


I 1 1 1 1
 1
1
1 1 1 1
1
1
D2 1 1 1 1

Rede Neural Artificial Aplicada ao Reconhecimento de Padrões usando o Método *Back-Propagation*

Divina Silva

Uberlândia, Dezembro/2000.

Rede Neural Artificial Aplicada ao Reconhecimento de Padrões usando o Método *Back-Propagation*

Divina Silva

Monografia apresentada ao Curso de Ciência da Computação do Centro Universitário do Triângulo - Unit, como requisito básico à obtenção do grau de Bacharel em Ciência da Computação, sob a orientação do Prof. Clarimundo Machado Moraes Júnior.

Uberlândia, Dezembro/2000.

Rede Neural Artificial Aplicada ao Reconhecimento de Padrões usando o Método *Back-Propagation*

Divina Silva

Monografia apresentada ao Curso de Ciência da Computação do Centro Universitário do Triângulo - Unit, como requisito básico à obtenção do grau de Bacharel em Ciência da Computação.

Clarimundo Machado Moraes Júnior,
Msc.

(Orientador)

Marcos Ferreira de Rezende, Msc.

(Coordenador de Curso)

Elmo Batista Faria, Msc.

(Avaliador)

Alfen Ferreira de Souza Júnior, Msc.

(Avaliador)

Uberlândia, Dezembro/2000.

Dedicatória

À meus pais, irmãs (Tina, Minha linda, Voíde, Meirinha),

À todos os meus professores, amigos e colegas,

Ao meu grande amigo 'Dib' que não está mais presente, mas que continua no meu coração.

Agradecimentos

À Deus, que muito me iluminou ,

À minha irmã Dimária, pelo essencial apoio e ajuda,




























Ao meu amigo Germano, pela valiosa dedicação e contribuição e


Ao meu orientador Clarimundo pela orientação na condução deste trabalho.


RESUMO

Este trabalho visa enfatizar os conceitos de Redes Neurais Artificiais, uma técnica de Inteligência Artificial, onde serão discutidos: conceitos, aplicações, componentes, processos, algoritmos, vantagens, desvantagens, histórico, topologias, neurocomputação e faz uma breve descrição sobre os principais tipos de redes usadas. Além disso, compreende-se tanto o funcionamento quanto o desenvolvimento do algoritmo *Back-propagation* e Reconhecimento de Padrões.

Sumário

	Introdução às Redes Neurais Artificiais	
1.1.	O que é uma Rede Neural?	
1.2.	O que é uma Rede Neural Artificial ?	
1.3.	Histórico	
1.4.	Por que usar Redes Neurais Artificiais?	
1.5.	Em que são usadas as Redes Neurais Artificiais ?	
1.6.	Características das Redes Neurais Artificiais	
1.7.	Tipos de Redes Neurais Artificiais	
1.8.	Topologia	 17
2.	Processos de Aprendizado de uma RNA	
2.1	Modelos de Redes Neurais Artificiais	
2.2.	Treinamento Supervisionado	
2.2.1.	Rede Perceptron.....	
2.2.2.	Regra Delta.....	
2.3.	Função de Transferência	
2.4.	Neurocomputação.....	
3.	Desenvolvimento de Aplicações.....	
3.1.	Coleta dos dados e separação em conjuntos	
3.2.	Configuração da rede	
3.3.	Treinamento	
3.4.	Teste	 34
3.5.	Integração	
4.	O Algoritmo Back-Propagation	
4.1.	As Deficiências do Back-Propagation	
5.	Conclusão.....	
6.	Referências Bibliográficas.....	

Apêndice A – Uma breve descrição de Reconhecimento de Padrões...

Apêndice B – Desenvolvimento do Back-Propagation em uma RNA.....



Lista de Figuras

Figura 1.1. Sinapse	2
Figura 1.2. Neurônio Biológico	2
Figura 1.3. Neurônio Biológico de McCulloch – Pitts	12
Figura 1.4. Organização em Camadas	13
Figura 1.5. Topologia Linear	18
Figura 1.6 Topologia Retroalimentada	18
Figura 1.7. Topologia Toda Conectada	19
Figura 1.8. Topologia Multinível	19
Figura 2.1. Regra Delta	24
Figura 2.2. Esquema de Treinamento do Perceptron	25
Figura 2.3. Funções de Ativação	26
Figura 4.1. Superfície de Erro de uma Rede Complexa	42
Figura A.1. Representação do Algoritmo ‘2’	49
Figura A.2. Treinamento para um Grupo de ‘10’ Algoritmos.....	51
Figura B.1. Gráfico da Superfície de Erro.....	53
Figura B.2. Gráfico com Peso e Inclinação Ajustados.....	54
Figura B.3. Gráfico dos Erros.....	55

Lista de Tabelas

Tabela 1.1. - Comparação entre tipos de Redes..... 15


Tabela 2.4. - Quadro comparativo entre computadores e neurocomputadores..... 29

Tabela 2.5. - Quadro comparativo entre computadores e neurocomputadores.....

1. Introdução às Redes Neurais Artificiais

1.1. O que é uma Rede Neural?

O cérebro humano é considerado o mais fascinante processador baseado em carbono existente, sendo composto por aproximadamente 100 bilhões de neurônios. Todas as funções e movimentos do organismo estão relacionados ao funcionamento destas pequenas células.

Os neurônios se comunicam através de **sinapses** (região onde dois neurônios entram em contato e através do qual os impulsos nervosos são transmitidos entre eles, como mostra a figura 1.1), juntos formam uma grande *Rede Neural*. As sinapses transmitem estímulos através de diferentes concentrações  Na^+ (Sódio) e K^+ (Potássio) e o resultado disto pode ser estendido por todo o corpo humano. [9]

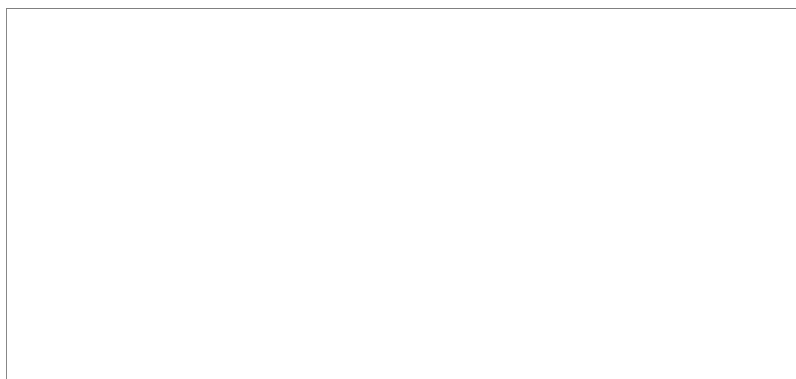


Figura 1.1- Sinapse.

Esta grande rede proporciona uma fabulosa capacidade de processamento e armazenamento de informação. Muitos até desconhecem sobre como os neurônios ensinam a si mesmos a processar

informações, até então, existem grandes quantidades de teorias que tentam explicar este fenômeno. No corpo humano, um neurônio coleta sinais de outros através de uma organização de finas estruturas chamadas dendritos (**dendrite**). O corpo do neurônio, também chamado de **somma**, é responsável por coletar e combinar informações vindas de outros neurônios. Finalmente, temos o axônio (**axon**), que é constituído de uma fibra tubular que pode alcançar até alguns metros, e é responsável por transmitir os impulsos para outras células. Como mostra a figura 1.2.



Figura 1.2 – Neurônio biológico.

1.2. O que é uma Rede Neural Artificial ?

Redes Neurais Artificiais nada mais são que, programas de computadores que simulam uma Rede Neural Biológica, ou seja, processamento de informações inspiradas em um caminho de sistemas nervosos biológicos, como o cérebro, processando informações. O elemento chave deste paradigma é a estrutura de processamento de informação de sistemas. É composto de um grande número de processamento de elementos interconectados trabalhando em união para resolver problemas específicos. As RNAs, como pessoas, aprendem com exemplos, e são configuradas por uma aplicação específica, semelhante ao reconhecimento de exemplos ou classificação de dados, através do processo de reconhecimento. Conhecimento em sistemas biológicos envolve ajustes para as conexões sinápticas que existem entre os neurônios. Uma RNA é baseada no conhecimento sobre sistemas nervosos biológicos.

As RNAs consistem em um método de solucionar problemas de *inteligência artificial*, construindo um sistema que tenha circuitos que simulem o cérebro humano, inclusive seu comportamento, ou seja, aprendendo, errando e fazendo descobertas. As RNAs são mais que isso, são técnicas computacionais que apresentam um modelo inspirado na estrutura neural de organismos inteligentes e que adquirem conhecimento através de experiência. Uma grande Rede Neural Artificial pode ter centenas ou milhares de unidades de processamento, enquanto que o cérebro de um mamífero pode ter muitos bilhões de neurônios[8].

1.3. Histórico

O primeiro neurônio artificial foi idealizado em 1943 pelo neurofisiologista Warren McCulloch e pelo lógico Walter Pitts. Mas a tecnologia disponível na época não permitiu que eles fizessem muito.

Para um breve histórico sobre Redes Neurais Artificiais pode se começar por três das mais importantes publicações iniciais, desenvolvidas por: McCulloch e Pitts (1943), Hebb (1949), e Roseblatt (1958). Estas publicações introduziram o primeiro modelo de redes neurais simulando

“máquinas”, o modelo básico de rede de auto-organização, e o modelo Perceptron de aprendizado supervisionado, respectivamente.

