Ograničeni Boltzmanov stroj

Doc.dr.sc. Marko Subašić

Generativni modeli

- Nije cilj naučiti funkciju koja povezuje uzorke iz skupa za učenje sa odgovarajućim oznakama
  - Diskrimativni modeli
  - Jednostavniji problem ali ...
- Generativni modeli mogu generirati nove uzorke koji odgovaraju skupu za treniranje
  - Iz uzoraka za učenje bitno je naučiti distribucije svih elemenata ulaznih uzoraka kako bi se generirali odgovarajući uzorci

Generativni modeli

- Modelira se distribucija vjerojatnosti više varijabli

$$p(\boldsymbol{x})$$

  - Kod nekih modela je moguće evaluirati model distribucije direktno, a kod nekih samo indirektno
- Neki potpadaju pod strukturirane stohastičke modele
  - Grafovi
- Generativno učenje
  - drugačiji pristup od diskriminativnog učenja

$p(\boldsymbol{x})$

$p(d|\boldsymbol{x})$

$$p(d|\boldsymbol{x}) = \frac{p(d)\,p(\boldsymbol{x}|d)}{p(\boldsymbol{x})}$$

Generativni modeli

- Poznavanje distribucije vjerojatnosti otvara neke nove mogućnosti:
  - Bolja, efikasnija reprezentacija sadržaja
    - Efikasnije od “sirovih” piksela za sliku
  - Pronalazak bitnih značajki
  - Kompresija
  - Prepoznati neuobičajeni uzorak
  - ...

Generativni modeli

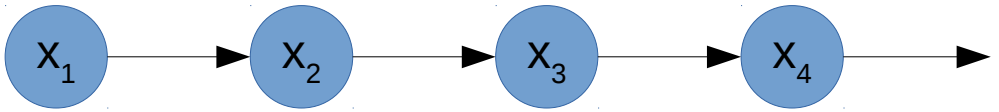
- Procjene vjerojatnosti u praksi rijetko idu direktno
- Često uključuju nekoliko ili puno uzimanja uzoraka kako bi na temelju zastupljenosti uzoraka procijenili funkciju vjerojatnosti
- Treba se naviknuti na to...
- ... a možda i ne
- Ako je algoritam efikasan uzorkovanja će biti malo

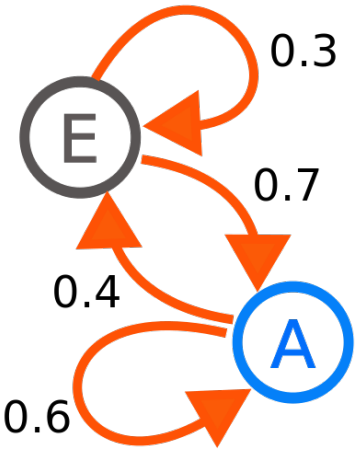
Korisni alati iz teorije vjerojatnosti

- Markovljevi lanci – Markov Chain
- Monte Carlo metoda
- Markov Chain Monte Carlo
- Gibbs sampling

Markov Chain

- Slučajni proces koji mijenja stanja
- Nema memorije – vjerojatnost slijedećeg stanja ovisi samo o trenutnom stanju
  - Jednostavnije nego da ovisi o drugim ili svim prošlim stanjima
- Uvjetna vjerojatnost slijedećeg stanja ovisi samo o trenutnom stanju





Monte Carlo metoda

- Eksperiment u kojem se **slučajno** uzimaju uzorci da bi se dobila kvalitetna procjena rezultat
  - Bitno je da se radi o stvarnoj slučajnosti
- Bitna primjena je uzorkovanje funkcije vjerojatnosti
  - Na temelju učestalosti pojedinih uzoraka zaključujemo o funkciji vjerojatnosti
- Što više uzoraka to bolja procjena rezultata

Markov Chain Monte Carlo

- Metode za uzorkovanje funkcija vjerojatnosti
- Konstruira se Markovljev lanac čija je ravnotežna distribucija (ekvilibrij) jednaka traženoj distribuciji
- Nakon inicijalizacije, mijenjaju se stanja kako bi dosegao ekvilibrij
  - Tada konkretno inicijalno stanje više nije bitno
- Nakon dosezanja ekvilibrija uzimaju se uzorci za procjenu distribucije
  - Kvaliteta uzorka se povećava sa brojem promjena stanja

Gibbsovo uzorkovanje

- MCMC algoritam za pribavljanje uzoraka združene funkcije vjerojatnosti za više slučajnih varijabli

$$p(x_1, x_2, x_3, \ldots, x_n)$$

- Koristi se kada direktno uzorkovanje nije moguće
- Rezultat je aproksimacija
- **Naizmjence se uzorkuju pojedine varijable** (x_n, jedna po jedna)
  - Mora biti poznata uvjetna vjerojatnost jedne varijable uz zadane sve ostale varijable

$$p(x_k | x_1, x_2, \ldots, x_n)$$

Gibbsovo uzorkovanje

- Nakon dosezanja ekvilibrija uzimamo uzorke koji dobro predstavljaju traženu distribuciju
- Susjedni uzorci su međusobno zavisni (Markovljevo polje)
  - Ako se traže nezavisni uzorci, preskačemo neki broj promjena stanja
- Može se ubrzati simuliranim hlađenjem
  - Veće vjerojatnosti u ranim iteracijama
- Ispravan način za pribavljanje kvalitetnih uzoraka za procjenu združene funkcije vjerojatnosti

Kako radi ograničeni Boltzmanov stroj

- Glavni princip rada je minimizacija energetske funkcije
- Neuroni su stohastički i binarni
- Nije feed forward mreža

Modeli temeljeni na energetskoj funkciji

- Definira se energetska funkciju na temelju stanja neurona u mreži i težina koje ih povezuju
- Na primjeru hopfieldove mreže pokazano je da binarni neuroni uz simetrične veze i potpunu povezanost iterativno idu prema (lokalnom) minimumu “pogodne” energetske funkcije

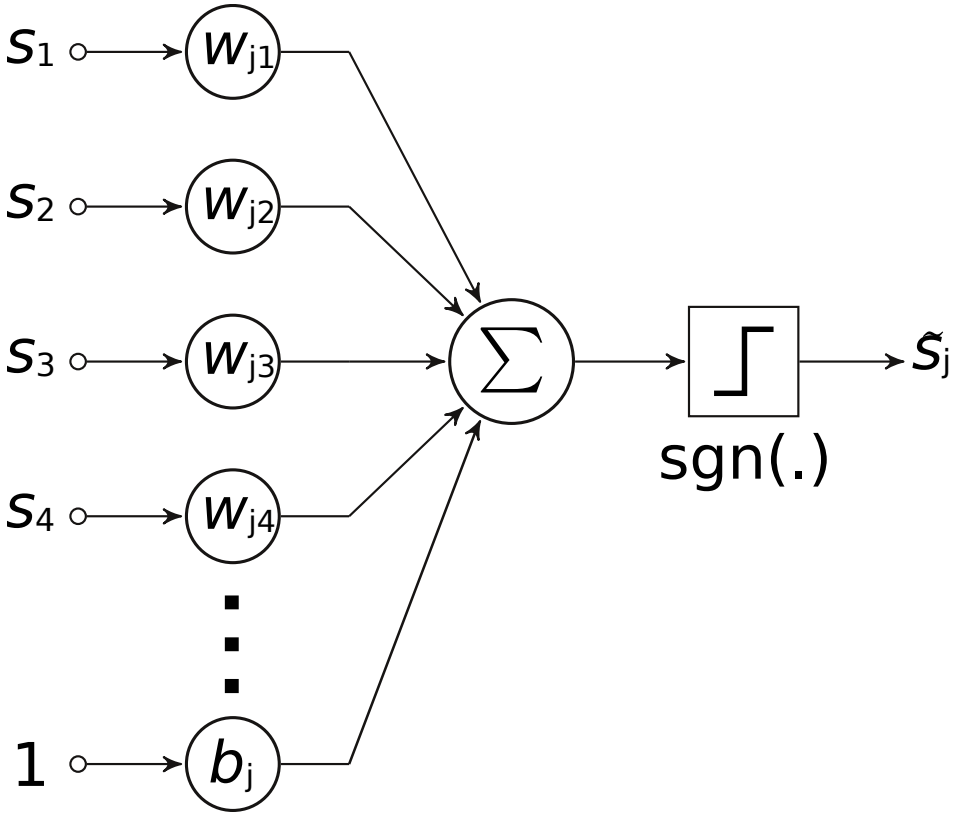
Hopfieldova mreža

- Memorije koje koriste lokalne minimume energetske funkcije
- Čitanje iz memorije – pronalazak lokalnog minimuma energetske funkcije
- Asocijativna memomirija
  - Mreža vraća naučeni vektor najsličniji ulaznom vektoru
  - Zašumljena naučena slika → naučena slika
  - Odrezana naučena slika → cijela naučena slika

Hopfieldova mreža

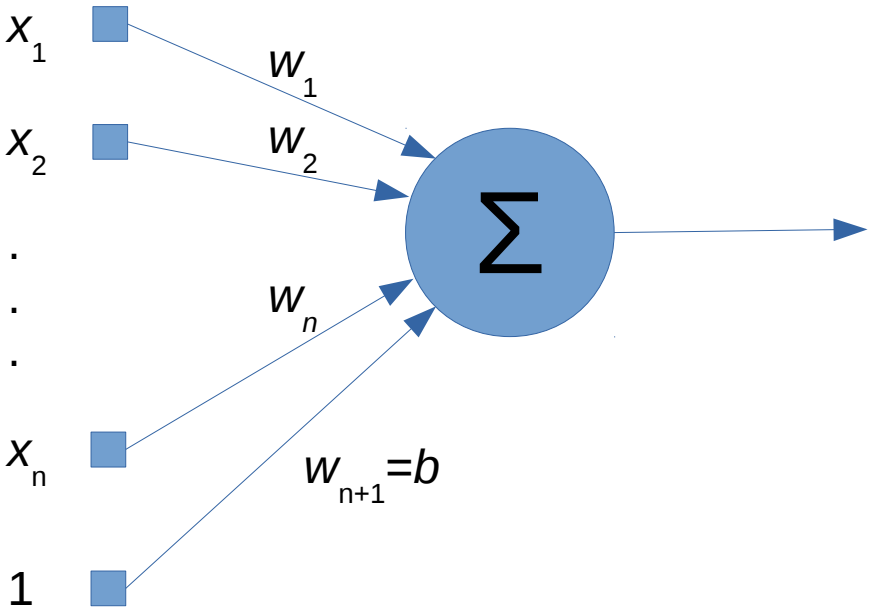
$$s_j = sgn\left(\sum_{i=1}^{N} w_{ji} s_i + b_j \right)$$

$$s_j = \{-1, +1\}$$



Sakrijmo pomak

- Radi preglednosti u nastavku se neće spominjati pomak *b* svakog neurona iako on uvijek postoji!
- Možemo ga sakriti među težine
  - Dodatna težina čiji je ulaz uvijek 1



Hopfieldova mreža

- Poslužit će nam kao primjer mreže čiji rad je zasnovan na minimizaciji energetske funkcije
- Neuroni nisu stohastički
- Koristi se funkcija praga
- Binarni neuroni (0,1) ili (-1,1)

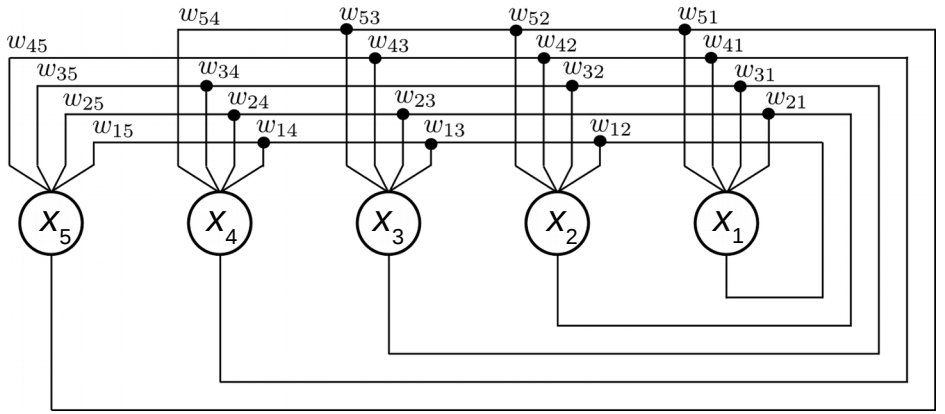
Binarni neuron

- Nelinearna funkcija praga kao aktivacijska funkcija
- Loše:
  - Gubitak informacija
- Dobro:
  - Efikasnost
  - Otpornost na šum
  - Robusnost
  - Generalizacija !!!
  - ...

Hopfieldova mreža

- Pamćenje *N* vektora
  - Pravilo vanjskog produkta
  - Generalizirano Hebbovo učenje
  - "Fire together, wire together"
  - Nema povratnih veza na isti neuron $w_{ii} = 0$
  - Simetrične težine $w_{ij} = w_{ji}$
  - Korelacija
  - Pomake b_i zanemarujemo, ali postoje

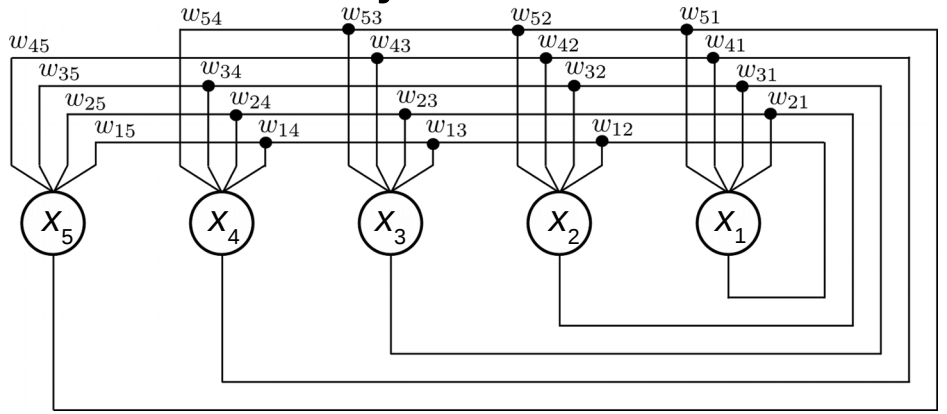
$$w_{ji} = \frac{1}{N} \sum_{m=1}^{N} x_{mj} x_{mi}$$



Hopfieldova mreža

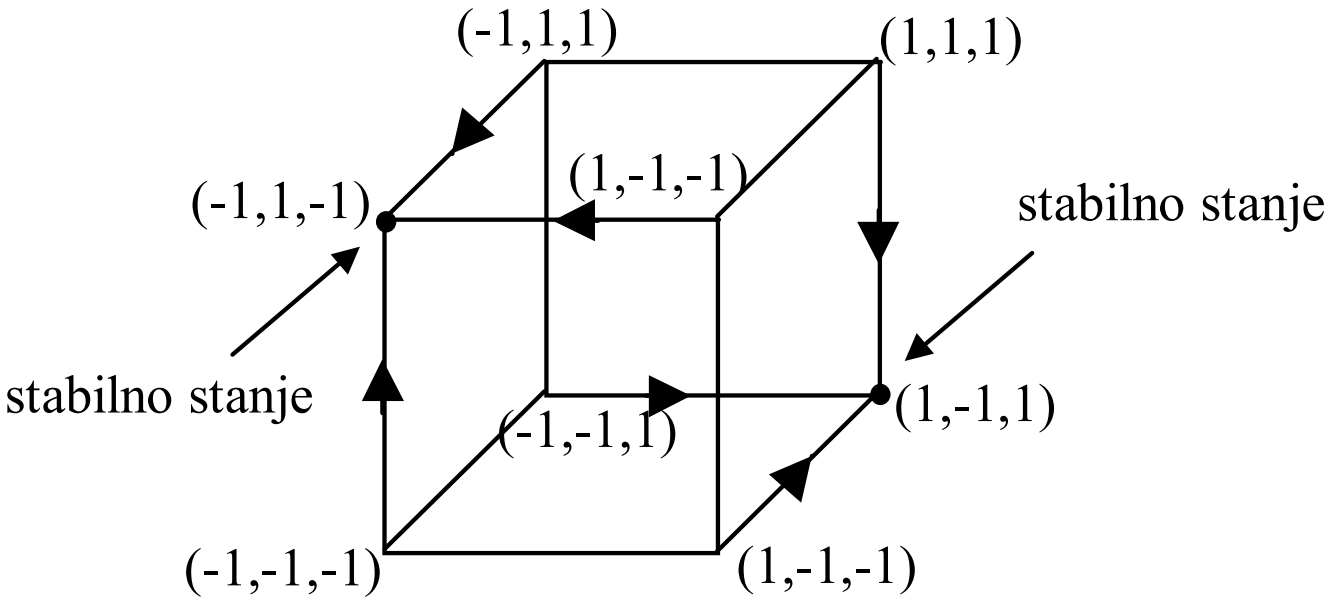
- Faza dohvata
  - Mreža se postavi u početno stanje jednako ulaznom vektoru
    - Npr. zašumljeni naučeni vektor
  - Iterativno određivanje novog stanja slučajno odabranog neurona do konvergencije
    - Konvergira ka jednom od naučenih stanja

$$x_j = sgn\left(\sum_{\substack{i=1 \\ i \neq j}}^{M} w_{ji} x_i + b_j\right)$$



Hopfieldova mreža

- Ilustracija konvergiranja



Hopfieldova mreža

- Energetska funkcija

$$E = -\frac{1}{2} \sum_{i=1}^{N} \sum_{\substack{j=1 \\ j \neq i}}^{N} w_{ji} x_j x_i$$

