Máquina Boltzmann Restrita

Deep Learning

1. O que é a máquina Boltzmann?

A Máquina Boltzmann foi inventada pela primeira vez em 1985 por Geoffrey Hinton, um professor da Universidade de Toronto. Ele é uma figura importante na comunidade de aprendizagem profunda e é referido por alguns como o “ Padrinho da aprendizagem profunda ”.

· Boltzmann Machine é um modelo gerativo não supervisionado, que envolve aprender uma distribuição de probabilidade de um conjunto de dados original e usá-lo para fazer inferências sobre dados nunca antes vistos.

· A Máquina Boltzmann tem uma camada de entrada (também chamada de camada visível) e uma ou várias camadas ocultas (também chamada de camada oculta).

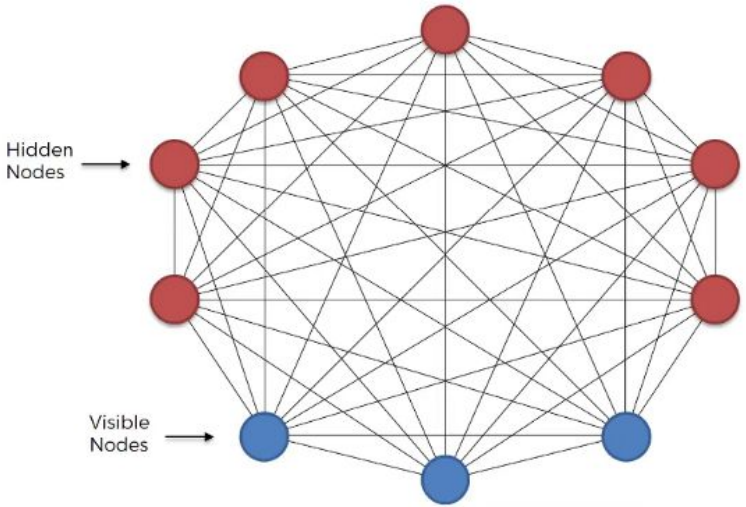
1. O que é a máquina Boltzmann?

A máquina Boltzmann é uma rede de nós conectados simetricamente

Os nós tomam decisões estocásticas para serem ativados ou desativados.

A máquina Boltzmann é um algoritmo de aprendizado de máquina não supervisionado. Ele ajuda a descobrir as características latentes presentes no conjunto de dados. O conjunto de dados é composto de vetores binários.

A conexão entre os nós não é direcionada. Cada nó na máquina Boltzmann está conectado a todos os outros nós.

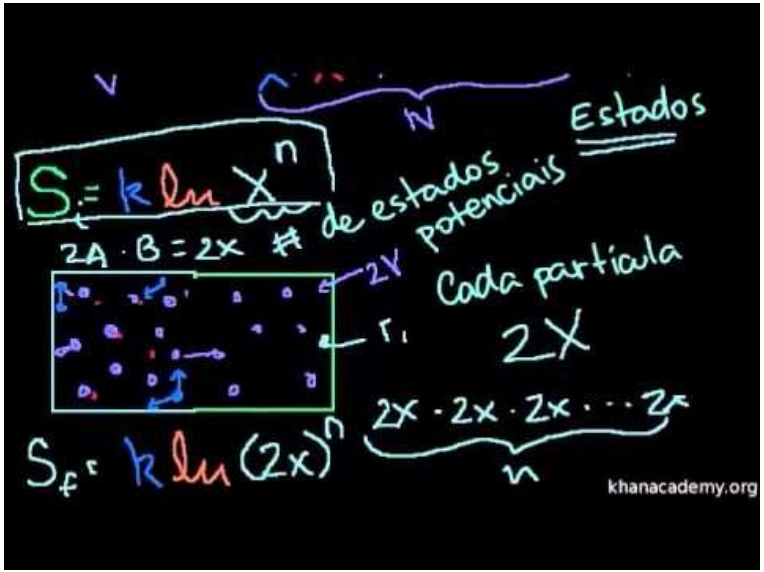


1. O que é a máquina Boltzmann?

Temos uma camada de entrada e uma camada oculta, mas nenhuma camada de saída . Isso porque cada nó é tratado da mesma forma. Todos os nós fazem parte do sistema. Cada nó gera estados do sistema e, portanto, é um modelo generativo .

Colocamos os dados na máquina Boltzmann. O modelo ajuda a aprender diferentes conexões entre nós e pesos dos parâmetros.

A máquina Boltzmann é um modelo baseado em energia.



1. O que é a máquina Boltzmann?

A máquina Boltzmann pode ser comparada a uma estufa. Em estufa, precisamos monitorar diferentes parâmetros de umidade, temperatura, fluxo de ar, luz

Como a máquina Boltzmann, a estufa é um sistema. Para a estufa, aprendemos a relação entre umidade, temperatura, luz e fluxo de ar. Compreender a relação entre diferentes parâmetros como umidade, fluxo de ar, condição do solo, etc., nos ajuda a entender o impacto no rendimento do efeito estufa.

1. O que é a máquina Boltzmann?

As máquinas Boltzmann são divididas principalmente em duas categorias: **Modelos baseados em energia (EBMs)** e **Máquinas Boltzmann restritas (RBM)** . Quando esses RBMs são empilhados uns sobre os outros, eles são conhecidos como **Deep Belief Networks (DBN)** .

2. Por que máquina de Boltzmann restrita?

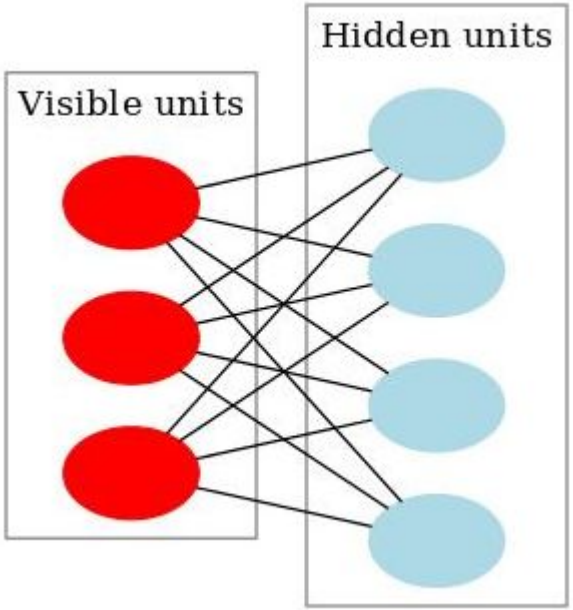
Na máquina Boltzmann, cada nó está conectado a todos os outros nós. As conexões entre todos os nós são não direcionadas. A máquina Boltzmann não tem se mostrado útil para problemas práticos de aprendizado de máquina.

A máquina Boltzmann pode se tornar eficiente colocando certas restrições. Restrições como nenhuma conexão entrelaçada tanto na camada visível quanto na camada oculta.

3. O que é a Máquina Boltzmann Restrita (RBM)?

O que torna os RBMs diferentes das máquinas Boltzmann é que os nós visíveis não estão conectados uns aos outros e os nós ocultos não estão conectados uns aos outros. Fora isso, os RBMs são exatamente iguais às máquinas Boltzmann.

Como você pode ver ao lado:



3. O que é a Máquina Boltzmann Restrita (RBM)?

- RBM é a rede neural que pertence ao modelo baseado em energia.

