O Perceptron de Rosenblatt

Nas décadas de 1950 e 1960, vários pesquisadores estavam propondo modelos de redes neurais contendo modificações no modelo original de McCulloch e Pitts e regras de modificação dos pesos sinápticos diferentes da lei de Hebb para tratar de problemas de aprendizado.

Talvez a regra de aprendizado proposta naquela época que tenha causado maior impacto seja a *Regra de Aprendizado do Perceptron*. Esta regra fazia parte de um modelo de rede neural desenvolvido pelo cientista da computação Frank Rosenblatt (1928 – 1971) entre 1958 e 1962, que foi chamado de **Perceptron**.

O Perceptron de Rosenblatt foi desenvolvido para lidar com o problema de *reconhecimento de padrões*. Este é um tipo de tarefa que os seres humanos fazem sem nenhum esforço aparente e de forma quase instantânea. Porém, é um dos problemas mais difíceis de serem resolvidos por uma máquina.

Em ciência da computação, o reconhecimento de padrões pode ser definido como o processo pelo qual um padrão ou sinal recebido por um sistema em sua entrada é classificado (rotulado) como pertencente a uma única classe de um conjunto de classes.

O conjunto de classes nos quais os padrões são classificados pode já ter sido definido *a priori*, antes do início da operação do sistema. Em tal caso, o sistema *aprende* a fazer a classificação durante uma fase de treinamento supervisionado (ver aula 3) em que padrões previamente escolhidos e classificados de acordo com a regra de classificação que se quer que o sistema aprenda são apresentados a ele.

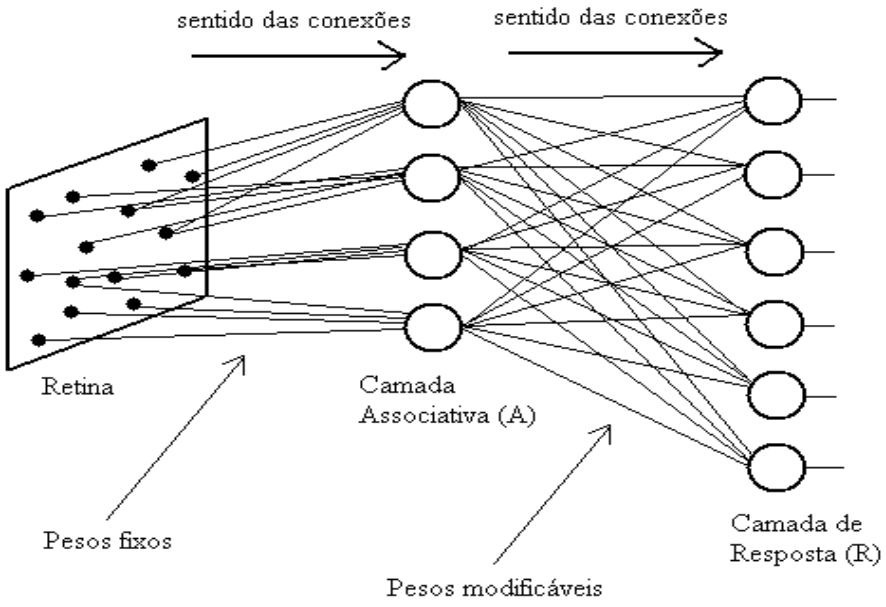
Outra forma pela qual o sistema pode aprender a classificar um conjunto de padrões é através de um aprendizado do tipo não-supervisionado (ver aula 3), no qual o próprio sistema tem que determinar as classes nas quais os padrões serão classificados, baseado em propriedades intrínsecas dos padrões.

Exemplos de problemas de reconhecimento de padrões são os seguintes:

- Determinar o sexo de uma pessoa ao ver o seu rosto;
- Reconhecer uma pessoa ao ouvir sua voz no telefone;
- Reconhecer sua mala no meio de outras em uma esteira rolante de aeroporto;
- Determinar a safra de um vinho pelo seu aroma e gosto;
- Reconhecer a chave que abre a porta da sua casa, no escuro, em seu molho de chaves quando você chega em casa depois de uma noitada.
- Pense em outros exemplos para por aqui ...

Essas são tarefas que um ser humano pode resolver sem grandes dificuldades ou com algum treinamento, mesmo em condições não-ideais, como, por exemplo, na presença de ruído, transformações no sinal, ou passado um tempo desde a última vez em que se teve contato com o padrão a ser reconhecido.

O Perceptron originalmente proposto por Rosenblatt consistia de uma camada de unidades sensoriais, denominada *retina*; uma camada de unidades ocultas, chamada de camada *associativa*; e uma camada de saída, chamada de camada de *resposta*. A figura abaixo ilustra o sistema.



As conexões entre os neurônios são sempre *feed-forward*: da retina para a camada associativa (camada A) e da camada A para a camada de resposta (camada R).

A conectividade entre a retina e a camada A não é total, mas parcial: um neurônio *i* da camada A recebe sinapses apenas de uma parte da retina, chamada de *campo receptivo* do neurônio *i*.

Os valores dos pesos das sinapses entre a retina e a camada A são fixos, valendo +1 para sinapses excitatórias e –1 para sinapses inibitórias.

A conectividade entre a camada A e a camada R é total: cada neurônio da camada R recebe conexões sinápticas de todos os neurônios da camada A.