- Promjena energije uslijed promjene stanja
  - Uvijek je negativna – traženje lokalnog minimum energetske funkcije

$$\Delta E = -\Delta x_j \sum_{\substack{i=1 \\ i \neq j}}^{N} w_{ji} x_i$$

Izraz za novo stanje x_j

Hopfieldova mreža

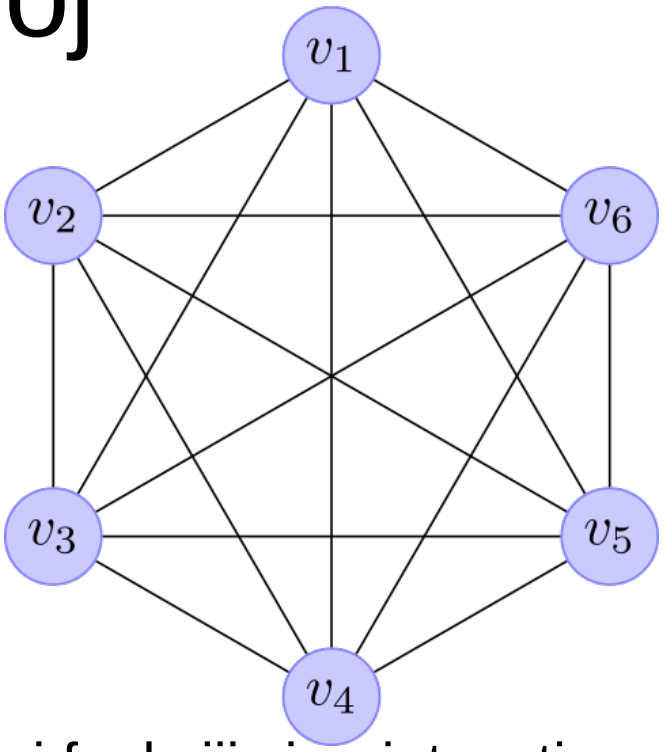
- Energija ovisi o stanju mreže uz neki dani skup težina
- Lokalni minimumi energetske funkcije predstavljaju memorirane uzorke
- Postoje i lažna stanja
  - Negativ memoriranog uzorka
  - Mješavine više uzoraka
- Kapacitet je $0.138N$
- Jedno rješenje za povećanje memorije: Unlearning - zaboravljanje
  - Nakon konvergencije iz slučajnog stanja uči se negativnim Hebbovim pravilom
  - Lažna stanja u pravilu imaju višu energiju

$$\Delta w_{ji} = -\varepsilon x_j x_i, \quad 0 < \varepsilon \ll 1$$

Hopfieldova mreža

- Binarni deterministički neuroni (-1, 1)
  - Ne može pobjeći iz lošeg lokalnog minimuma
- Faza dohvata = minimizacija energije
- Naučena stanja = lokalni minimumi energije
- Zaboravljanje neželjenih stanja kako bi bolje zapamtili željena
- Dvosmjerne veze
  - Bez povratnih veza

Boltzmanov stroj



- Binarni neuroni
- Potpuna povezanost bez povratnih veza
- Dvosmjerne simetrične veze
- Stohastička verzija Hopfieldove mreže
  - Promjena stanja je slučajna – prema naučenoj funkciji vjerojatnosti
  - Parametri mreže određuju vjerojatnost slijedećeg stanja, a ne slijedeće stanje
  - Otvara mogućnost bijega iz lošeg lokalnog minimuma

Boltzmanov stroj

- Vjerojatnost definirana pomoću energetske funkcije
- Boltzmanova (Gibbsova) distribucija $P(\boldsymbol{x}) \propto e^{\frac{-E(\boldsymbol{x})}{kT}}$
  - k – Boltzmanova konstanta
  - T – Termodinamička temperatura – ignoriramo je za sada

$$P(\boldsymbol{x};\boldsymbol{W}) = \frac{e^{-E(\boldsymbol{x})/T}}{\sum_{\boldsymbol{x}} e^{-E(\boldsymbol{x})/T}} = \frac{e^{\frac{1}{2}\boldsymbol{x}^T \boldsymbol{W} \boldsymbol{x}}}{Z(\boldsymbol{W})} \qquad \sum_{\boldsymbol{x}} P(\boldsymbol{x}) = 1$$

  - $Z(\boldsymbol{W})$ je particijska funkcija (partition function)
    - Uključuje **sva** stanja mreže

$$E(\boldsymbol{x}) = -\frac{1}{2}\sum_{i=1}^{N}\sum_{\substack{j=1 \\ j\neq i}}^{N} w_{ji}\, x_j\, x_i = -\frac{1}{2}\boldsymbol{x}^T \boldsymbol{W} \boldsymbol{x}$$

Boltzmanov stroj

- Energetska razlika za stanja 0 i 1

$$\Delta E_j = E_{x_j=0} - E_{x_j=1} = -\frac{1}{2}\sum_{\substack{i=1\\k\neq i}}^{N}\sum_{\substack{k=1\\k\neq i\\k\neq j}}^{N} w_{ki}\,x_k\,x_i + \frac{1}{2}\sum_{\substack{i=1\\k\neq j}}^{N}\sum_{\substack{k=1\\k\neq i\\k\neq j}}^{N} w_{ki}\,x_k\,x_i + \sum_{\substack{i=1\\i\neq j}}^{N} w_{ji}\,x_i$$

$$= \sum_{i=1}^{N} w_{ji}\,x_i$$

- Promjena stanja neurona

$$x_j = \begin{cases} 1, & \text{s vjerojatnošću } \dfrac{1}{1+e^{-\Delta E_j/T}} = \dfrac{1}{1+e^{-\sum_{i=1}^{N} w_{ji}\,x_i}} \\ 0, & \text{inače} \end{cases}$$

  - T – temperatura sustava (uključuje Boltzmanovu konstantu)

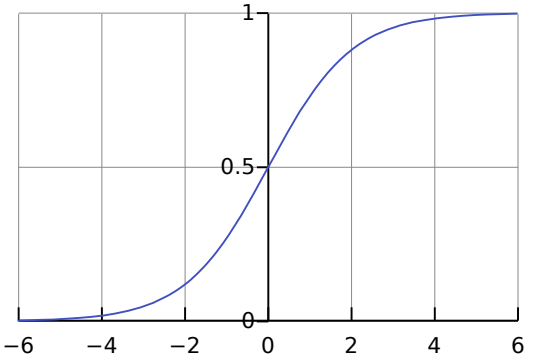
Boltzmanov stroj

- Promjena stanja neurona – uzorkovanje!

$$x_j = \begin{cases} 1, & \text{s vjerojatnošću } \dfrac{P(x_j=1, \boldsymbol{x}_{\not j})}{P(x_j=0, \boldsymbol{x}_{\not j}) + P(x_j=1, \boldsymbol{x}_{\not j})} = \dfrac{e^{\sum_{i \neq j} w_{ji} x_i}}{1 + e^{\sum_{i \neq j} w_{ji} x_i}} = \dfrac{1}{1 + e^{-\sum_{i \neq j} w_{ji} x_i}} \\ 0, & \text{inače} \end{cases}$$

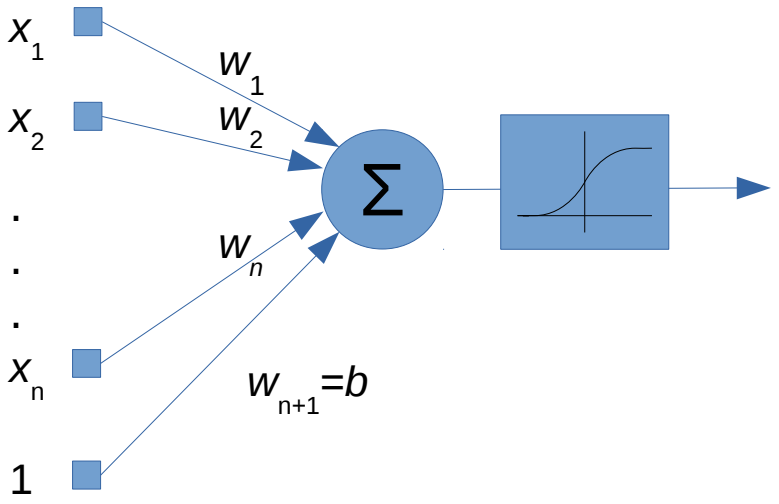
  - Logistička funkcija

$$f(x) = \frac{1}{1 + e^{-k(x - x_0)}}$$



Stohastički neuron

- Struktura vrlo slična perceptronu, ali je funkcionlnost malo drugačija



Boltzmanov stroj

- Učenje prilagođava distribuciju podacima za učenje
  - Maksimiziranje vjerojatnosti za podatke za učenje

$$P(\boldsymbol{x};\boldsymbol{W}) = \frac{e^{-E(\boldsymbol{x})/T}}{\sum_{\boldsymbol{x}} e^{-E(\boldsymbol{x})/T}} = \frac{e^{\frac{1}{2}\boldsymbol{x}^T \boldsymbol{W}\boldsymbol{x}}}{Z(\boldsymbol{W})}$$

- Radi preglednosti privremeno zaboravljamo T mada ju u konačnici možemo koristiti
- Maksimizirati možemo i logaritam vjerojatnosti

$$\ln\left[\prod_{n=1}^{N} P(\boldsymbol{x}^{(n)};\boldsymbol{W})\right] = \sum_{n=1}^{N}\left[\frac{1}{2}\boldsymbol{x}^{(n)^T}\boldsymbol{W}\,\boldsymbol{x}^{(n)} - \ln Z(\boldsymbol{W})\right]$$

  - Linearna funkcija težina
  - Jednostavno određivanje gradijenta
  - Jednostavno optimiziranje

Boltzmanov stroj

$$\frac{\partial}{\partial w_{ij}} \ln \left[\prod_{n=1}^{N} P\left(\boldsymbol{x}^{(n)}; \boldsymbol{W}\right) \right] = \sum_{n=1}^{N} \left[x_i^{(n)} x_j^{(n)} - \sum_{\boldsymbol{x}} x_i x_j P\left(\boldsymbol{x}; \boldsymbol{W}\right) \right]$$

$$= N \left[\langle x_i x_j \rangle_{Data} - \langle x_i x_j \rangle_{P(\boldsymbol{x}; \boldsymbol{W})} \right]$$

- Možemo koristiti stohastic gradient descent
- Učenje mijenja težine kako bi se povećalo umnožak logaritama vjerojatnosti stanja mreže na slijedeći način:

$$\Delta w_{ij} = \eta \left[\langle x_i x_j \rangle_{Data} - \langle x_i x_j \rangle_{P(\boldsymbol{x}|\boldsymbol{W})} \right]$$

  - η je faktor učenja
  - Pratimo gradijent

Komponente gradijenta

- Empirijska korelacija
  - Jednostavno se određuje – iz trenutne mini grupe
  - Wake rule
  - Istovjetno učenju Hopfieldove mreže
    - Hebbovo učenje
  - Koje vrijednosti može poprimiti $x_i x_j$?

$$\langle x_i x_j \rangle_{Data} = \frac{1}{N} \sum_{n=1}^{N} \left[x_i^{(n)} x_j^{(n)} \right]$$

Komponente gradijenta

- Korelacija prema modelu $\langle x_i x_j \rangle_{P(x;W)} = \sum_x x_i x_j P(x;W)$
  - Komponenta dolazi od normalizacijskog člana *Z* koji bi trebao uključivati sva stanja
  - Određivanje nije jednostavno
  - Estimira se pomoću Monte Carlo metode
    - Prosječna vrijednost x_i x_j dok BM iterira
    - Treba prvo doći u ekvilibrij
  - Sleep rule – zaboravljanje (Hebbovo učenje)

$$\Delta w_{ij} = \eta \left[\langle x_i x_j \rangle_{Data} - \langle x_i x_j \rangle_{P(x|W)} \right]$$

Thermal equilibrium

- Pojam posuđen iz fizike
- Ne odnosi se na stacionarno stanje mreže
- Odnosi se na stacionarnost funkcije gustoće vjerojatnosti
  - Konvergencija funkcije gustoće vjerojatnosti
  - Za zadanu temperaturu *T*
- Teško je odrediti kada je dosegnut

Thermal equilibrium

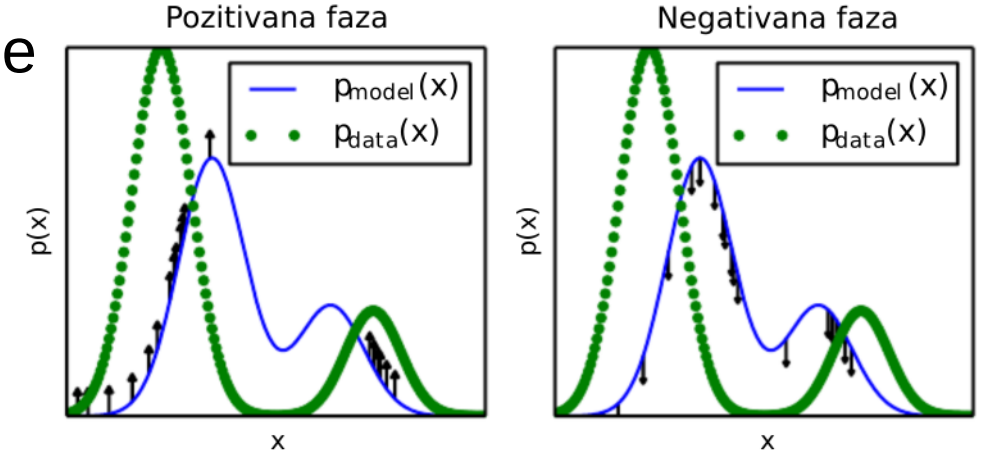
- Vjerojatnost konfiguracije je eksponencijalna funkcija energije samo u termalnom ekvilibriju
  - Ne ovisi o početnom stanju

$$P(\boldsymbol{x}) \propto e^{\frac{-E(\boldsymbol{x})}{kT}}$$

- Smanjenjem temperature postižemo se da energije stanja mreže približavaju lokalnom minimumu
  - Simulirano hlađenje

Boltzmanov stroj

- Wake (poztivna) faza – povećava se vjerojatnost uzoraka iz skupa za treniranje
  - Ne svih uzoraka već iz dotične mini grupe
- Sleep (negativna) faza – smanjuju se vjerojatnosti "svih" uzoraka (stanja) modela
  - Ne svih stanja već karakterističnog uzorka stanja prema "mišljenju" modela
- Konvergencija kada su ove dvije faze u ravnoteži
  - Kada model distribucija modela odgovara distribuciji skupa za treniranje
- Samo wake faza vodi u zasićenje
  - Sleep faza to sprječava

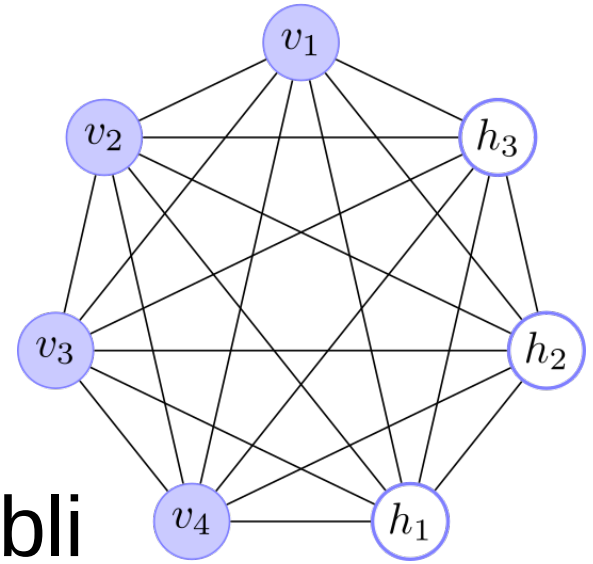


Boltzmanov stroj

- Problemi:
- Uzorkovanje za sleep fazu je potencijalno dugotrajno
- Istrenirani model se temelji na korelaciji drugog reda
  - Za opisivanje kompleksnijih odnosa trebaju i korelacije viših redova $\langle x_i x_j x_k \ldots \rangle$
    - Rješenje je BM višeg reda – broj parametara (težina) modela brzo raste

$$P'(\boldsymbol{x}; \boldsymbol{W}, \boldsymbol{V}, \ldots) = \frac{e^{\frac{1}{2}\sum_{ij} w_{ij} x_i x_j + \frac{1}{6}\sum_{ijk} v_{ijk} x_i x_j x_k + \ldots}}{Z'}$$

Boltzmanov stroj



- Uvode se skrivene varijable
- Njihov zadatak je ubaciti korelacije višeg reda vidljivih varijabli u model koji koristi samo korelaciju drugog reda
- Jedno stanje mreže čine i vidljive i skrivene varijable
- Skriveni neuroni obično se označavaju sa *h*
- Vidljivi neuroni obično se označavaju sa *v*

Skriveni neuroni

- Nemaju zadane vrijednosti pa mogu poprimiti bilo koje vrijednosti
- Ulaze u model distribucije i ponašaju se u skladu s modelom
- Mogu imati korisne uloge i otvoriti nove mogućnosti
  - Recimo mogu raditi ekstrakciju značajki
- Ukupno stanje mreže

$$x = (v, h)$$

Skriveni sloj

- Kako su neuroni skrivenog sloja slobodni, maksimizira se vjerojatnost vidljivih neurona