O Psicólogo Donald Hebb (1949), demonstrou que a capacidade da aprendizagem em redes neurais vem da alteração da eficiência sináptica, isto é, a conexão somente é reforçada se tanto as células pré-sinápticas quanto as pós-sinápticas estiverem excitadas.

Roseblatt (1958) mostrou em seu livro (Principles of Neurodynamics) o modelo dos "Perceptrons". Nele, os neurônios eram organizados em camada de entrada e saída, onde os pesos das conexões eram adaptados a fim de se atingir a eficiência sináptica.

Em 1960 surgiu a rede ADALINE(ADaptative LInear NETwork) e o MADALINE(Many ADALINE) perceptron, proposto por Widrow e Hoff. O ADALINE/MADALINE utilizou saídas analógicas em uma arquitetura de três camadas.

Alguns históricos sobre a área costumam “saltar” os anos 60 e 70 e apontam um reinício da área com a publicação dos trabalhos de Hopfield (1982) relatando a utilização de redes simétricas para otimização e de Rumelhart, Hinton e Williams que introduziram o poderoso método Backpropagation.

Entretanto, para se ter um histórico completo, devem ser citados alguns pesquisadores que realizaram, nos anos 60 e 70, importantes trabalhos sobre modelos de redes neurais em visão, memória, controle e auto-organização como: Amari, Anderson, Cooper, Cowan, Fukushima, Grossberg, Kohonen, von der Malsburg, Werbos e Widrow.

Seguindo o período inicial de entusiasmo, este campo sobreviveu a um período de frustração e má reputação. Durante este período quando o financiamento e o suporte profissional eram mínimos, importantes avanços foram feitos por relativamente poucos pesquisadores[3].

Na ordem cronológica ocorreu o seguinte:

- **1943:** McCULLOUGH e PITTS estabeleceram as bases da neurocomputação, com modelos matemáticos.
- **1949:** HEBB traduziu matematicamente a sinapse dos neurônios biológicos.
- **1951:** MINSKI construiu o Snark, primeiro neurocomputador com capacidade de aprendizado, ou seja, ajustava automaticamente os pesos entre as sinapses. Não executou nenhuma função útil.
- **1957:** ROSENBLATT concebeu o "perceptron", que era uma rede neural de duas camadas, usado no reconhecimento de caracteres.
- **1962:** WIDROW desenvolveu um processador para redes neurais e fundou a primeira empresa de circuitos neurais digitais, a Memistor Corporation.
- **1967:** Fim das verbas destinadas à pesquisa de redes neurais.
- **1974:** WERBOS lançou bases para o algoritmo de retropropagação (*backpropagation*).

As simulações de Redes Neurais Artificiais, parecem ser de um recente descobrimento. Entretanto, este campo foi notado antes da invenção de computadores, e tem sobrevivido no mínimo em grande atraso a várias eras. É muito importante ressaltar o avanço que teve com o uso de execuções acessíveis de comandos de computador. [8] e [2]

1.4. Por que usar Redes Neurais Artificiais?

As RNAs com sua notável habilidade de derivar significados de dados complicados ou imprecisos, pode ser usada para extrair exemplos, modelos e detectar direções/tendências que são tão complexas e são notificadas tanto por humanos quanto por outras técnicas de computador. Uma RNA pode ser entendida como uma categoria inteligente de informação utilizada para análise e que pode gerar resultados em situações novas com base no aprendizado. Elas possuem algumas vantagens e desvantagens:

VANTAGENS:

- **Qualidade superior:** as RNAs permitem análises superiores às conseguidas com técnicas estatísticas.
- **Competitividade:** empresas que tenham conseguido RNAs bem elaboradas possuem maior poder de fogo frente aos seus concorrentes, dado que essa é uma tecnologia ainda nova e pouco conhecida.
- **Auto-aprendizado:** não necessitam de conhecimentos de especialistas para tomar decisões; elas se baseiam unicamente nos exemplos históricos que lhes são fornecidos: não é necessário informar porque tal situação resultou em tal decisão no passado, ou porque tal decisão resultou em tal consequência.
- **Implementação mais rápida:** para algumas redes, o tempo necessário para se implementar uma RNA é menor que o utilizado para a construção de um sistema especialista equivalente, além do menor custo envolvido.
- **Imunidade a falhas:** como as unidades de rede operam em paralelo, a destruição ou defeito em um de seus modos não torna a rede inoperante, podendo até mesmo não causar grandes problemas em seu funcionamento.
- **Capacidade de generalização:** mesmo com dados incompletos ou imprecisos as RNAs podem preencher as lacunas sem sofrer degradação; é algo parecido com a interpolação e extrapolação da estatística, porém operacionalmente muito diferente.
- **Imunidade à ruídos:** os dados reais sempre contém ruídos (variações aleatórias adicionadas aos valores originais); as RNAs conseguem separar o ruído da informação irrelevante, tendo sido utilizadas mesmo como filtros de dados.
- **Adaptabilidade:** uma vez construída uma rede eficiente em dada aplicação, ela pode ser utilizada em aplicações de tempo-real, sem necessidade de ter sua arquitetura alterada a cada atualização; basta que seja retreinada com base nos novos dados históricos que formem surgindo.
- **Democratização:** através de arquiteturas e algoritmos gerais que podem ser inseridos em softwares e hardwares de rede, os executivos poderão se utilizar dessa ferramenta, à medida em que as RNAs se tornarem mais conhecidas, sem necessidade de construir outros tipos de redes ou de desenvolver novos algoritmos.

-

DESVANTAGENS:

- **Resultados desconcertantes:** as RNAs podem chegar a conclusões que contrariem as regras e teorias estabelecidas, bem como considerar dados irrelevantes como básicos; somente o bom senso do profissional experiente saberá tratar tais casos.
- **Treinamento demorado:** o treinamento de uma RNA, dependendo da aplicação pode ser demorado (horas ou mesmo dias).
- **Hardware high-tec:** o hardware necessário para o treinamento deve ser de processamento rápido: um PC pode ser suficiente, mas um Cray (um grande computador, que possui por volta de 120 processadores ou mais) é uma boa possibilidade de uso; podem ser necessárias placas especiais de processamento paralelo, além do inevitável coprocessador aritmético.
- **Caixa-preta:** é impossível saber porque a RNA chegou a tal conclusão; seus critérios decisórios são encriptados, não se sabendo até o momento que pesos são relevantes à tomada de uma dada decisão; os milhares de pesos não aceitam interpretação e nem são passíveis de interpretação lógica:

sabe-se apenas que funcionam.

- **Volume grande de dados:** para uma RNA poder aprender corretamente necessita de milhares de dados históricos; a carência de dados passados relevantes em quantidade suficiente torna a rede inaplicável.
- **Preparação de dados:** os dados de entrada necessitam de tratamento prévio: devem ser normalizados, em alguns casos fuzificados (usando lógica fuzzy) e devem ser cuidadosamente selecionados para que a rede seja corretamente ensinada a agir: dados de má qualidade produzem resultados falhos (como por exemplo omitir algum parâmetro importante no treinamento).
- **Trabalho artesanal:** não há regras gerais para se determinar o volume de dados de entrada para treinamento, quantas camadas devem ser utilizadas, a melhor estratégia de treinamento, que percentagem de dados deve ser destinada ao treinamento e ao teste da rede: esses parâmetros só podem ser estabelecidos através de bom-senso, experiência com redes e de tentativa e erro (heurísticas).

1.5. Em que são usadas as Redes Neurais Artificiais ?

As aplicações se dividem em várias categorias principais:

Classificação:

Negócios

- Classificação de crédito e estimativa de riscos, avaliação de seguros de riscos, detecção de fraudes.
- Detecção de comportamento pertencentes a atividades internas, análise de marketing.
- Verificação de assinaturas, invenção de controles.
- Reconhecimento de voz.

Planejamentos

- Diagnose de defeitos de dispositivos, máquinas, processamento de sinais e reconhecimento de caracteres, dígitos.
- Supervisão de processos, análise de erros de processos, reconhecimento de voz.
- Visão de máquinas, classificação de sinais de radar.

Segurança

- Reconhecimento de face, verificação de fala, análise de impressão digital, etc.

Medicina

- Diagnose geral, detecção de falha do coração.

Ciência

- Reconhecimento de genes, classificação botânica, identificação de bactérias.

Prognósticos

- Vendas futuras, requerimento de produções, performance de mercado.
- Indicadores econômicos, requerimento de energia, variáveis baseadas na determinação do tempo.

Deteção de Novidades

- Inspeção de falha, performance, detecção de fraude.
- Detectando características futuras.

1.6. Características das Redes Neurais Artificiais

Uma Rede Neural Artificial é composta por várias unidades de processamento, cujo funcionamento é bastante simples. Essas unidades, geralmente são conectadas por canais de comunicação que estão associados a determinado peso. As unidades fazem operações apenas sobre seus dados locais, que são entradas recebidas pelas suas conexões. O comportamento inteligente de uma Rede Neural Artificial vem das iterações entre as unidades de processamento da rede.

A operação de uma unidade de processamento, proposta por McCulloch e Pitts em 1943, pode ser resumida da seguinte maneira:

- sinais são apresentados à entrada;
- cada sinal é multiplicado por um número, ou peso, que indica a sua influência na saída da unidade;
- é feita a soma ponderada dos sinais que produz um nível de atividade;
- se este nível de atividade exceder um certo limite (threshold), a unidade produz uma determinada resposta de saída.