Os pesos das conexões de A para R são modificáveis durante uma *fase de treinamento*, ficando fixos depois. Antes do início da fase de treinamento, os pesos recebem valores aleatórios escolhidos do intervalo entre –1 e +1.

Os neurônios das camadas A e R são neurônios de McCulloch-Pitts: a saída de um neurônio *j* é +1 se $u_j = \sum_{i=1}^{n} w_{ji} x_i - b_j \geq 0$ e –1 se $u_j = \sum_{i=1}^{n} w_{ji} x_i - b_j < 0$, onde b_j é o viés do neurônio *j*.

A arquitetura do perceptron de Rosenblatt possui alguma semelhança (na realidade, é inspirada nele) com a do sistema visual de mamíferos.

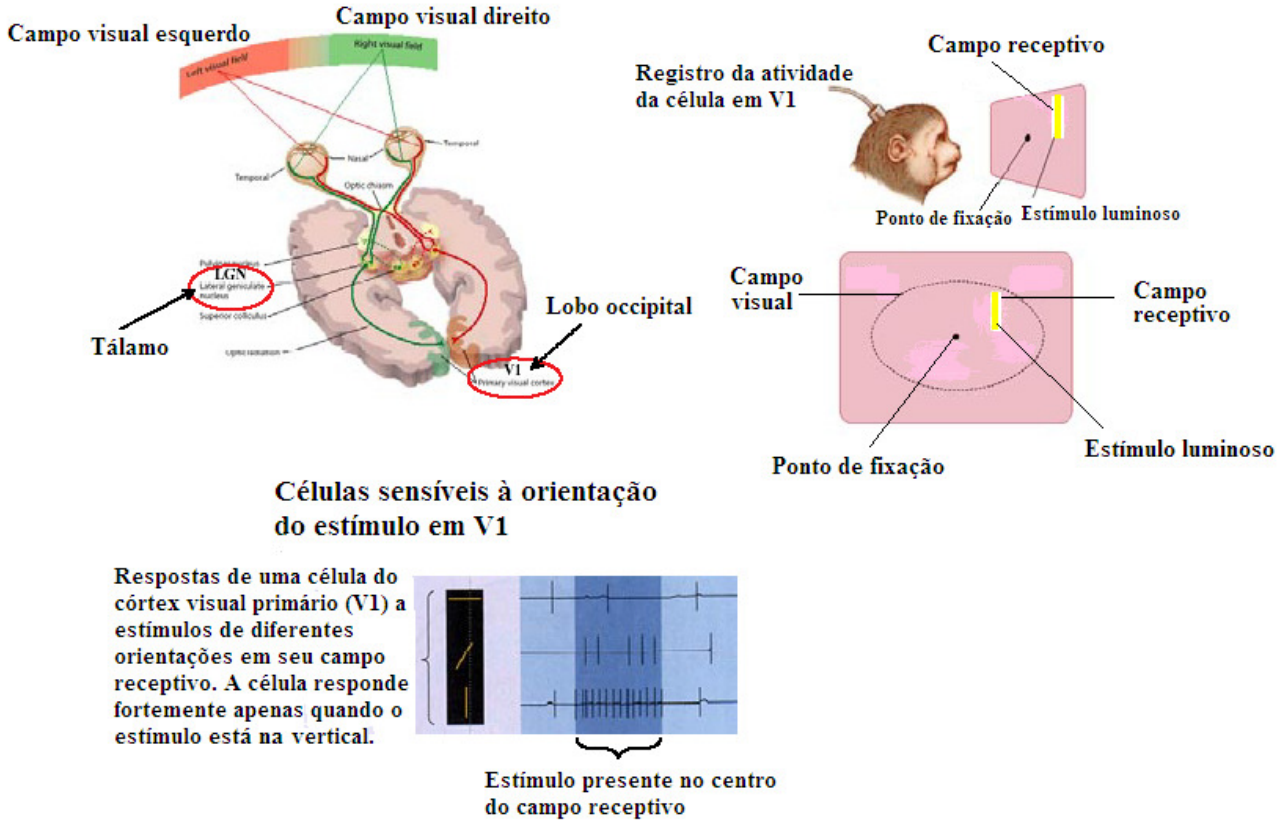
A camada *retina* é inspirada na retina do sistema visual. É ali que são projetados os sinais vindos do mundo exterior.

Os neurônios da camada A seriam *detectores de características*.

Neurônios especializados na detecção de certas características também existem no sistema visual. Por exemplo, sabe-se que no córtex visual primário (área V1) de mamíferos existem neurônios especializados na detecção de estímulos luminosos na forma de barras com certa orientação: se uma barra luminosa com uma dada orientação aparece no *campo receptivo* de um neurônio sensível àquela orientação, ele dispara potenciais de ação com a máxima freqüência possível; caso contrário, ele não dispara ou dispara com uma freqüência baixa.