- É um algoritmo de aprendizado de máquina profundo probabilístico, não supervisionado e generativo.

- O objetivo do RBM é encontrar a distribuição de probabilidade conjunta que maximize a função log-verossimilhança.

- RBM não é direcionado e tem apenas duas camadas, camada de entrada e camada oculta

3. O que é a Máquina Boltzmann Restrita (RBM)?

- Todos os nós visíveis são conectados a todos os nós ocultos. O RBM tem duas camadas, camada visível ou camada de entrada e camada oculta, portanto, também é chamado de gráfico bipartido simétrico.

- Nenhuma conexão intralayer existe entre os nós visíveis. Também não há conexão intralamada entre os nós ocultos. Existem conexões apenas entre nós de entrada e ocultos.

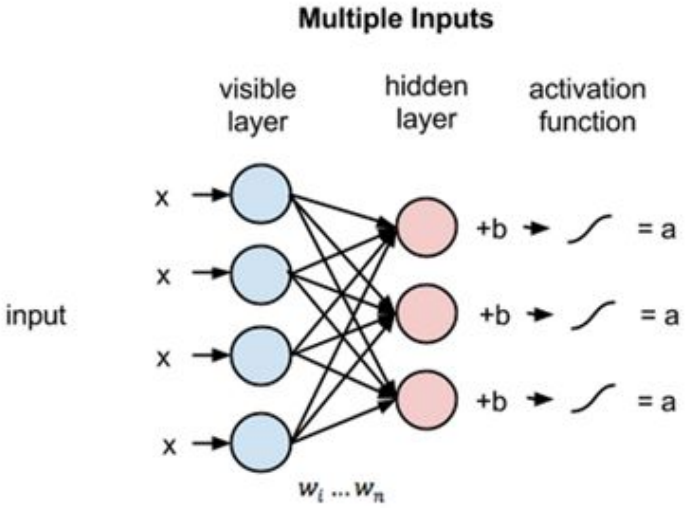
- Como os RBMs não são direcionados, eles não ajustam seus pesos por meio da descida do gradiente e da retropropagação. Eles ajustam seus pesos por meio de um processo chamado divergência contrastiva . No início deste processo, pesos para os nós visíveis são gerados aleatoriamente e usados para gerar os nós ocultos. Esses nós ocultos usam os mesmos pesos para reconstruir os nós visíveis. Os pesos usados para reconstruir os nós visíveis são os mesmos por toda parte. No entanto, os nós gerados não são os mesmos porque não estão conectados uns aos outros.

4. Como funciona a máquina Boltzmann restrita?

RBM é uma Rede Neural Estocástica, o que significa que cada neurônio terá algum comportamento aleatório quando ativado. Existem duas outras camadas de unidades de polarização (tendência oculta e tendência visível) em um RBM. Isso é o que torna os RBMs diferentes dos codificadores automáticos. A polarização oculta RBM produz a ativação no passe para frente e a polarização visível ajuda o RBM a reconstruir a entrada durante um passe para trás. A entrada reconstruída é sempre diferente da entrada real, pois não há conexões entre as unidades visíveis e, portanto, nenhuma maneira de transferir informações entre si.

4. Como funciona a máquina Boltzmann restrita?

A imagem ao lado mostra a primeira etapa no treinamento de um RBM com várias entradas. As entradas são multiplicadas pelos pesos e, em seguida, somadas ao enviesamento. O resultado é então passado por uma função de ativação sigmóide e a saída determina se o estado oculto é ativado ou não. Os pesos serão uma matriz com o número de nós de entrada como o número de linhas e o número de nós ocultos como o número de colunas. O primeiro nó oculto receberá a multiplicação vetorial das entradas multiplicadas pela primeira coluna de pesos antes que o termo de polarização correspondente seja adicionado a ele.



4. Como funciona a máquina Boltzmann restrita?

E se você está se perguntando o que é uma função sigmóide, aqui está a fórmula:

$$S(x) = \frac{1}{1 + e^{-x}} = \frac{e^{x}}{1 + e^{x}}$$

Portanto, a equação que obtemos nesta etapa seria,

$$\mathbf{h}^{(1)} = S(\mathbf{v}^{(0)T} W + \mathbf{a})$$

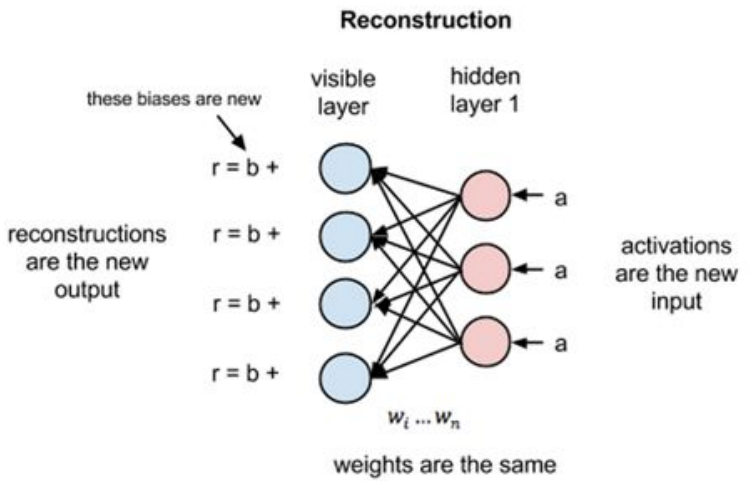
onde **h (1)** e **v (0)** são os vectores correspondentes (matrizes de coluna) para o escondido e as camadas visíveis com o sobrescrito como a iteração v (0) meios de entrada que fornecem para a rede) e **um** é o escondido vetor de polarização de camada.

4. Como funciona a máquina Boltzmann restrita?

Agora, esta imagem mostra a fase reversa ou a fase de reconstrução . É semelhante à primeira passagem, mas na direção oposta. A equação acaba sendo:

$$\mathbf{v}^{(1)} = S(\mathbf{h}^{(1)}W^T + \mathbf{b})$$

onde v (1) e h (1) são os vectores correspondentes (matrizes de coluna) para o visível e as camadas escondidas com o sobrescrito como a iteração e b é o vector de polarização camada visível.



4.1. O processo de aprendizagem

Agora, a diferença **v (0) −v (1)** pode ser considerada como o erro de reconstrução que precisamos reduzir nas etapas subsequentes do processo de treinamento. Então os pesos são ajustados a cada iteração de forma a minimizar esse erro e isso é o que essencialmente é o processo de aprendizagem. Agora, vamos tentar entender esse processo em termos matemáticos, sem nos aprofundarmos muito na matemática. Na passagem para a frente, estamos calculando a probabilidade de saída **h (1)** dada a entrada **v (0)** e os pesos **W** denotados por:

$$p(\mathbf{h}^{(1)} \mid \mathbf{v}^{(0)}; W)$$

E na passagem para trás, ao reconstruir a entrada, estamos calculando a probabilidade de saída **v (1)** dada a entrada **h (1)** e os pesos **W** denotados por:

$$p(\mathbf{v}^{(1)} \mid \mathbf{h}^{(1)}; W)$$

4.1. O processo de aprendizagem

Os pesos usados no passe para frente e para trás são os mesmos. Juntas, essas duas probabilidades condicionais nos levam à distribuição conjunta dos insumos e das ativações:

$$p(\mathbf{v}, \mathbf{h})$$

A reconstrução é diferente da regressão ou classificação porque estima a distribuição de probabilidade da entrada original em vez de associar um valor contínuo / discreto a um exemplo de entrada. Isso significa que ele está tentando adivinhar vários valores ao mesmo tempo. Isso é conhecido como aprendizado generativo, em oposição ao aprendizado discriminativo que acontece em um problema de classificação (mapeamento de entrada para rótulos).