Figura 1.3. Neurônio biológico de McCulloch - Pitts.

O neurônio de McCulloch (figura 1.3) era extremamente simples, diante das informações que já estavam disponíveis sobre o comportamento elétrico da célula nervosa. Era um dispositivo binário, onde, a sua saída poderia ser pulso ou não pulso, e as suas várias entradas tinham ganho arbitrário e poderiam ser excitatórias ou inibitórias. Para determinar a saída do neurônio, calculava-se a soma ponderada das entradas com os respectivos ganhos como fatores de ponderação, positivos nos casos excitatórios e negativos nos casos inibitórios. Portanto, ele fazia uma analogia entre células vivas e o processo eletrônico, simulando o comportamento do neurônio natural, onde o neurônio possuía apenas uma saída, que era uma função de entrada (threshold) da soma do valor de suas diversas entradas.

Suponha que se tenha p sinais de entrada X_1, X_2, \dots, X_p e pesos w_1, w_2, \dots, w_p e limitador t ; com sinais assumindo valores booleanos (0 ou 1) e pesos valores reais.

Neste modelo, o nível de atividade a é dado por:

$$a = w_1X_1 + w_2X_2 + \dots + w_pX_p$$


A saída y é dada por:

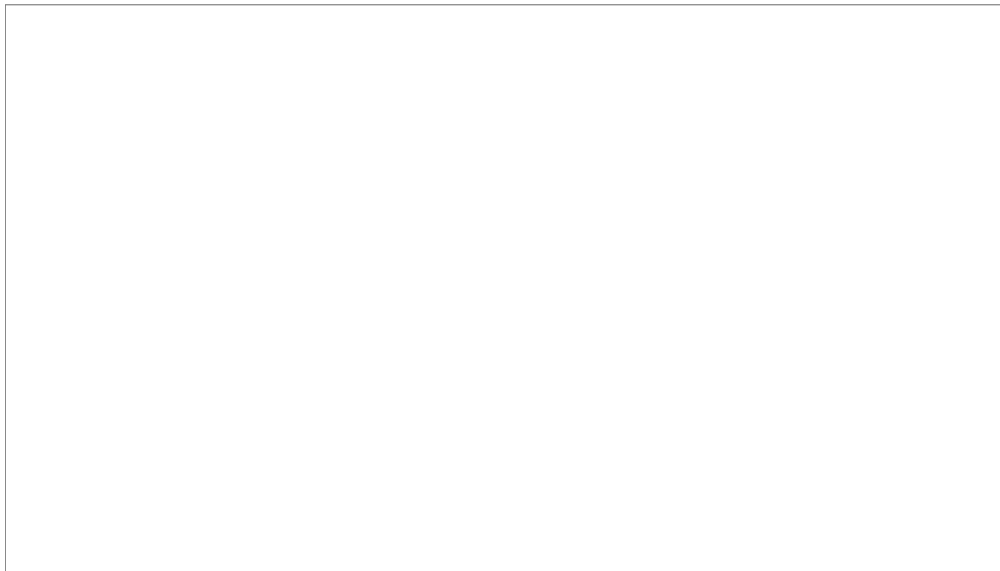
$$y = 1, \text{ se } a \geq t \text{ ou}$$

$$y = 0, \text{ se } a < t.$$

A maioria dos modelos de redes neurais possui alguma regra de treinamento, onde os pesos de suas conexões são ajustados de acordo com os padrões apresentados. Em outras palavras, elas aprendem através de exemplos.

Arquiteturas neurais são tipicamente organizadas em camadas, com unidades que podem estar conectadas às unidades da camada posterior. Como se pode visualizar na figura 1.4. :

Figura 1.4. Organização em camadas 



Usualmente as camadas são classificadas em três grupos:

- **Camada de Entrada (Input Layer):** onde os padrões são apresentados à rede;
- **Camadas Intermediárias ou Escondidas (Hidden layer):** onde é feita a maior parte do processamento, através das conexões ponderadas; podem ser consideradas como extratoras de características;
- **Camada de Saída (Output layer):** onde o resultado final é concluído e apresentado.

Os neurônios que recebem diretamente as entradas da rede constituem o que se chama de **camada de entrada**.

Os neurônios que recebem como entradas as saídas da camada de entrada, constituem a **segunda camada** e assim sucessivamente até a camada final que é a **camada de saída**.

As camadas internas que não são nem a camada de entrada e nem a camada de saída são geralmente referidas como **camadas ocultas**.

Quando Redes Neurais Artificiais de uma só camada são utilizadas, os padrões de treinamento apresentados à entrada são mapeados diretamente em um conjunto de padrões de saída da rede, ou seja não é possível a formação de uma representação interna. Neste caso, a codificação proveniente do mundo exterior deve ser suficiente para implementar esse mapeamento.

Tal restrição implica que padrões de entrada similares resultem em padrões de saída similares, o que leva o sistema à incapacidade de aprender importantes mapeamentos. Como resultado, padrões de entrada com estruturas similares, fornecidos do mundo externo, que levem a saídas diferentes não são possíveis de serem mapeados por redes sem representações internas, isto é, sem camadas intermediárias.

Minsky e Papert analisaram matematicamente o Perceptron e demonstraram que redes de uma camada não são capazes de solucionar problemas que não sejam linearmente separáveis. Como não acreditavam na possibilidade de se construir um método de treinamento para redes com mais de uma camada, eles concluíram que as Redes Neurais Artificiais seriam sempre suscetíveis a essa limitação.

Contudo, o desenvolvimento do algoritmo de treinamento *back-propagation*, por Rumelhart, Hinton e Williams em 1986, precedido por propostas semelhantes ocorridas nos anos 70 e 80, mostrou que é possível treinar eficientemente redes com camadas intermediárias, resultando no modelo de Redes

Neurais Artificiais mais utilizado atualmente, as redes Perceptron Multi-Camadas (MLP), treinadas com o algoritmo *back-propagation*.

Nessas redes, cada camada tem uma função específica. A camada de saída recebe os estímulos da camada intermediária e constrói o padrão que será a resposta. As camadas intermediárias funcionam como extratoras de características, seus pesos são uma codificação de características apresentadas nos padrões de entrada e permitem que a rede crie sua própria representação, mais rica e complexa, do problema.

Uma Rede Neural Artificial é especificada, principalmente pela sua topologia, pelas características dos nós e pelas regras de treinamento.

1.7. Tipos de Redes Neurais Artificiais

Alguns dos melhores tipos de Redes Neurais Artificiais conhecidos, estão descritos na tabela a seguir (veja tabela 1.1) [4]:

Tabela 1.1. – Comparação entre tipos de redes

Rede	Aplicações	Limitações	Comentário
Adaptive Resonance Theory	Reconhecimento de exemplos, especialmente se é complicado ou não familiar aos humanos(radar, timbre de voz)	Sensível a traduções, distorções, mudanças em escala.	Muito sofisticada, ainda não é aplicada a muitos problemas.
Rede	Aplicações	Limitações	Comentário
Avalanche	Reconhecimento contínuo da fala; ensinamento de comandos de motores para armas robóticas	Audição literal da gravação de sequência do motor – nenhum simples caminho altera a velocidade ou a interpolação dos movimentos.	Classe de Redes – De forma alguma uma única rede pode fazer todas estas tarefas.
Back-propagation	Junção da fala de um texto; controle adaptável armas robóticas; aplicações de empréstimos bancários.	Somente aprendizado supervisionado – exemplos corretos de entradas/saídas devem ser predominantes.	A rede mais popular hoje – funciona bem e é simples para aprender.
Bidirectional Associative Memory (BAM)	Endereçamento/ conteúdo para memória associativa	Baixa densidade de armazenamento; os dados devem estar corretamente codificados.	Rede muito fácil de aprender – bom instrumento educacional; une pares fragmentados de objetos a pares completos.
Boltzmann and Cauchy machines	Reconhecimento de exemplos para imagens,	Máquinas: Boltzmann tempo longo de	Redes simples na Qual função é usada para

	sondas, radares.	aprendizado. Cauchy: Geração de ruídos na estatística adequada de distribuição.	encontrar o mínimo global.
Brain state in a box	Extração de conhecimento de bases de dados.	Estrutura de decisão one-shot – não há raciocínio repetitivo.	Similar a BAM nas saídas fragmentadas e completadas.
Cerebellatron	Controle das ações de motores para armas robóticas.	Exige complexo controle de entrada.	Avalanche; pode combinar comandos com pesos diferentes interpolar movimentos precisos.
Counterpropagation	Compressão de imagens; análise estatística; pontuação de pedido de empréstimo.	Número grande de processamento de elementos e conexões exigidas a altas precisões p/ qualquer tamanho de problema.	Funçõam como uma auto programação; similar ao <i>back-propagation</i> , portanto mais simples, embora poderosa também.
Hopfield	Retorno de dados completos ou fragmentos de imagens.	Não aprende – pesos devem ser ajustados antecipadamente.	Pode ser implementada em uma grande escala.
Madaline Continuação Madaline...	Adaptável a anulação de radares de voz, modems, niveladores (imita suspensões de linhas telefônicas).	Assume um relacionamento linear entre entrada e saída.	Colocam abreviaturas para múltiplos elementos adaptáveis, potente lei de aprendizado, uso comercial há mais de 20 anos.
Neocognitron	Reconhecimento de impressão de caracteres.	Raramente exige um número grande de processamento de elementos e conexões.	Rede mais complicada já desenvolvida; não sensível a escalas diferentes, tradução, rotação; hábil a identificar caracteres complexos (como o chinês)....
Perceptron	Reconhecimento de caracteres digitados.	Não pode reconhecer caracteres complexos (como o chinês); sensível a diferença em escala; tradução, distorções.	É a RNA conhecida mais antiga; foi construída em hardware; raramente usada hoje.
Self-organizing map	Traça regiões geométricas (ex.: rede elétrica retangular) sobre outra (ex.: aeronave)	Exige aprendizado extensivo.	Mais efetivo que muitas técnicas de algoritmos para cálculo numérica corrente aerodinâmico.