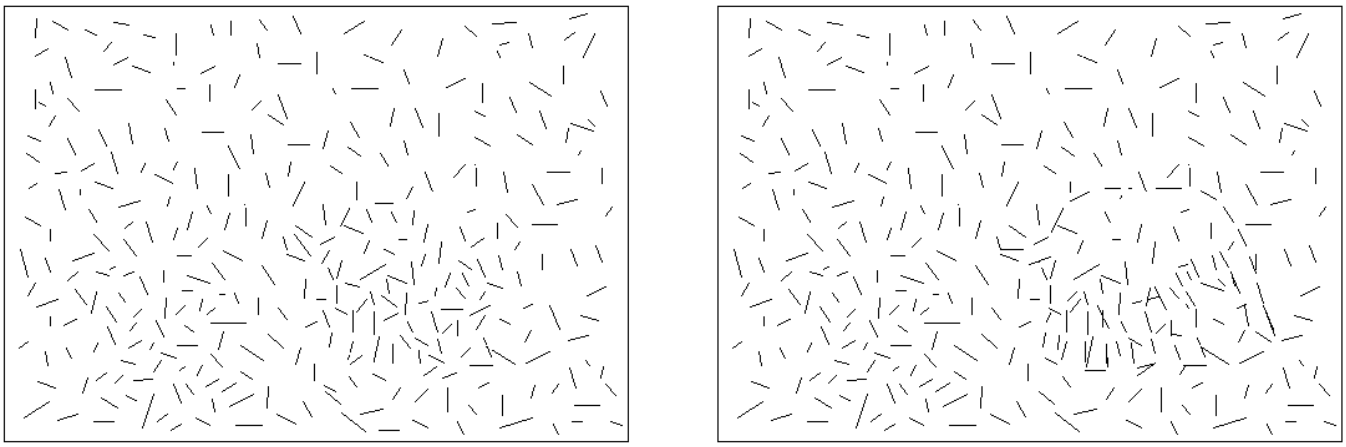
O *campo receptivo* de um neurônio de V1 é definido como a região da retina ou do campo visual que, se for estimulada, faz o neurônio disparar.

O desenho abaixo ilustra o conceito de campo receptivo e dá um exemplo de célula sensível à orientação vertical no córtex visual primário (V1).



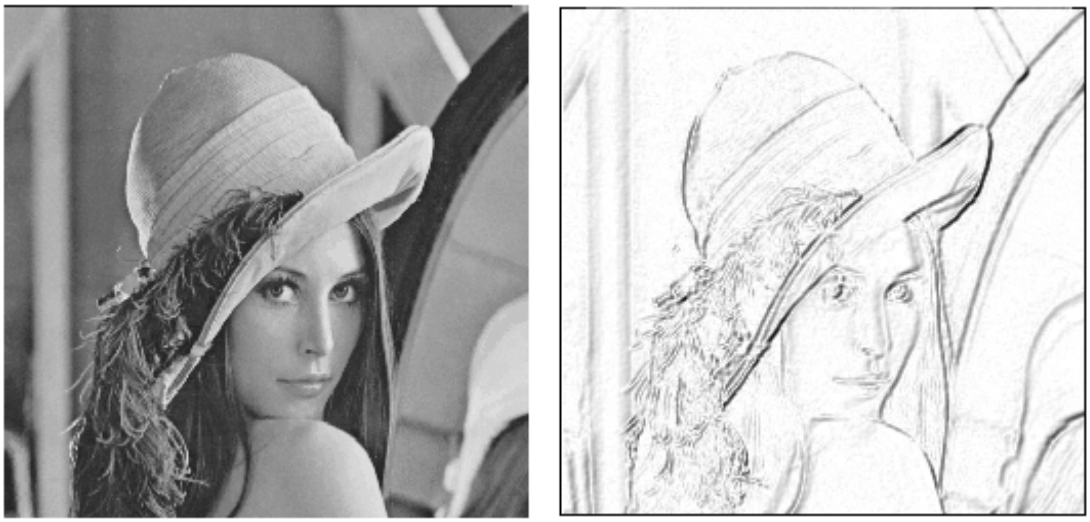
A razão pela qual a área visual primária do córtex (a primeira estação processadora de informação visual do córtex) tem neurônios especializados em detectar barras luminosas orientadas ainda é objeto de especulação.

Pode-se imaginar que, ao longo da evolução, era importante para nossos ancestrais reconhecer formas (por exemplo, de predadores) em meio a outras formas. Uma maneira de destacar uma forma é pelo seu *contorno*, que pode ser caracterizado por uma sequência de pequenas linhas ou barras "emendadas" umas às outras (veja a figura abaixo).



Em uma imagem, as mudanças mais abruptas de intensidade luminosa ocorrem nas *fronteiras* entre os objetos. Fronteiras também podem ser aproximadas por barras de diferentes orientações.

Em outras palavras, os elementos de uma imagem podem ser decompostos em termos de *características* simples e, de todas as possíveis, as barras orientadas, talvez por permitirem a determinação dos contornos dos objetos e das fronteiras entre os elementos de uma imagem, foram selecionadas pela evolução como importantes para a sobrevivência (veja outro exemplo na figura a seguir).



A figura acima mostra um exemplo de uma imagem e, ao lado, apenas as fronteiras separando os elementos da imagem. As linhas representando as fronteiras podem ser imaginadas como resultantes de uniões de pequenas linhas retas com diferentes orientações. Será que é assim que o sistema visual representa uma imagem, pelo menos em alguma etapa do processamento?