4.1. O processo de aprendizagem

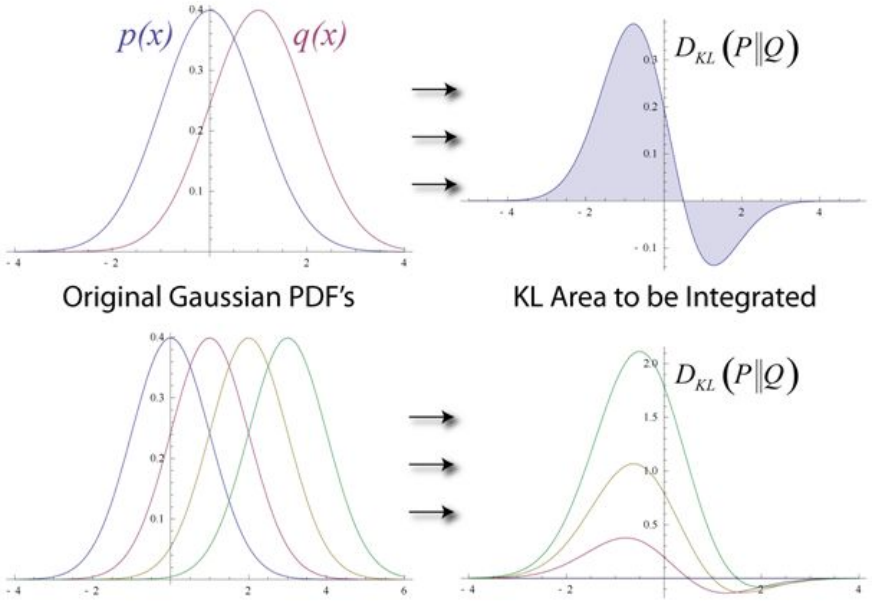
Vamos tentar ver como o algoritmo reduz a perda ou, simplesmente, como ele reduz o erro a cada etapa. Suponha que temos duas distribuições normais, uma dos dados de entrada (**denotada por p (x)**) e uma da aproximação de entrada reconstruída (**denotada por q (x)**). A diferença entre essas duas distribuições é nosso erro no sentido gráfico e nosso objetivo é minimizá-lo, ou seja, aproximar os gráficos o máximo possível. Essa ideia é representada por um termo denominado divergência de Kullback-Leibler .

$$DkL^{(p(x)\|q(x))} = P(x)\log\frac{p(x)}{q(x)}$$

4.1. O processo de aprendizagem

A divergência KL mede as áreas não sobrepostas sob os dois gráficos e o algoritmo de otimização do RBM tenta minimizar essa diferença alterando os pesos para que a reconstrução se pareça muito com a entrada. Os gráficos à direita mostram a integração da diferença nas áreas das curvas à esquerda.

Isso nos dá intuição sobre nosso termo de erro. Agora, para ver como isso realmente é feito para RBMs, teremos que mergulhar em como a perda está sendo calculada. Todos os algoritmos de treinamento comuns para RBMs aproximam o gradiente de log-verossimilhança dados alguns dados e realizam a ascensão do gradiente nessas aproximações.



5. Divergência contrastiva.

Máquinas Boltzmann (e RBMs) são modelos baseados em energia e uma configuração conjunta, **(v, h)** das unidades visíveis e ocultas tem energia dada por:

$$E(\mathbf{v}, \mathbf{h}) = -\sum_{i \in visible} a_i v_i - \sum_{j \in hidden} b_j h_j - \sum_{i,j} v_i h_j w_{ij}$$

Onde vi, hj são os estados binários da unidade visível ii e da unidade oculta j, ai, bj são seus vieses e wij é o peso entre eles.

A probabilidade de que a rede atribui a um vetor visível, v , é dada pela soma de todos os vetores ocultos possíveis:

5. Divergência contrastiva.

Z aqui é a função de partição e é dada pela soma de todos os pares possíveis de vetores visíveis e ocultos:

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

Isso nos dá:

$$p(\mathbf{v}) = \frac{\sum_{\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})}}{\sum_{\mathbf{v},\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})}}$$

5. Divergência contrastiva.

O gradiente de probabilidade logarítmica ou a derivada da probabilidade logarítmica de um vetor de treinamento em relação ao peso é surpreendentemente simples:

$$\frac{\partial log p(\mathbf{v})}{\partial w_{ij}} = \langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{model}$$

Onde os colchetes angulares são usados para denotar as expectativas sob a distribuição especificada pelo subscrito a seguir. Isso leva a uma regra de aprendizado muito simples para realizar a subida mais íngreme estocástica na probabilidade de log dos dados de treinamento:

$$\Delta w_{ij} = \alpha(\langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{model})$$

onde **α** é uma taxa de aprendizado.

5. Divergência contrastiva.

Para obter mais informações sobre o que significam as equações acima ou como são derivadas, consulte o Guia de treinamento RBM de Geoffrey Hinton . O importante a notar aqui é que, como não há conexões diretas entre unidades ocultas em um RBM, é muito fácil obter uma amostra imparcial de ⟨vihj⟩data . Obter uma amostra imparcial do modelo ⟨vihj⟩ .

Porém, é muito mais difícil. Isso ocorre porque exigiria que executássemos uma cadeia de Markov até que a distribuição estacionária fosse alcançada (o que significa que a energia da distribuição é minimizada - equilíbrio!) Para aproximar o segundo termo.

5. Divergência contrastiva.

Então, em vez de fazer isso, executamos a Amostragem de Gibbs da distribuição. É um algoritmo Markov Chain Monte Carlo (**MC**) para obter uma sequência de observações que são aproximadas de uma distribuição de probabilidade multivariada especificada quando a amostragem direta é difícil (como em nosso caso). A cadeia de Gibbs é inicializada com um exemplo de treinamento **v (0)** do conjunto de treinamento e produz a amostra **v (k)** após **k** etapas.

Cada etapa t consiste em amostrar h (t) de p (h|v (t)) e amostrar v (t + 1) de p (v|h (t)) subsequentemente (o valor k = 1 surpreendentemente funciona muito bem) .

A regra de aprendizagem agora se torna:

$$\Delta w_{ij} = \alpha(\langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{recon})$$

5. Divergência contrastiva.

O aprendizado funciona bem, embora esteja apenas aproximando grosseiramente o gradiente da probabilidade de log dos dados de treinamento. A regra de aprendizado se aproxima muito mais do gradiente de outra função objetivo chamada de divergência contrastiva, que é a diferença entre duas divergências de Kullback-Liebler.

Quando aplicamos isso, obtemos:

$$\mathbf{CD}_k(W, \mathbf{v}^{(0)}) = -\sum_{\mathbf{h}} p(\mathbf{h} \mid \mathbf{v}_k) \frac{\partial E(\mathbf{v}_k, \mathbf{h})}{\partial W} + \sum_{\mathbf{h}} p(\mathbf{h} \mid \mathbf{v}_k) \frac{\partial E(\mathbf{v}_k, \mathbf{h})}{\partial W}$$

Onde o segundo termo é obtido após cada **k** etapas da Amostragem de Gibbs.

