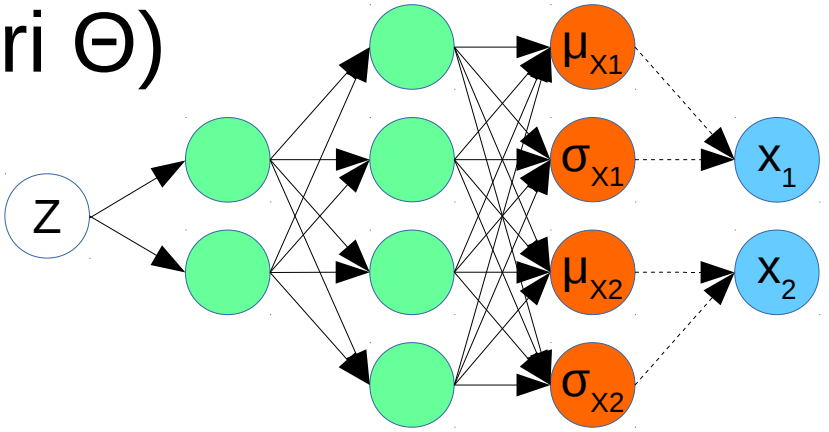


- Pretpostavimo da je za svaki izlazni element *x*

$$p_\theta(x|z) = N(\mu_x(z), \sigma_x(z))$$

  - Dijagonalna matrica kovarijance

- Parametre normalne distribucije određuje NM (parametri Θ)



VAE enkoder

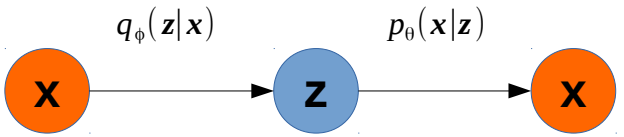
- Enkoderski dio je problematičan
  - Teško je odrediti

$$p_\theta(\boldsymbol{z}|\boldsymbol{x})$$

- Rješenje je u aproksimaciji sa novom distribucijom

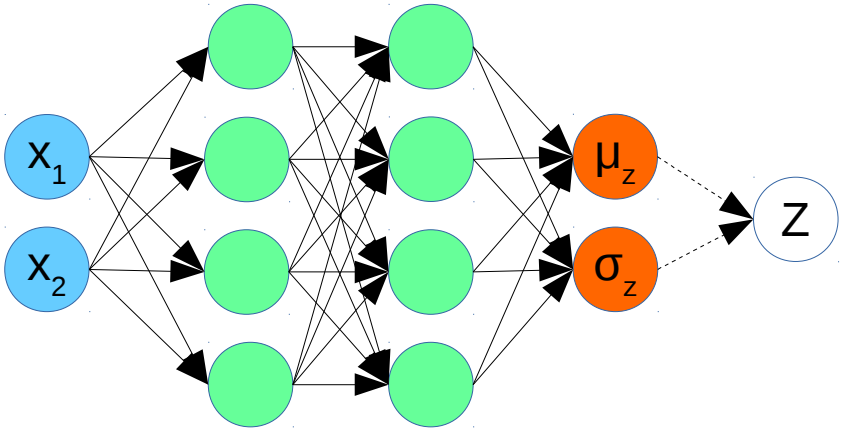
$$q_\phi(\boldsymbol{z}|\boldsymbol{x}) = N(\boldsymbol{\mu}_z(\boldsymbol{x}), \boldsymbol{\sigma}_z(\boldsymbol{x}))$$

  - Slično kao kod wake-sleep algoritma



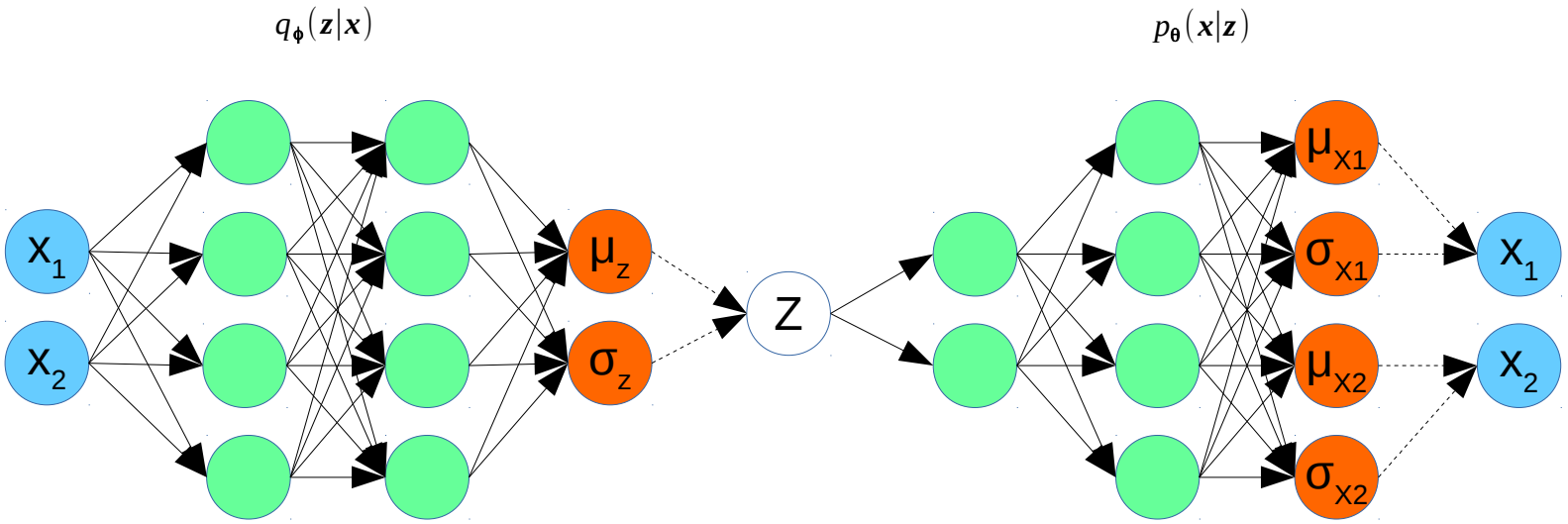
VAE enkoder

- Enkoder je sličan kao i dekoder
- Enkoder generira distribuciju q(**z**|**x**)
  - Iz koje se mogu uzimati uzorci