Uma teoria de representação sensorial para explicar porque o cérebro usa a decomposição de uma imagem em termos de características simples logo no primeiro estágio de processamento é a seguinte: os neurônios da área V1 funcionariam como *detectores de características*, ou seja, seriam capazes apenas de reconhecer pequenos elementos de uma imagem. Em um estágio posterior de processamento, em outra área cerebral, outros neurônios receberiam as saídas dos neurônios de V1 e fariam uma combinação delas montando uma imagem. Como isto é feito, e se é que é feito, é um problema conhecido como "problema da integração" ou da "ligação" (*binding problem*).

Também existem evidências comportamentais de que trabalhamos com detectores de características. Por exemplo, em um experimento de busca visual feito pelo psicólogo estadunidense Ulric Neisser (1928 -) em 1964, a tarefa dos sujeitos era localizar a letra Z contra um fundo composto, ou por letras "curvas", ou por letras "angulares" (veja a figura a seguir).

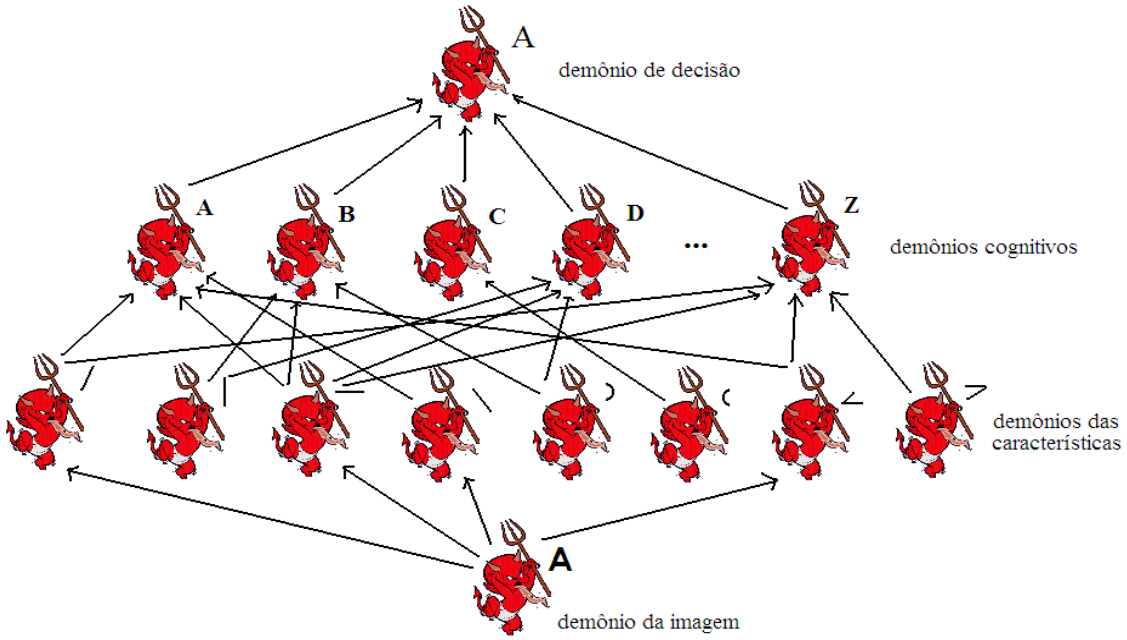
Letras Curvas	Letras Angulares
CBDUZGRO	IVMXZWEF

O tempo de resposta foi medido para os dois casos e constatou-se que ele é menor (cai pela metade) no caso do fundo de letras curvas do que no caso do fundo de letras angulares. Como só há uma letra Z por seqüência, ocupando a mesma posição nas duas, Neisser concluiu que as pessoas deviam estar monitorando as *características* da letra, e não a própria letra. Como a letra Z é uma letra angular, seria mais fácil detectar as suas características contra um fundo de letras com características diferentes do que contra um fundo de letras com as mesmas características.

Em outras palavras, se o que procuramos em um padrão são as suas características, seria mais fácil confundir padrões com características similares entre si (tempo de reação maior para a detecção) do que padrões com características diferentes.

Um modelo para reconhecimento de padrões muito parecido com o perceptron, mas que possui unidades detectoras de características que declaram *explicitamente* os padrões aos quais são sensíveis, é o *Pandemonium*, proposto pelo cientista da computação Oliver Selfridge (1926 – 2008) em 1958 (o mesmo ano em que Rosenblatt publicou seu primeiro artigo sobre o Perceptron).

Uma versão do *Pandemonium* para reconhecer letras do alfabeto é mostrada na figura a seguir, para comparação com o Perceptron.