5. Divergência contrastiva.

Aqui está o pseudocódigo para o algoritmo do **CD** :

Algorithm 1. k-step contrastive divergence

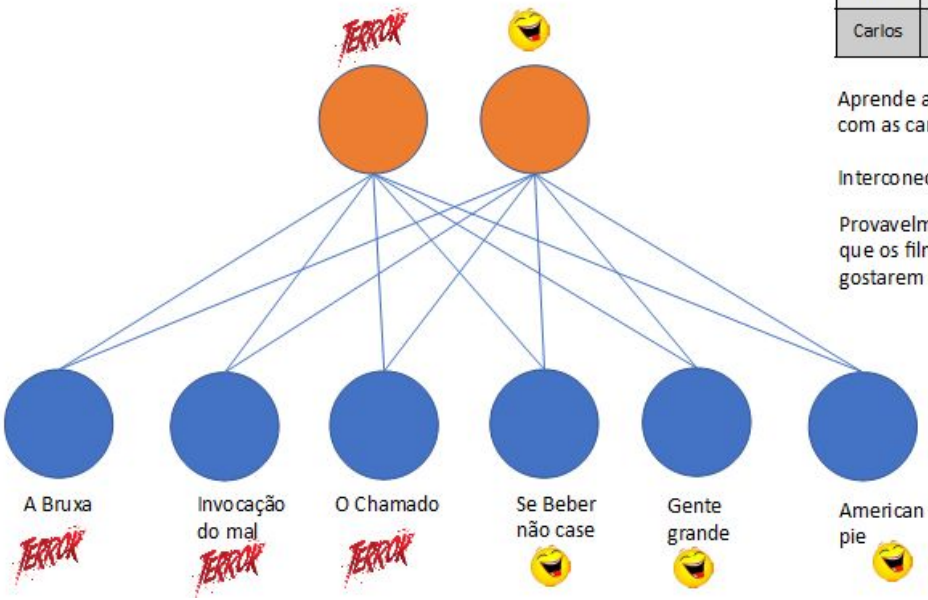
Input: RBM $(V_1, \ldots, V_m, H_1, \ldots, H_n)$, training batch S
Output: gradient approximation Δw_{ij}, Δb_j and Δc_i for $i = 1, \ldots, n$, $j = 1, \ldots, m$

init $\Delta w_{ij} = \Delta b_j = \Delta c_i = 0$ for $i = 1, \ldots, n$, $j = 1, \ldots, m$
forall the $\boldsymbol{v} \in S$ **do**
  $\boldsymbol{v}^{(0)} \leftarrow \boldsymbol{v}$
  for $t = 0, \ldots, k-1$ **do**
    for $i = 1, \ldots, n$ **do** sample $h_i^{(t)} \sim p(h_i \mid \boldsymbol{v}^{(t)})$
    for $j = 1, \ldots, m$ **do** sample $v_j^{(t+1)} \sim p(v_j \mid \boldsymbol{h}^{(t)})$
  for $i = 1, \ldots, n$, $j = 1, \ldots, m$ **do**
    $\Delta w_{ij} \leftarrow \Delta w_{ij} + p(H_i = 1 \mid \boldsymbol{v}^{(0)}) \cdot v_j^{(0)} - p(H_i = 1 \mid \boldsymbol{v}^{(k)}) \cdot v_j^{(k)}$
    $\Delta b_j \leftarrow \Delta b_j + v_j^{(0)} - v_j^{(k)}$
    $\Delta c_i \leftarrow \Delta c_i + p(H_i = 1 \mid \boldsymbol{v}^{(0)}) - p(H_i = 1 \mid \boldsymbol{v}^{(k)})$

Exemplo: Sistema de Recomendação de Filmes

Restricted Boltzmann Machines

Ana	1		1	0		0
Marcos		0	1			
Paulo	1		1			0
Priscila		0			1	1
Maria	0			1	0	1
Carlos	0		1		0	1

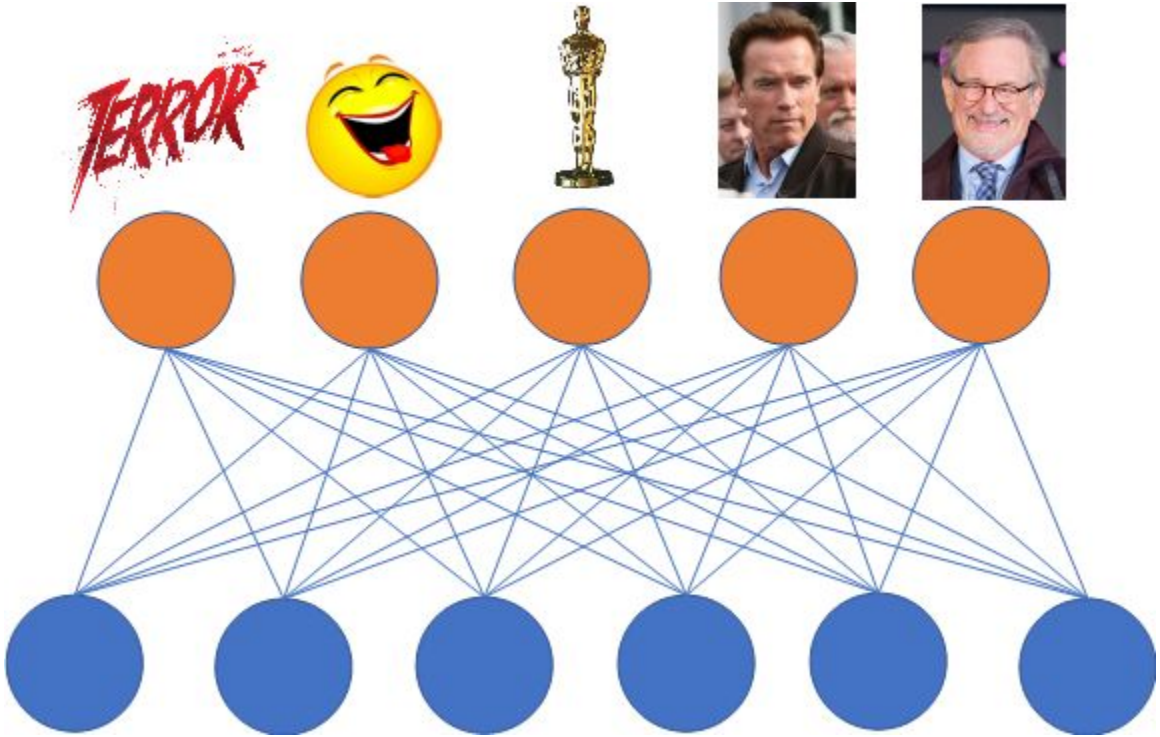


Aprende alocar os nós escondidos de acordo com as características (cada nó é um padrão)