Varijacijski autoenkoderi

- Kompletni VAE

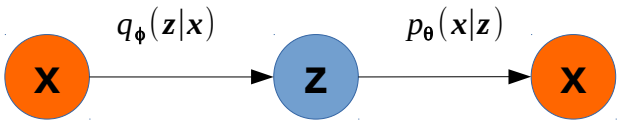


Varijacijski autoenkoderi

- Log vjerojatnost pojedinog ulaznog uzorka može se izraziti kao

$$\log(p(x)) = -D_{KL}(q(z|x)\|p(z)) + \mathrm{E}_{q(z|x)}(\log(p(x|z))) + D_{KL}(q(z|x)\|p(z|x))$$

- D_{KL} je Kullback–Leibler divergencija – mjera sličnosti dviju distribucija (0 kada su identične, uvijek ≥ 0, nije simetrična)



Varijacijski autoenkoderi

$$
\begin{aligned}
\log(p(x)) &= \sum_z q(z|x)\log(p(x)) \\
&= \sum_z q(z|x)\log\left(\frac{p(z,x)}{p(z|x)}\right) \\
&= \sum_z q(z|x)\log\left(\frac{p(z,x)}{q(z|x)}\frac{q(z|x)}{p(z|x)}\right) \\
&= \sum_z q(z|x)\log\left(\frac{p(z,x)}{q(z|x)}\right) + \sum_z q(z|x)\log\left(\frac{q(z|x)}{p(z|x)}\right) \\
&= L + D_{KL}(q(z|x)\|p(z|x)) \\
&\geq L
\end{aligned}
$$

L – Donja varijacijska granica

$$
\begin{aligned}
L &= \sum_z q(z|x)\log\left(\frac{p(z,x)}{q(z|x)}\right) \\
&= \sum_z q(z|x)\log\left(\frac{p(x|z)\,p(z)}{q(z|x)}\right) \\
&= \sum_z q(z|x)\log\left(\frac{p(z)}{q(z|x)}\right) + \sum_z q(z|x)\log\left(p(x|z)\right) \\
&= -D_{KL}(q(z|x)\|p(z)) + \mathrm{E}_{q(z|x)}(\log(p(x|z)))
\end{aligned}
$$

Varijacijski autoenkoderi

- Log vjerojatnost ne može biti manja od L

$$\log_{\boldsymbol{\theta}} p\left(\boldsymbol{x}^{(i)}\right) = D_{KL}\left(q_{\boldsymbol{\phi}}\left(\boldsymbol{z}|\boldsymbol{x}^{(i)}\right)\|p_{\boldsymbol{\theta}}\left(\boldsymbol{z}|\boldsymbol{x}^{(i)}\right)\right) + L\left(\boldsymbol{\theta},\boldsymbol{\phi};\boldsymbol{x}^{(i)}\right)$$

- Umjesto vjerojatnosti, optimizira se donja varijacijska granica L

$$L\left(\boldsymbol{\theta},\boldsymbol{\phi},\boldsymbol{x}^{(i)}\right) = -D_{KL}\left(q_{\boldsymbol{\phi}}\left(\boldsymbol{z}|\boldsymbol{x}^{(i)}\right)\|p\left(\boldsymbol{z}\right)\right) + \mathrm{E}_{q_{\boldsymbol{\phi}}\left(\boldsymbol{z}|\boldsymbol{x}^{(i)}\right)}\left[\log\left(p_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}|\boldsymbol{z}\right)\right)\right]$$

- Očekivanje log(p(**x**|**z**)) je maksimalan log(1) kada varijabla **z** omogućava savršenu rekonstrukciju
  - Možemo ga gledati kao mjeru uspješnosti rekonstrukcije
  - Kada bi funkcija cilja imala samo ovu komponentu, to bi bio običan autoenkoder s mogućim kopiranjem ulaz-izlaz
- -D_{KL} potiče izjednačavanje distribucija q(**z**|**x**) i p(**z**)
  - Smatra se regularizacijskom komponentom

$$\sum_z q(z|x)\log\left(\frac{p(z)}{q(z|x)}\right) = \sum_z q(z|x)\left[\log\left(p(z)\right) - \log\left(q(z|x)\right)\right]$$

Varijacijski autoenkoderi

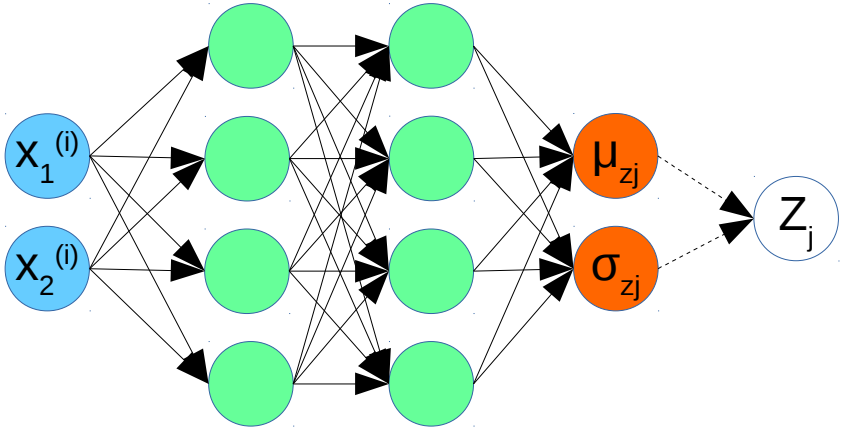
- Pretpostavke

$$q(\boldsymbol{z}|\boldsymbol{x}) = N(\boldsymbol{\mu}_{\boldsymbol{z}}, \boldsymbol{\sigma}_{\boldsymbol{z}})$$