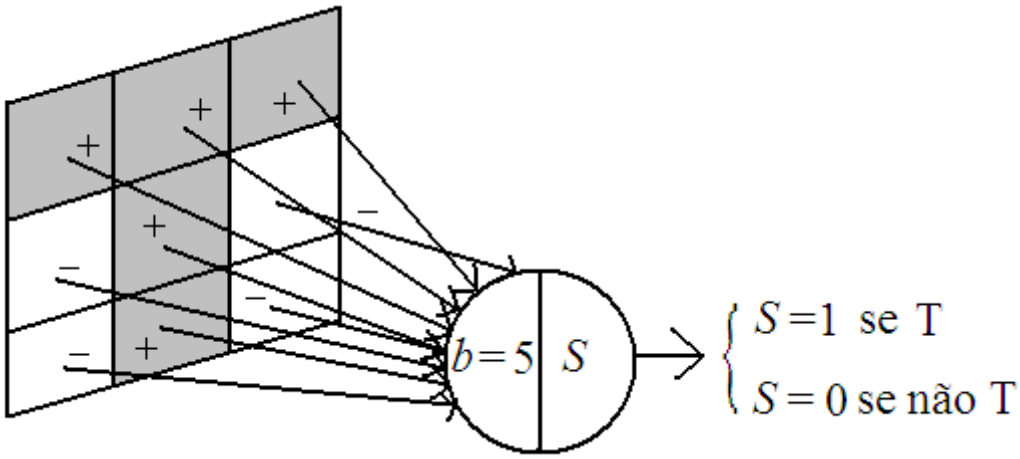


As unidades do modelo foram chamadas de “demônios” (*Pandemonium* é a palavra grega para “conjunto dos demônios”) e, na concepção metafórica de Selfridge, emitiriam sinais através de gritos.

No fim de um processo de reconhecimento, o demônio de decisão grita o nome da letra que está sendo apresentada pelo demônio da imagem. O demônio de decisão dá a sua resposta em função dos gritos que ele ouve dos demônios cognitivos, cada um gritando uma letra diferente com uma intensidade que depende da informação que lhe é passada pelos demônios das características na camada anterior. Os demônios das características são detectores de características específicas presentes nas letras e gritam sempre que encontram a sua característica na letra sendo apresentada pelo demônio da imagem.

O Perceptron de Rosenblatt é parecido com o *Pandemonium*, pois as unidades da camada A são similares aos demônios detectores de características. As conexões sinápticas entre a retina e a camada A têm seus pesos fixos para indicar que as unidades da camada A estão “sintonizadas” para detectar *características bem definidas* que apareçam nos seus campos receptivos.

Por exemplo, o neurônio de McCulloch-Pitts da figura abaixo detecta Ts em seu campo receptivo 3x3.



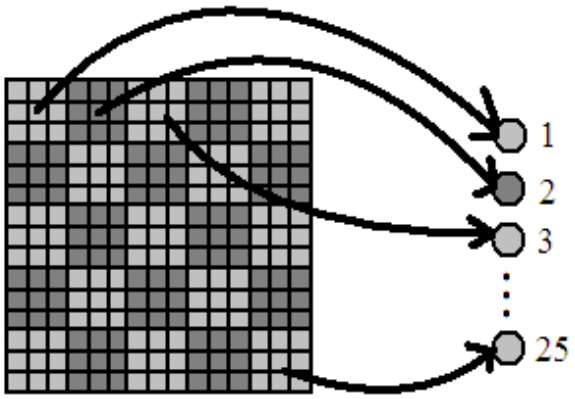
A operação de extração de características executada na passagem da retina para a camada A é um exemplo de *pré-processamento* feito no padrão bruto projetado na retina antes do seu reconhecimento.

O pré-processamento consiste em uma série de operações e transformações feitas sobre os padrões brutos vindos do mundo real para deixá-los em condições de ser reconhecidos pelos neurônios da camada R. No caso de letras, por exemplo, antes de se executar a etapa de extração de características poder-se-ía implementar um outro tipo de pré-processamento, responsável pela conversão de letras escritas à mão por diversas pessoas, com várias formas, tamanhos e inclinações, em estruturas com dimensões padronizadas projetadas na retina.

Uma das características de um pré-processamento é realizar uma *redução na dimensionalidade* dos padrões de entrada. Em geral, os padrões do mundo real que se quer classificar ou reconhecer são matrizes ou vetores com um número muito grande de dimensões (pense, por exemplo, em imagens digitais de 1024x1024 pixels com cada pixel sendo codificado por um número de 8 bits). Em tais casos, não é muito conveniente fornecer todos os valores de um padrão diretamente como entrada para um sistema de reconhecimento de padrões realizar a sua tarefa.

No caso do Perceptron a redução da dimensionalidade é feita pela camada A, pois um processo de extração de características implica quase sempre em uma redução da dimensionalidade.

A figura a seguir ilustra um exemplo de redução na dimensão do padrão projetado na retina, uma matriz com 15x15 = 225 elementos. O padrão bruto é convertido em um vetor de 25 elementos na camada A. Cada elemento da camada A é uma unidade extratora de características com campo receptivo de 3x3 unidades sem sobreposição com os campos receptivos das demais unidades.



Matriz 15x15 = 225

Note que o processo de extração de características, com a conseqüente redução na dimensionalidade do padrão de entrada, implica em uma nova *representação*, ou *codificação*, do padrão de entrada original.

O pré-processamento constitui uma etapa *fundamental* para o bom desempenho de um sistema de reconhecimento de padrões, seja ele uma rede neural ou não. Por exemplo, se as características escolhidas (determinadas pelos pesos fixos das conexões chegando aos neurônios da camada A) não forem adequadas, ou o número de características for muito pequeno ou muito grande, o problema pode não ter solução. Neste caso, a codificação feita não terá sido a mais adequada para o problema.