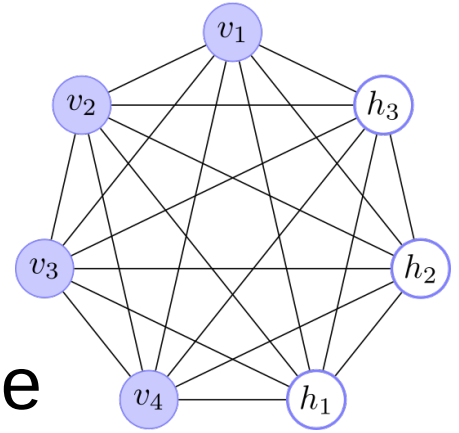
$$P(\boldsymbol{v};\boldsymbol{W})=\sum_{\boldsymbol{h}} P(\boldsymbol{v},\boldsymbol{h};\boldsymbol{W})=\sum_{\boldsymbol{h}} \frac{e^{\frac{1}{2}\boldsymbol{x}^T \boldsymbol{W}\boldsymbol{x}}}{Z(\boldsymbol{W})}$$

  - A ne vjerojatnost stanja svih neurona

$$P(\boldsymbol{x};\boldsymbol{W})=P(\boldsymbol{v},\boldsymbol{h};\boldsymbol{W})$$

  - Opet maksimiziramo logaritam vjerojatnosti

Skriveni neuroni



- Gradijent težina opet ima dvije komponente

$$\frac{\partial}{\partial w_{ij}} \ln \left[\prod_{n=1}^{N} P\left(\boldsymbol{v}^{(n)}; \boldsymbol{W}\right) \right] = N \left[\left\langle x_i x_j \right\rangle_{P(\boldsymbol{h}|\boldsymbol{v}^{(n)};\boldsymbol{W})} - \left\langle x_i x_j \right\rangle_{P(\boldsymbol{v},\boldsymbol{h};\boldsymbol{W})} \right]$$

- Prva komponenta je korelacija stanja mreže ako je vidljivi sloj fiksiran na ulazni uzorak
  - Skriveni sloj je slobodan i varira prema modelu
- Druga komponenta je korelacija kada BM generira uzorke iz svoje distribucije
  - I vidljivi i skriveni sloj su slobodni

Skriveni neuroni

- Konačni izraz za treniranje je

$$\Delta w_{ij} = \eta \left[\langle x_i x_j \rangle_{P(\boldsymbol{h}|\boldsymbol{v}^{(n)};\boldsymbol{W})} - \langle x_i x_j \rangle_{P(\boldsymbol{v},\boldsymbol{h};\boldsymbol{W})} \right]$$

- Relativno je jednostavan zahvaljujući
  - Linearnoj ovisnosti logaritma vjerojatnosti o energiji stanja
  - Linearnoj ovisnosti energije o težinama

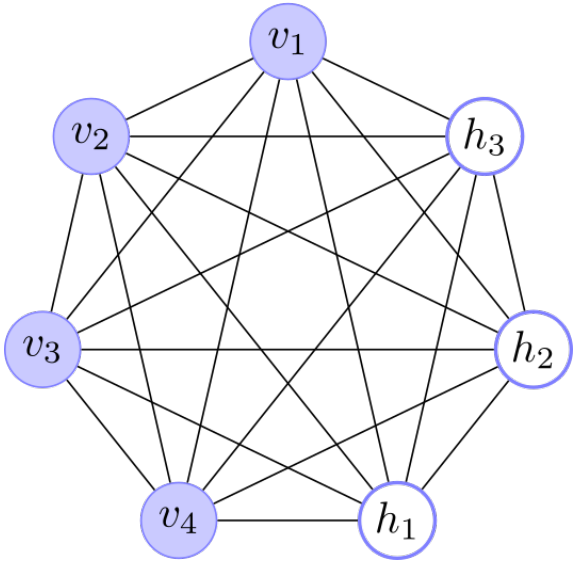
$$P(\boldsymbol{v};\boldsymbol{W}) = \sum_{\boldsymbol{h}} P(\boldsymbol{v},\boldsymbol{h};\boldsymbol{W}) = \sum_{\boldsymbol{h}} \frac{e^{-E(\boldsymbol{x})}}{Z(\boldsymbol{W})} = \sum_{\boldsymbol{h}} \frac{e^{\frac{1}{2}\boldsymbol{x}^T \boldsymbol{W} \boldsymbol{x}}}{Z(\boldsymbol{W})}$$

Skriveni neuroni

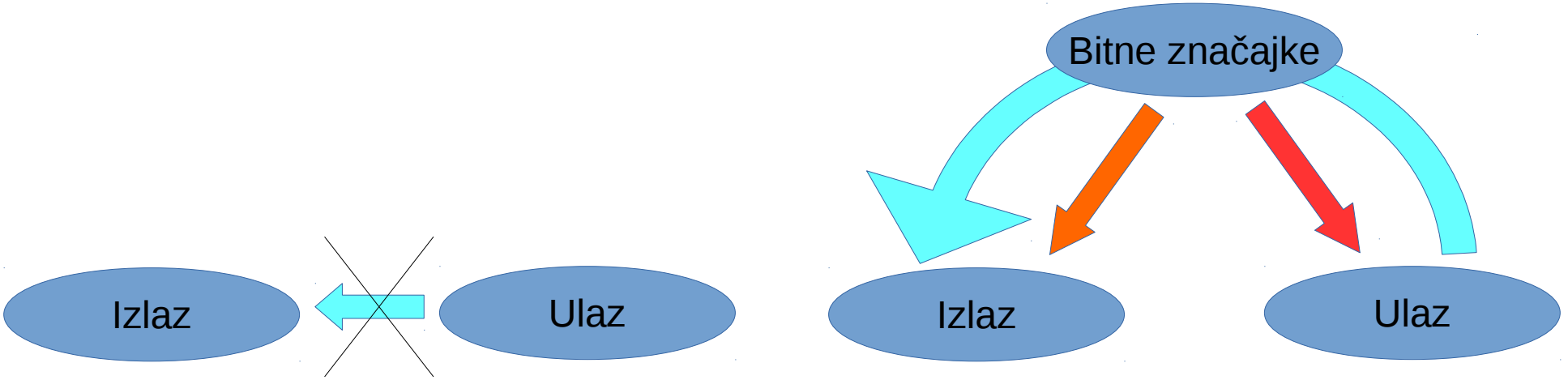
- Obje komponente estimiraju se pomoću Monte Carlo metode
  - Dugotrajno
  - Simulacije potrebno napraviti za svaki korak

$$\Delta w_{ij} = \eta \left[\langle x_i x_j \rangle_{P(\boldsymbol{h}|\boldsymbol{v}^{(n)};\boldsymbol{W})} - \langle x_i x_j \rangle_{P(\boldsymbol{v},\boldsymbol{h};\boldsymbol{W})} \right]$$

- Skriveni sloj preuzima ulogu detektora značajki i omogućuje modeliranje kompleksnih povezanosti



Bitne značajke



Energetska funkcija

- Određena je binarnim stanjima neurona i težinama koje ih povezuju
- Jednadžba sa međusobnim vezama skrivenih i međusobnim vezama vidljivih

$$E(\boldsymbol{v}, \boldsymbol{h}) = -\boldsymbol{v}^T \boldsymbol{R} \boldsymbol{v} - \boldsymbol{v}^T \boldsymbol{W} \boldsymbol{h} - \boldsymbol{h}^T \boldsymbol{S} \boldsymbol{h}$$

- Dvije komponente odbacujemo kod RBM

Treniranje Boltzmanovog stroja

- Izraz za korekciju težina

$$\Delta w_{ij} = \eta \left[\langle x_i x_j \rangle_{P(\boldsymbol{h}|\boldsymbol{v}^{(n)};\boldsymbol{W})} - \langle x_i x_j \rangle_{P(\boldsymbol{v},\boldsymbol{h};\boldsymbol{W})} \right]$$

- Simulirano hlađenje kombinirano sa gradient descent
- Lokalnog je karaktera – određivanje težina ovisi samo o neuronima koje ona povezuje (korelacije)
  - Moguća paralelizacija kada ne bi svi bili međusobno povezani
  - Efikasno, praktično – prihvatljiva interpretacija bioloških mreža
- Faktor učenja mora biti malen zbog
  - Šuma stohastičkog uzorkovanja
  - Uvest ćemo još neke aproksimacije
    - To će nas malo udaljiti od cilja – maksimizacije p(v)

Treniranje Boltzmanovog stroja

k = broj Gibbsovih koraka za doseg ekvilibrija

dok nema konvergencije

uzorkuj minibatch $\{v^{(1)}, \ldots, v^{(m)}\}$ iz skupa za treniranje

\+

inicijaliziraj uzorke $\{h^{(1)}, \ldots, h^{(m)}\}$, $x^{(n)} = [v^{(n)}, h^{(n)}]$

for n = 1 do m

for i = 1 do k

$h^{(n)} \leftarrow$ gibbs_update($h^{(n)} \mid v^{(n)}$)

$g_{ij} \leftarrow 1/m \sum x_i^{(n)} x_j^{(n)}$

\-

inicijaliziraj m slučajnih uzoraka $\{\tilde{x}(1), \ldots, \tilde{x}(m)\}$

for n = 1 do m

for i = 1 do k

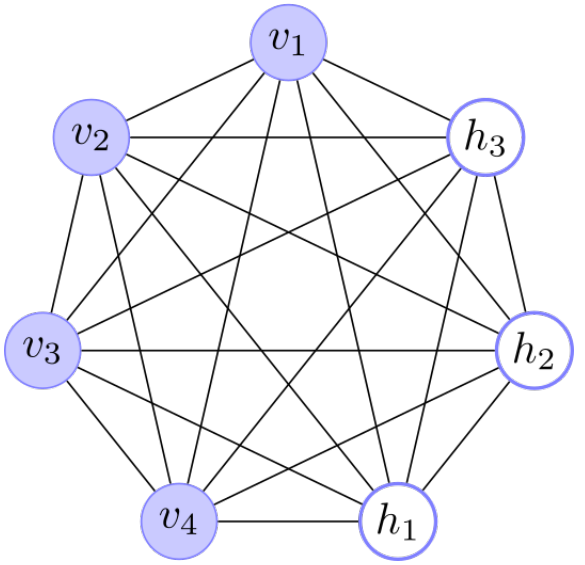
$\tilde{x}^{(n)} \leftarrow$ gibbs_update($\tilde{x}^{(n)}$)

$g_{ij} \leftarrow g_{ij} - 1/m \sum \tilde{x}_i^{(n)} \tilde{x}_j^{(n)}$

$w_{ij} = w_{ij} + \eta g_{ij}$

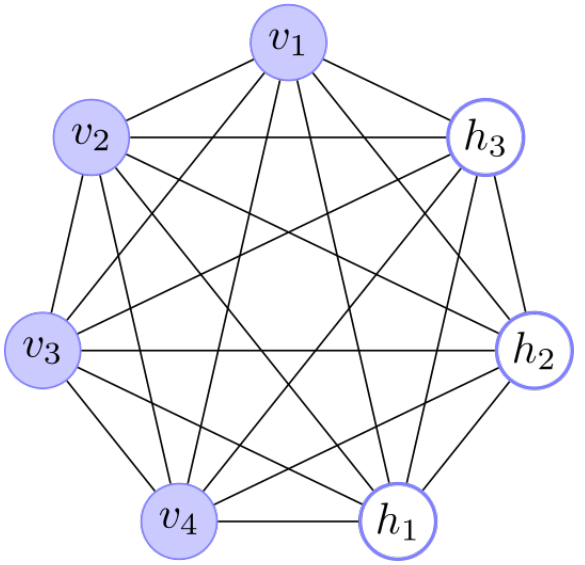
Korištenje BM

- Na vidljivi sloj fiksira se ulazni uzorak, skriveni sloj se slučajno inicijalizira
- Uzorkuju se skriveni neuroni
  - Mijenjaju se jedan po jedan u skladu s trenutnom energijom
- Nakon nekog vremena dostiže se termalni ekvilibrij
- Skriveni uzorci koji se često pojavljuju (imaju veliku vjerojatnost p(h|v)), dobro predstavljaju ulazni uzorak
  - Oni više ne ovise o početnom stanju skrivenih neurona



Korištenje BM

- Generiranje novih uzoraka
  - skriveni sloj je fiksiran – zadan ili slučajno određen
  - Vidljivi sloj nije fiksiran – MC uzorkovanje
  - Može potrajati do termalnog ekvilibrija
    - Zbog međusobnih ovisnosti neurona vidljivog sloja



Razlika u odnosu na backpropagation

- Traženje značajki
  - MLP, konvolucijske mreže traže značajke koje omogućuju dobro predviđanje oznaka
  - BM traži značajke koje dobro opisuju što se događa u skupu za treniranje
    - Ulazni vektori i oznake posljedica su toga što se događa pa ovaj pristup ima smisla
- BP ima dva posve drugačija koraka
  - Prolaz unaprijed
  - Propagacija greške unazad
- BM ima dva identična koraka koja se koriste u treniranju

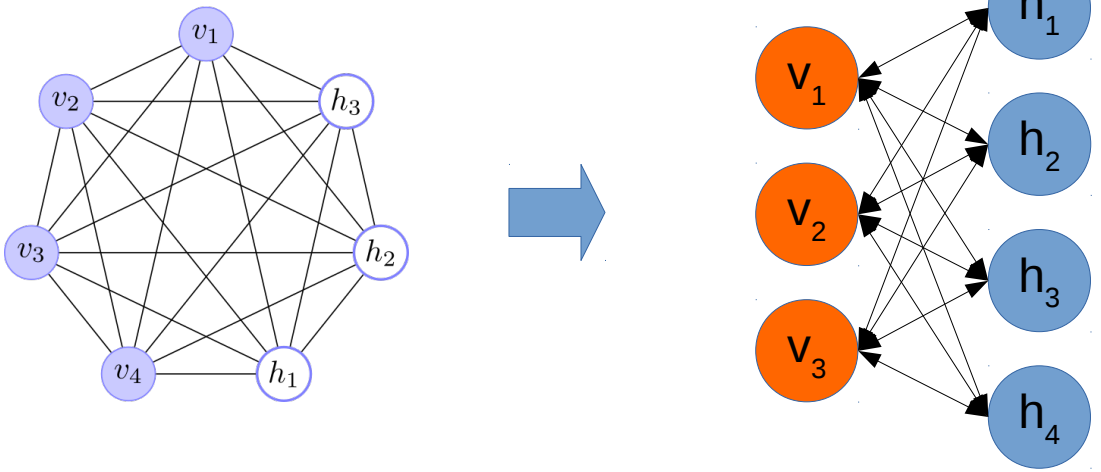
$$\Delta w_{ij} = \eta \left[\langle x_i x_j \rangle_{P(\boldsymbol{h}|\boldsymbol{v}^{(n)};\boldsymbol{W})} - \langle x_i x_j \rangle_{P(\boldsymbol{v},\boldsymbol{h};\boldsymbol{W})} \right]$$

Razlika u odnosu na backpropagation

- Lokaliziranost
  - Nova vrijednost težine u BM ovisi samo o stanjima neurona koje povezuje
  - Nova vrijednost težina kod BP ovisi o svim neuronima iz slijedećeg sloja
- Lokalizirana varijanta je jednostavnija i stoga bolji kandidat za interpretaciju bioloških neuronskih mreža

Ograničeni Boltzmanov stroj

- Izumio ga je Paul Smolensky 1986.
- Zasluge se pripisuju Geoffreyu Hintonu koji je predstavio brze algoritme učenja za RBM
- Može se trenirati pod nadzorom ili bez nadzora
- Bipartitni graf
  - Vidljivi sloj
  - Skriveni sloj



Ograničeni Boltzmanov stroj

- Svaki neuron skrivenog sloja povezan je sa svakim neuronom vidljivog sloja
- Ograničenje na međusobne povezanosti skrivenih i međusobne povezanosti vidljivih neurona
  - Povlači i ograničenu funkcionalnost, ali i bitnu prednost
    - Odbacujemo korelacijske veze drugog reda!
    - 7Skriveni neuroni su nezavisni uz zadani vidljivi sloj!