$$p(z) = N(0,1)$$

- Tada je

$$-D_{KL}(q_{\phi}(\boldsymbol{z}|\boldsymbol{x}^{(i)})||p(\boldsymbol{z})) = \frac{1}{2}\sum_{j}\left(1+\log(\sigma_{z_j}^{(i)2}) - \mu_{z_j}^{(i)2} - \sigma_{z_j}^{(i)2}\right)$$



Varijacijski autoenkoderi

- Aproksimacija očekivanja - uzorkovanje

$$\mathrm{E}_{q_{\phi}(\boldsymbol{z}|\boldsymbol{x}^{(i)})}[\log(p_{\theta}(\boldsymbol{x}^{(i)}|\boldsymbol{z}))] \approx \frac{1}{K}\sum_{k=1}^{K}\log(p_{\theta}(\boldsymbol{x}^{(i)}|\boldsymbol{z}^{(i,k)}))$$

  - Obično se K postavlja na 1 dokle god je veličina minibatch-a M dovoljno velika (npr. 100)

Varijacijski autoenkoderi

- Funkcija cilja

$$L(\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{x}^{(i)}) = -D_{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}^{(i)}) \| p(\boldsymbol{z})) + \mathrm{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}^{(i)})}[\log(p_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}|\boldsymbol{z}))]$$

  - Propagira se unazad kroz mrežu

- komponente

$$\mathrm{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}^{(i)})}[\log(p_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}|\boldsymbol{z}))] \approx \frac{1}{K}\sum_{k=1}^{K}\log(p_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}|\boldsymbol{z}^{(i,k)}))$$

$$-\log(p_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}|\boldsymbol{z}^{(i,k)})) = \sum_j \frac{1}{2}\log(\sigma_{x_j}^{(i,k)2}) + \frac{(x_j^{(i)} - \mu_{x_j}^{(i,k)})^2}{2\sigma_{x_j}^{(i,k)2}}$$

$$-D_{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}^{(i)}) \| p(\boldsymbol{z})) = \frac{1}{2}\sum_j \left(1 + \log(\sigma_{z_j}^{(i)2}) - \mu_{z_j}^{(i)2} - \sigma_{z_j}^{(i)2}\right)$$

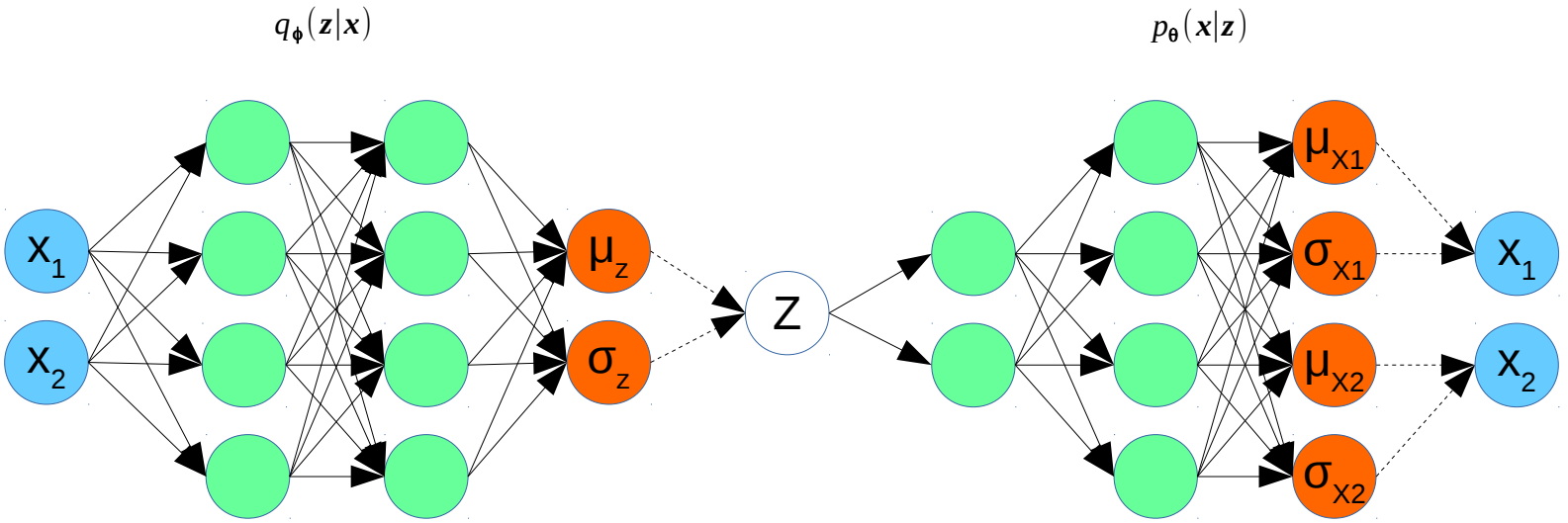
Varijacijski autoenkoderi

- Funkcija cilja za jedan uzorak $x^{(i)}$

$$L(\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{x}^{(i)}) = -D_{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}^{(i)}) || p(\boldsymbol{z})) + \mathrm{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}^{(i)})}[\log(p_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}|\boldsymbol{z}))]$$