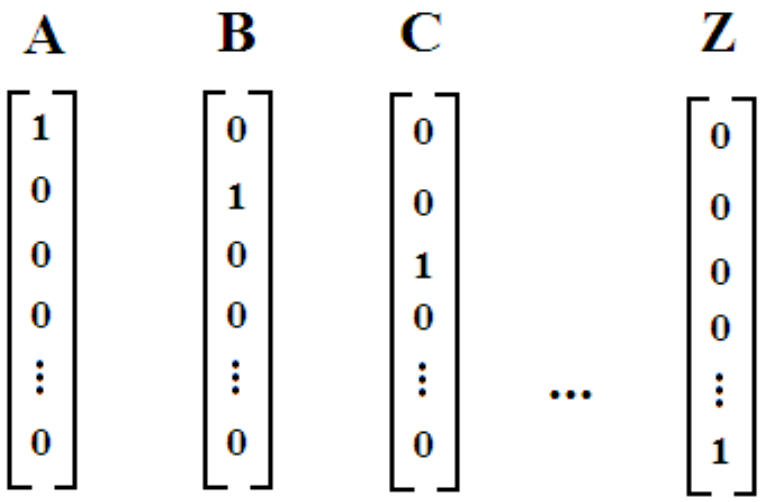
A camada R do Perceptron é o estágio em que ocorre a combinação das várias características detectadas na camada A em um padrão único, a resposta do sistema.

Esse padrão deve ser uma *representação* do padrão apresentado na retina que corresponda a um reconhecimento dele de acordo com algum código previamente estabelecido (o Perceptron é um sistema que aprende de maneira supervisionada). Portanto, a tarefa de *reconhecer* o padrão apresentado na retina deve ser feita pelos neurônios da camada R. Eles são os *reconhecedores de padrões* propriamente ditos.

A partir da informação que é fornecida e pré-processada pelos detectores de características da camada A, os neurônios da camada R devem reconhecer qual é o padrão apresentado na retina. Como cada um dos neurônios da camada R é independente dos demais (note que não há conexões entre eles), cada um deve funcionar como detector da presença de um certo padrão projetado na retina, dando como saída o valor 1 quando o padrão estiver na retina e o valor 0 quando ele não estiver.

Por exemplo, cada neurônio da camada R pode ser o detector de uma letra do alfabeto, como no caso do exemplo do *Pandemonium*. Em tal caso, a camada R teria 26 unidades. Quando uma dada letra, por exemplo, B, for projetada na retina, a unidade da camada R responsável pela detecção da letra B deve ter atividade +1 e todas as outras unidades devem ter atividade 0. Isto indicaria que a letra presente na retina é a letra B. Já quando esta letra for o M, por exemplo, somente a unidade responsável pela detecção de Ms deveria ter atividade 1 e as demais deveriam ter atividade 0.

Desta forma, cada letra do alfabeto seria indicada por um padrão de atividades dos neurônios da camada R em que haveria apenas um neurônio com atividade 1 e os demais com atividade 0, conforme mostrado na figura a seguir.



Note que o número de unidades na camada R define o *número de classes* nas quais o Perceptron pode classificar padrões. Para classificar todas as letras do alfabeto, ele deve ter 26 unidades. Se tiver menos, alguma(s) letra(s) ficarão sem ser classificadas.

O problema do Perceptron *per se*, supondo que um bom pré-processamento foi feito, é o de encontrar um algoritmo de modificação dos pesos das sinapses que conectam os neurônios da camada A aos neurônios da camada R tal que a execução desse algoritmo durante a fase de treinamento faça os valores dos pesos convergir para valores que gerem a classificação correta de todos os padrões.

A solução de Rosenblatt para este problema é **regra de aprendizado do perceptron**, cujo algoritmo é o seguinte:

- Durante a fase de treinamento, apresentam-se *exemplos* ao Perceptron, constituídos por um padrão de entrada na retina e pela resposta desejada. Por exemplo, quando a letra "A" for projetada na retina a camada R deve ser informada de que a sua resposta correta deve ser o padrão em que o primeiro neurônio tem saída igual a 1 e os demais têm saídas iguais a 0.

- Toda vez que a resposta de uma unidade *i* da camada R não for igual à sua resposta desejada, isto é, quando a unidade *i* cometer um erro, os valores dos pesos das conexões da camada A para o neurônio *i* devem ser modificados pela seguinte regra:

$$\Delta \vec{\omega}_i = \eta \left(R_i^{\text{correto}} - R_i \right) \vec{a} \ ,$$

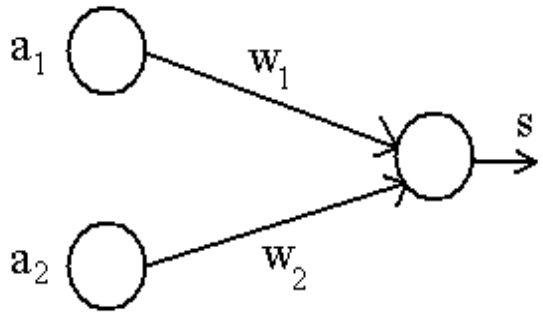
onde R_i^{correto} é a saída desejada da unidade i, R_i é a saída que a unidade *i* fornece de fato, $\vec{a}$ é o vetor de atividades existente na camada A e η é uma constante dando o tamanho da modificação.

Note que a regra acima se parece com a regra de Hebb, com a atividade do neurônio pós-sináptico R_i substituída pela diferença, ou erro, $\left(R_i^{\text{correto}} - R_i \right)$.