Ograničeni Boltzmanov stroj

- I dalje se maksimizira vjerojatnost vidljivih uzoraka iz skupa za treniranje

$$P(\boldsymbol{v}) = \frac{\sum_{\boldsymbol{h}} e^{-E(\boldsymbol{v},\boldsymbol{h})/T}}{Z}$$

$$P(\boldsymbol{v},\boldsymbol{h}) = \frac{e^{-E(\boldsymbol{v},\boldsymbol{h})/T}}{Z}$$

$$E(v,h) = -\boldsymbol{v}^T \boldsymbol{R} \boldsymbol{v} - \boldsymbol{v}^T \boldsymbol{W} \boldsymbol{h} - \boldsymbol{h}^T \boldsymbol{S} \boldsymbol{h}$$

$$E(v,h) = -\frac{1}{2} \boldsymbol{v}^T \boldsymbol{W} \boldsymbol{h}$$

$$Z = \sum_{\boldsymbol{v}} \sum_{\boldsymbol{h}} e^{-E(\boldsymbol{v},\boldsymbol{h})}$$

RBM particijska funkcija

$$P(\boldsymbol{v},\boldsymbol{h})=\frac{e^{-E(\boldsymbol{v},\boldsymbol{h})/T}}{Z} \qquad Z=\sum_{\boldsymbol{v}}\sum_{\boldsymbol{h}} e^{-E(\boldsymbol{v},\boldsymbol{h})}$$

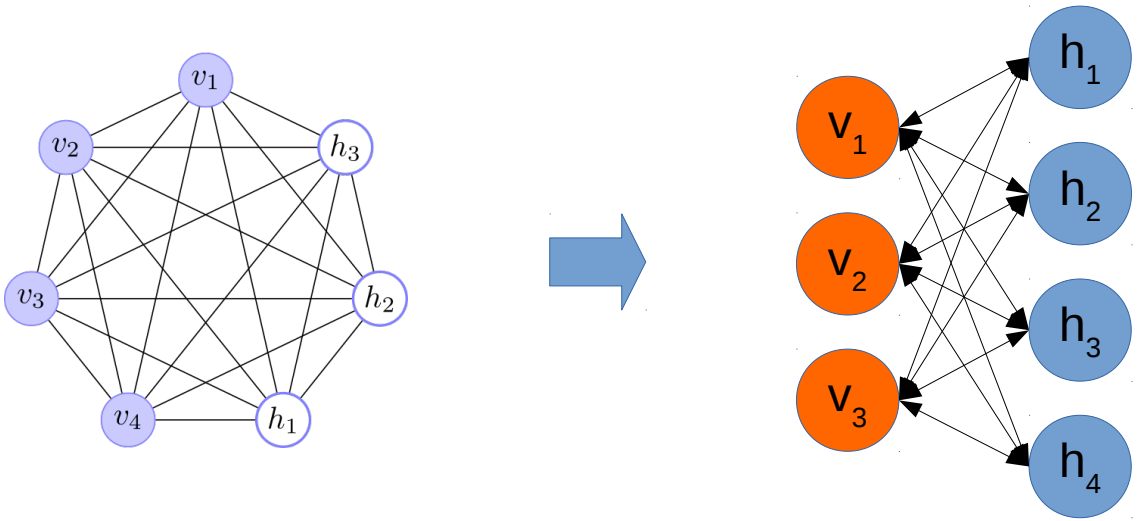
- Njena svrha je normalizirati vjerojatnosti

$$\sum_{\boldsymbol{x}} P(\boldsymbol{x})=1$$

- Particijska funkcija BM-a postaje prekompleksna za praktični izračun s povećanjem broja neurona
- Stoga nije moguće izračunati ni vjerojatnost vidljivog uzorka P(**v**)
- Potrebna je mala pomoć koju donosi sama struktura RBM

RBM partition function

- p(**h**|**v**) i p(**v**|**h**) se mogu faktorizirati, jednostavno se mogu uzorkovati i izračunati
  - To je posljedica strukture RBM-a i odabrane energetske funkcije
  - Izbjegavamo potrebu za puno uzorkovanja



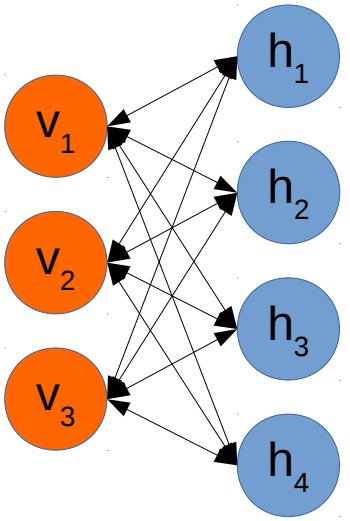
Treniranje RBM-a

- Trenira se isto kao i običan BM

$$\Delta w_{ij} = \eta \left[\langle x_i x_j \rangle_{P(\boldsymbol{h}|\boldsymbol{v}^{(n)};\boldsymbol{W})} - \langle x_i x_j \rangle_{P(\boldsymbol{v},\boldsymbol{h};\boldsymbol{W})} \right]$$

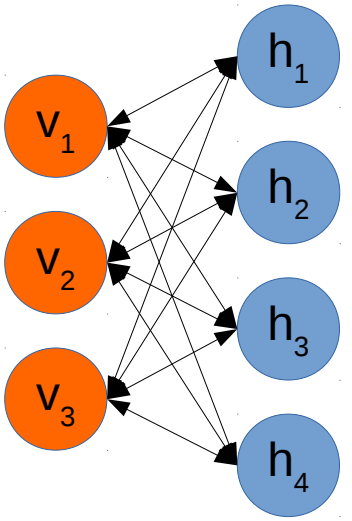
$$\Delta w_{ij} = \eta \left[\langle v_i h_j \rangle_{P(\boldsymbol{h}|\boldsymbol{v}^{(n)};\boldsymbol{W})} - \langle v_i h_j \rangle_{P(\boldsymbol{v},\boldsymbol{h};\boldsymbol{W})} \right]$$

- Razlika u arhitekturi omogućuje jednostavnije treniranje pozitivne faze
- Negativna faza je još uvijek problem



Pozitivna faza

- Vidljivi sloj je vezan (fixiran) na ulazne uzorke
- Odgovarajući skriveni sloj određuje se u jednoj iteraciji jer nema međusobnih veza u skrivenom sloju (ograničenje kod RBM)
- "Instantni termalni ekvilibrij!!"
- Može se interpretirati kao prilagođavanje distribucije modela vidljivim uzorcima za treniranje
  - Maksimizira se brojnik p(v)



Negativna faza

- Ispravan način povećava energije svih vjerojatnih stanja
  - Nećemo ga baš slijediti u potpunosti...
- Minimizira se nazivnik p(v)
  - Suprotn efekt od pozitivne faze

$$P(\boldsymbol{v}) = \frac{\sum_{\boldsymbol{h}} e^{-E(\boldsymbol{v},\boldsymbol{h})/T}}{\sum_{\boldsymbol{v}}\sum_{\boldsymbol{h}} e^{-E(\boldsymbol{v},\boldsymbol{h})/T}}$$

Negativna faza

- Uzorkovanje distirbucije modela
  - Onu u što model vjeruje
- Pretpostavka je da je model u krivu pa to treba zaboraviti
  - To je točno za lažna stanja
  - Nije istina za stanja čiji vidljivi sloj odgovara uzorcima za treniranje
  - Konvergencija nastaje kada su pozitivna i negativna faza u ravnoteži
- MCMC sa slučajnom inicijalizacijom -> termalni ekvilibrij
  - nije praktično
- Moguće su aproksimacije ali one neće povećavati energiju tamo gdje treba
  - Možda će ipak dobro funkcionirati

Neefikasno treniranje RBM

k = broj Gibbsovih koraka za doseg ekvilibrija

dok nema konvergencije

uzorkuj minibatch $\{v^{(1)}, \ldots, v^{(m)}\}$ iz skupa za treniranje

\+

$h^{(j)} \leftarrow$ gibbs_update($h^{(n)} \mid v^{(n)}$)

$g_{ij} \leftarrow 1/m \sum x_i^{(n)} x_j^{(n)}$

\-

inicijaliziraj m slučajnih uzoraka $\{\tilde{x}(1), \ldots, \tilde{x}(m)\}$

for n = 1 do m

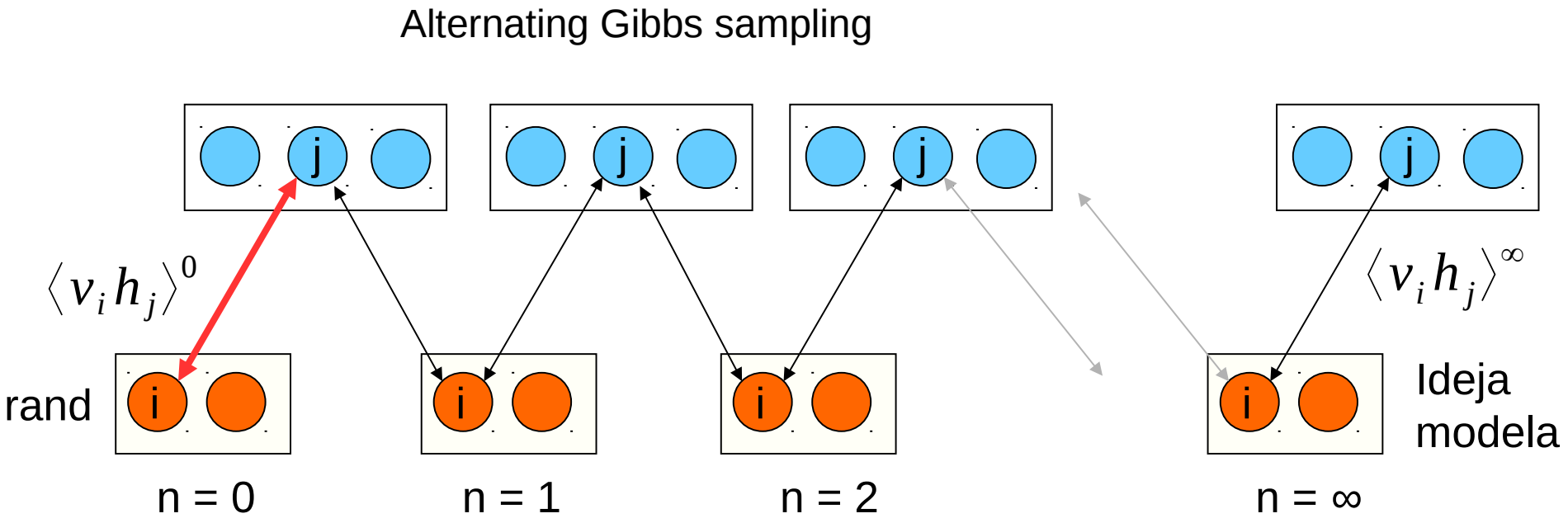
for i = 1 do k

$\tilde{x}^{(n)} \leftarrow$ gibbs_update($\tilde{x}^{(n)}$)

$g_{ij} \leftarrow g_{ij} - 1/m \sum \tilde{x}_i^{(n)} \tilde{x}_j^{(n)}$

$w_{ij} = w_{ij} + \eta g_{ij}$

Neefikasno treniranje RBM

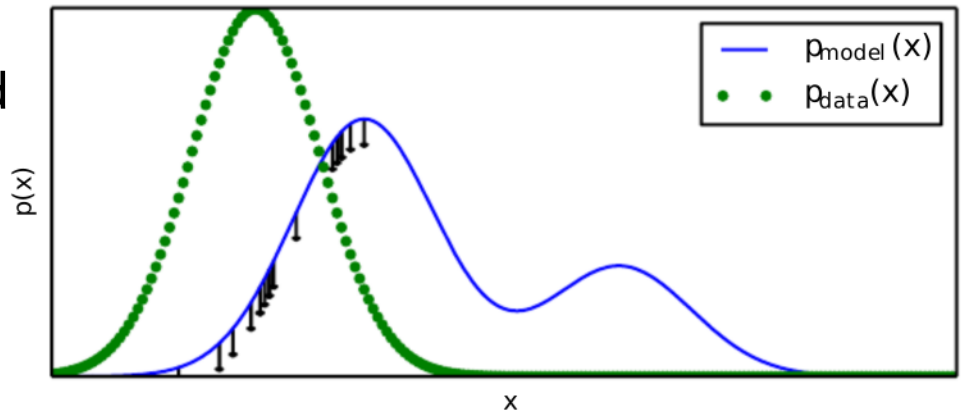


$$\Delta w_{ij} = \eta \left[\langle v_i h_j \rangle_{P(\boldsymbol{h}|\boldsymbol{v}^{(n)};\boldsymbol{W})} - \langle v_i h_j \rangle_{P(\boldsymbol{v},\boldsymbol{h};\boldsymbol{W})} \right]$$

$$\Delta w_{ij} = \eta \left[\langle v_i h_j \rangle^0 - \langle v_i h_j \rangle^\infty \right]$$

Contrastive divergence (CD)

- Modifikacija u negativnoj fazi
- MCMC se inicijaliziraju sa ulaznim uzorcima
  - Radi se manji broj (*k*) Gibsovih uzorkovanja (CD-*k)*
  - Što veći *k*, manje pristrana estimacija gradijenta
    - k=1 je obično dobro
  - To nije ispravno uzorkovanja distribucije modela
- Negativna faza treba zaboravljati ono u što model vjeruje
  - Ulazni podaci nisu on u što model vjeruje
  - Pozitivna faza će to postići s vremenom pa CD postaje korisnija
  - Ne korigiraju se lažna stanja daleko od uzoraka za treniranje



Contrastive divergence (CD)

- Samo približna aproksimacija negativne faze
  - Ne postižemo gradient descent – aproksimiramo ga
  - Mreža možda neće konvergirati
- Ipak radi dosta dobro i brzo
- Zašto radi dobro
  - Nije potrebno doći do termalnog ekvilibrija
  - Dovoljno je da model odluta od ulaznog uzorka u ulaznom sloju da bi ga znali korigirati
  - Dovoljna je jedna ili par iteracija
  - Nesavršenost vodi do loše detekcije značajki
  - dodatne informacije propuštaju se u više slojeve kod dubokih mreža?

Contrastive divergence (CD)

k = manji broj Gibbsovih koraka nedostatan za doseg ekvilibrija

dok nema konvergencije

uzorkuj minibatch $\{v^{(1)}, \ldots, v^{(m)}\}$ iz skupa za treniranje

+

$h^{(n)} \leftarrow$ gibbs_update($h^{(n)} \mid x^{(n)}$)

$g_{ij} \leftarrow 1/m \sum x_i^{(n)} x_j^{(n)}$

-

for n = 1 do m

$\tilde{x}^{(n)} \leftarrow x^{(n)}$

for n = 1 do m

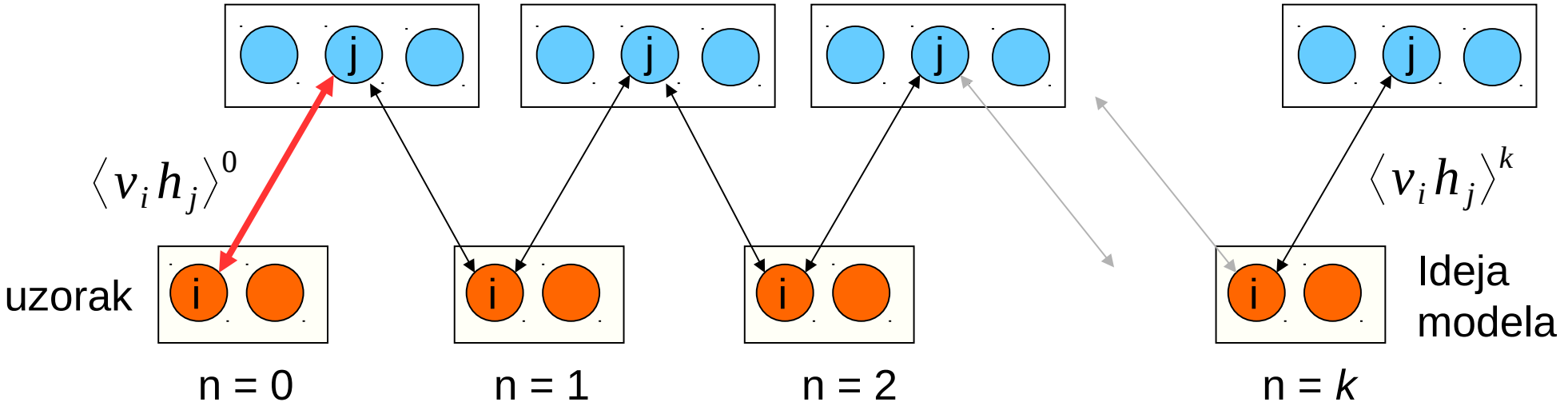
for i = 1 do k

$\tilde{x}^{(n)} \leftarrow$ gibbs_update($\tilde{x}^{(n)}$)

$g_{ij} \leftarrow g_{ij} - 1/m \sum \tilde{x}_i^{(n)} \tilde{x}_j^{(n)}$

$w_{ij} = w_{ij} + \eta g_{ij}$

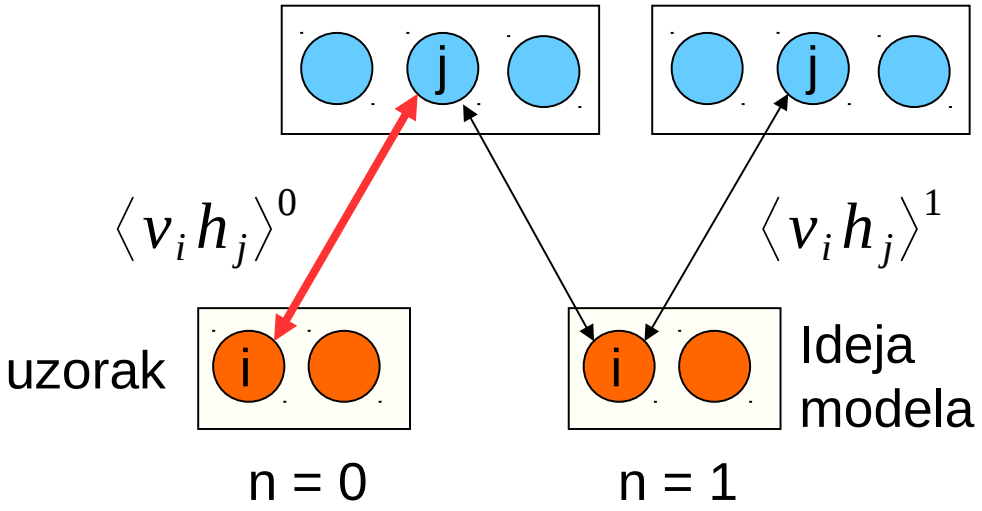
Contrastive divergence (CD-k)



$$\Delta w_{ij} = \eta \left[\langle v_i h_j \rangle_{P(\boldsymbol{h}|\boldsymbol{v}^{(n)};\boldsymbol{W})} - \langle v_i h_j \rangle_{P(\boldsymbol{v},\boldsymbol{h};\boldsymbol{W})} \right]$$

$$\Delta w_{ij} = \eta \left[\langle v_i h_j \rangle^0 - \langle v_i h_j \rangle^k \right]$$