Interconectividade entre a nota dos usuários

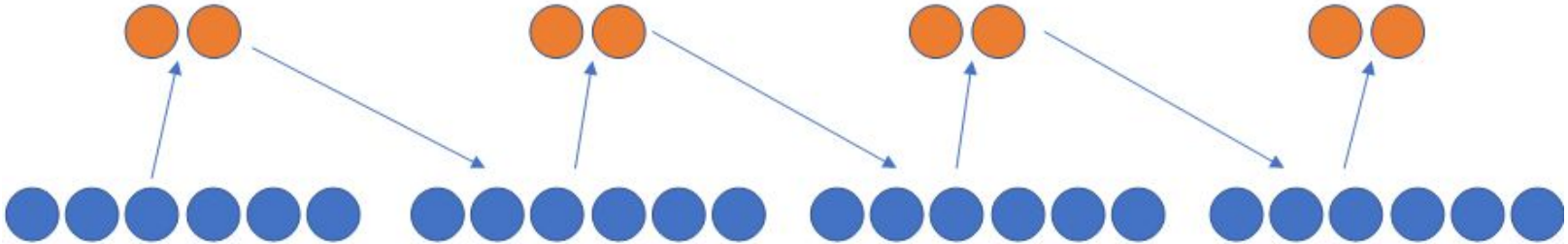
Provavelmente existe alguma característica que os filmes possuem que faz as pessoas gostarem (pessoas gostam das características)

Exemplo: Sistema de Recomendação de Filmes



Exemplo: Sistema de Recomendação de Filmes

Constrative Divergence (aprendizagem)



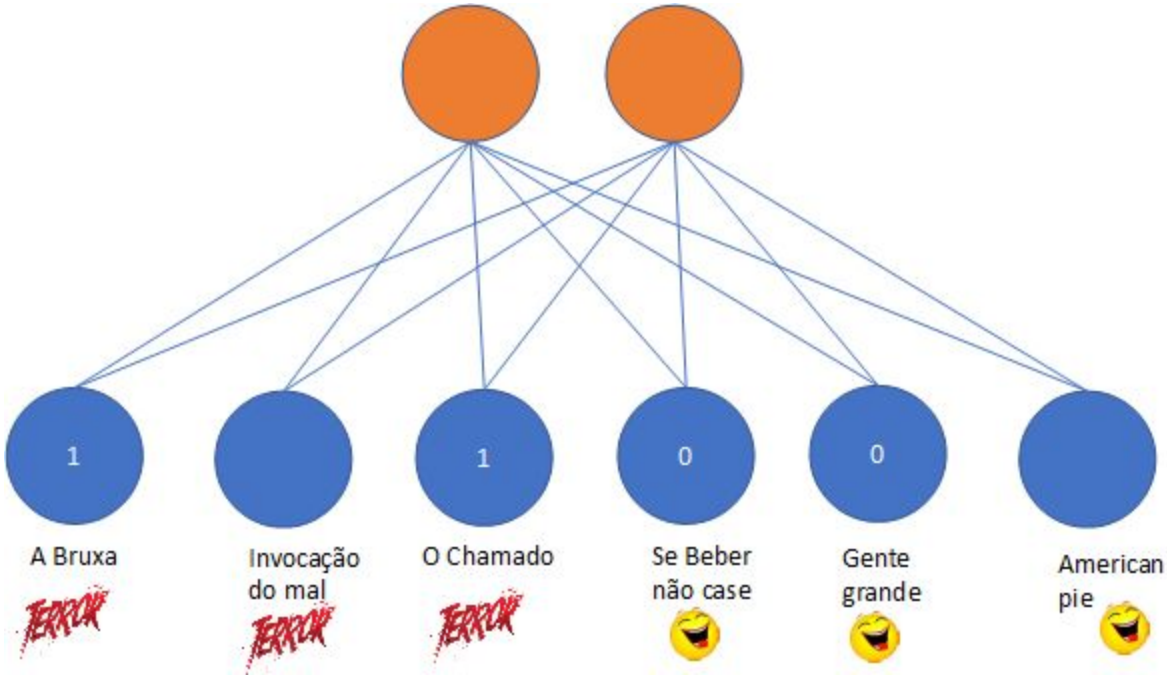
Reconstrução do nó (gibbs sampling)

A execução termina quando os valores são os mesmos (ou quando o número de épocas é atingido)

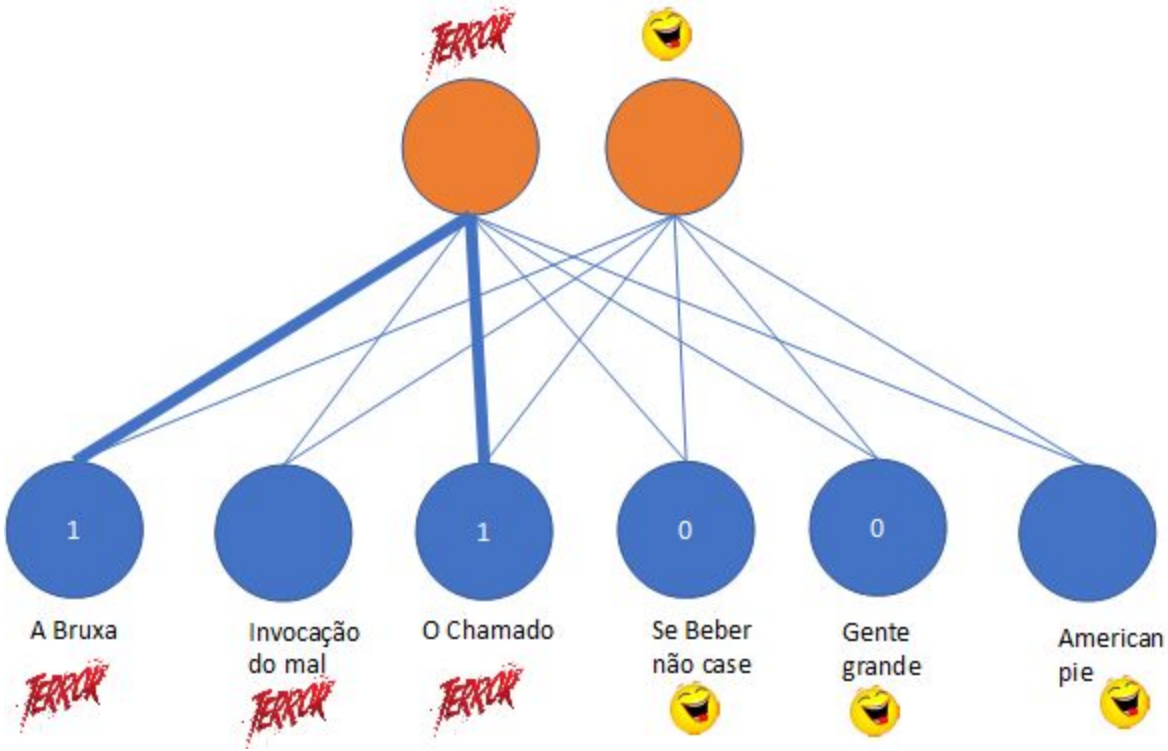
O nó é reconstruído usando todos os nós da camada oculta