A regra de aprendizado do Perceptron é mais fácil de ser entendida de maneira gráfica.

Vamos dar um exemplo para ilustrá-la. Para simplificar, vamos supor que existem apenas dois neurônios na camada A, com saídas a_1 e a_2. Como os neurônios da camada R são independentes uns dos outros, podemos limitar, sem perda de generalidade, nossa análise do Perceptron a apenas uma unidade na camada R.

Nosso Perceptron fica, então, reduzido à estrutura da figura abaixo.



Vamos supor, também sem perda de generalidade, que o viés da unidade de saída vale $b = 0$. Portanto, o nível de ativação da unidade de saída é

$$u = \sum_{i=1}^{2} \omega_i a_i = \omega_1 a_1 + \omega_2 a_2$$

e a sua saída é

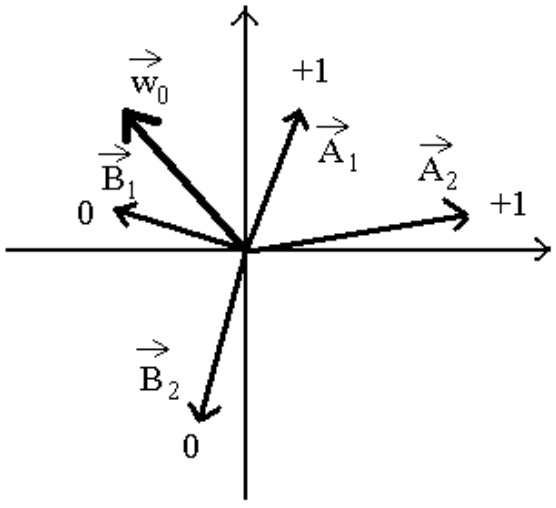
$$S = f(u) = \begin{cases} 1 \text{ se } u \geq 0 \\ 0 \text{ se } u < 0 \end{cases}$$

.

Podemos pensar nesse problema como o de identificação do sexo de uma pessoa a partir de apenas dois parâmetros. Por exemplo, valores medidos de dois elementos da face (pensem em alguns que vocês achem que permitiriam uma boa identificação dos sexos). Como temos dois valores possíveis para a saída, o valor 1 poderia indicar o sexo feminino e o 0 o sexo masculino.

O gráfico abaixo mostra um exemplo em que quatro padrões, os valores de (a_1, a_2) para quatro possíveis faces de pessoas, são representados por vetores em coordenadas cartesianas em duas dimensões. Os pontos estão categorizados em duas classes, A e B.

Os pontos da classe A devem fazer o neurônio de saída do Perceptron dar a resposta +1 (portanto, a classe A corresponde ao sexo feminino). Já os pontos da classe B devem fazer o neurônio de saída dar a resposta 0 (correspondendo ao sexo masculino).



Vamos supor que os valores dos quatro padrões são:

$$\vec{A}_1 = (0{,}3;0{,}7),\ \vec{A}_2 = (0{,}7;0{,}3);\ \vec{B}_1 = (-0{,}6;0{,}3);\ \vec{B}_2 = (-0{,}2;-0{,}8).$$

Também se encontra representado no gráfico o vetor de pesos inicial, cujos valores foram escolhidos aleatoriamente entre –1 e +1:

$$\vec{\omega}_0 = (-0{,}6;0{,}8).$$

O primeiro passo no treinamento do Perceptron é a aplicação de um dos quatro padrões, por exemplo o $\vec{A}_1$. Após a aplicação desse padrão, o nível de ativação da unidade de saída é:

$$u = \sum \omega_{0i} A_{1i} = \vec{\omega}_0 . \vec{A}_1 = (-0{,}6).(0{,}3) + (0{,}8).(0{,}7) = -0{,}18 + 0{,}56 = 0{,}38.$$

Como esse valor é positivo e o limiar do neurônio de saída é 0, a saída do neurônio será +1. A saída desejada para este padrão também é +1. Portanto, o erro cometido $\left(s^{\text{correto}} - s\right)$ é zero neste caso e o vetor de pesos não é modificado: $\vec{\omega}_1 = \vec{\omega}_0$.

Vamos aplicar agora outro padrão, por exemplo $\vec{A}_2$. Como o vetor de pesos ainda é o mesmo, o nível de ativação da unidade de saída é:

$$u = \sum \omega_{0i} A_{2i} = \vec{\omega}_0 . \vec{A}_2 = (-0{,}6).(0{,}7) + (0{,}8).(0{,}3) = -0{,}42 + 0{,}24 = -0{,}18.$$

Como este valor é negativo, a saída da rede será 0. O valor de resposta desejado para o padrão $\vec{A}_2$ é +1. Portanto, agora há um erro:

$$\left(s^{\text{correto}} - s\right) = (1 - 0) = 1,$$

e o vetor de pesos será modificado por

$$\Delta\vec{\omega} = \eta . 1 . \vec{A}_2 .$$

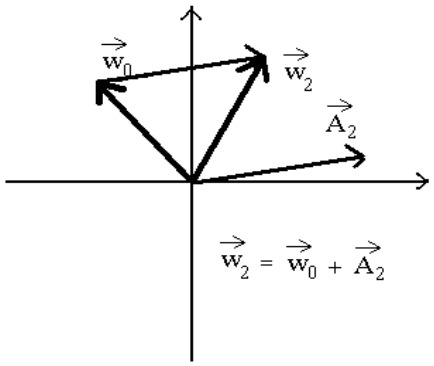
O novo vetor de pesos será dado por

$$\vec{\omega}_2 = \vec{\omega}_1 + \Delta\vec{\omega} = \vec{\omega}_0 + \eta . \vec{A}_2 .$$