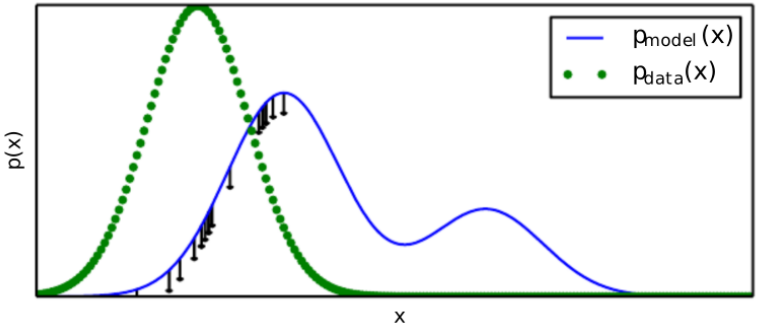
Contrastive divergence (CD-1)



$$\Delta w_{ij} = \eta \left[\langle v_i h_j \rangle_{P(\boldsymbol{h}|\boldsymbol{v}^{(n)};\boldsymbol{W})} - \langle v_i h_j \rangle_{P(\boldsymbol{v},\boldsymbol{h};\boldsymbol{W})} \right]$$

$$\Delta w_{ij} = \eta \left[\langle v_i h_j \rangle^0 - \langle v_i h_j \rangle^1 \right]$$

Poboljšanja CD



- Postepeno povećavati broj iteracija *k* (CD-*k*)
  - Cilj je doseći udaljena lažna stanja
- Persistive CD
  - Inicijalizirati trenutni korak PCD na zadnje stanje prethodnog koraka PCD (vidljiva i skrivena stanja)
  - Potrebno je pamtiti stanja između koraka (vidljiva i skrivena)
  - Između dva koraka se ne očekuje velika promjena modela pa će iz prošlog stanja MCMC brzo doći u ekvilibrij
    - Ne funkcionira dobro ako je promjena modela prebrza
    - Ne funkcionira dobro kod minibatch učenja
  - Traženje lažnih stanja nastavlja tamo gdje se prije stalo pa je moguće doći do dalekih lažnih stanja

Persistive Contrastive divergence (PCD)

k = manji broj Gibbsovih koraka nedostatan za doseg ekvilibrija

dok nema konvergencije

uzorkuj minibatch $\{v^{(1)}, \ldots, v^{(m)}\}$ iz skupa za treniranje

+

$h^{(n)} \leftarrow$ gibbs_update($h^{(n)} \mid v^{(n)}$)

$g_{ij} \leftarrow 1/m \sum x_i^{(n)} x_j^{(n)}$

-

if $\tilde{x}(v^{(n)})$ prazan

$\tilde{x}(v^{(n)}) \leftarrow x^{(n)}$ #samo prvi puta

for n = 1 do m

for i = 1 do k

$\tilde{x}(v^{(n)}) \leftarrow$ gibbs_update($\tilde{x}(v^{(n)})$)

$g_{ij} \leftarrow g_{ij} - 1/m \sum \tilde{x}_i(v^{(n)}) \tilde{x}_j(v^{(n)})$

$w_{ij} = w_{ij} + \eta g_{ij}$

Poboljšanja CD

- Mean-field aproksimacija
  - Vjerojatnost da će x_j biti 1 ovisi o preostalim stohastičkim binarnim stanjima x_i

$$p(x_j=1)=\frac{1}{1+e^{-\sum_{i\neq j} w_{ji} x_i}}$$

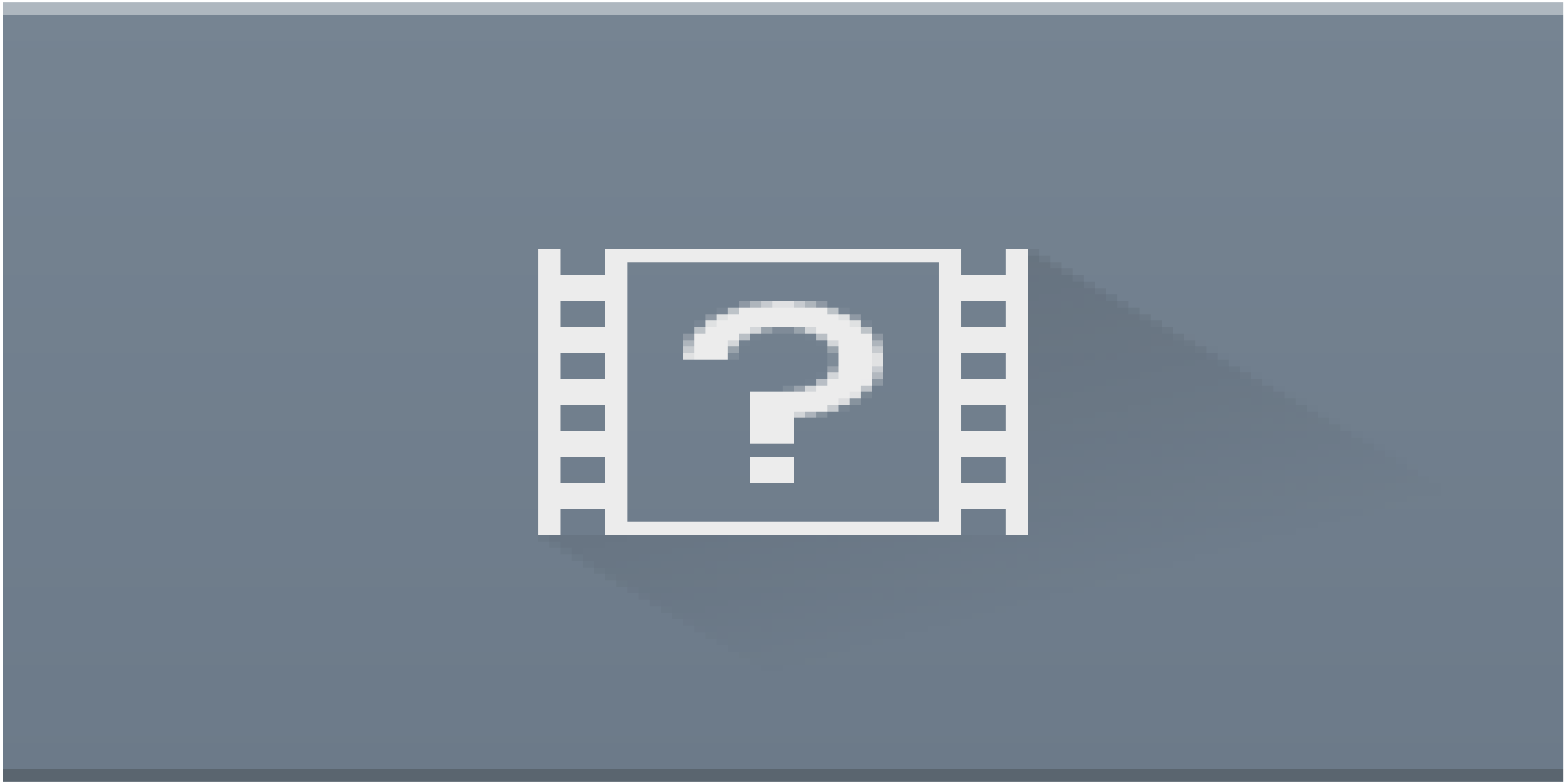
  - Umjesto stohastičkih binarnih stanja pamtimo realne iznose vjerojatnosti
    - Konvergencija (termalini ekvilibrij) se može detektirati i nastupa kada se vjerojatnosti više ne mijenjaju

$$p_j(n+1)=\frac{1}{1+e^{-\sum_{i\neq j} w_{ji} p_i(n)}}$$

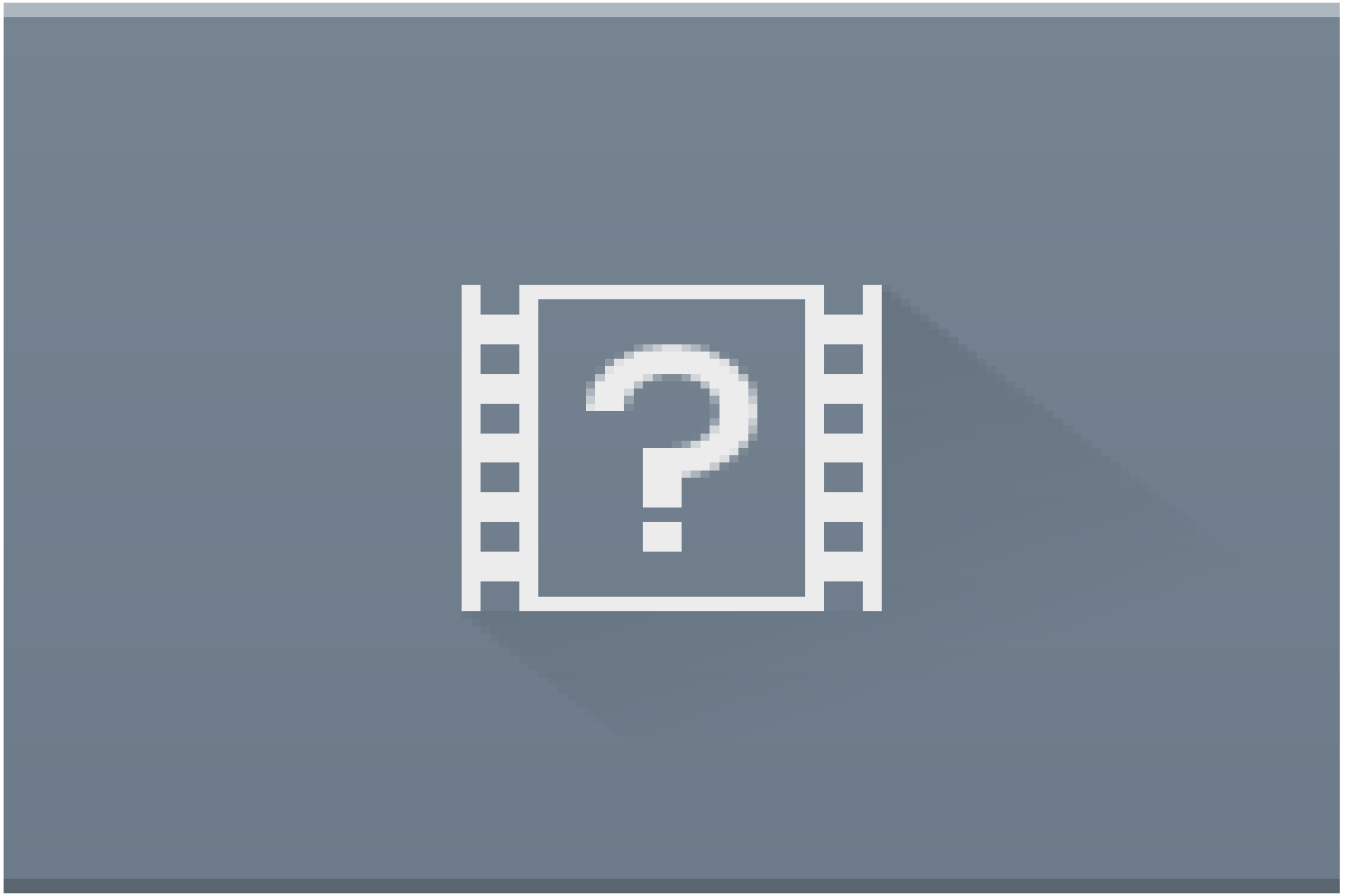
Učenje pomoću mini grupa

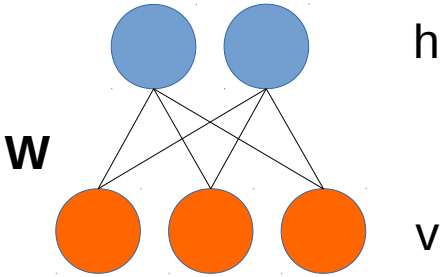
- Efikasnije učenje
- Koristi se manji dio skupa za učenje
  - Slučajni odabir
- Svaki uzorak doprinosi korekciji težina
  - Usrednjavanje korekcije
- Praktično rješenje – ubrzavanje izračuna
- Problem je što podskup uzoraka u pravilu ne predstavlja savršeno distribuciju čitavog skupa za učenje
  - Unosi se šum u postupak traženja maksimalne vjerojatnosti p(v)
  - To je zapravo dobro i korisno za MCMC!

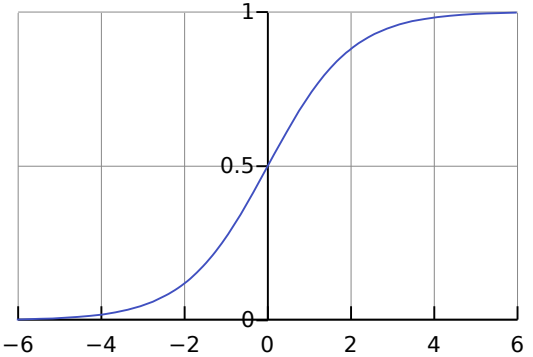
Primjer treniranja



Primjer rekonstrukcije



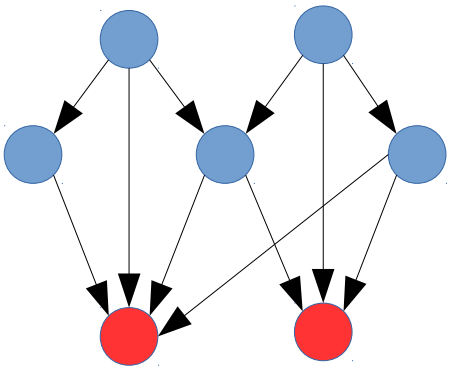




$$p(v_j = 1 | \boldsymbol{h}) = \frac{1}{1 + e^{-\sum_{i \neq j} w_{ji} h_i}}$$

Belief Nets

- Rijetko povezane mreže s usmjerenim vezama
  - Efikasni mehanizmi određivanja vjerojatnosti
- Kombinacije s generativnim modelima omogućavaju kvalitetno učenje s manjim brojem označenih uzoraka za učenje
- Rješavanje problema backporagacije u dubokim mrežama

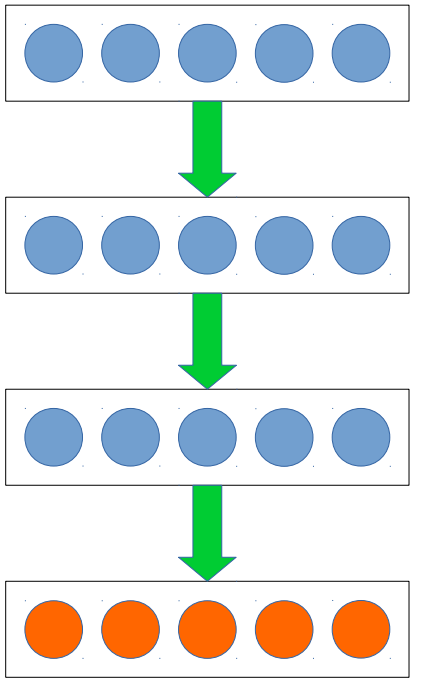


Sigmoid beleief nets

- Kauzalna beleif mreža
- Jednosmjerna mreža
- Vjerojatnost aktivacije je sigmoidna funkcija
- Binarni stohastički neuroni
- Lakše se treniraju od BM
  - Ne postoji negativna faza – ne bavimo se particijskom funkcijom
  - Maksimizira se vjerojatnost uzoraka za treniraje
    - Isto kao kod RBM maksimizira se log(p(v))

Sigmoid beleief nets

- Generiranje uzoraka
  - Stohastičko određivanje stanja najvišeg sloja
  - Slijedeći slojevi se jednostavno generiraju pomoću prethodnog sloja
  - Zadnji se generira vidljivi sloj
- Uzrokovanje onoga u što mreža vjeruje
- Pretpostavlja se da su neuroni najvišeg sloja nezavisni – iako nisu



Sigmoid beleief nets

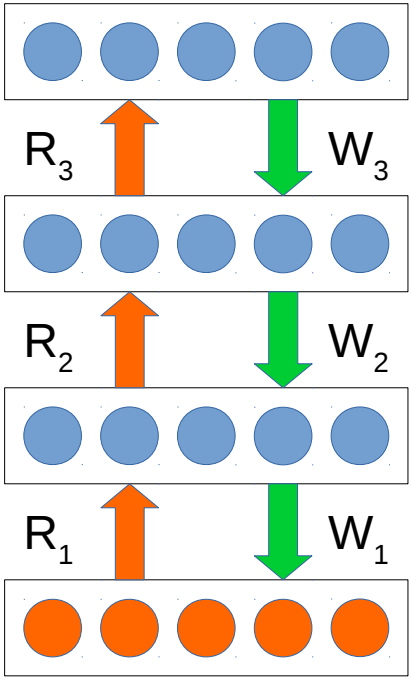
- Treniranje je nešto kompliciranije nego kod RBM-a
- Teško je odrediti p(h|v)
  - Treba nam da bi odredili generatinve težine
  - Neuroni u skrivenom sloju nisu neovisni
  - Teško je i uzimati uzorke od h uz poznati v
    - Kada bi imali uzorke, treniranje bi bilo jednostavno

