iMobilis

SEVA

Treinamento e Aprendizado em RNAs

André Siqueira Ruela

UFOP
Universidade Federal de Ouro Preto

decom
departamento de computação

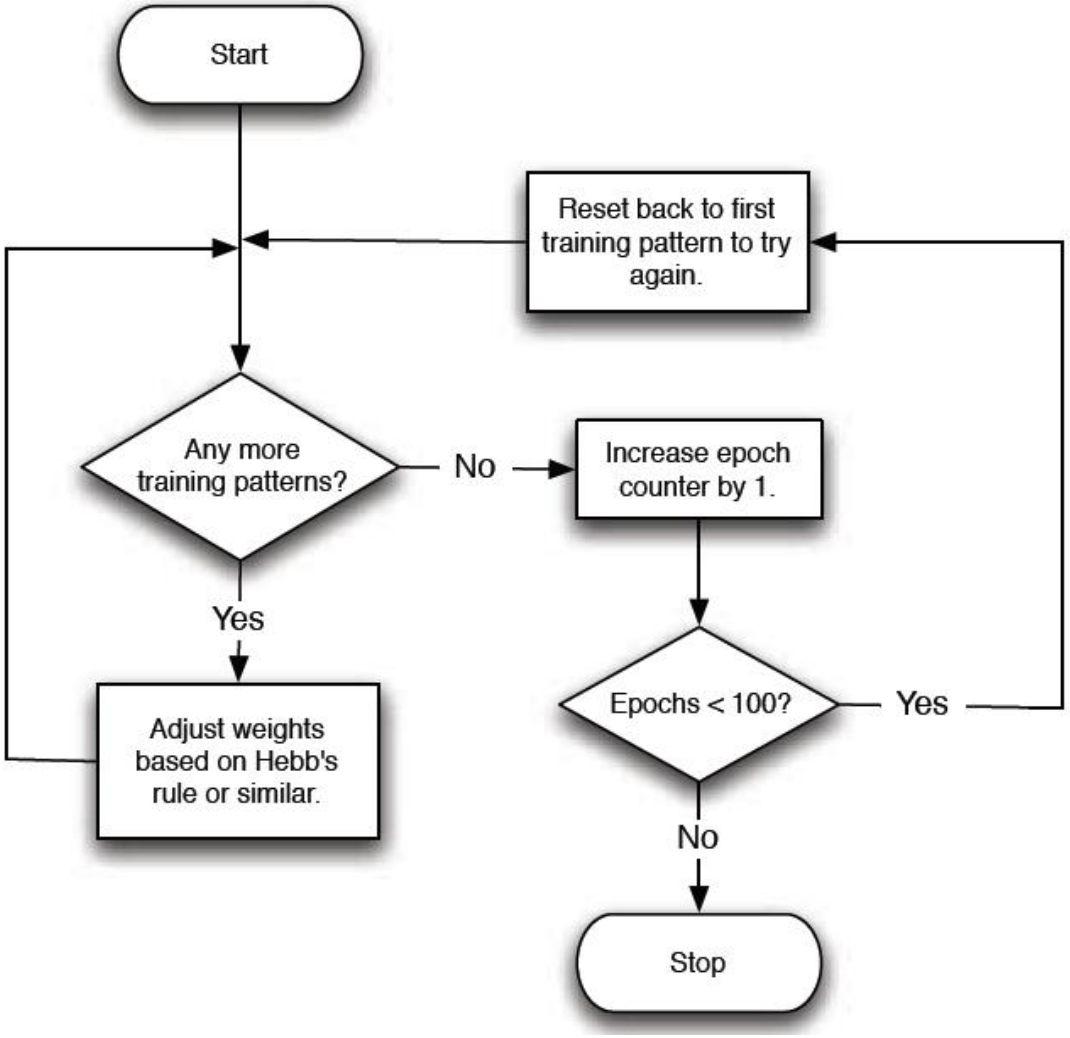
Sumário

- Treinamento Não Supervisionado
- Treinamento Supervisionado
- Cálculo do Erro
- Regra de Hebb
- Regra do Delta

Treinamento Não Supervisionado

- Nenhuma saída antecipada ou resposta desejada é fornecida.
- Usualmente utilizado para a classificação dos padrões de entrada.
- Em geral, apenas um único neurônio na camada de saída dispara uma resposta.
- Tal resposta classifica o padrão de entrada.