Pesos não são atualizados nesse processo

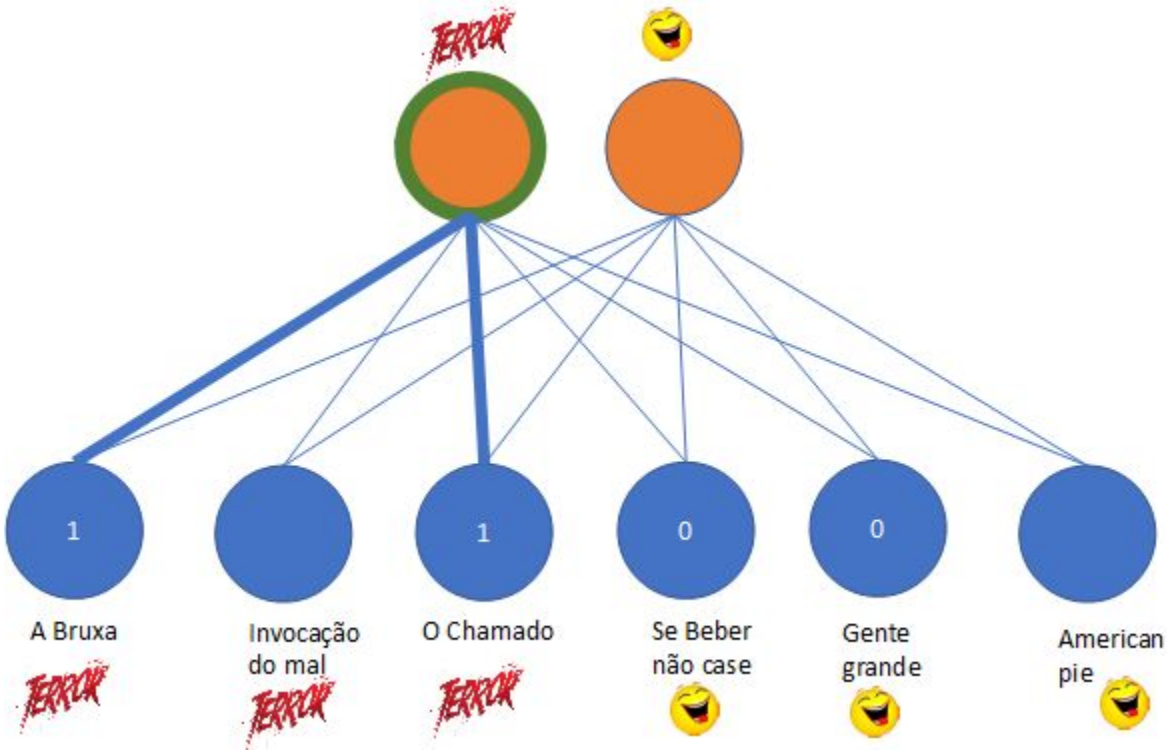
Exemplo: Sistema de Recomendação de Filmes



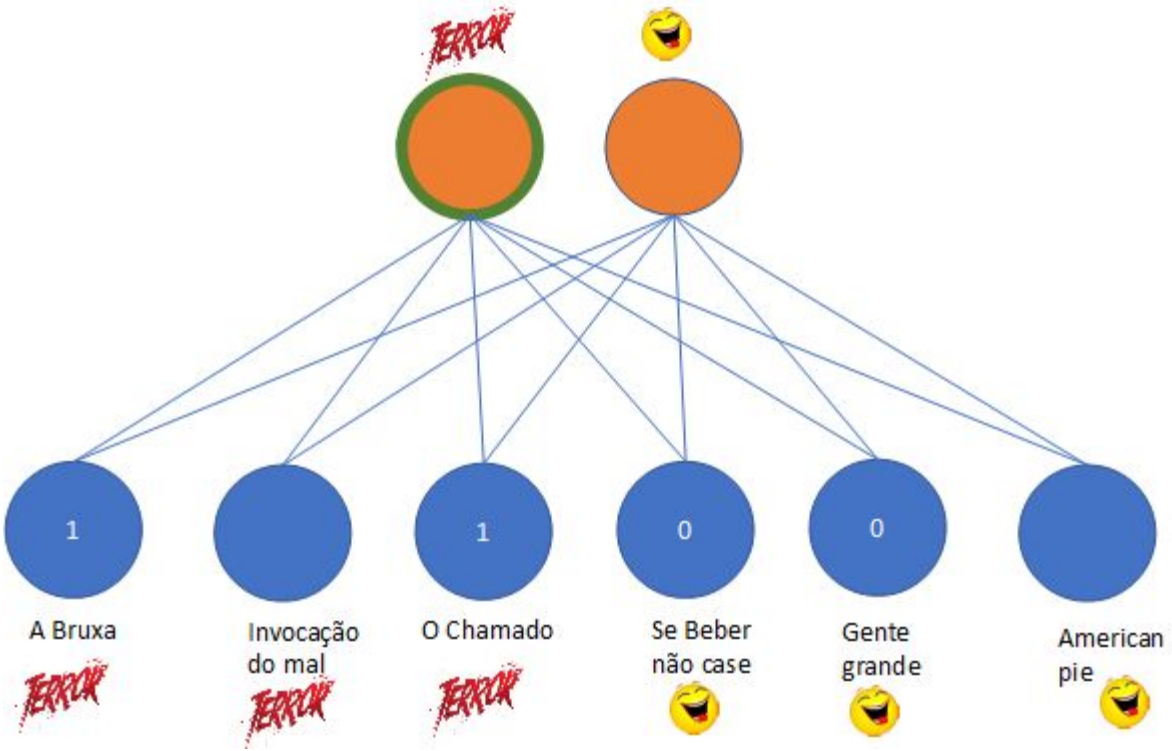
Exemplo: Sistema de Recomendação de Filmes



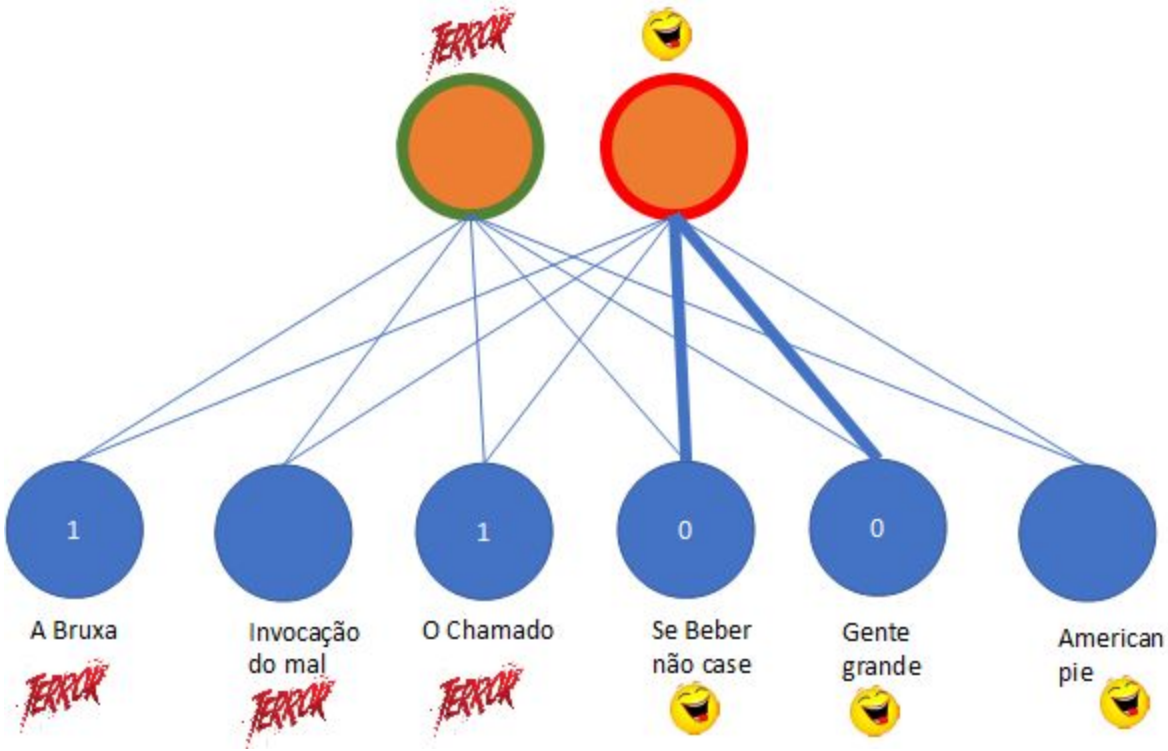
Exemplo: Sistema de Recomendação de Filmes