Sigmoid beleief nets

- Pojednostavi kompliciranо
  - Pretpostavimo da su neuroni u jednom skrivenom sloju nezavisni
  - Koristi se "pogrešna" distribucia kao zamjena za p(h|v)
    - Varijacijska metoda
    - Uzorci h se uzimaju iz zamjenske distribucije
    - Time se podiže donja varijacijska granica log vjerojatnosti, a ne sama vjerojatnost ulaznih uzoraka
- Wake sleep algoritam

Wake – sleep

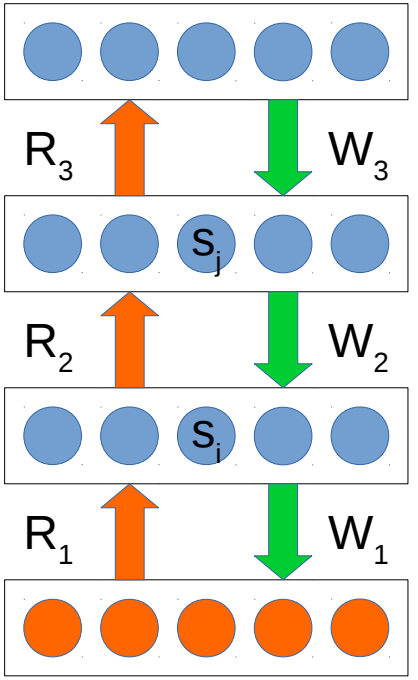
- Dva skupa težina
  - W – određuju distribuciju p(v|h)
  - R – određuju aproksimaciju p(h|v)
- Wake faza – unaprijedni prolaz
  - Određuju se uzorci u skrivenim sljevima pomoću težina R
  - Korigiraju se težine W
- Sleep faza – prolaz unazad
  - Generiraju se uzorci pomoću težina W
  - Korigiraju se težine R



Wake faza

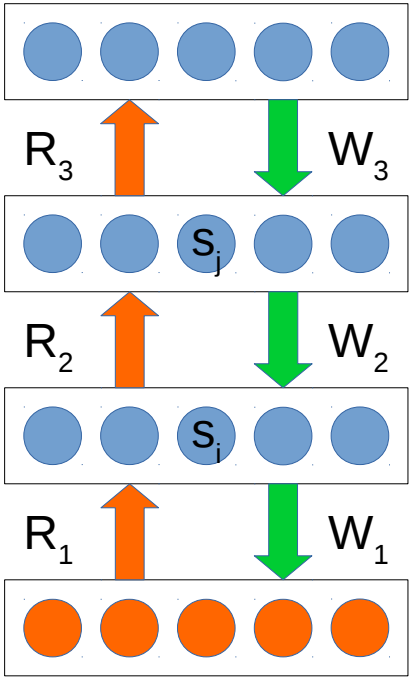
- Bottom-up
  - Kreće se od vidljivog sloja
  - Određuju se stanjanja u sloju iznad pomoću težina R
  - Korigiraju se težine W
    - Pokušava se rekonstruirati donji sloj iz gornjeg
- Korekcije se rade samo pomoću pozitivne faze

$$\Delta w_{ij} = s_j \left(s_i - p_i \right)$$



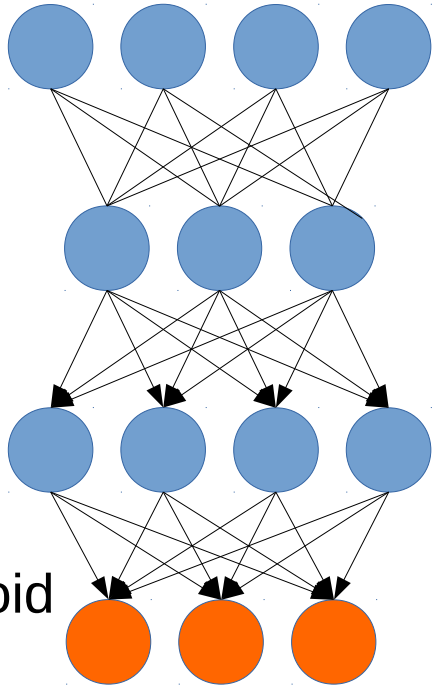
Sleep faza

- Up-down prolaz prolaz
  - Određuju se stanjanja u sloju ispod pomoću težina W
  - Korigiraju se težine R
    - Pokušava se što bolje rekonstruirati sloj iznad iz onog ispod
    - Ovo učenje nije u skladu maksimizacijom vjerojatnosti!!



$$\Delta r_{ij} = s_i \left(s_j - p_j \right)$$

Deep belief networks



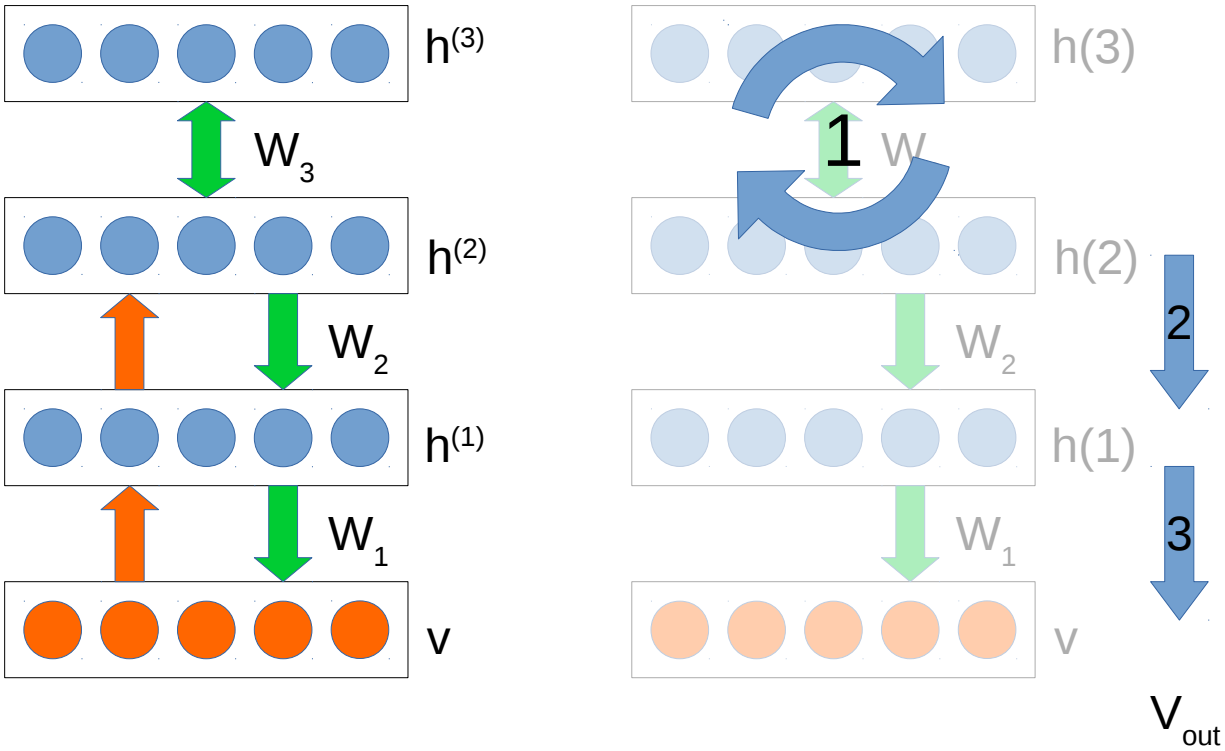
- Prvi uspješni duboki modeli izuzev konvolucijskih
- Više skrivenih slojeva
  - Binarni stohastički neuroni
  - Dva najgornja sloja imaju dvosmjerne veze – slojevi čine RBM
  - Ostale veze između slojeva su su jednosmjerne – sigmoid bleief network
- Nema povezanosti unutar slojeva
- Povezanost neurona sa svim neuronima iz susjednih slojeva
- Neuroni u vidljivom sloju mogu biti i realni (ne binarni)

Deep belief networks

- Kod generiranja uzoraka, veze između dva najviša sloja su dvosmjerne ostale su jednosmjerne – prema dolje
- Pohlepno treniranje
  - Sloj po sloj
- Dodatni slojevi poboljšavaju generatvni model
  - Poboljšavaju vjerojatnosti uzoraka iz skupa za treniranje p(v)
- Dodatni slojevi mogu poboljšati i diskriminativnost mreže
- Dovoljno je treniranje slojeva sa CD-1
  - Dobro je da niži slojevi propuštaju i nešto šuma u gornje slojeve

Generiranje uzoraka

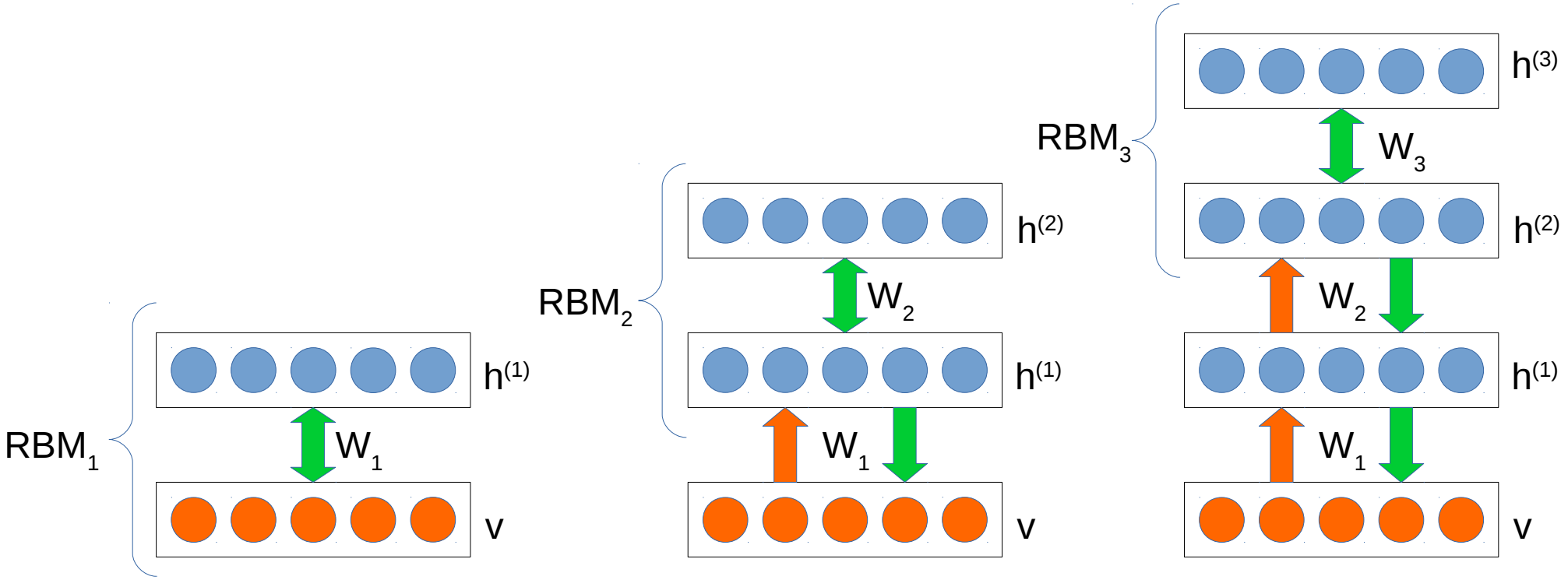
- Gibbsovo uzorkovanje između dva gornja sloja
  - Uzimanje uzorka iz gornjeg RBM-a
- Uzorkovanje prema dolje bez alternirajećeg Gibbsovog uzorkovanja – direktno do vidljivog sloja



Treniranje DBN

- Treniranje svih slojeva od jednom nije praktično
  - Dobra inicijalizacija je ključna
- Problem je u tome što više nije istina da su skriveni neuroni jednog sloja međusobno nezavisni
- Kod Sigmoid belief mreža to se jednostavno ignorira
  - Sleep wake algoritam
  - Obično dobro radi

Pohlepno treniranje sloj po sloj



Pohlepno treniranje sloj po sloj

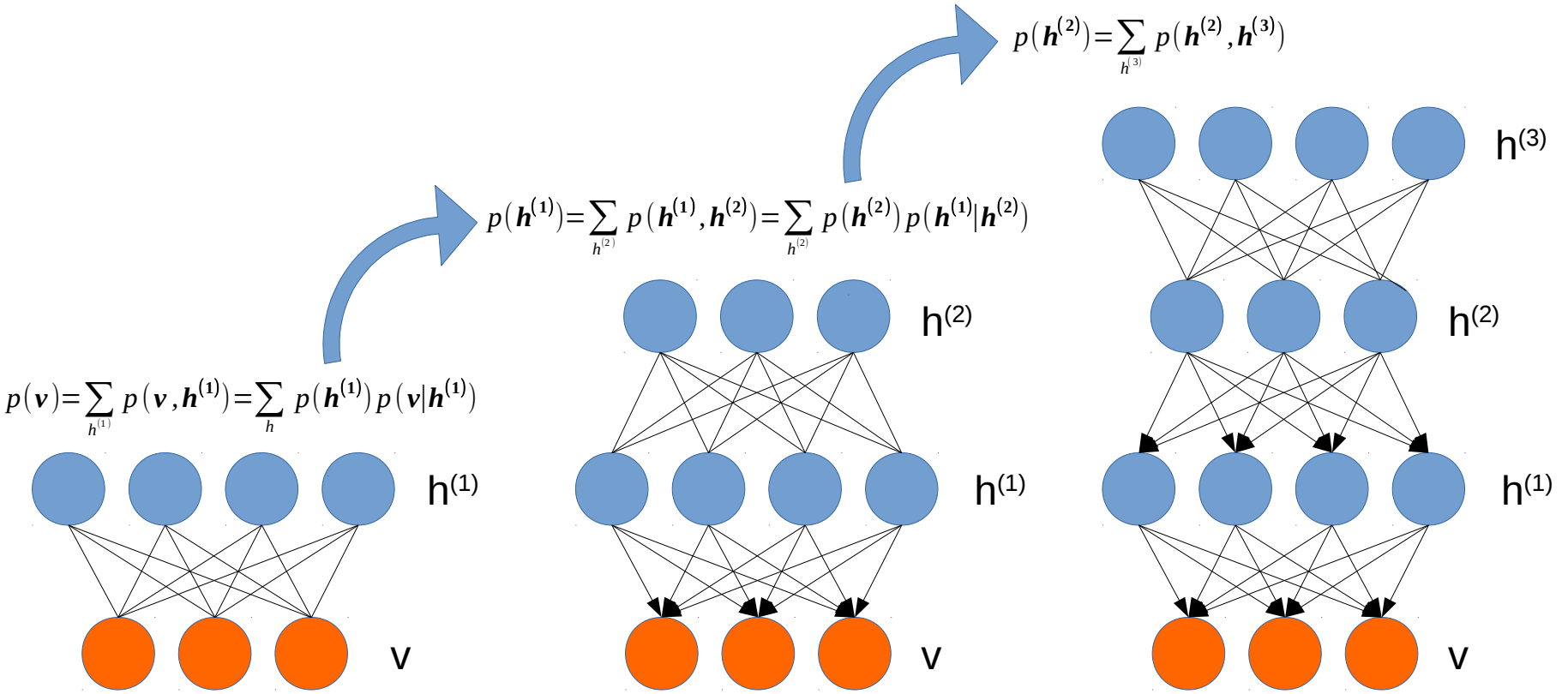
- Pronalaženje značajki na temelju prethodno pronađenih značajki
  - Značajke više razine
  - Čini se logičnim, ali...
- Nije maksimiziranje vjerojatnosti
  - Ne maksimizira se p(v)!
- Može se pokazati, u određenim uvjetima, dodavanje RBM slojeva podiže donju granicu vjerojatnosti p(v)
  - Uvjeti ne dozvoljavaju korištenje CD koji se u praksi koristi
  - Nema garancije da se doista povećava p(v)
  - Samo pokazatelj da se ipak može očekivati povećanje p(v)
  - Vjerojatno proizvodi dobre značajke za npr. klasifikaciju

Treniranje DBN

- Treniranje sloj po sloj počevši od dna
- U najnižem sloju maksimizira se vjerojatnost ulaznih uzoraka u vidljivom sloju
- U svakom slijedećem sloju maksimizira se vjerojatnost prethodnog skrivenog sloja
  - Modelira se distribucija prethodnog skrivenog sloja $p(h)$
  - Traži se bolji model prethodnog skrivenog sloja