1.8. Topologia

A modelagem da topologia da rede corresponde a descrição das interconexões entre os neurônios da rede. A topologia da rede tem reflexos significativos nas características e propriedades apresentadas pelo modelo.

Existem diversos tipos de topologias, entre as quais podemos citar: totalmente conectada, linear, retro-alimentada, multi-nível, matricial ou livre. Um modelo de Rede Neural Artificial pode também corresponder a uma combinação das topologias citadas acima.

As conexões entre neurônios, que caracterizam a topologia de rede, são normalmente representadas através de uma matriz de pesos $P = p_{ij}$. Nesta matriz, o elemento p_{ij} representa o peso da conexão que sai do neurônio i e entra no neurônio j . Seu valor pode ser inibitório (negativo) ou excitatório (positivo). Se não há conexão entre o neurônio i e o neurônio j , então, $p_{ij} = 0$.

As figuras abaixo apresentam o esquema de algumas topologias de interconexão entre neurônios.

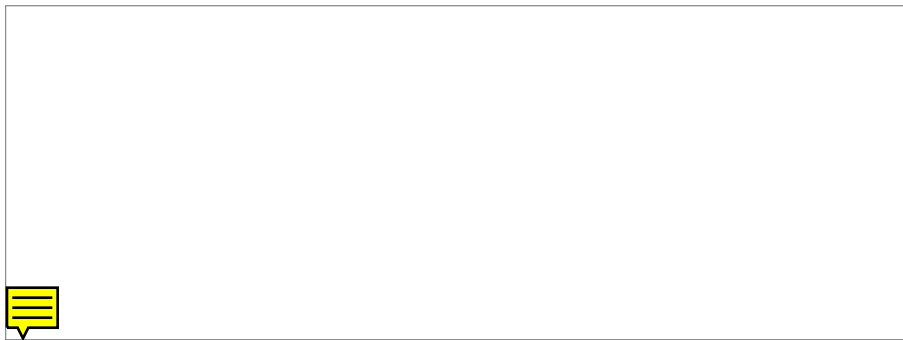


Figura 1.5 - Topologia Linear.

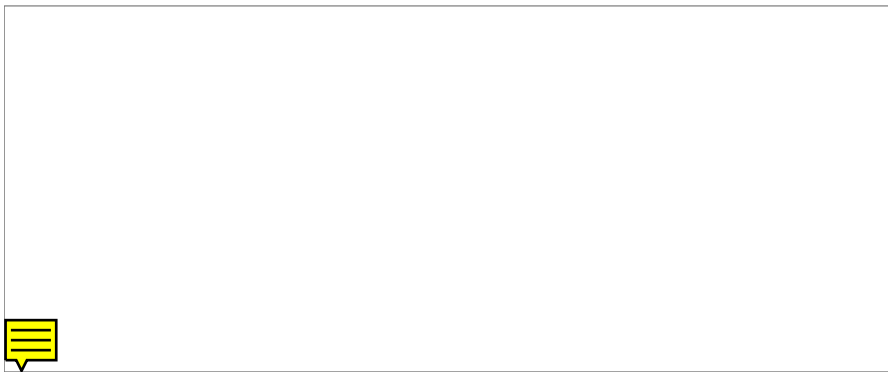


Figura 1.6 - Topologia Retroalimentada.

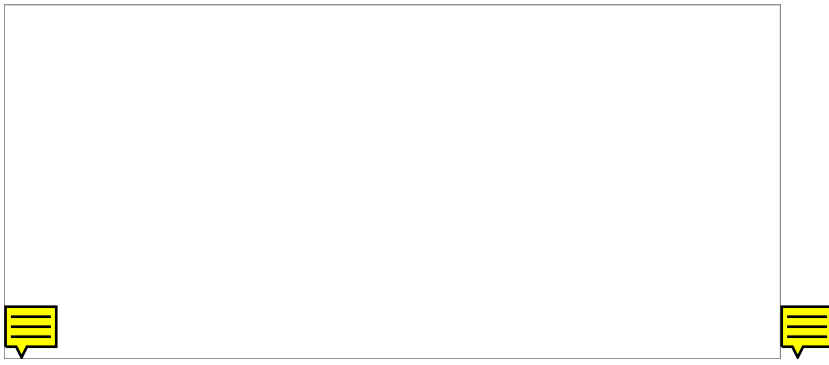


Figura 1.7 - Topologia toda conectada.

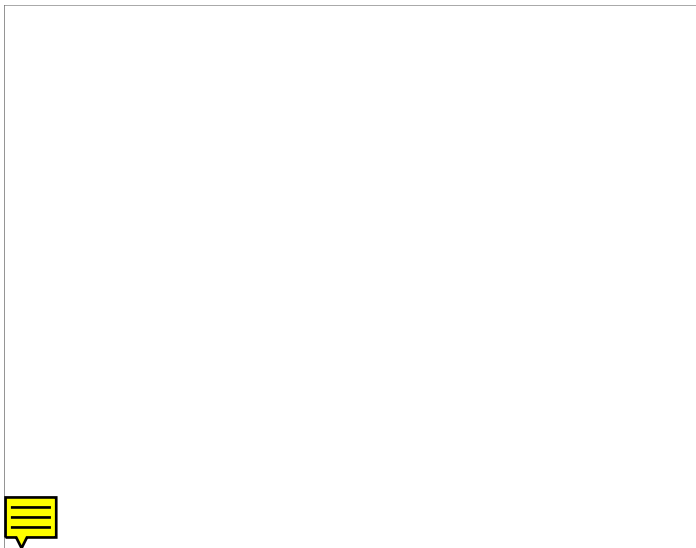


Figura 1.8 - Topologia multinível.

2. Processos de Aprendizado de uma RNA

Nesta etapa, serão analisados os processos de aprendizado, ou plasticidade[7].

A propriedade mais importante das RNAs é a habilidade de aprender de seu ambiente e com isso melhorar seu desempenho. Isso é feito através de um processo iterativo de ajustes aplicado a seus pesos (“treinamento”). O aprendizado ocorre quando a RNA atinge uma solução generalizada para uma classe de problemas.

Denomina-se algoritmo de aprendizado a um conjunto de regras bem definidas para a solução de um problema de aprendizado. Existem muitos tipos de algoritmos de aprendizado específicos para determinados modelos de RNAs, sendo que estes algoritmos diferem entre si principalmente pelo modo como os pesos são modificados.

Outro fator importante é a maneira pela qual uma RNA se relaciona com o ambiente. Categoriza-se as situações de aprendizado em dois tipos distintos. São eles:

- **Aprendizado Supervisionado:** ou aprendizado por associação, é treinado com fornecimento de exemplos compatíveis de entradas e saídas. Estas entradas-saídas em pares podem ser fornecidas por um professor (agente) externo, ou por um sistema na qual contém uma rede(auto

supervisão).

- **Aprendizado Não Supervisionado:** ou auto-organização, na qual uma unidade de saída é treinada a responder a um grupo de exemplos com entradas. Neste paradigma é suposto que o sistema descobrirá estatisticamente características consideráveis de dados de entradas. Diferente do paradigma de aprendizado supervisionado, não se sabe anteriormente o conjunto de categorias na qual os exemplos são classificados; Especialmente o sistema deve desenvolver a representação do próprio estímulo de entrada dele.

Diz-se que uma RNA aprende *off-line* se o aprendizado e a operação são de fases distintas. Uma RNA aprende *on-line* se ela aprende e opera ao mesmo tempo. Usualmente, o aprendizado supervisionado é executado off-line, e o aprendizado não supervisionado é executado on-line.

Denomina-se ciclo, como sendo uma apresentação de todos os N pares (entrada e saída) do conjunto de treinamento no processo de aprendizado. A correção dos pesos num ciclo pode ser executado de dois modos:

- 1) **Modo Padrão:** A correção dos pesos acontece a cada apresentação à rede de um exemplo do conjunto de treinamento. Cada correção de pesos baseia-se somente no erro do exemplo apresentado naquela iteração. Assim, em cada ciclo ocorrem N correções.
- 2) **Modo Batch:** Somente uma correção é feita por ciclo. Todos os exemplos do conjunto de treinamento são apresentados à rede, seu erro médio é calculado e a partir deste erro fazem-se as correções dos pesos.

2.1 Modelos de Redes Neurais Artificiais

Existe uma série de processos ou modelos de redes neurais artificiais dentre os quais se destacam os processos de aprendizado supervisionado por correção de erros. A maioria dos processos restantes é baseada em variações deste processo ou modelo básico.

