Redes Neurais Artificiais com Processamento Temporal

Um estudo sobre redes estáticas temporais, redes recorrentes e aplicações

Karen Braga Enes

JUIZ DE FORA

AGOSTO, 2013

Redes Neurais Artificiais com Processamento Temporal

Um estudo sobre redes estáticas temporais, redes recorrentes e aplicações

Karen Braga Enes

Universidade Federal de Juiz de Fora
Instituto de Ciências Exatas
Departamento de Ciência da Computação
Bacharelado em Ciência da Computação

Orientador: Prof. D.Sc. Raul Fonseca Neto

JUIZ DE FORA
AGOSTO, 2013

REDES NEURAIS ARTIFICIAIS COM PROCESSAMENTO TEMPORAL

Um estudo sobre redes estáticas temporais, redes recorrentes e aplicações

Karen Braga Enes

MONOGRAFIA SUBMETIDA AO CORPO DOCENTE DO INSTITUTO DE CIÊNCIAS EXATAS DA UNIVERSIDADE FEDERAL DE JUIZ DE FORA, COMO PARTE INTEGRANTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE BACHAREL EM CIÊNCIA DA COMPUTAÇÃO.

Aprovada por:

Prof. D.Sc. Raul Fonseca Neto

Prof. D.Sc. Custódio Gouvêa Lopes da Motta

Prof. D.Sc. Luciana C. Dias Campos

JUIZ DE FORA

30 DE AGOSTO, 2013

In memorian:

Aos meus avós Lea e Hélio.

Resumo

O presente trabalho apresenta um estudo sobre Redes Neurais Artificiais com enfoque em redes com processamento temporal. Inicialmente, é feita uma introdução do conceito de redes neurais, sua motivação inicial e o embasamento biológico quando se trata da origem do estudo dos neurônios artificiais. As principais topologias de redes neurais e principais classificações são descritas de uma maneira genérica. Estudou-se nesse trabalho duas topologias diferentes para tratamento temporal de dados: a Time-lagged Feedforward Network e a Time Delay Neural Network. Além disso, duas redes neurais recorrentes também são apresentadas, a Rede de Elman e a Rede de Jordan. Apresenta-se duas variações do algoritmo de aprendizado supervisionado Backpropagation para treinamento dessas topologias. Ao final, espera-se que o estudo do modelos permita uma avaliação comparativa resultando na identificação dos modelos adequados de acordo com a necessidade de cada problema.

Palavras-chave: Inteligência Artificial, Redes Neurais Artificiais, Processamento Temporal, Backpropagation, Redes Recorrentes, Time Delay Neural Network,Time-lagged Feedforward Network, Rede de Elman, Rede de Jordan.

Abstract

The present work shows a study about Artificial Neural Networks, focus on temporal processing networks. Initially, there is an introduction of Neural Networks concepts, its first motivation and the biological foundation when it comes from the origin of the studies of artificial neurons. The main neural networks topologies and the main types of classification were discribed in a generic form. Moreover, two different topologies for temporal data treatment were study, the Time-lagged Feedforward Network and the Time Delay Neural Network. This work as also presents two recurrent networks, Elman Network and Jordan Network. Besides, two variations of supervised learning algorithm Backpropagation for training these topologies were studied. Finally, it's expected, after the study models a comparative evaluantion that will result in a suitable identification models according to the needs of each different problem.

Keywords: Artificial Inteligence, Artificial Neural Networks, Temporal Processing, Backpropagation, Recurrent Networks, Time Delay Neural Network,Time-lagged Feedforward Network, Elman Network, Jordan Network.

Agradecimentos

Antes de mais nada, é preciso dizer que não existem palavras capazes de traduzir o tamanho da minha alegria em chegar até aqui e, principalmente, o tamanho da minha gratidão a todos aqueles que, de alguma forma, contribuiram para que concluísse esse trabalho e completasse mais essa etapa da minha vida.