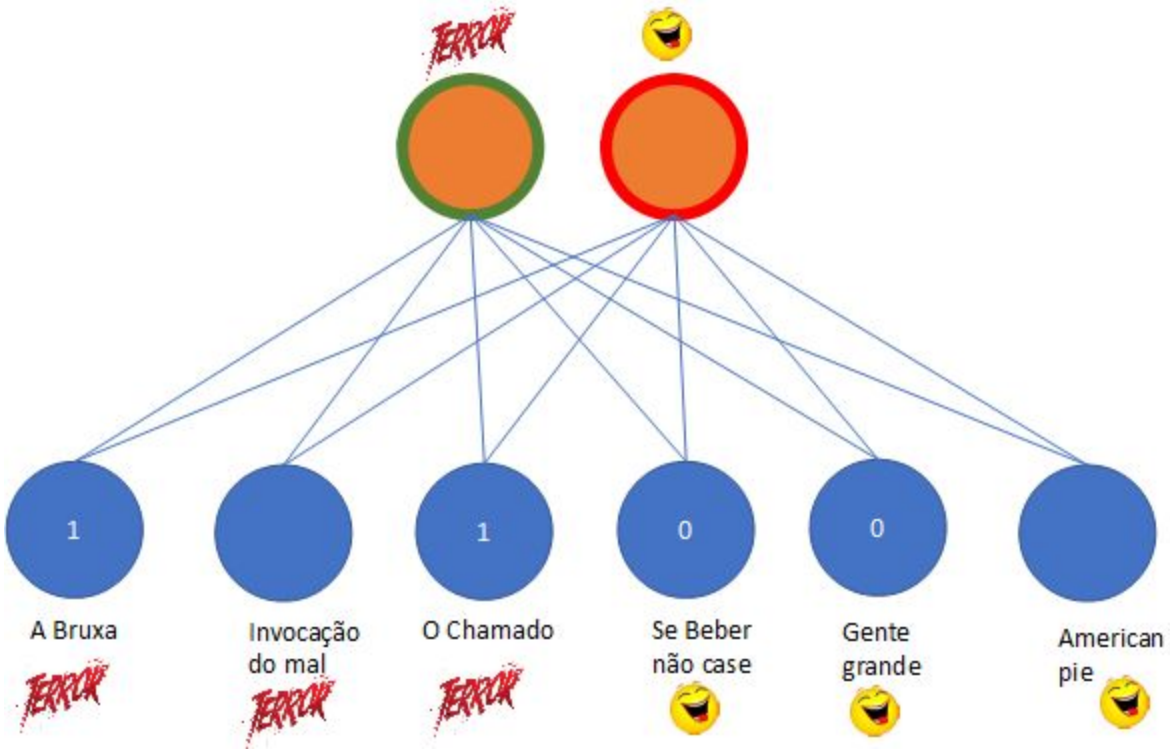
Exemplo: Sistema de Recomendação de Filmes



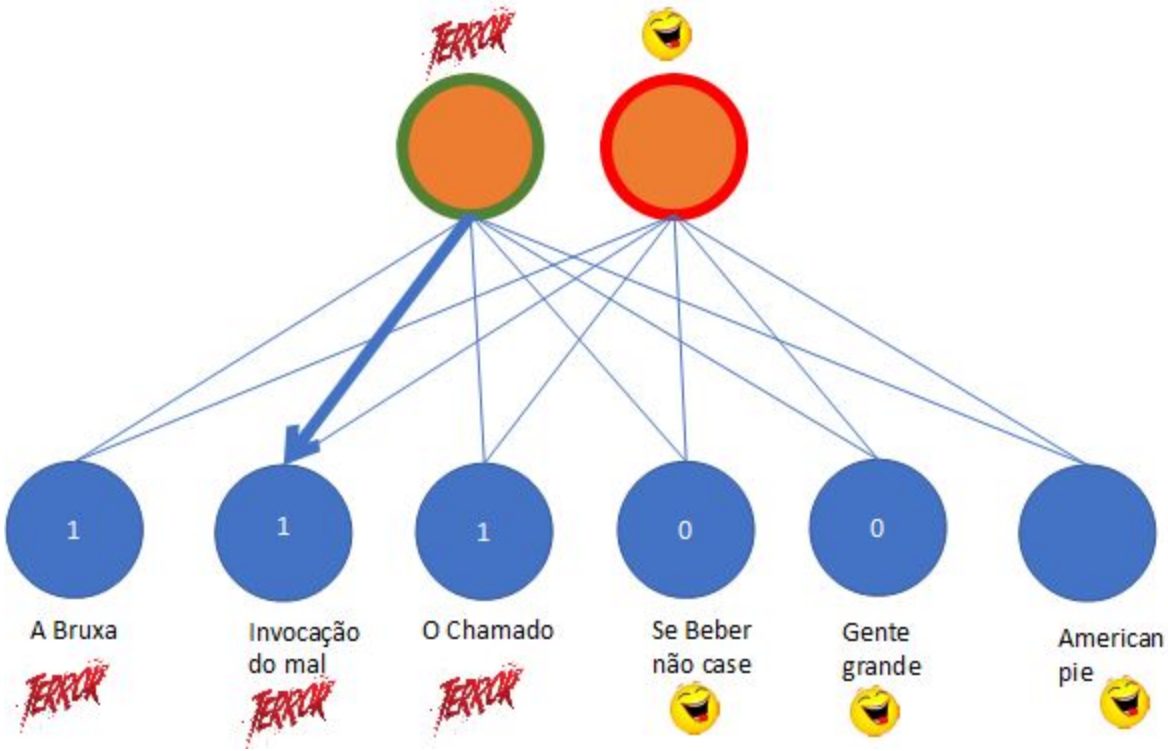
Exemplo: Sistema de Recomendação de Filmes