Pode-se dividir os processos de redes em diversos grupos, de acordo com:

- Treinamento: pode ser auto-aprendizado (sem supervisão) ou aprendizado supervisionado;
- Regra de aprendizado: algoritmo competitivo ou algoritmo adaptativo por correção de erros;
- Interconexões dos neurônios: redes com neurônios sem realimentação ou com realimentação;
- Organização da rede: um nível único ou em diversos níveis(camadas);
- Distribuição das memórias: localizada ou distribuída;
- Classificação: separador linear ou não linear.

Dentre as divisões citadas acima, a regra de aprendizado que merece uma ênfase maior é a regra

de aprendizado do tipo competitivo encontrada nas redes de Hopfield e Kohonen. Este algoritmo é caracterizado pelas conexões laterais dos neurônios com seus vizinhos, estabelecendo assim uma “competição entre os neurônios” que levará a rede a um estado estável. Já as redes com aprendizado do tipo correção de erros são baseadas no princípio da adaptação e correção dos pesos de atuação de cada neurônio visando minimizar o erro na saída da rede, até que este responda da maneira desejada. A correção de erros está diretamente ligada ao aprendizado do tipo supervisionado ou tutorado.

2.2. Treinamento Supervisionado

2.2.1. Rede Perceptron

Genuína rede neural de múltiplos neurônios. Redes Neurais estudadas por Rosenblatt, Block, Minsky Papert e outros. O termo é frequentemente usado para referir uma camada única na classificação de exemplos de uma rede com unidades *threshold* e lineares [1].

O treinamento supervisionado do modelo de rede Perceptron, consiste em ajustar os pesos e os *thresholds* (quando o nível de atividade excede um certo limite) de suas unidades para que a classificação desejada seja obtida

Para a adaptação do *threshold* juntamente com os pesos podemos considerá-lo como sendo o peso associado a uma conexão, cuja entrada é sempre igual à -1 e adaptar o peso relativo a essa entrada.

Quando um padrão é inicialmente apresentado à rede, ela produz uma saída. Após medir a distância entre a resposta atual e a desejada, são realizados os ajustes apropriados nos pesos das conexões de modo a reduzir esta distância. Este procedimento é conhecido como **Regra Delta** (veja figura 2.1).



Figura 2.1. Regra Delta

2.2.2. Regra Delta

Tem-se, assim, o seguinte esquema de treinamento .

1. Iniciar todas as conexões com pesos aleatórios;
2. Repita até que o erro E seja satisfatoriamente pequeno ($E = e$).

2.1. Para cada par de treinamento (X, d) , faça:

2.1.1. Calcular a resposta obtida O ;

2.1.2. Se o erro não for satisfatoriamente pequeno $E > e$, então:

2.1.3. Atualizar pesos: $W_{\text{novo}} := W_{\text{anterior}} + \text{meta } E \cdot X$

Onde:

- O par de treinamento (X, d) corresponde ao padrão de entrada e a sua respectiva resposta desejada;
- O erro E é definido como: Resposta Desejada - Resposta Obtida ($d - O$);
- A taxa de aprendizado é uma constante positiva, que corresponde à velocidade do aprendizado (Veja figura 2.2)

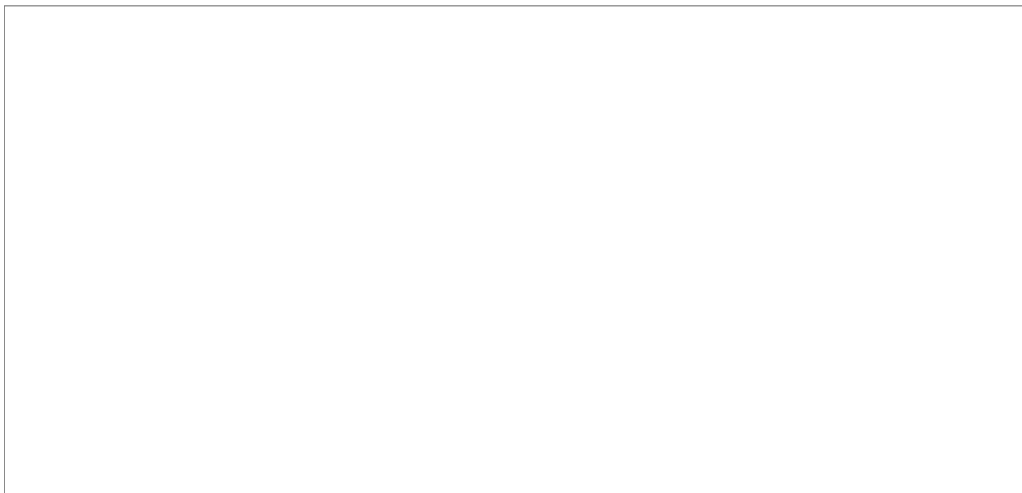


Figura 2.2. - Esquema de treinamento do Perceptron.

As respostas geradas pelas unidades são calculadas através de uma função de ativação(ou transferência). Existem vários tipos de funções de ativação (veja figura 2.3), as mais comuns são: Hard Limiter, Threshold Logic e Sigmoid.





Figura 2.3 - Funções de ativação

2.3. Função de Transferência

O comportamento de uma RNA (Rede Neural Artificial) depende de ambos os pesos e da função *input-output* (função de transferência ou ativação), que é especificado para as unidades e que se encaixa em uma das três unidades a seguir:

1. linear (ou rampa)
 2. *threshold* (limite)
 3. *sigmoid*
- Para as unidades lineares, a atividade de saída é proporcional para o total de saída.
 - Para as unidades de limite (*threshold*), a saída é fixada à um dos dois níveis, dependendo de se o total de entrada é maior ou menor que algum valor de limite (*threshold*).
 - Para unidades *sigmoid*, a saída varia continuamente, mas não tão linearmente quanto as alterações de entrada. As unidades *Sigmoid* parecem-se mais aos neurônios biológicos do que as unidades linear e *threshold* (estas facilitam o cálculo), mas todas três devem ser consideradas como aproximações.

Para fazer uma rede neural que executa alguma tarefa específica, deve-se escolher como as unidades serão conectadas umas as outras (veja figura 1.2), e fixar os pesos nas conexões apropriadamente. As conexões determinam se é possível para uma unidade influenciar outra. Os pesos especificam a força da influência[8].

Pode-se ensinar uma rede de três camadas executar uma tarefa particular usando o procedimento seguinte:

1. Apresenta-se a cadeia com exemplos de treinamento, que consiste em um padrão de atividades para as unidades de entrada junto com o padrão desejado de atividades para as unidades de saída (*input* e *output*).

2. Determina-se como que a atual saída da rede seja compatível a saída desejada.
3. Muda-se o peso de cada conexão de forma que a rede produz uma aproximação melhor da saída desejada.

- **Exemplo:**

Assuma que se queira uma rede para identificar dígitos de escrita à mão. Pode-se usar uma ordem de, 256 sensores, cada um registrando a presença ou ausência de tinta em uma pequena área de um único dígito. A rede então precisaria de 256 unidades de entrada (um para cada sensor), 10 unidades de saída (um para cada espécie de dígito) e um número de várias unidades escondidas.

Para cada espécie de dígito registrado pelos sensores, a rede devia produzir alta atividade na unidade de saída apropriada e baixa atividade nas outras unidades de saída.

Para treinar a rede, apresenta-se uma imagem de um dígito e compara-se a atividade atual das 10 unidades de saída com a atividade desejada. Calcula-se então o erro, que é definido como o quadrado da diferença entre as atividades atuais e as atividades desejadas. O próximo passo é mudar o peso de cada conexão para que se reduza o erro. Repete-se este processo de treinamento para muitas imagens diferentes de cada espécie diferente de dígito até que a rede classifique toda a imagem corretamente.

Para implementar este procedimento precisa-se calcular o erro quadrático médio para o peso (EW) para alterar o peso por uma medida que seja proporcional ao valor da alteração do erro e ao peso que é alterado.

Outra maneira para calcular o EW é utilizar o algoritmo *Back-propagation*, onde tem se tornado ultimamente um dos mais importantes instrumentos para treinamento de Redes Neurais Artificiais. Este método foi desenvolvido independentemente por dois grupos, um deles, Fogelman-Soulie, Gallinari e Le Cun, o outro, Rumelhart, Hinton e Williams, nos E.U.A.

2.4. Neurocomputação

“Neurocomputação é fundamentalmente um novo e diferente paradigma de processamento de informações – a primeira alternativa para algoritmos de programação. Sempre que ela é aplicada, totalmente novas habilidades de processamento de informações podem ser desenvolvidos, e o custo de desenvolvimento e o tempo frequentemente diminuem por uma grandeza de ordem” [4].


Pode-se fazer as seguintes comparações entre o cérebro humano e o computador:

Tabela 2.4. - Quadro comparativo entre cérebro e o computador.

 Parâmetros	Cérebro	Computador
Material	Orgânico	Metal e plástico
Velocidade	Milisegundos	Nanosegundos
Tipo de Processamento	Paralelo	Sequencial
Armazenamento	Adaptativo	Estático
Controle de Processos	Distribuído	Centralizado

O mesmo pode ser feito comparando o computador com as Redes Neurais Artificiais. Para tanto, faz-se a comparação com o paradigma predominante nos computadores atuais.

Tabela 2.5. - Quadro comparativo entre computadores e neurocomputadores.