Os meus maiores e melhores agradecimentos sempre serão para os meus pais, Katia e Marcos, além de meus melhores amigos, são também meus maiores incentivadores, dividindo comigo os momentos mais felizes e difíceis da minha vida. À minha mãe, eu só tenho a agradecer pela calma, carinho e afeto dispensados a mim em todas as etapas da minha vida, comemorando comigo cada nova etapa cumprida, cada mínima vitória alcançada. Ao meu pai, agradeço por me trazer calma nos momentos difíceis, pelo carinho e cuidado de sempre, pelas noites de estudo sem dormir, pela companhia diária e incondicional. Eu, com certeza, não teria chegado aqui sem vocês. E se eu consegui foi por vocês. Obrigada por tudo!

Às minhas irmãs, Karine e Gabrielle, agradeço pelas risadas, pelos momentos de descontração, por muitas vezes desempenharem o papel de irmã mais velha, quando devia ser ao contrário. Pela paciência, atenção e carinho durante todos esses anos. À Karine, em especial, por ser minha empresária particular. Ao meu priminho Matheus, pelo carinho e pelas brincadeiras.

Ao meu tio Ricardo, pelas referências biológicas para esse trabalho, por ser uma referência muito significativa na minha vida e o melhor biólogo e professor do mundo. As minhas tias Mônica e Valéria e ao meu tio Edvaldo, por serem exemplos de pessoas batalhadoras e guerreiras, agradeço por todo carinho, atenção e preocupação. Vocês são imprescindíveis na minha vida. Obrigada por torcerem sempre por mim!

Aos melhores amigos que alguém pode ter, meu muito obrigada! À Taylla, primeiramente, por todos esses anos de amizade e paciência e por compartilhar todos os momentos comigo desde o berço. À Marina, pelos almoços, por cuidar de mim como só

uma mãe cuidaria e principalmente, pela amizade desde quase o primeiro dia de faculdade. À Flávia e a Carol pela amizade inquestionável, pelo carinho, pela companhia e pela compreensão durante os períodos de ausência. Ao Luiz e Marcelo pelos passeios pós aula, pela amizade, pela preocupação de pai e pela companhia de sempre. Ao Roberto por todo carinho, atenção, cuidado, companheirismo e preocupação durante esses anos.

Aos Professores do DCC, em especial ao Raul pela orientação, parceria, cuidado, pela calma de sempre e pelas várias disciplinas ministradas durante a gradução. Ao Barrére por todo apoio, pela orientação na IC, por me acalmar nos momentos de desespero, pela torcida, por todo incentivo, pela paciência, apoio, enfim, por tudo! Ao Jairo, pela disciplina mais difícil durante a graduação, pelo apoio e dedicação de sempre. À Julieta, pela atenção, carinho e por acreditar em mim desde o primeiro ajuste de matrícula.

À todos que divivem comigo essa conquista, meu muito obrigada. Vocês foram fundamentais para a conclusão dessa etapa e eu não sei o que seria de mim sem vocês! Obrigada!

"O sucesso nasce do querer, da determinação e persistência em se chegar a um objetivo. Mesmo não atingindo o alvo, quem busca e vence obstáculos, no mínimo fará coisas admiráveis.".

José de Alencar

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviações

BP	Backpropagation
BPTT	Backpropagation Through Time
LMS	Least Squares Method
MAPE	Erro Médio Percentual Absoluto
MLP	Multilayer Perceptron
MSE	Mean Square Error
NMSE	Erro Médio Quadrado Normalizado
PE	Elemento de Processamento
RBF	Radial Base Function
RMSE	Raiz do Erro Médio Quadrado
RNA	Rede Neural Artificial
TDNN	Time Delay Neural Network
TLFN	Time-lagged Feedforward Network

1 Introdução

Este capítulo introduz o problema abordado, iniciando pela apresentação do tema e sua contextualização, desde o início dos estudos de redes neurais artificiais (RNAs), até os tópicos abordados por este trabalho. A seguir, apresenta-se a justificativa do tema, os conceitos básicos sobre RNAs e a importância em se tratar esse tema. Por fim, os objetivos a serem alcançados com o trabalho.