Treinamento Não Supervisionado



UFOP
Universidade Federal de Ouro Preto

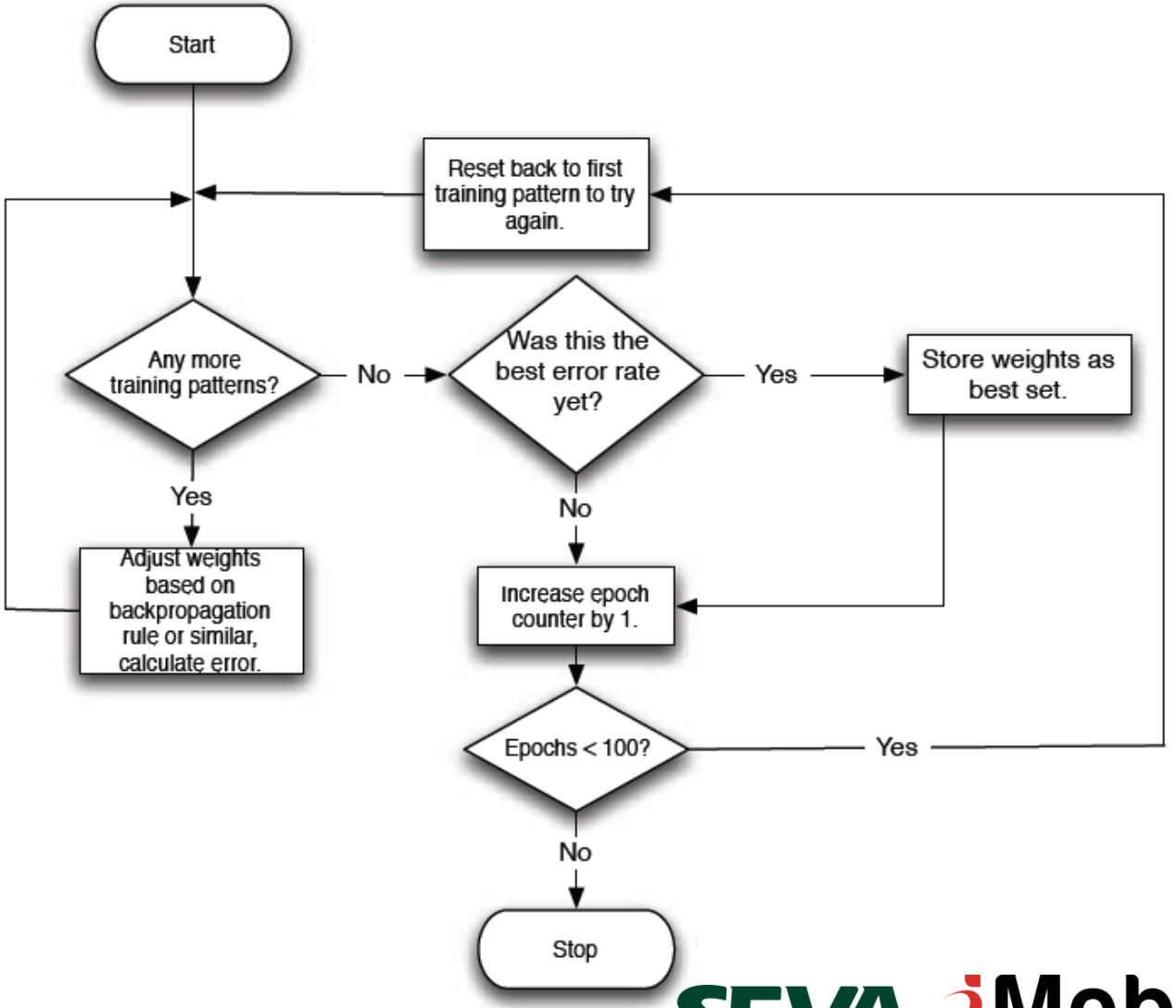
SEVA iMobilis

decom
departamento de computação

Treinamento Supervisionado

- Um conjunto de resultados esperados é fornecido.
- A diferença entre a saída atual e a saída desejada consiste no erro.
- O cálculo do erro permite que a matriz de pesos seja ajustada.
- A rede é aperfeiçoada a cada série de treinamentos.

Treinamento Supervisionado



Cálculo do Erro

- Erros existem tanto em treinamentos supervisionados, quanto em não supervisionados.

- Sob certo ponto de vista, o objetivo de todo treinamento é minimizar a taxa de erro na realização de uma tarefa.

UFOP
Universidade Federal de Ouro Preto

SEVA iMobilis

decom
departamento de computação

Cálculo do Erro em Treinamento Supervisionado

- O erro pode ser calculado para cada elemento do conjunto de entrada.
- Após o cálculo de todos os erros individuais, pode-se calcular a raiz da média dos quadrados (RMS).
- O RMS atua como uma taxa de erro global de toda a RNA.
- ErrorCalculation Class.