 Computadores	Neurocomputadores
Executa programas	Aprende
Executa operações lógicas	Executa operações não lógicas, transformações, comparações
Depende do modelo ou do programador	Descobre as relações ou regras dos dados e exemplos
Testa uma hipótese por vez	Testa todas as possibilidades em paralelo

3. Desenvolvimento de Aplicações

Há alguns passos necessários para o desenvolvimento de aplicações utilizando Redes Neurais Artificiais. São eles :

3.1. Coleta dos dados e separação em conjuntos

Os dois primeiros passos do processo de desenvolvimento de RNAs são a coleta de dados relativos ao problema e a sua separação em um conjunto de treinamento e um conjunto de testes. Esta tarefa requer uma análise cuidadosa sobre o problema para minimizar ambigüidades e erros nos dados. Além disso, os dados coletados devem ser significativos e cobrir amplamente o domínio do problema; não devem cobrir apenas as operações normais ou rotineiras, mas também as exceções e as condições nos limites do domínio do problema.

Normalmente, os dados coletados são separados em duas categorias: dados de treinamento, que serão utilizados para o treinamento da rede e dados de teste, que serão utilizados para verificar sua performance sob condições reais de utilização. Além dessa divisão, pode-se usar também uma subdivisão do conjunto de treinamento, criando um conjunto de validação, utilizado para verificar a eficiência da rede quanto a sua capacidade de generalização durante o treinamento, e podendo ser empregado como critério de parada do treinamento.

Depois de determinados estes conjuntos, eles são geralmente colocados em ordem aleatória para prevenção de tendências associadas à ordem de apresentação dos dados. Além disso, pode ser necessário pré-processar estes dados, através de normalizações, escalonamentos e conversões de formato para torná-los mais apropriados à sua utilização na rede.

3.2. Configuração da rede

O terceiro passo é a definição da configuração da rede, que pode ser dividido em três etapas:

1. Seleção do paradigma neural apropriado à aplicação.
2. Determinação da topologia da rede a ser utilizada - o número de camadas, o número de unidades em cada camada, etc.
3. Determinação de parâmetros do algoritmo de treinamento e funções de ativação. Este passo tem um grande impacto na performance do sistema resultante.

Existem metodologias, "dicas" e "truques" na condução destas tarefas. Normalmente, estas escolhas são feitas de forma empírica. A definição da configuração de Redes Neurais Artificiais é ainda considerada uma arte, que requer grande experiência dos projetistas.

3.3. Treinamento

O quarto passo é o treinamento da rede. Nesta fase, seguindo o algoritmo de treinamento escolhido, serão ajustados os pesos das conexões. É importante considerar, nesta fase, alguns aspectos tais como a inicialização da rede, o modo de treinamento e o tempo de treinamento.

Uma boa escolha dos valores iniciais dos pesos da rede pode diminuir o tempo necessário para o treinamento. Normalmente, os valores iniciais dos pesos da rede são números aleatórios uniformemente distribuídos, em um intervalo definido. A escolha errada destes pesos pode levar a uma saturação prematura. Nguyen e Widrow encontraram uma função que pode ser utilizada para determinar valores iniciais melhores que valores puramente aleatórios.

Quanto ao modo de treinamento, na prática é mais utilizado o modo padrão devido ao menor armazenamento de dados, além de ser menos suscetível ao problema de mínimos locais, devido à pesquisa de natureza estocástica que realiza. Por outro lado, no modo batch se tem uma melhor estimativa do vetor gradiente, o que torna o treinamento mais estável. A eficiência relativa dos dois modos de treinamento depende do problema que está sendo tratado.

Quanto ao tempo de treinamento, vários fatores podem influenciar a sua duração, porém sempre será necessário utilizar algum critério de parada. O critério de parada do algoritmo *back-propagation* não é bem definido, e geralmente é utilizado um número máximo de ciclos, mas devem ser considerados a taxa de erro médio por ciclo, e a capacidade de generalização da rede. Pode ocorrer que em um determinado instante do treinamento a generalização comece a degenerar, causando o problema de “*over-training*”, ou seja, a rede se especializa no conjunto de dados do treinamento e perde a capacidade de generalização.

O treinamento deve ser interrompido quando a rede apresentar uma boa capacidade de generalização e quando a taxa de erro for suficientemente pequena, ou seja, menor que um erro admissível. Assim, deve-se encontrar um ponto ótimo de parada com erro mínimo e capacidade de generalização máxima.

3.4. Teste

O quinto passo é o teste da rede. Durante esta fase o conjunto de teste é utilizado para determinar a performance da rede com dados que não foram previamente utilizados. A performance da rede, medida nesta fase, é uma boa indicação de sua performance real.

Devem ser considerados ainda outros testes como análise do comportamento da rede utilizando entradas especiais e análise dos pesos atuais da rede, pois, se existirem valores muito pequenos as conexões associadas podem ser consideradas insignificantes e assim serem eliminadas. De modo inverso, valores substantivamente maiores que os outros poderiam indicar que houve “*over-training*” da rede.

3.5. Integração

Finalmente, com a rede treinada e avaliada, ela pode ser integrada em um sistema do ambiente operacional da aplicação. Para maior eficiência da solução, este sistema deverá conter facilidades de utilização como interface conveniente e facilidades de aquisição de dados através de planilhas eletrônicas, interfaces com unidades de processamento de sinais, ou arquivos padronizados. Uma boa documentação do sistema e o treinamento de usuários são necessários para o sucesso do mesmo.

Além disso, o sistema deve periodicamente monitorar sua performance e fazer a manutenção da rede quando for necessário ou indicar aos projetistas a necessidade de um novo treinamento. Outras melhorias poderão ainda ser sugeridas quando os usuários forem se tornando mais familiares com o sistema.

4. O Algoritmo Back-Propagation

A técnica de *Back-Propagation* conseguiu aprimorar as Redes Neurais Artificiais, pois permite o aprendizado das redes neurais baseadas no Perceptron com qualquer número de níveis. É difícil dizer quem é o real criador da solução para redes multi-nível (o *Back-Propagation*), pois vários trabalhos foram feitos em lugares diferentes na mesma época. No entanto foi Rumelhart quem mais difundiu este método, cuja grande importância é o fato de ser uma REGRA DELTA GENERALIZADA para múltiplos níveis [13].

É bom ressaltar uma diferença fundamental nas superfícies de erro encontradas através da Regra Delta (*Delta Rule*) e *Back-Propagation*. Na primeira a superfície de erro é côncava, ou seja, um mínimo local é o mínimo global. Na segunda, a superfície pode conter muitos pontos de mínimo local, o que pode fazer com que o algoritmo fique preso em um ponto de mínimo local, não convergindo para o(s) ponto(s) de mínimo global.

Uma rede que utiliza o modelo *Back-Propagation* é uma rede com neurônios em três ou mais níveis: um nível de entrada (*input layer*), um ou mais níveis intermediários ou ocultos (*hidden layers*) e um nível de saída (*output layer*). A função transferência é uma função do tipo função sigmóide (tem forma de um 'S' deformado - *sigmoid*) e a regra de aprendizado é derivada da **Regra Delta**, mas aplicável

a uma rede multi-nível.

Para treinar uma RNA a executar alguma tarefa, deve-se ajustar os pesos de cada unidade de modo que o erro entre a saída desejada e a saída atual é reduzido. Este processo exige que a RNA compute o erro derivado dos pesos (EW). Em outras palavras, isto deve calcular como o erro muda de acordo com cada peso que é aumentado ou decrementado ligeiramente. O Algoritmo *Back-Propagation* é o método mais largamente usado por determinar o EW.

O algoritmo *Back-Propagation* é mais fácil de entender se todas as unidades na rede são lineares[9].

*“....A vantagem no uso de camadas escondidas é a possibilidade de implementação de superfícies de decisão mais complexas, ou seja, aumentando o poder de representação da rede. Em compensação, o aprendizado torna-se uma tarefa mais difícil. O princípio básico é uma busca do tipo gradient descent, seguindo o mesmo procedimento da Delta Rule, só que agora também nas unidades escondidas (por esta razão, o Back-Propagation é conhecido como **Generalized Delta Rule**). A dificuldade existe no fato de que não existe um erro pré-definido para as unidades escondidas....” [7].*

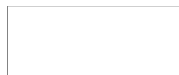
Descrição:

As unidades são conectadas umas as outras. Há um número real associado com cada conexão, na qual é chamada de peso da conexão. Denota-se por W_{ij} o peso da conexão da unidade u_i até a unidade u_j . É conveniente representar o padrão de medida da capacidade dos computadores trabalharem simultaneamente na rede por uma matriz de peso W cujos elementos são os pesos W_{ij} .

Dois tipos de conexão são usualmente distintos: excitatório e inibitório. Um peso positivo representa uma conexão excitatória, onde um peso negativo representa uma conexão inibitória. O padrão de medida da capacidade dos computadores trabalharem simultaneamente na rede caracteriza a arquitetura de rede [14].

Uma unidade na camada de saída (*output*) determina sua atividade pelos dois seguintes procedimentos :

1. 



Calcula-se o total de entradas armazenadas (*weighted input*) X_j , usando a fórmula:

Onde, y_i é o nível de atividade da unidade j na camada anterior e W_{ij} é o peso da conexão entre a unidade i e j .

2. 