1.1 Apresentação do Tema e Contextualização

O estudo de redes neurais teve sua motivação inicial através do reconhecimento que o cérebro humano funciona de maneira completamente diferente dos computadores digitais convencionais. O cérebro é, ainda, altamente complexo e não linear. Ele possui a capacidade de organizar seus componentes estruturais, conhecidos como neurônios, de modo a realizar certas computações muitas vezes mais rápido que o mais veloz computador digital existente (HAYKIN, 2009).

Esse reconhecimento gerou a criação do neurônio artificial, um modelo matemático similar em sua morfologia ao neurônio humano. O modelo composto por um conjunto de sinapses, conhecidos também como pesos sinápticos, uma função de soma que combina os pesos, uma função de ativação, com intuito de limitar a amplitude da saída, e a saída.

O estudo de redes neurais possui aplicação em diversas áreas dentro da computação, no entanto, esse trabalho visa abordar especificamente as RNAs que introduzem alguma noção de temporalidade e sua capacidade de aplicação em diversos problemas.

Uma vantagem do processamento temporal é evidenciado quando se trata da extração em tempo real de informações. Esses sistemas de aprendizado podem fazer uso de filtros (informação de domínio de frequência) para alcançar metas de processamento (PRINCIPE *et al.*, 2000).

Entre os tópicos que serão estudados, destacam-se:

- Arquitetura e conceitos básicos de RNAs;

- Time-lagged feedforwards networks (TLFNs)e Time delay neural networks (TDNNs), como RNAs estáticas aplicadas em problemas temporais;
- Redes de Elman e redes de Jordan, como redes recorrentes específicas para processamento temporal;
- Backprogation through time (BPTT);
- e aplicações.

Os problemas temporais abrangem uma gama de aplicações possíveis, entre elas é possível destacar as aplicações relacionadas ao mercado financeiro e ao mercado de ações da bolsa de valores, como em MOSELEY (2003) e POMMERANZENBAUM (2009). Além disso, existem aplicações em sistemas estruturais como mostrado em ROCHA *et al.* (2007), aplicações em sistemas de fluxo de dados, MIGUEL (2011), entre outros.

1.2 Justificativa

Dentro da inteligência artificial, por anos o objetivo principal foi estudar o funcionamento do cérebro humano de modo a reproduzir seu funcionamento. No entanto, o comportamento do cérebro é algo não trivial e modelar esse funcionamento também não é simples.

O estudo biológico do cérebro humano e seu funcionamento estimularam a criação do neurônio artificial, um modelo matemático semelhante ao neurônio humano, porém simplificado, que quando agrupado gera uma rede. Essas redes, ou conjuntos de neurônios artificiais são conhecidas como redes neurais artificiais (RNAs).

Existem inúmeras definições para RNAs, que por vezes evidenciam as características de natureza tecnológica e outras vezes as de natureza essencialmente matemática. Em PRINCIPE *et al.* (2000) define-se RNAs como sistemas "distribuídos, adaptáveis, geralmente máquinas de aprendizado não-lineares construídas a partir de diferentes elementos de processamento (PEs)."

O sistema de aprendizado de uma RNA é similar ao aprendizado do cérebro humano. O neurônio artificial aprende através do método de tentativa e erro e ao errar ocorre aprendizado.

As RNAs podem ser aplicadas em várias classes de problemas, sejam eles de regressão, classificação, predição, além de reconhecimento de padrões, entre outros. Além disso, essas aplicações podem ser feitas de maneira estática ou dinâmica, utilizando dados de um repositório ou ainda extraindo dados em tempo real. A aplicação das RNAs em problemas com processamento temporal, bem como o estudo de redes recorrentes e suas aplicações são assuntos ainda não explorados totalmente, e portanto, cabe ainda contribuição.

"Em problemas temporais as medições não são mais um conjunto de amostras de entrada mas sim funções de tempo. Um sensor único produz uma sequência de medições que são ligadas por uma relação ordenada, a estrutura de sinal de tempo. Caso seja alterada a ordem das amostras, o sinal de tempo é distorcido e o seu conteúdo de frequência é alterado, sendo assim, a ordem das amostras deve ser preservada dentro do processamento temporal"(PRINCIPE *et al.*, 2000).