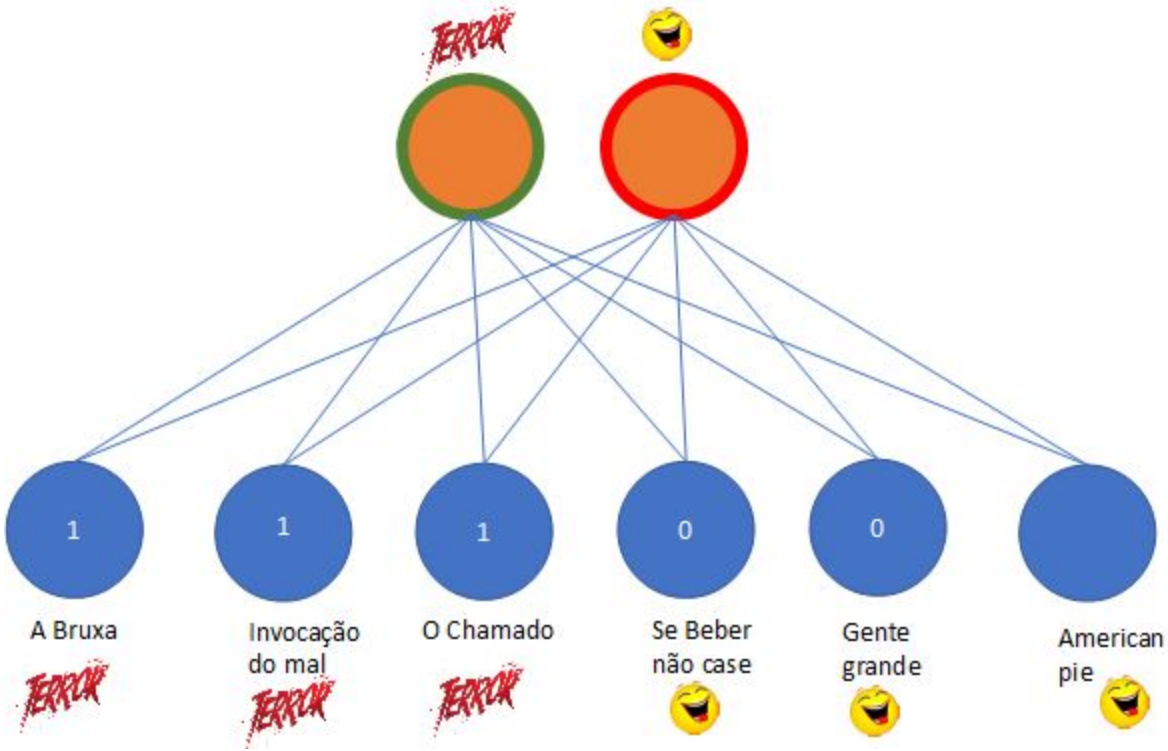
Exemplo: Sistema de Recomendação de Filmes



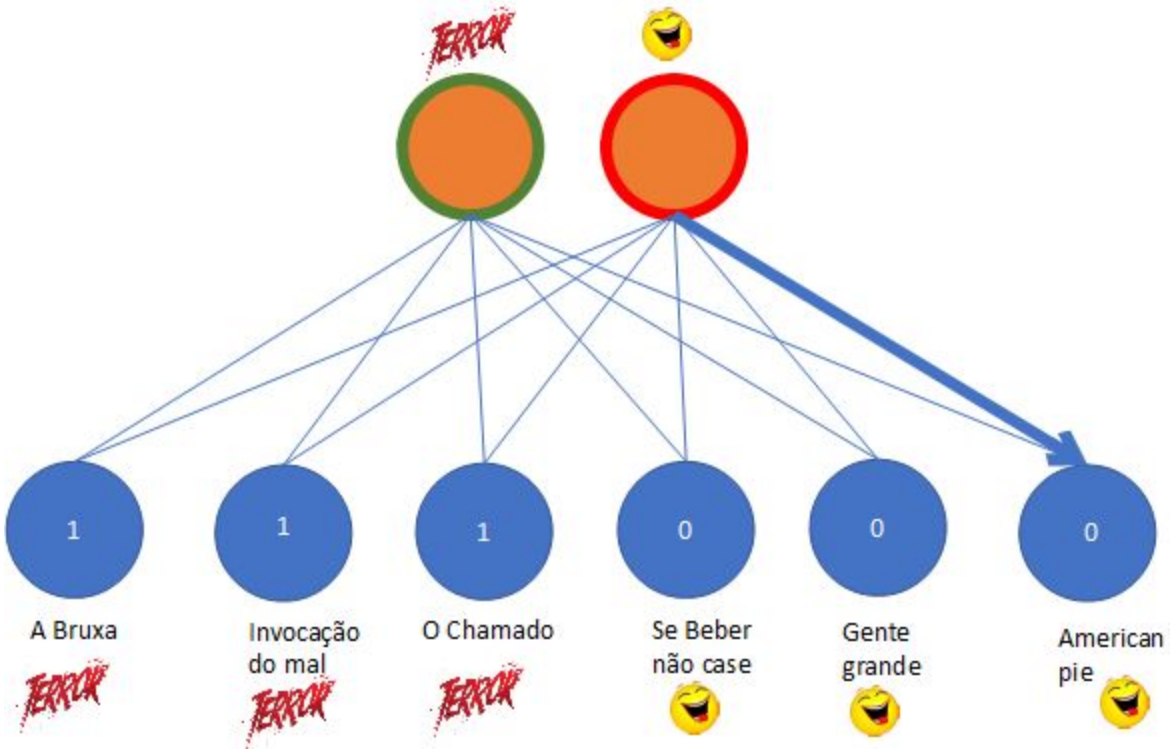
Exemplo: Sistema de Recomendação de Filmes



Exemplo: Sistema de Recomendação de Filmes



Exemplo: Sistema de Recomendação de Filmes



A mostly complete chart of

Neural Networks

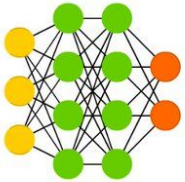
- Backfed Input Cell
- Input Cell
- Noisy Input Cell
- Hidden Cell
- Probablistic Hidden Cell
- Spiking Hidden Cell
- Output Cell
- Match Input Output Cell
- Recurrent Cell
- Memory Cell
- Different Memory Cell
- Kernel
- Convolution or Pool

Perceptron (P)

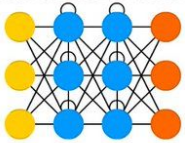
Feed Forward (FF)

Radial Basis Network (RBF)

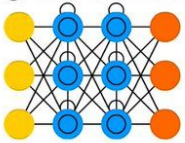
Deep Feed Forward (DFF)



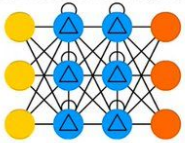
Recurrent Neural Network (RNN)



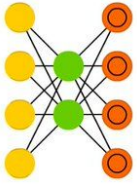
Long / Short Term Memory (LSTM)



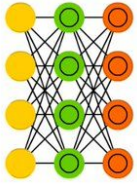
Gated Recurrent Unit (GRU)



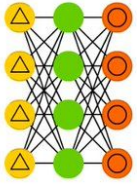
Auto Encoder (AE)



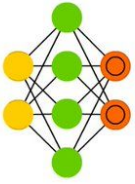
Variational AE (VAE)



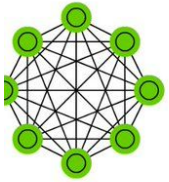
Denoising AE (DAE)



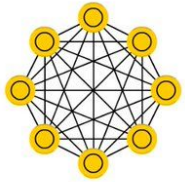
Sparse AE (SAE)



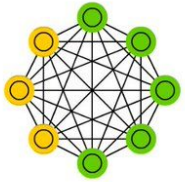
Markov Chain (MC)



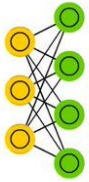
Hopfield Network (HN)



Boltzmann Machine (BM)



Restricted BM (RBM)



Deep Belief Network (DBN)