- Funkcija cilja za minibatch X^M (M slučajnih uzoraka od N iz skupa za treniranje)

$$L^M(\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{X}^M) = \frac{N}{M} \sum_{i=1}^{M} L(\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{x}^{(i)})$$



Varijacijski autoenkoderi

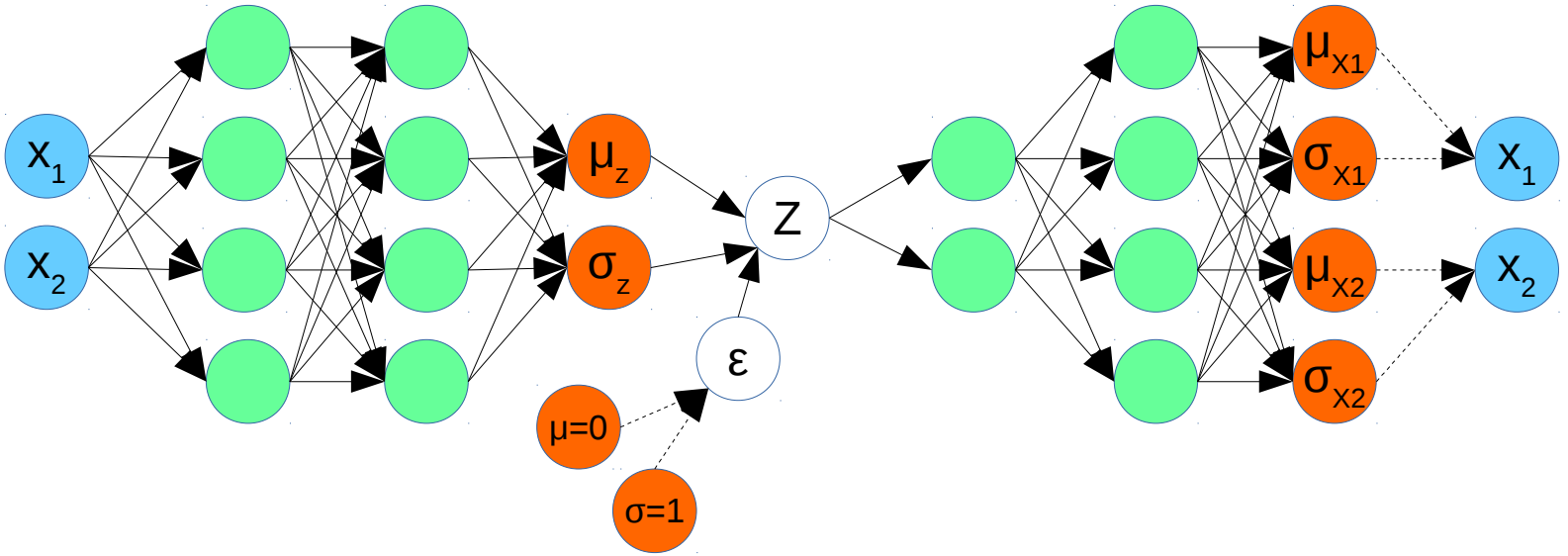
- Reparametrizacijski trik

$$z \sim N\left(\mu_z^{(i)}, \sigma_z^{(i)}\right)$$

$$z^{(i,k)} = \mu_z^{(i)} + \sigma_z^{(i)} \times \varepsilon^{(k)} \qquad \varepsilon_i \sim N(0,1)$$

$$z^{(i,k)} = g\left(\mu_z^{(i)}, \sigma_z^{(i)}, \varepsilon^{(k)}\right)$$

- z ima istu distribuciju, ali je sada moguće odrediti gradijent - backpropagacija