Cálculo do Erro em Treinamento Supervisionado

Equation 4.1: Root Mean Square Error (RMS)

$$x_{rms} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2} = \sqrt{\frac{x_1^2 + x_2^2 + \ldots + x_n^2}{n}}$$

Equation 4.2: RMS for a Neural Network

$$x_{rms} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} (actual_i - ideal_i)^2}$$

UFOP Universidade Federal de Ouro Preto

SEVA iMobilis

decom departamento de computação

Cálculo do Erro em Treinamento Não Supervisionado

- O cálculo do erro em treinamento não supervisionado não é uma tarefa óbvia.
- O ideal é que um neurônio dispare em um nível elevado para uma entrada específica.
- Se este não é o caso, os pesos entre os neurônios são ajustados para que as conexões sejam fortalecidas.

Algoritmos de Treinamento

- O treinamento ocorre quando os pesos das conexões entre os neurônios são modificados.
- O algoritmo de treinamento ou as “regras de aprendizado” determinam como estes pesos serão modificados.
- O treinamento é um processo iterativo (época) realizado sobre o mesmo conjunto de entrada.

Algoritmos de Treinamento

- As regras de aprendizado modificam os pesos gradativamente a cada época.
- matrixPesos[i][j] += regraAprendizado(...);
- Os pesos anteriores não são descartados.
- O treinamento continua até que a taxa de erro seja reduzida a uma margem aceitável ou até atingir o número máximo de épocas.

Algoritmos de Treinamento

- Quando uma taxa de erro é apresentada ao algoritmo de treinamento, trata-se de um treinamento supervisionado.

- Em um treinamento não supervisionado, as regras de aprendizagem permitem que a rede corrija o seu erro sozinha.

Regras de Hebb (não supervisionado)

```
protected double learningRule(
  double rate, double input, double output)
{
  return rate*input*output;
}
```

- A regra de Hebb tem como parâmetro uma taxa de aprendizagem, uma entrada do neurônio “i” e uma saída do neurônio “j”.

- O resultado é somado ao peso entre os neurônios “i” e “j”.

Regras de Hebb

- *"Neurons that fire together, wire together".*
- Fortalece as conexões entre neurônios de ativações similares.
- Enfraquece as conexões entre neurônios de ativações distintas.

Case	Neuron i Value	Neuron j Output	Hebb's Rule	Weight Delta
Case 1	+1	-1	1*1*-1	-1
Case 2	-1	+1	1*-1*1	-1
Case 3	+1	+1	1*1*1	+1

Regra de Hebb

- Hebb Class.

```
***Beginning Epoch #1***
Presented [-1.0,-1.0] result=0.0,delta w1=-0.0,delta w2=-0.0
Presented [-1.0,1.0] result=-1.0,delta w1=1.0,delta w2=-1.0
Presented [1.0,-1.0] result=2.0,delta w1=2.0,delta w2=-2.0
Presented [1.0,1.0] result=0.0,delta w1=0.0,delta w2=0.0
```

- A segunda e terceira entradas produzem resultados “fortes”.

- A regra de Hebb tende a fortalecer as saídas nas direções para as quais já havia uma tendência sendo seguida.

Regra do Delta (supervisionado)

Equation 4.4: The Delta Rule

$$\Delta w_{ij} = 2 \mu x_i (ideal - actual)_j$$

- As respostas desejadas são fornecidas.
- O peso da conexão entre os neurônios “i” e “j” é ajustado de acordo com o cálculo do erro.

(ideal – atual)

UFOP Universidade Federal de Ouro Preto

SEVA iMobilis

decom departamento de computação

Regra do Delta

```
protected double trainingFunction(
    double rate, double input, double ideal, double actual)
{
  return rate*input*(ideal-actual);
}
```

- Delta Class.

- O algoritmo treina por 100 épocas.

```
***Beginning Epoch #100***
Presented [0.0,0.0,1.0] anticipated = 0.0, actual = -0.33333333131711973, error = 0.33333333131711973
Presented [0.0,1.0,1.0] anticipated = 0.0, actual = 0.333333333558949, error = -0.333333333558949
Presented [1.0,0.0,1.0] anticipated = 0.0, actual = 0.33333333370649876, error = -0.33333333370649876
Presented [1.0,1.0,1.0] anticipated = 1.0, actual = 0.6666666655103011, error = 0.3333333448969893
```

Regra do Delta

- A rede aproxima 0.333 para o valor antecipado 0 e 0.666 para o valor antecipado 1.

- A rede neural nunca consegue retornar exatamente a resposta desejada, mas pode chegar perto.

- Apesar de ser eficiente no ajuste dos pesos, a regra do delta não é comumente usada.