Como visto em KOSKELA *et al.* (2000), as redes neurais devem conter memória para promover o processamento temporal da informação. Existem dois caminhos básicos de construir memória dentro das redes neurais (HAYKIN, 1994). O primeiro caminho é introduzir atrasos na rede e ajustar os parâmetros durante a fase de aprendizado. O segundo caminho é usar feedback positivo, o que torna a rede recorrente. Para caracterizar memórias em diferentes arquiteturas, duas dimensões foram propostas: profundidade e resolução. Embora esses procedimentos tenham sido elaborados há algum tempo, ainda hoje há lacunas a serem preenchidas relacionadas ao aprofundamento e aperfeiçoamento de estudos nessa área.

"Atualmente, o campo de aplicações que as redes neurais artificiais (RNA) cobrem é bem amplo, variando de reconhecimento óptico de caracteres e problemas de diagnóstico na medicina até a utilização em processamento de imagens. No campo da economia, destacam-se problemas como a previsão de quebras de empresas e predição de indicadores ou tendências no mercado financeiro" (POMMERANZENBAUM, 2009). Fica evidenciado ainda, a importância do aprofundamento das pesquisas para garantir o aumento da gama de aplicações práticas possíveis nas mais diversas áreas de conhecimento.

1.3 Objetivos

O objetivo desse trabalho é apresentar um estudo comparativo entre 4 topologias de redes neurais com processamento temporal. Duas dessas topologias são representadas por sistemas estáticos. A Time-lagged Feedforward Network e a Time Delay Neural Network. Essas topologias são treinadas com o algoritmo padrão do Backpropagation. Outras duas topologias são recorrentes e apresentadas como sistemas dinâmicos. Para esse tipo de rede é necessário um algoritmo de treinamento que entenda as recorrências da rede. A rede de Elman e a rede de Jordan são treinadas com o algoritmo Backpropagation Through Time.

Após a comparação teórica desses modelos e seus respectivos algoritmos de aprendizado, segue-se com algumas aplicações desses modelos, objetivando uma comparação entre resultados obtidos relacionados às duas topologias estáticas. Além disso, são feitas aplicações com os modelos recorrentes e os dois algoritmos de aprendizagem estudados, com intuito de comparar a eficiência.

Por fim, espera-se que a partir dessa análise comparativa seja possível avaliar qual modelo deve ser usado por diferentes problemas.

1.4 Estruturação

Primeiramente, o capítulo introdutório apresenta uma ideia geral do tema a ser abordado. A apresentação do tema é seguida de uma contextualização histórica e motivação para o tema. A seguir é apresentada a justificativa de estudo, alguns conceitos básicos e a importâcia desse tema abordado. O segundo capítulo apresenta a fundamentação teórica do trabalho, apresentando um levantamento da evolução das pesquisas na área de redes neurais, a fundamentação biológica para o estudo e as principais características das RNAs, bem como suas diferentes topologias e classificações. O terceiro capítulo tem como objetivo explorar duas topologias de redes feedforward adaptadas para processamento temporal de dados, a primeira topologia abordada é a Time-lagged Feedforward Network(TLFN) e a segunda é um tipo especial de TLFN conhecida como Time Delay Neural Network (TDNN). O quarto capítulo explora duas topologias de redes recorren-

tes, a rede de Elman e a rede de Jordan. Tais redes foram desenvolvidas especificamente como soluções para tratamento temporal. O capítulo cinco conta com a ampliação dos conceitos de redes recorrentes, apresentando o algoritmo Backpropagation Through Time (BPTT). Finalmente, o capítulo seis apresenta o estudo de caso com as aplicações das topologias de rede estudadas e algoritmos de aprendizado. O último capítulo conta com as considerações finais.

2 Fundamentação Teórica

Eesse capítulo aborda os principais conceitos sobre Redes Neurais Artificiais. Começando por um breve histórico envolvendo as linhas de pesquisa em Inteligência Artificial e sua relação com o estudo de RNAs. A seguir, é proposta uma comparação entre as redes neurais biológicas e artificiais de forma mais aprofundada e os principais conceitos iniciais sobre as RNAs, tendo em vista que as redes neurais biológicas e a necessidade de compreensão de seu funcionamento foram as principais motivações para o início dos estudos nessa área. E, ao final do capítulo, um estudo sobre as principais topologias e classificação das diferentes arquiteturas das RNAs.