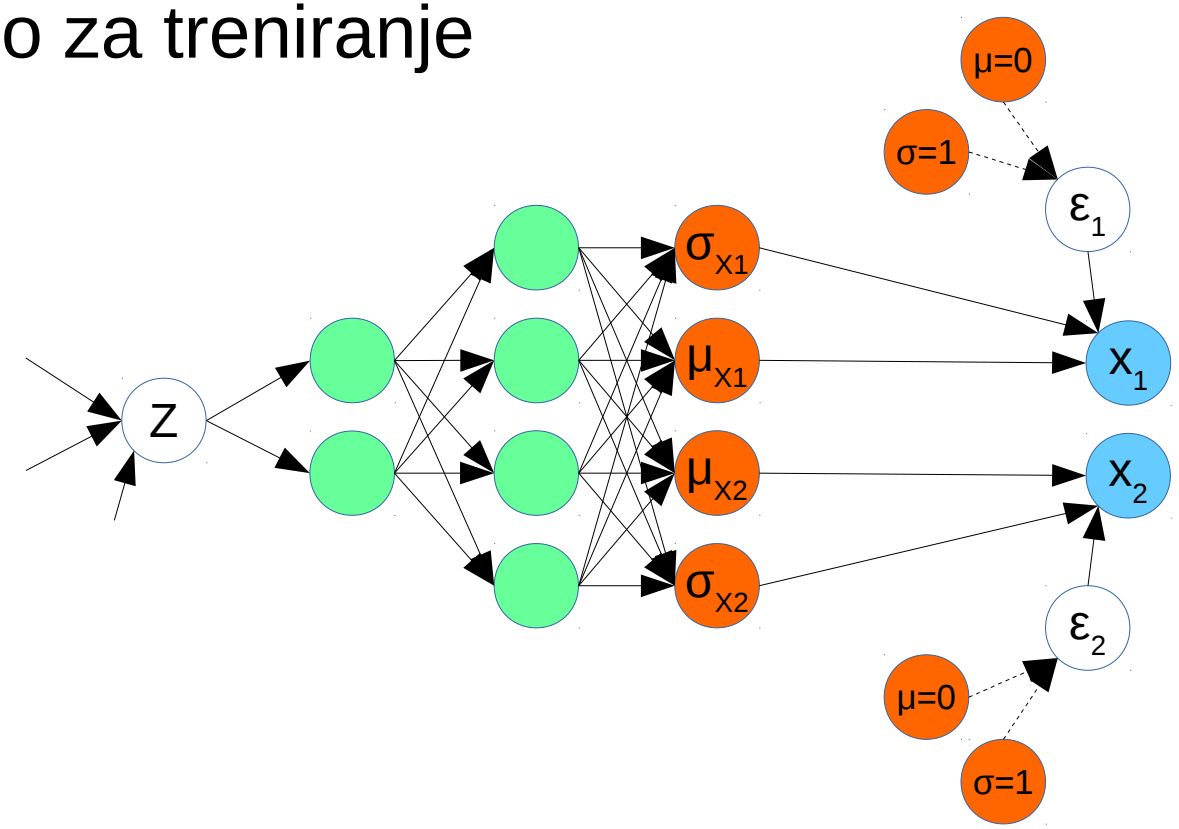


Varijacijski autoenkoderi

- Algoritam
  - Inicijaliziraj parametre Θ i Φ
  - Ponavljaj
    - Odaberi slučajni minibarch $\mathbf{X}^M$
    - Uzorkuj $\boldsymbol{\varepsilon}$
    - Odredi gradijent od L s obzirom na Θ i Φ
    - Izračunaj nove vrijednosti za Θ i Φ prema gradijentu
  - Dok Θ i Φ ne konvergiraju

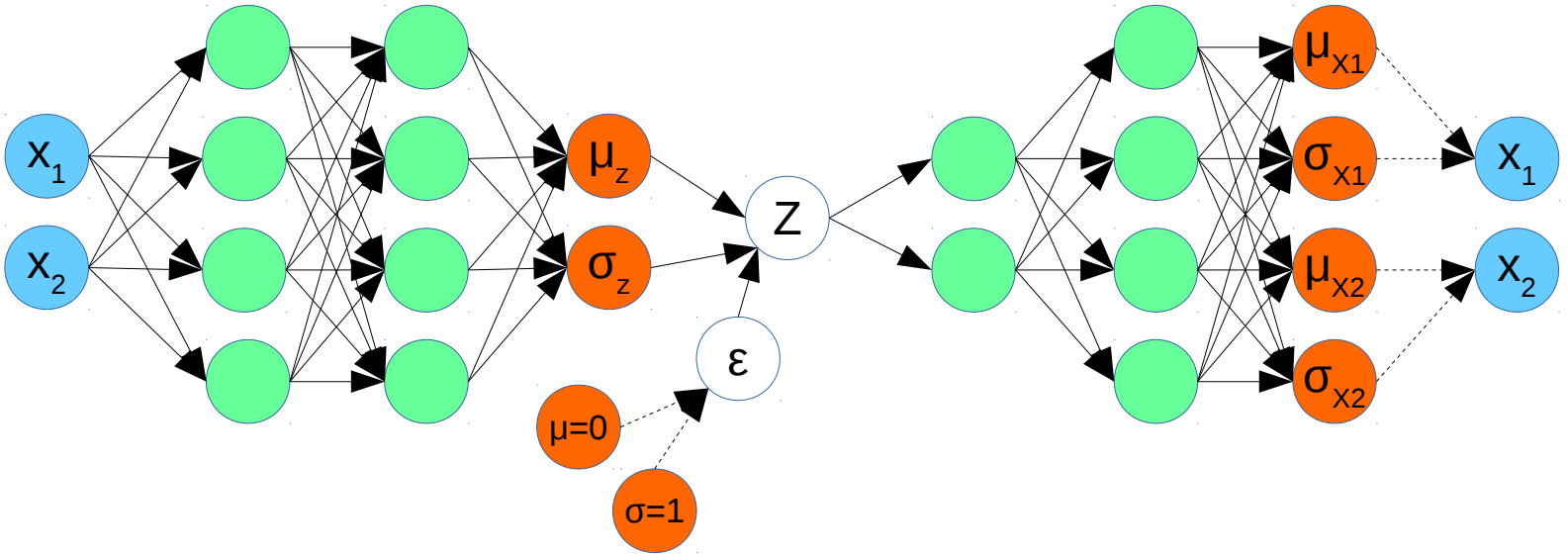
Kako generirati uzorke

- Npr. $x = \mu_x + \sigma_x \times \varepsilon$ $\varepsilon_i \sim N(0,1)$
  - Nije bitno za treniranje