Calcula-se a unidade da atividade y_j usando alguma função do total de entrada armazenadas. Normalmente usa-se a função *sigmoid*:

Uma vez que as atividades de todas as unidades de saída foram determinadas, a rede calcula o erro E , que é definido pela expressão a seguir:



Onde, y_j é o nível de atividade da unidade j na camada posterior e d_j é a saída desejada da unidade j .

O Algoritmo *Back-Propagation* consiste em quatro passos :

1.



Calcule quanto tempo o erro muda a medida em que a atividade de uma unidade de saída é alterada. Este derivativo de erro (E_A) é a diferença entre a real e a atividade desejada.

2. Calcule quanto tempo o erro altera a medida em que o total recebido de entrada por uma unidade de saída é alterada. Esta quantidade (E_I) é a resposta do passo 1 multiplicado pela taxa na qual a saída de uma unidade altera a medida em que seu total de entrada é alterado.





3. Calcule quanto tempo o erro altera a medida em que o peso na conexão para uma unidade de saída é alterada. Esta quantidade (**EW**) é a resposta do passo 2 multiplicado pelo nível de atividade da unidade da qual a conexão provém.



4. Calcule quanto tempo o erro altera a medida em que a atividade de uma unidade na camada anterior é alterada. Este passo crucial permite a retro-propagação ser aplicada a redes multicamadas (*multilayer*). Quando a atividade de uma unidade na camada anterior altera, afeta a atividade de todas as unidades de saída na qual está conectada. Então para calcular o efeito global no erro, soma-se ao mesmo tempo todos estas ações separadas nas unidades de saída. Mas cada ação é simples para calcular. É a resposta ao passo 2 multiplicado pelo peso da conexão a unidade de saída.



Usando os passos 2 e 4, pode-se converter os **EAs** de uma camada das unidades em **EAs** para a camada anterior. Este procedimento pode ser repetido para conseguir os **EAs** para muitas camadas anteriores a medida que são desejadas. Uma vez que se sabe o EA de uma unidade, pode-se usar os passos 2 e 3 para calcular os **EWs** em suas conexões entrantes.

4.1. As Deficiências do Back-Propagation

A desgosto do aparente sucesso do algoritmo de aprendizado *back-propagation*, há alguns aspectos que não fazem o uso dele ser totalmente garantido. Mais preocupante é o longo processo de

treinamento. Este pode não ser o melhor resultado de aprendizado avaliado no momento. Muitos algoritmos avançados baseados no aprendizado de *back-propagation* tem alguns métodos eficazes para adaptar a avaliação deste aprendizado. Mais claramente, as falhas de treinamento geralmente aparecem em duas origens: *Network paralysis* e *Local Mínima* [3].

Network Paralysis (Paralisação da Rede) - A medida em que uma rede treina, os pesos podem ser ajustados para valores muito grandes. O total de entradas de uma unidade oculta (*hidden units*) ou unidade de saída pode então alcançar valores muito altos (positivo ou negativo), e devido a função de ativação *sigmoid*, a unidade terá uma ativação muito próxima a 0 ou muito próxima a 1. O processo de treinamento pode vir a ficar na realidade paralisado (suspensão).

Local Mínima (Mínimo Local) – A superfície de erro de uma rede complexa é cheia de montes e vales (veja figura 5.1). Por causa da inclinação gradual (*gradient descent*) a rede pode se concentrar em uma depressão mínima local, deixando de se concentrar em uma depressão próxima muito mais profunda, onde o erro final seria mais reduzido. Existem alguns métodos probabilísticos que podem ajudar a escapar deste problema.

Mas tendem a ser lentos. Outra possibilidade sugerida é aumentar o número de unidades escondidas (*hidden units*). Causará trabalho, pelo fato da alta dimensão do espaço de erro e as chance de cair no problema ser menor, ele poderá mostrar um limite superior de número de unidades escondidas, o que, quando excedidas, novamente resulta em sistema com problemas de **mínimo local**.

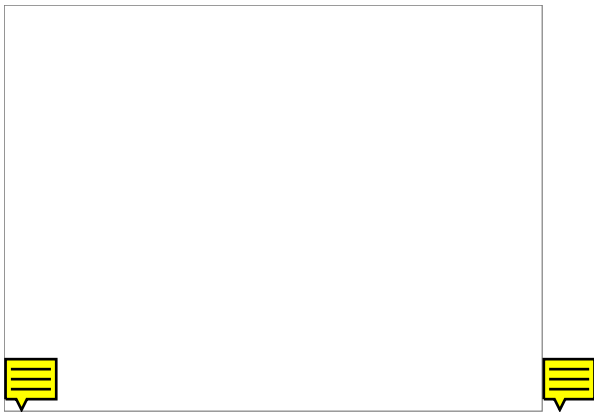


Figura 4.1 Superfície de Erro de uma Rede Complexa.

5. Conclusão

Todas as informações aqui expostas levam a crer que o campo de RNAs é acima de tudo extremamente vasto e promissor. Por ser um assunto que surgiu a muito tempo atrás, ganhou muita credibilidade, e devido à novas descobertas relacionadas a ela a cada instante, tornou-se bastante atrativo para profissionais de domínios distintos, tornando-se um assunto interdisciplinar.

Durante o desenvolvimento deste trabalho, notou-se que o mundo da computação está novamente voltado para as RNAs. A habilidade das RNAs aprenderem com exemplos faz delas muito flexíveis e poderosas. Além disso não há necessidade de desenvolver um algoritmo em ordem para executar uma tarefa específica, pois não seria necessário entender os mecanismos internos de uma tarefa. Elas estão também bem adaptadas para sistemas em tempo real, devido a rápida resposta e tempo de computação, o que é devido a arquitetura de processamento paralelo.

Redes Neurais Artificiais também contribuem com outras áreas de pesquisas, tais como a Neurologia e Psicologia. Elas são regularmente usadas como modelos de vida para organismos e para

investigar os mecanismos internos do neurônio biológico.

Possivelmente o aspecto mais instigante das RNAs é a possibilidade que daqui a alguns tempos, redes ‘conscientes’ poderão ser produzidas. Há um número de cientistas discutindo que a consciência é uma característica mecânica e que Redes Neurais ‘conscientes’ podem ser possibilidades reais.

Os neurônios artificiais já trazem grandes semelhanças com os neurônios biológicos, e é certo também que essas semelhanças já produzem resultados significativos quando aplicados de forma devida. Entretanto muitos ajustes ainda são necessários para assemelhar as RNAs ainda mais às Redes Neurais Biológicas. Desse modo, aproximando esses dois sistemas, acredita-se que ambientes neurais artificiais possam ser úteis para elucidar mecanismos neurais complexos e ainda desconhecidos.

Finalmente, é bom ressaltar que, mesmo que RNAs tenham um gigantesco potencial, só consegue-se o melhor delas quando elas estão integradas com a outras áreas da computação. Inteligência Artificial, Lógica Fuzzy e outros assuntos próximos.

6. Referências Bibliográficas

Livros

- [1] . Fausett, Laurene, Fundamentals of Neural Networks: Architectures, Algorithms And Applications, Flórida, Prentice Hall, Englewood Cliffs, 458p
- [2] . L. Kovács, Zsolt, Redes Neurais Artificiais: Fundamentos e Aplicações, 2ª Edição, São Paulo, Collegium Cognitio, 1996, 174p.
- [3] . Krose ,Ben J. A., Smagt ,P. Patrick Van Der, An Introduction to Neural Networks, 7ª Edição, Amsterdam, University of Amsterdam, Dezembro 1995, 131p.
- [4] . Nelson, Marylin McCord, Illingworth, W.T., How Do You Move from Theory to Applications?, In: _____. A Pratical Guide to Neural Nets, USA: Sponsoring Editor, Ted Buswick, Agosto 1990, p. 159-163.

Periódicos

- [5] . Hetch – Nielsen, Robert, Neurocomputing: picking the human brain, San Diego - Califórnia, p. 36-42, março 1988.

-

Teses

- [6] . Ribeiro, Leilton, Redes Neurais Artificiais, São José dos Campos (SP), Instituto Tecnológico de Aeronáutica – ITA, Dezembro, 1991, 28p (Relatório Final de Graduação).
- [7] . Maranhão, Fábio Martins, Paula, Roberto Uchôa de, Redes Neurais aplicadas a problemas de reconhecimento perceptual e tomada de decisão, São José dos Campos (SP), Instituto Tecnológico de Aeronáutica – ITA, Novembro 1996, 50p (Relatório Final de Graduação).

Sites

- [8] . <http://www.epub.org.br/cm/n05/tecnologia/rna.htm>
- [9] . http://www.dse.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/cs11/report.html
- [10] . <http://www.tecnet.pt/dmn/dmn1336.html>
- [11] . <http://www.mathworks.com/products/neuralnet/>
- [12] . <http://elsevier.com>
- [13] . <http://speech.inesc.pt/>
- [14] . <http://www.google.com/>
- [15] . http://www.informatik.uniosnabrueck.de/papers_html3/wtc_95_frank/node7.html
- [16] . <http://www.uku.fi/laitokset/ympkem/airquality/hybrid/poster/back1.html>

Apêndice A – Uma breve descrição de Reconhecimento de Padrões

Aplicações de imaginação para RNAs aparentam ser uma combinação natural. Poderia se concluir que RNAs ‘adoram’ fazer classificação de exemplos. O treinamento supervisionado, *off-line*, é muito rápido e normalmente é livre de erros [4].