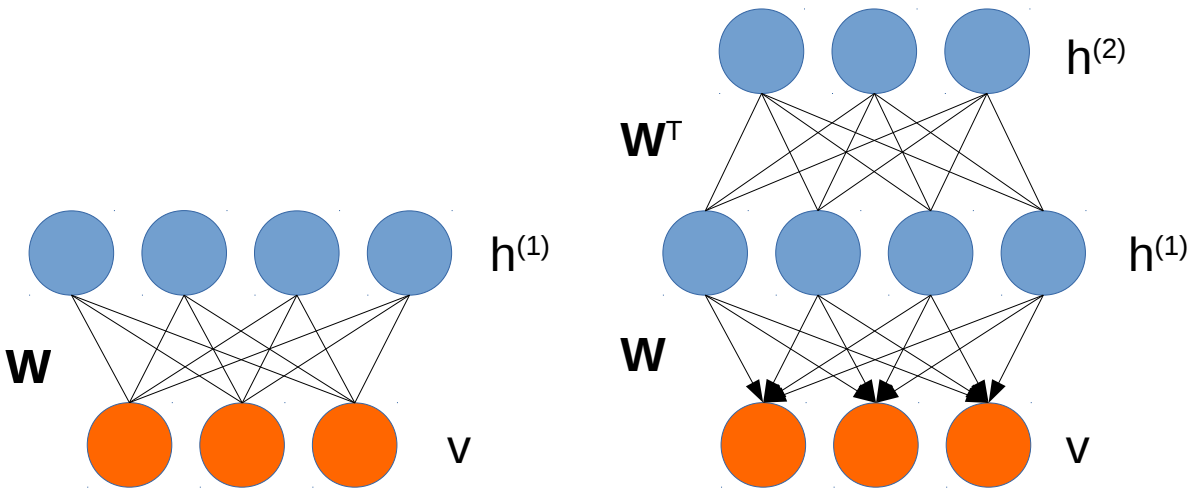
$$p(\boldsymbol{v}) = \sum_{h} p(\boldsymbol{h})\, p(\boldsymbol{v}|\boldsymbol{h})$$

Treniranje DBN



Treniranje DBN

- Kod dodavanja slijedećeg sloja, težine novog sloja se inicijaliziraju pomoću težina prethodnog sloja
  - Time se osigurava da će novi sloj poboljšati $p(h^{(1)})$, a onda i $p(v)$
  - Ako je broj novih skrivenih varijabli veći ($N_{h(2)} > N_v$), preostale težine se inicijaliziraju na 0
  - Nije nužan korak i često se ne prakticira

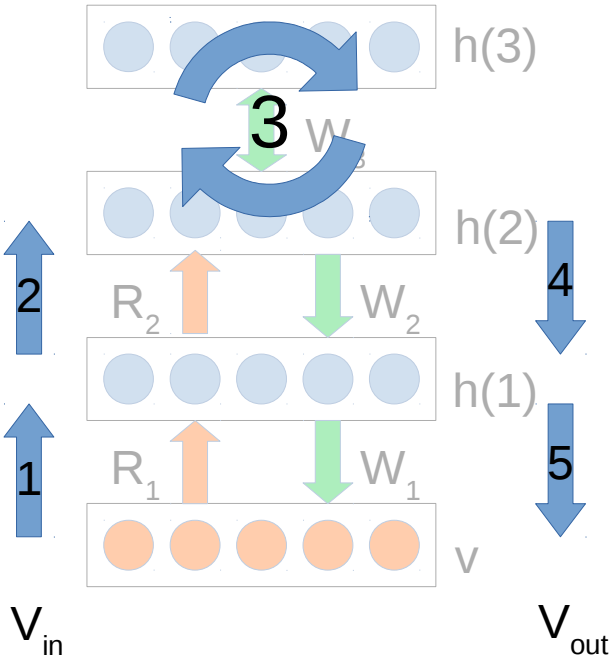


Generativni fine-tunning

- Moguće je raditi fine-tuning za bolje generiranje uzoraka – bolji model
- Nakon pohlepnog treniranja slojeva, rezultat nije optimalan p(v)
- Dvosmjerne težine pretvaraju se u dva skupa jednosmjernih težina koji se korigiraju zasebno
  - Generativne težine – generiraju podatke u sloju ispod
  - Težine prepoznavanja – generiraju podatke u sloju iznad
  - Slično kao sigmoidne duboke mreže

Generativni fine-tuning

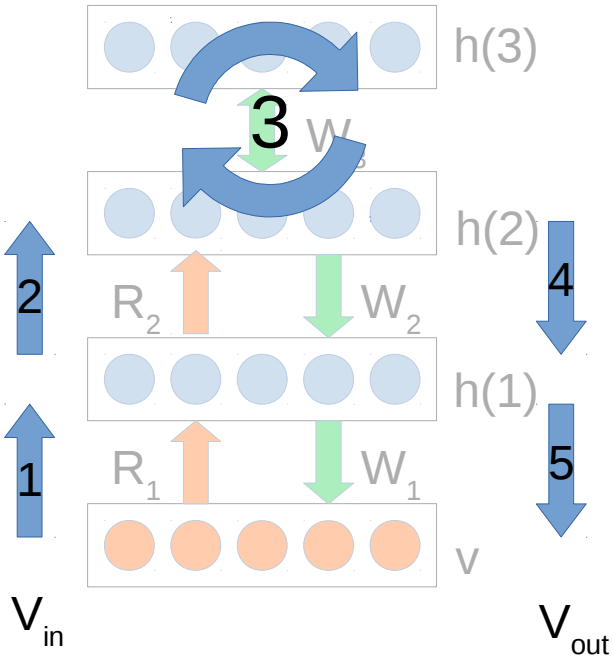
- Cuntrastive wake sleep
  1. Prolaz od dna prema vrhu (wake)
     - Modificiranje težina prema dolje (generativne težine) kako bi poboljšali rekonstrukciju u nižem sloju
  2. CD iteracije u gornjem RBM
     - Modifikacija u odnosu na Sigmoid beleief nets
     - Pruža bolji model zadnjeg skrivenog sloja
  3. Prolaz od vrha prema dnu (sleep)
     - Modifikacije težina prema gore (težine prepoznavanja) kako bi poboljšali rekonstrukciju prema gore

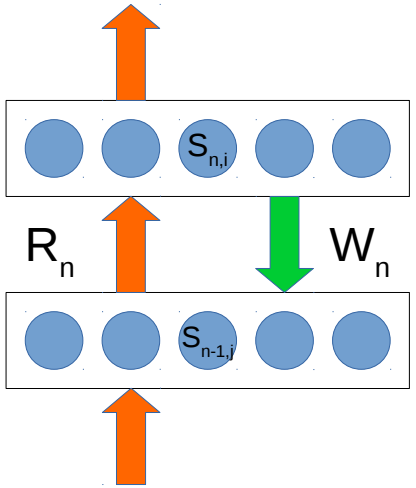


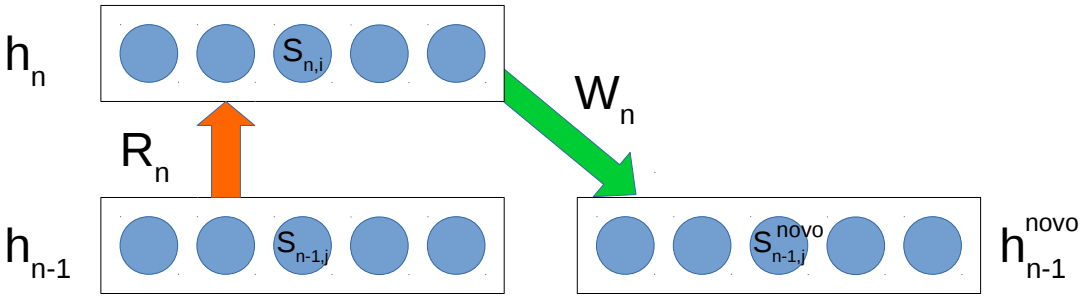
Generativni fine-tuning

- Wake faza
  - Ako je stvarnost (niži sloj) drugačija od onoga što mreža misli (rekonstrukcija na temelju sloja iznad) – modificiraj generativne težine da stvarnost postane slična mišljenju

$$\Delta w_{n,ij} = \mu s_{n,i}\left(s_{n-1,j} - s^{novo}_{n-1,j}\right) \qquad w_{n,ij} := \left(W_n\right)_{ij}$$



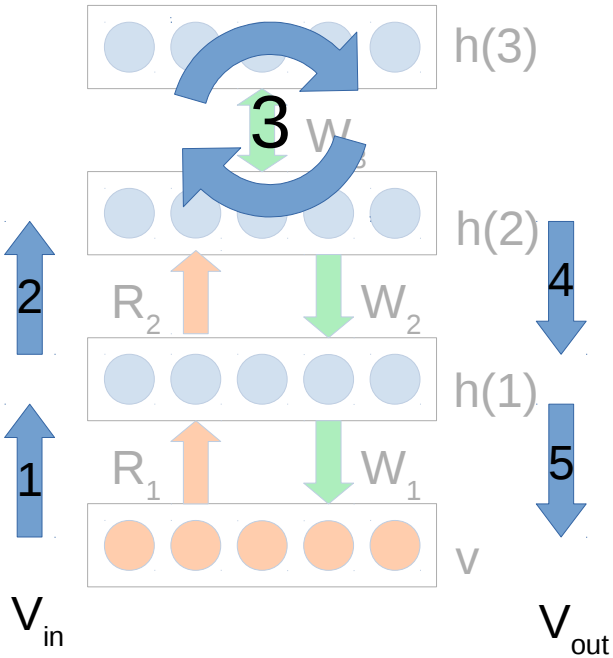


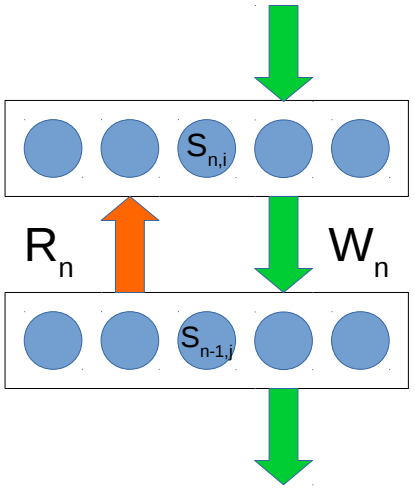


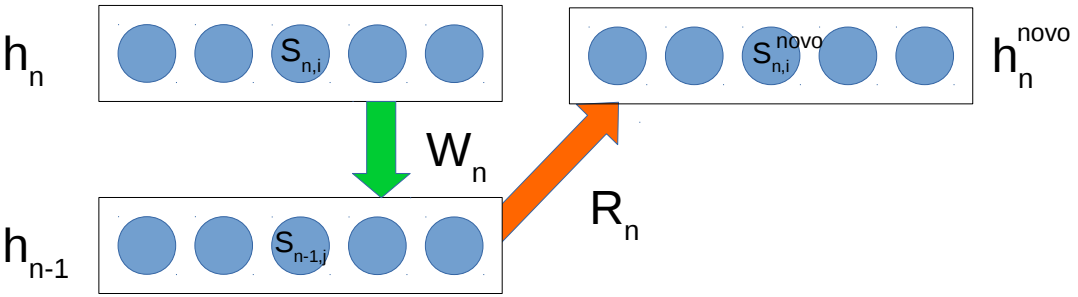
Generativni fine-tuning

- Sleep faza
  - Ako je zamišljeno drugačije od stvarnosti, modificiraj težine prepoznavanja da mišljenje postane slično stvarnosti

$$\Delta r_{n,ij} = \mu\, s_{n-1,j}\left(s_{n,i} - s^{novo}_{n,i}\right) \qquad r_{n,ij} := \left(R_n\right)_{ij}$$







Generativni fine-tuning

- Za razliku od Sigmoid beleief mreža, treniranje kreće iz vrlo dobre inicijalizacije
  - Pohlepno treniranje RBM-ova je samo predtreniranje
- Fine-tunning cijele mreže nakon čega mreža generira uzorke koji više nalikuju skupu za treniranje
- Kod predtreniranja težine u nižim slojevima nisu "svjesne" utjecaja težina u višim slojevima – to se rješava fine-tuningom
- RBM na vrhu predstavlja puno bolji model za bitne značajke
  - Bolji generativni model

Lower variational bound

$$\log(p(x)) = \sum_h q(h|x)\log(p(x))$$
$$= \sum_h q(h|x)\log\left(\frac{p(h,x)}{p(h|x)}\right)$$
$$= \sum_h q(h|x)\log\left(\frac{p(h,x)}{q(h|x)}\frac{q(h|x)}{p(h|x)}\right)$$
$$= \sum_h q(h|x)\log\left(\frac{p(h,x)}{q(h|x)}\right) + \sum_h q(h|x)\log\left(\frac{q(h|x)}{p(h|x)}\right)$$
$$= L + D_{KL}(q(h|x)\|p(h|x))$$

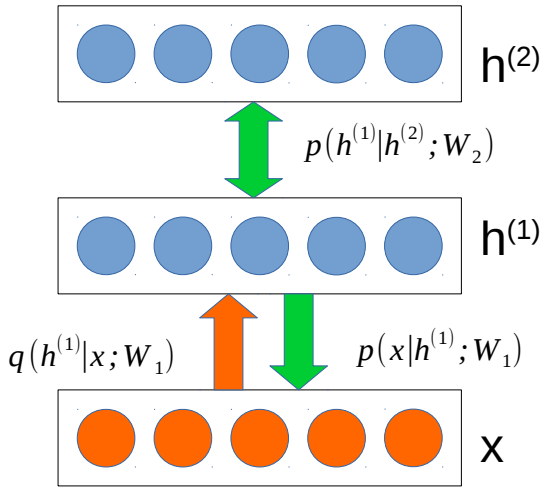
Što je q(h|x) bliže p(h|x), manji je odmak donje granice L od log(p(x)), D_{KL} = 0 (Kullback–Leibler divergencija)

$$L = \sum_h q(h|x)\log\left(\frac{p(h,x)}{q(h|x)}\right)$$
$$= \sum_h q(h|x)\log\left(\frac{p(x|h)\,p(h)}{q(h|x)}\right)$$
$$= \sum_h q(h|x)\log(p(h)) + \sum_h q(h|x)\log(p(x|h)) - \sum_h q(h|x)\log(q(h|x))$$