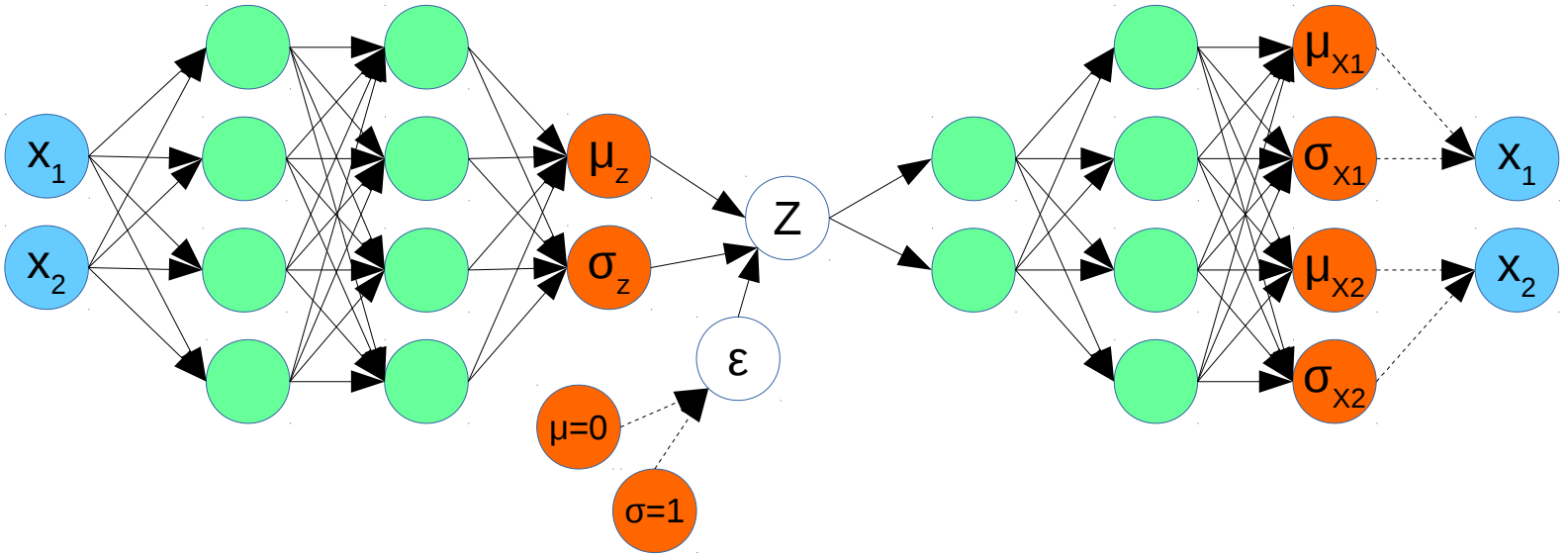
Iterpretacija VAE

- Enkoder uči koliko šuma (σ_z) dodati ulaznim podacima da bi dekoder bio dobar generator podataka

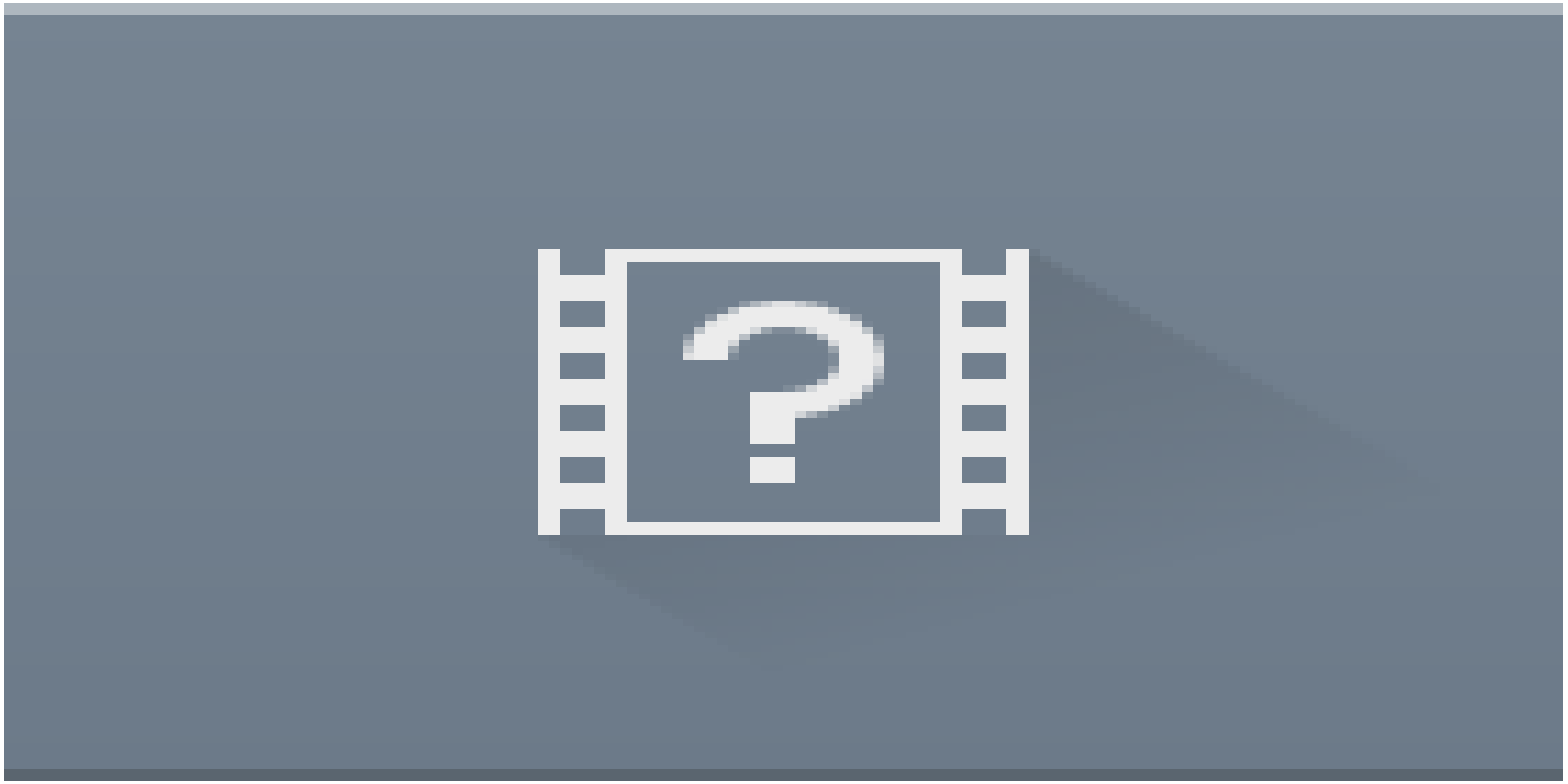


Iterpretacija VAE

- Koliko je sve ovo nepoznato:
  - Backpropagacija?
  - Dodavanje šuma negdje u mreži?
  - Regularizacijski član vezan za skriveni sloj? (sparse AE)