Permitindo-se mostrar através de uma aplicação de imaginação simples, ou classificação de exemplos, o objetivo é desenvolver uma RNA que possa reconhecer os algarismos de 0 até 9. Cada algarismos destes podem facilmente ser representados e reconhecidos em uma matriz 4 por 7, usando exatamente duas sombras acinzentadas, ou seja, preto e branco. Alternativamente, pode-se representar cada algarismo por uma matriz de 1s(uns) e 0s(zeros). Cada imagem ou algarismo, então requer uma entrada contendo 28 dígitos compostos de 0s e 1s.

Esta experiência mostra que quando se dispõe de dados de entrada para ambos os grupos de treinamento e o teste de grupo na aplicação de imagens, grande quantidade é conquistada por representação de dados em forma de matriz, como mostra a figura A.1.

A imagem do algarismo '2' na figura A.1., o 0 (zero) representa a ausência de luz e o 1 (um) representa o preto. Preenchendo os espaços onde estão os 1s e não preenchendo onde estão os 0s, obtém-se a forma do 2.

Para o software de uma RNA, para reconhecer a entrada de dados, deve-se entrar com cada imagem ou número em ordem sucessivamente. Para a imagem do algarismo '2', deve-se ter:

I 1 1 1 1 0 0 0 1 0 0 0 1 1 1 1 1 0 0 0 1 0 0 0 1 1 1 1
D 2

Onde, I refere-se a entrada (*input*) e o D é a saída desejada (*output*). Devido ter que representar os dez algarismos com as saídas desejadas entre 0 e 9, é conveniente determinar que, cada imagem do algarismo é o valor numérico dele. Ou seja, 0=0, 1=1, 2=2...9=9.

1 1 1 1	1 1 1 1
0 0 0 1	1
0 0 0 1	1
1 1 1 1	1 1 1 1
1 0 0 0	1
1 0 0 0	1
1 1 1 1	1 1 1 1

Figura A.1. – Representação do Algarismo '2'.

Quando o treinamento do grupo é montado, é necessário treinar a rede com as corretas imagens dos algarismos de entradas; não serão relacionados com não-algarismos durante o treinamento. O próximo passo é preparar os dados para a apresentação do programa de software de desenvolvimento

da RNA. O exemplo a seguir mostra um método curto de matrizes, entrando com as imagens. Para o algarismo '2', primeiro se escreve o seguinte:



Então preenche-se os espaços com zeros :



I

1 1 1 1

0 0 0 1

0 0 0 1

1 1 1 1

1 0 0 0

1 0 0 0



1 0 0 0

0 1 1 1

0 1 0 0

1 0 0 0

0 0 0 0



0 0 0 0

0 0 0 0

0 0 0 0

1 1 1 1

1 1 1 1

1 1 1 1

1 1 1 1

0 0 1 0

1 0 0 0

0 0 0 1

0 0 0 1

0 0 0 1

0 0 1 0

1 0 0 0

0 0 0 1

0 0 0 1

0 0 0 1

0 1 1 1

1 1 1 1

1 1 1 1

0 0 0 1

1 1 1 1

D1

D2

D3

D4

D5

I 1 0 0 0

I 1 1 1 1

I 1 1 1 1

I 1 1 1 1

I 1 1 1 1

1 0 0 0

0 0 0 1

1 0 0 1

1 0 0 1

1 0 0 1

1 0 0 0

0 0 0 1

1 0 0 1

1 0 0 1

1 0 0 1

1 1 1 1

0 0 0 1

1 1 1 1

1 1 1 1

1 0 0 1

1 0 0 1	0 0 0 1	1 0 0 1	0 0 0 1	1 0 0 1
1 0 0 1	0 0 0 1	1 0 0 1	0 0 0 1	1 0 0 1
1 1 1 1	0 0 0 1	1 1 1 1	0 0 0 1	1 1 1 1
D6	D7	D8	D9	D10

Figura A.2. Treinamento para um Grupo de ‘10’ Algarismos

Apêndice B – Desenvolvimento do Back-Propagation em uma RNA

Utilizando-se a *toolbox* de Redes Neurais Artificiais do Matlab, pôde-se constatar que:

- As regras de aprendizado são utilizadas para ajustar os pesos e as inclinações das redes, de forma a minimizar a soma proporcional do erro na rede. Isto é feito através da mudança contínua de valores dos pesos e inclinações da rede, na direção da inclinação excessiva, de acordo com o erro. Isto é chamado de procedimento de inclinação gradual (*gradient descent*).
- Cita-se um simples problema com 1 vetor de entrada e 1 de saída, ambos contendo 2 elementos:

$$P = [-3.0 ; + 2.0];$$

$$T = [+0.4; + 0.8];$$

Utiliza-se uma única camada da rede para este problema. Nota-se que há somente 2 variáveis da rede que são otimizadas, o peso e a inclinação.

- Como há somente duas variáveis, pode-se esboçar o gráfico (Figura B.1.) abaixo, da soma proporcional aos erros, na escala de inclinações (Biases) e pesos (*weights*):



Figura B.1. – Gráfico da Superfície do Erro.

- Para treinar uma rede, primeiro se inicializa o peso(W) e a inclinação(B). Ajusta-se os parâmetros de treinamento, para, acompanhar com que frequência o progresso é mostrado (*Disp_freq*), limitar o número de iterações de aprendizado (*Max_epoch*), definir um erro aceitável (*Err_goal*) e ajustar a taxa de aprendizado(Lr):

$W_0 = (-2.1617)$ e $B_0 = (-1.7862)$

$Disp_freq = 1$;

$Max_epoch = 100$;

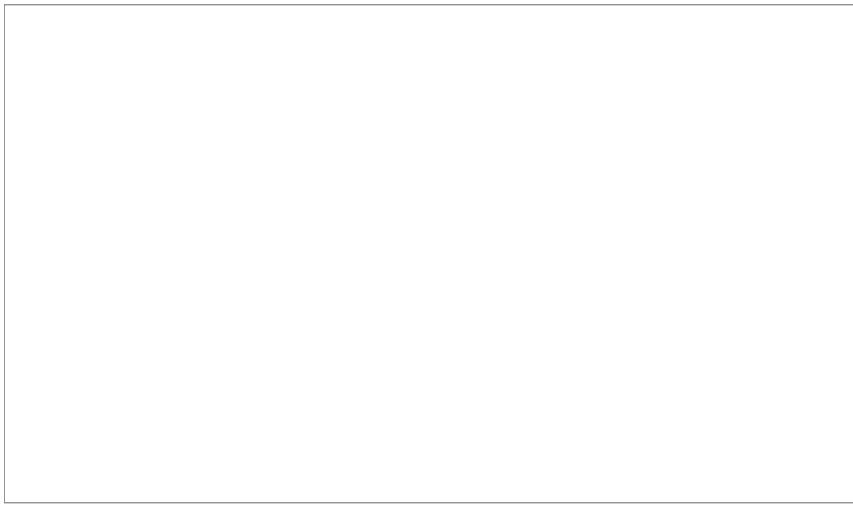
$Err_goal = 0.001$;

$Lr = 1$.

- A rede está pronta para ser treinada.

Figura B.2. – Gráfico com Peso e Inclinação ajustados.





Conforme o aprendizado ocorre, o peso e a inclinação são ajustados para se obter uma inclinação mínima no gráfico, cujo erro está abaixo do valor aceitável (*Err-goal*) que vale 0,001 (veja figura B.2).

- O aprendizado grava a variável ‘erros’ (guarda o erro da rede) durante a toda a duração do mesmo.
- No gráfico abaixo (figura B.3), mostra-se o plano desses erros. Note que a descida permanente do erro, é executada pelo algoritmo *back-propagation*.

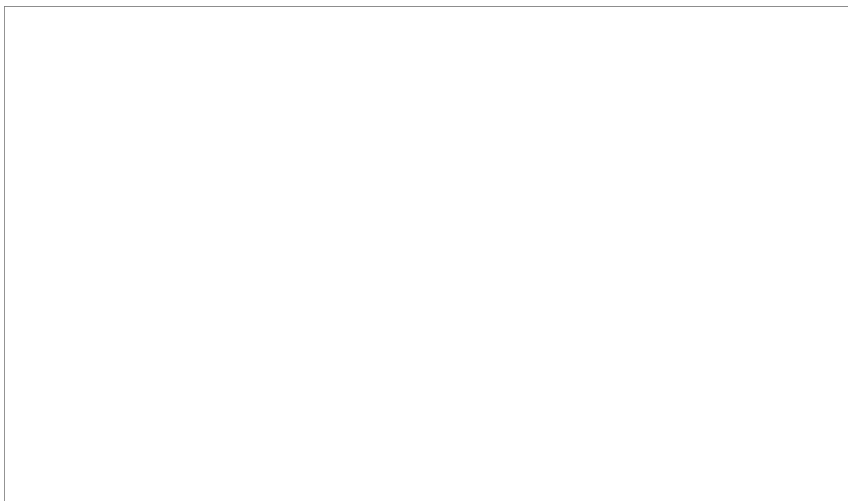


Figura B.3. – Gráfico dos erros.

- O plano de erros pára com 71 (setenta e uma) iterações, quando a soma ajustada do erro é calculada em 0.00096, o que é menos que o erro aceitável inicialmente (0.001). Isto mostra uma convergência do erro para o valor sugerido.
- Os resultados finais são:

$W = 0.3346$. (*Weight*)

$B = 0.5473$. (*Bias*)