O que esta fórmula diz é que o vetor $\vec{\omega}_0$ deve ser "arrastado" na direção do vetor $\vec{A}_2$ por um fator η. Vamos fazer $\eta = 1$ para simplificar. A operação a ser feita é então:

$$\vec{\omega}_2 = \vec{\omega}_1 + \vec{A}_2 = (-0{,}6;\ 0{,}8) + (0{,}7;\ 0{,}3) = (0{,}1;\ 1{,}1).$$

A figura abaixo ilustra graficamente esta operação, usando vetores:



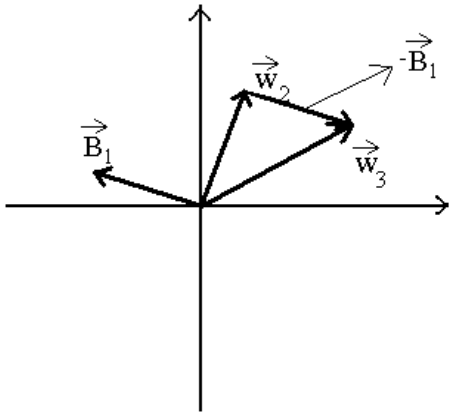
Vamos agora aplicar o padrão $\vec{B}_1$ à rede. Temos que:

$$u = \sum \omega_{2i} B_{1i} = \vec{\omega}_2 . \vec{B}_1 = (-0{,}6).(0{,}1) + (0{,}3).(1{,}1) = -0{,}06 + 0{,}33 = 0{,}27 > 0 \Rightarrow s = +1.$$

Houve um erro e a correção nos pesos será dada por $\Delta\vec{\omega} = -\vec{B}_1$:

$$\vec{\omega}_3 = \vec{\omega}_2 + \Delta\vec{\omega} = \vec{\omega}_2 - \vec{B}_1 = (0{,}1;1{,}1) - (-0{,}6;0{,}3) = (0{,}7;0{,}8).$$

Graficamente:



Vamos aplicar agora o padrão $\vec{B}_2$. O nível de ativação é:

$$u = \sum \omega_{3i} B_{2i} = \vec{\omega}_3 . \vec{B}_2 = (0,7).(-0,2) + (0,8).(-0,8) = -0,14 - 0,64 = -0,78 < 0 \Rightarrow s = 0.$$

Não há erro e, portanto, $\vec{\omega}_4 = \vec{\omega}_3$.

A aplicação de todos os padrões possíveis em seqüência é denominada uma *época de treinamento* (ver aula 3).

Vamos agora aplicar outra época de treinamento. Comecemos com $\vec{A}_1$. O nível de ativação é:

$$u = \vec{A}_1 . \vec{\omega}_4 = (0,7).(0,3) + (0,8).(0,7) = 0,21 + 0,56 = 0,77 > 0 \Rightarrow s = +1.$$

O erro é nulo e o vetor de pesos não muda: $\vec{\omega}_5 = \vec{\omega}_4$.

Vejamos agora o resultado da aplicação de $\vec{A}_2$. O nível de ativação é:

$$u = \vec{A}_2 . \vec{\omega}_5 = (0,7).(0,7) + (0,8).(0,3) = 0,49 + 0,24 = 0,73 > 0 \Rightarrow s = +1.$$

O erro é nulo e o vetor de pesos não muda: $\vec{\omega}_6 = \vec{\omega}_5$.

Note que agora não há mais erro na resposta do Perceptron quando o padrão $\vec{A}_2$ é apresentado. A regra de aprendizado do Perceptron mudou o vetor de pesos de tal maneira que o erro desapareceu e a resposta passou a ser a resposta desejada.

Vamos agora aplicar o padrão $\vec{B}_1$, para o qual também houve erro na primeira época. O nível de ativação é:

$$u = \vec{\omega}_6 . \vec{B}_1 = (0,7).(-0,6) + (0,8).(0,3) = -0,42 + 0,24 = -0,18 < 0 \Rightarrow s = 0.$$

O erro agora é nulo e $\Delta\vec{\omega} = 0$. Também para este padrão a regra de mudança dos pesos funcionou, fazendo o erro convergir para zero.

Falta agora aplicar o padrão $\vec{B}_2$ à rede. Como o vetor de pesos não mudou até agora nesta segunda época de treinamento, o valor atual de $\vec{\omega}$, $\vec{\omega}_7$, é igual a $\vec{\omega}_4$. Como já vimos que $\vec{\omega}_4 \cdot \vec{B}_2 < 0 \Rightarrow$ s = 0, o erro é nulo e o vetor de pesos não precisa ser modificado. O Perceptron acertou.

Portanto, neste exemplo, em apenas uma época de treinamento a regra de aprendizado do Perceptron fez o vetor de pesos conectando as duas unidades de entrada à de saída convergir para um vetor capaz de fornecer as saídas corretas. Em outras palavras, o Perceptron foi *treinado* para fornecer a classificação correta dos padrões.