Primjer treniranja s 2 skrivena elementa



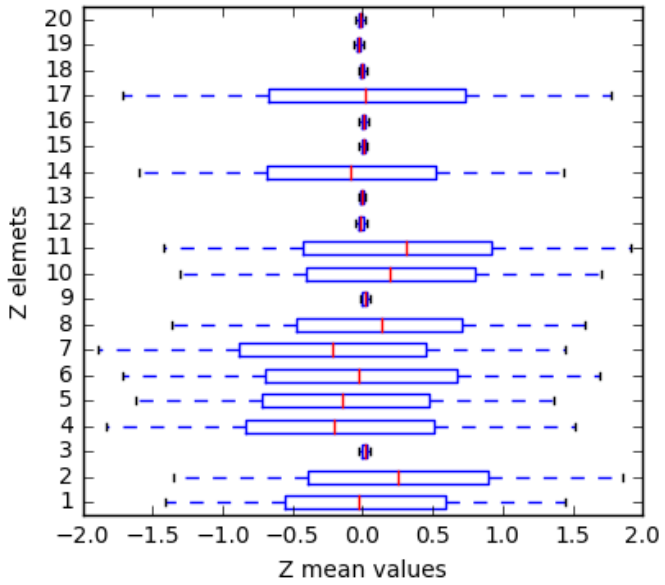
Component collapsing

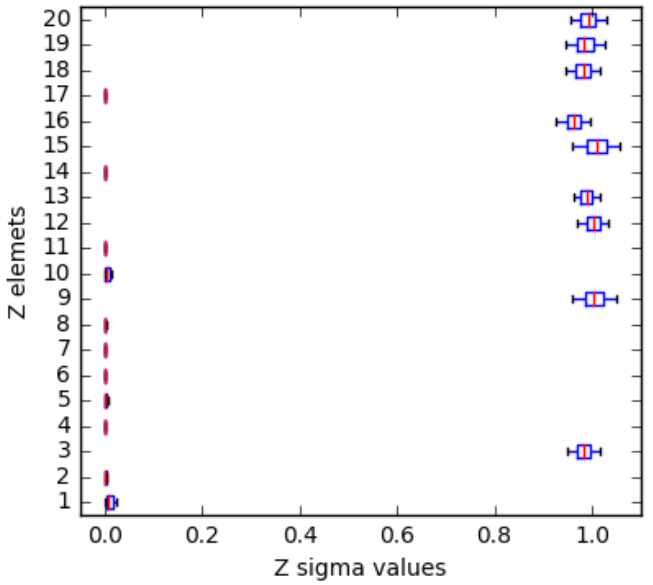
- Što ako regularizacijski član uspije?
  - Za neke z_i postigao je da $q(\mathbf{z}|\mathbf{x}_j) = p(\mathbf{z})$ za sve $\mathbf{x}_j$
- Ti skriveni elementi z_i više ne nose nikakvu korisnu informaciju
  - Isti su za svaki ulaz
  - $\mu_{zi} = 0$ – približno
  - $\sigma_{zi} = 1$ – približno
  - KL divergencija koja je komponenta funkcije cilja tada je = 0
- Dekoder te skrivene elemente ne može upotrijebiti za išta smisleno pa ih "isključuje"
  - Težine koje povezuju z_i sa prvim slojem dekodera postavlja na nulu – približno