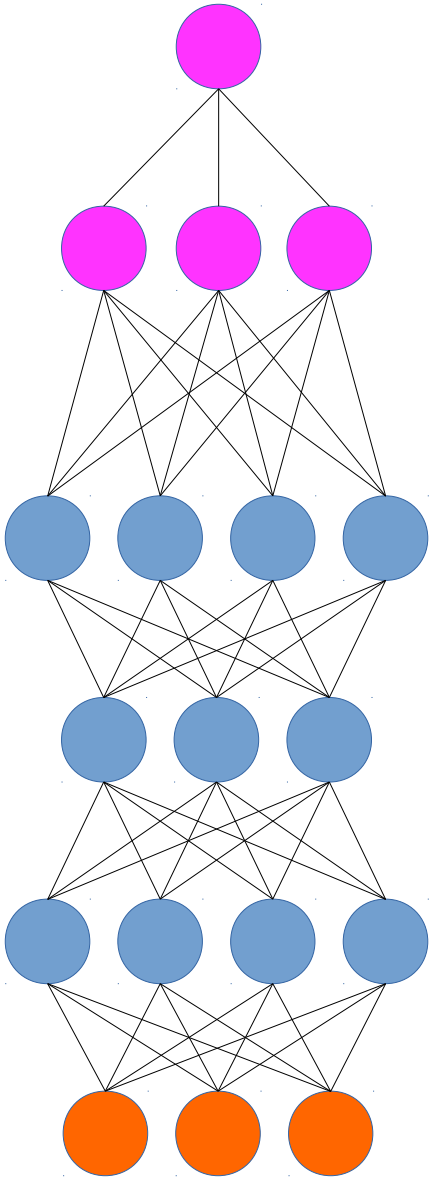
Maksimizira se ova komponenta

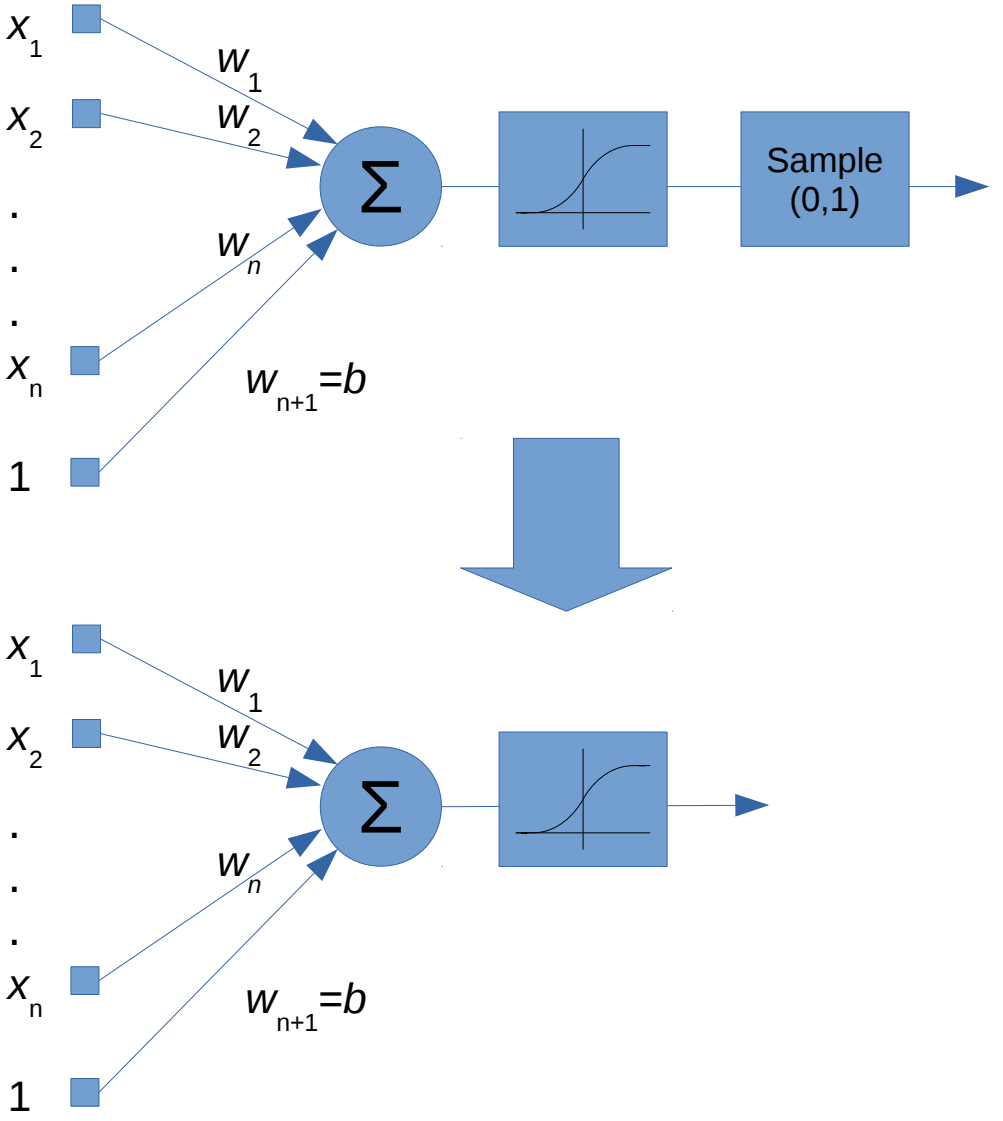
$$\sum_h q(h|x)\log(p(h))$$

Modelira se novim RBM slojem



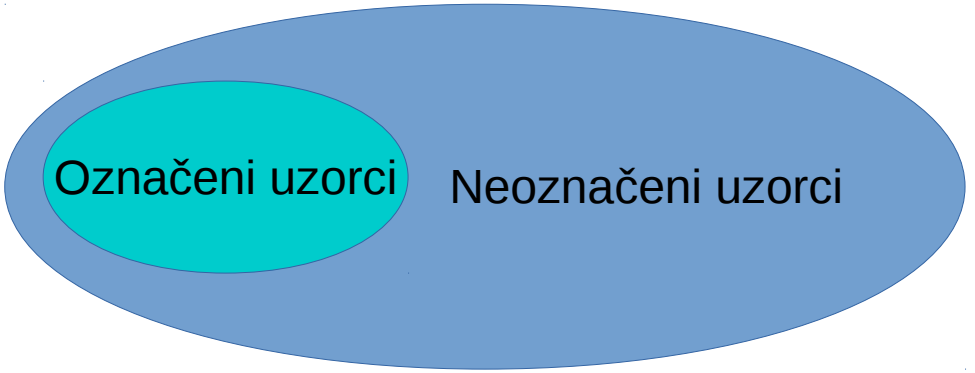
Diskriminativni fine-tunnung





Diskriminativni fine-tunnung

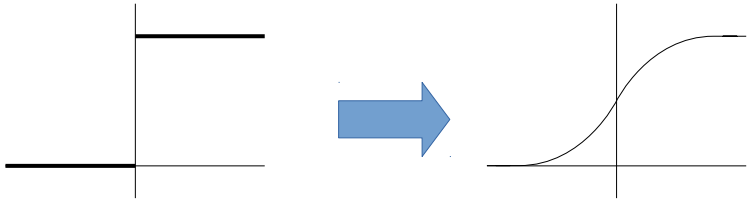
- Duboki generativni modeli se često koriste za klasifikaciju
- Naučene težine koriste se za inicijalizaciju MLP-a
  - Slijedi backpropagation pomoću označenih uzoraka za učenje
    - Nužni su označeni uzorci, ali označen može biti samo manji dio skupa za treniranje!!
    - Treniranje generativnog modela ne traži označene uzorke!!
  - Heurističko rješenje

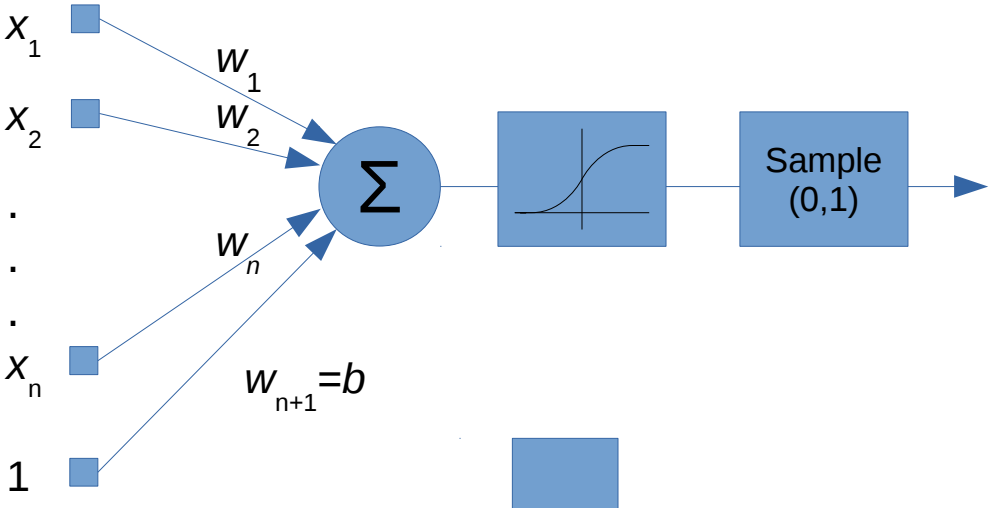


Diskriminativni fine-tunnung

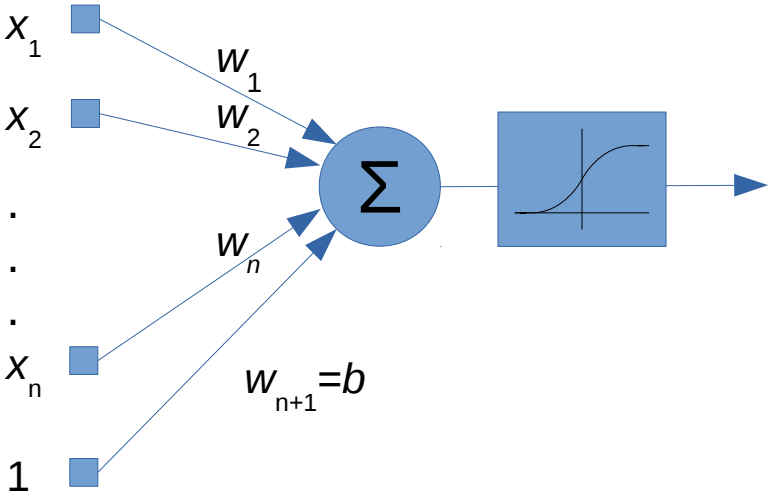
- Treniranje DBN može se gledati kao predtreniranje MLP-a
  - Koristi se za pametnu inicijalizaciju težina
    - Težine su podešene predtreniranjem na traženje dobrih značajki što bi trebalo biti korisno za klasifikaciju
  - Neke značajke neće biti korisne za klasifikaciju, ali neke hoće!
    - Te korisne značajke mogu dobiti veće težine za klasifikaciju, a kroz fine-tuning se mogu malo i poboljšati
  - MLP i backpropagation ne moraju tražiti značajke – fino podešavaju granice klasa
    - Očekuju se manje promjene težina – fine-tunning
  - Predtreniranjem se izbjegavaju uobičajeni problemi Backpropagationa na dubokoj mreži
    - Modeliranje podataka u fazi predtreniranja povećava šansu za dobru generalizaciju

Skok sa DBN na MLP?

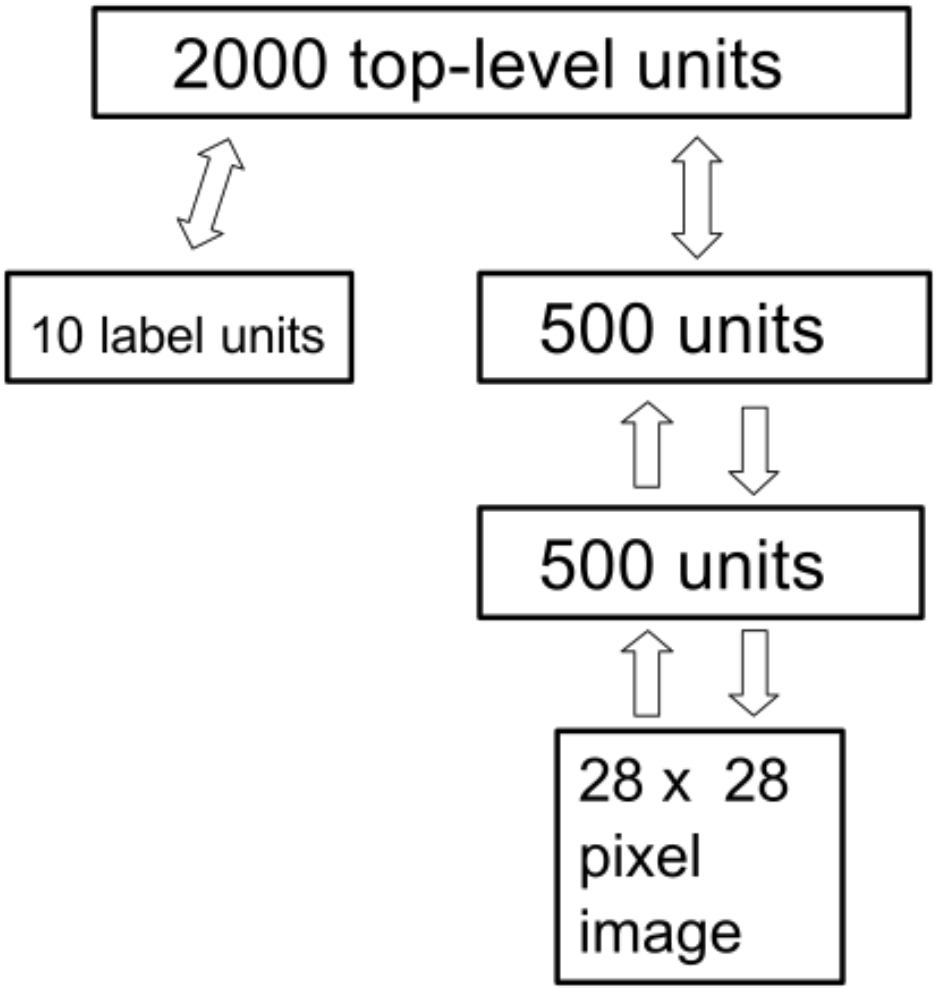




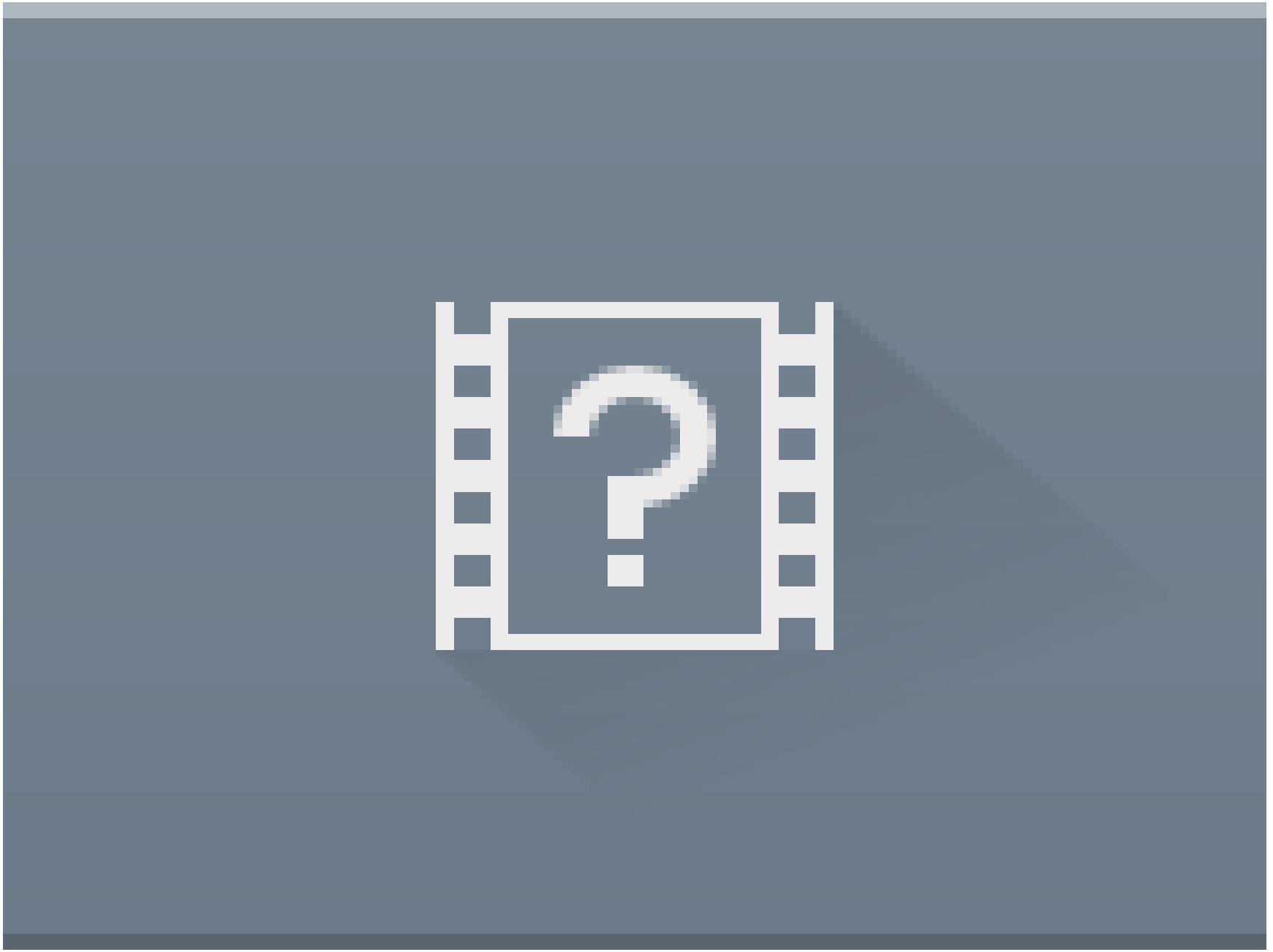
$$x = \{0,1\}$$
$$E(x) = p(x)$$



Primjer DBN - Hinton

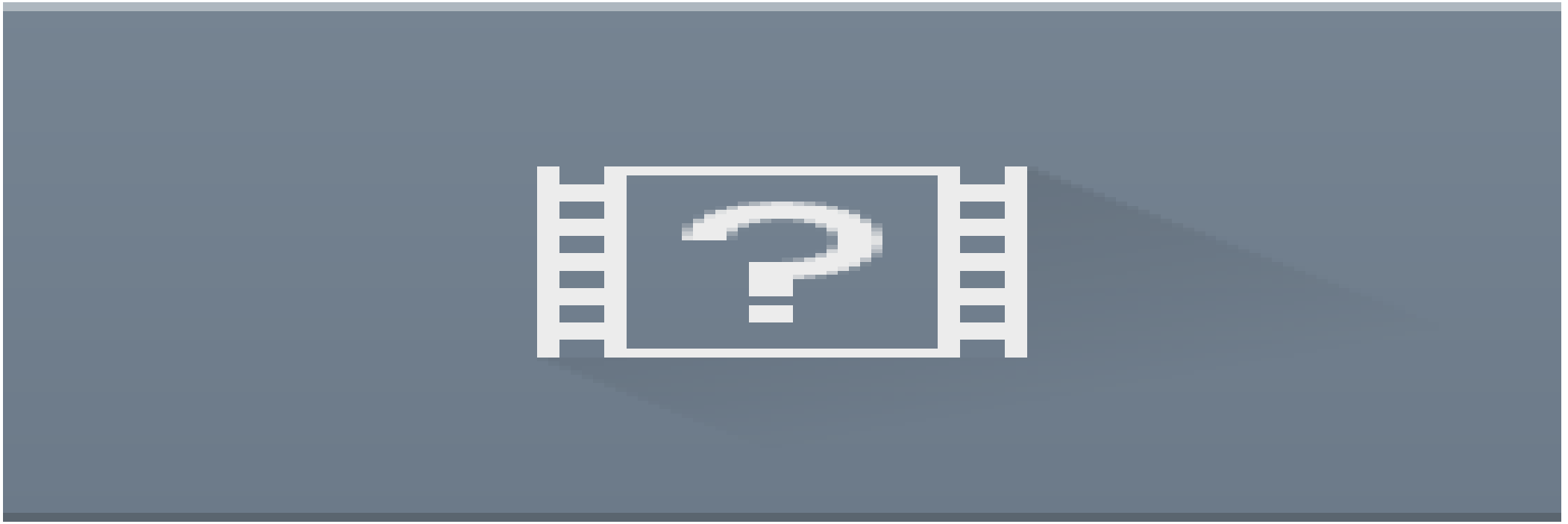


Primjer DBN - Hinton



Je li DBN bolji od RBM?

- Rekonstrukcije nakon višekratnih Gibbsovih uzorkovanja
- Neparni stupci – početne vjerojatnosti vidljivog sloja
- Parni stupci – trenutne vjerojatnosti vidljivog sloja



Kuda dalje?

- Ako razmotamo RBM dobijemo strukturu koja podsjeća na autoenkodere...