2.1 Histórico

Denominamos nossa espécie *Homo sapiens* - homem sábio - porque nossas capacidades mentais são muito importantes para nós. Durante milhares de anos, procuramos entender como pensamos; isto é, como um mero punhado de matéria pode perceber, compreender, prever e manipular um mundo muito maior e mais complicado que ela própria. O campo da Inteligência Artificial, vai ainda mais além: ele tenta não apenas compreender, mas também construir entidades inteligentes (RUSSEL e NORVIG, 2002).

O termo Inteligência Artificial, ou IA, foi empregado pela primeira vez logo após a Segunda Guerra Mundial, em 1956 em uma conferência de verão em Dartmouth College, NH, USA. Atualmente, existem duas principais linhas de pesquisa em IA: a simbólica e a conexionista. A primeira "parte do pressuposto que o aparato mental é essencialmente 'um dispositivo lógico que pode ser descrito por meio de um conjunto de computações abstratas, onde o que importa são as propriedades formais dos símbolos que são manipulados' (TEIXEIRA, 1998). Para essa vertente, a inteligência poderia ser definida como a capacidade de resolver problemas" (PRIMO, 2003). Já a linha conexionista "visa à modelagem da inteligência humana através da simulação dos componentes do cérebro, isto é, de seus neurônios, e de suas interligações" PRIMO (2003). É dentro dessa linha, a conexionista,

que se desenvolveram os estudos sobre Redes Neurais Artificiais, que teve sua origem em "um primeiro modelo de rede neuronal, isto é, um conjunto de neurônios interligados, que foi proposto por Rosenblatt, e foi chamado Perceptron" BITTENCOURT (1996).

2.2 Redes Neurais Biológicas

Em GUYTON e HALL (2012), entende-se o sistema nervoso humano como aquele que controla e coordena as atividades sensoriais e os estímulos nervosos do corpo humano. As chamadas células nervosas, ou neurônios constituem o principal componente do sistemas nervoso. Ele é dividido em duas partes, o sistema nervoso central (SNC) e o sistema nervoso periférico (SNP). O SNC é composto pelo encéfalo e pela medula espinhal, enquanto o SNP é formado por receptores, nervos e gânglios. O encéfalo, por sua vez é formado por vários órgãos como o cérebro, o diencéfalo, o bulbo e o cerebelo. O cérebro é o centro do intelecto e do pensamento humano. Contém aproximadamente 12 milhões de células nervosas e é dividido em dois hemisférios: o direito e o esquerdo. As células nervosas estão presentes em todo o sistema nervoso humano e são o objeto de estudo desse trabalho.

Existem aproximadamente 100 bilhões de células nervosas no sistema nervoso humano, elas se ligam umas as outras formando uma rede interligada de neurônios. Cada um desses neurônios tem a função de processar impulsos elétricos, químicos ou ambos e se comunicar com outros milhares de neurônios, paralelamente. O neurônio é formado por três principais componentes: os dendritos, que tem a função de receber os estímulos transmitidos pelos outros neurônios. O corpo do neurônio, responsável por recolher e agrupar informações vindas de outros neurônios. O axônio, ou fibra nervosa, que é constituído de um fibra tubular responsável por transmitir os estímulos para as outras células. A fisiologia básica de um nerônio é ilustrada na Figura 2.1.

A comunicação entre os neurônios é feita através de um mecanismo chamado sinapse. Os neurotransmissores das sinapses, chamados também de neurormônios, saem do axônio e entram no corpo celular ou dendritos de um outro neurônio ou de uma outra célula, como uma célula muscular, por exemplo. A essa transferência de estímulos, é dado o nome de sinapse.