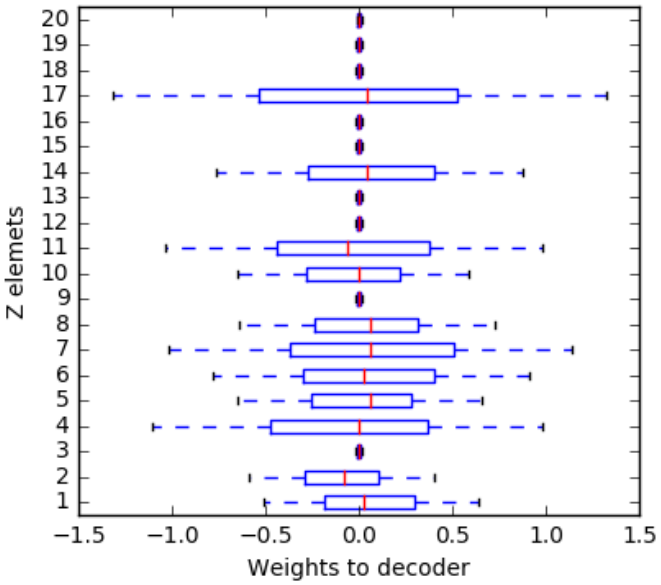
Component collapsing

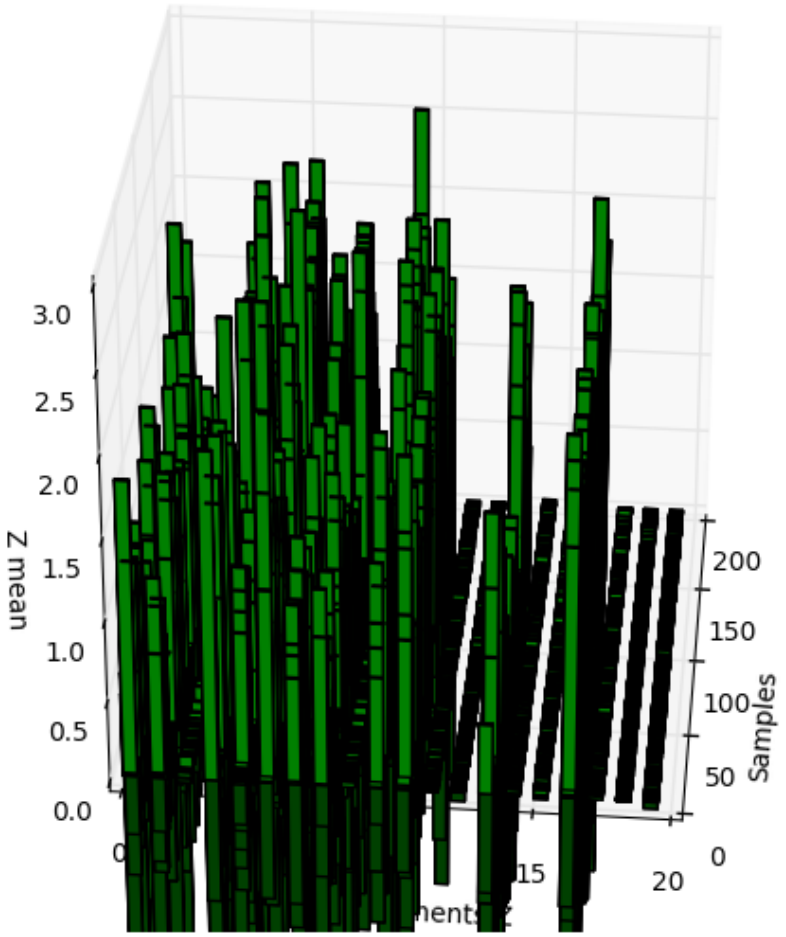
- Sličan efekt kao nametanje rijetkosti (sparsity constraint), ali su aktivni uvijek isti skriveni elementi
- Te skrivene varijable samo donose šum – ne nose korisnu informaciju
- Veća dubina mreže -> manje korištenih skrivenih varijabli
- Nije iskorišten puni kapacitet mreže!
  - Skrivene elemente koji se ne koriste možemo obrisati
- Dobro je znati ako se skriveni sloj namjerava koristiti za npr. diskriminaciju

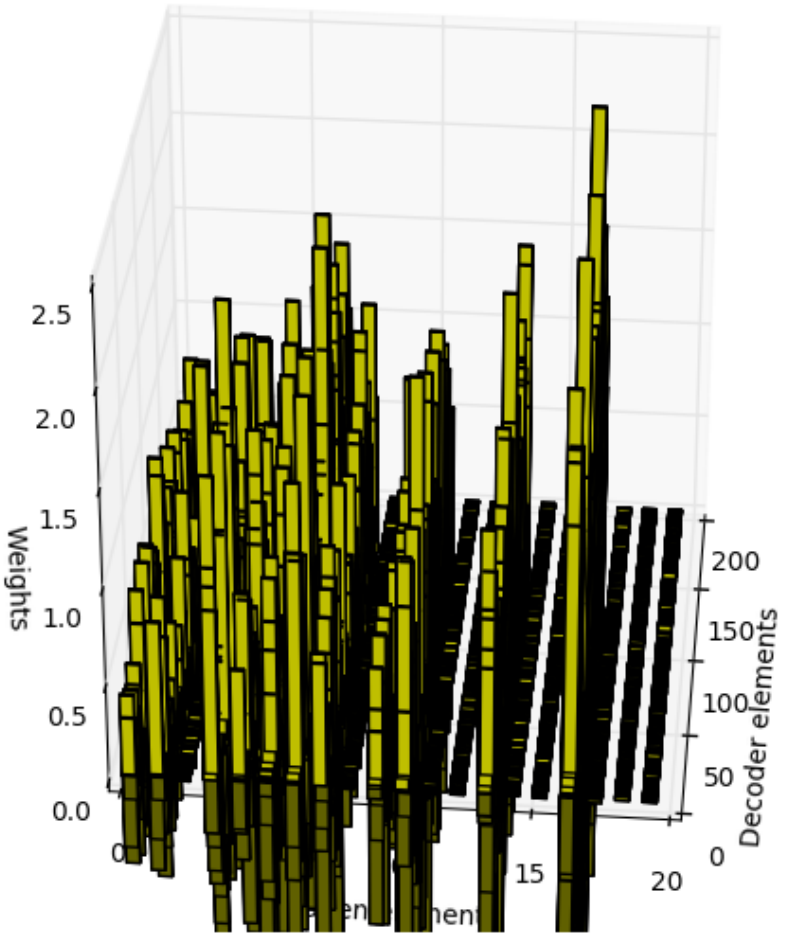
Component collapsing








3.0
2.5
2.0
1.5
1.0
0.5
0.0
Z mean
0
15
20
200
150
100
50
0
Samples


2.5
2.0
1.5
1.0
0.5
0.0
Weights
0
15
20
200
150
100
50
0
Decoder elements

Varijacijski autoenkoderi

- Prednosti
  - Efikasno uzorkovanje – samo jednom je dovoljno
  - Nema hiperparametara kao što su reguzlarizacijska konstanta ili distribucija šuma
  - Bolji generativni model od običnog AE – jednostavno generiranje uzoraka
- Nedostaci
  - Maksimizira se donja varijaciska granica a ne p(x) – maksimum donje granice ne mora odgovarati maksimumu p(x)

Dubiki AE

- VAE
  - Problem dubokog učenja za početne slojeve riješava novom funkcijom greške samo za enkoderski dio – početne slojeve
- Duboki AE
  - Sporo učenje dubokih slojeva ostaje, ali to nije problem zbog dobre inicijalizacije kroz pohlepno predtreniranje