Desta forma, os neurônios realizam a maioria das funções cerebrais e conseguem

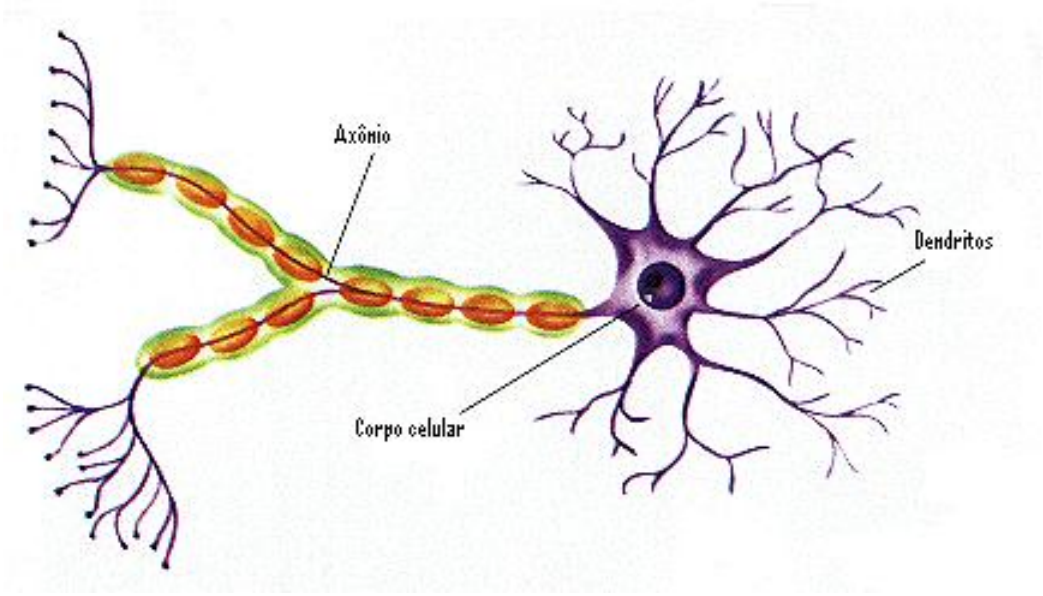

Figura 2.1: Neurônio Biológico

desempenhar as funções complexas de processamento operando em paralelo com todos os outros neurônios.

O avanço dos estudos relacionados a fisiologia celular do cérebro humano desencadeou interesses em diversas áreas de pesquisas, inclusive na computação, estimulando o avanço dos estudos e pesquisas relacionadas ao tema RNAs dentro da Inteligência Artificial.

2.3 Redes Neurais Artificiais

A partir do estudo das redes neurais biológicas entende-se o estudo das redes neurais artificiais como forma de aproximação do processamento dos computadores ao processamento do cérebro humano.

Basicamente, uma rede neural artificial é uma máquina de aprendizado desenvolvida com intuito de simular o sistema de aprendizado do cérebro humano, ou como ele pode realizar uma determinada ação. Além disso, existem muitas definições sobre redes neurais artificiais. Em PRINCIPE *et al.* (2000), tem-se que as RNAs são máquinas de aprendizado distribuído, adaptável e geralmente não lineares construídas a partir de diferentes elementos de processamento (PEs), nas quais cada PEs recebe conexões de outros PEs e dele mesmo. Além disso, as RNAs são compostas também por nodos de rede, ou

neurônios, que calculam as funções de soma, e ligados de forma unidirecional os chamados pesos sinápticos, ou simplesmente pesos, que além de medir o peso das entradas recebidas pela rede, são capazes também de armazenar conhecimento.

No entanto, existem diferenças substanciais, entre elas, destaca-se a capacidade extremamente superior de processamento do cérebro se comparado ao melhor computador digital convencional existente atualmente. Existem ainda diferenças relacionadas a forma e as características de processamento. Por exemplo, o número de elementos processados no cérebro humano é da ordem de 10^{11} enquanto no computador digital é no máximo 10^6, as ligações entre os elementos processados no computador digital não passa de 10 enquanto no cérebro é aproximadamente 10^4. Por outro lado, a velocidade de processamento do computador é obviamente superior sendo da ordem de nanosegundos, enquanto no cérebro é da ordem de milissegundos, além de outros parâmetros, como observado em SILVA (2003).

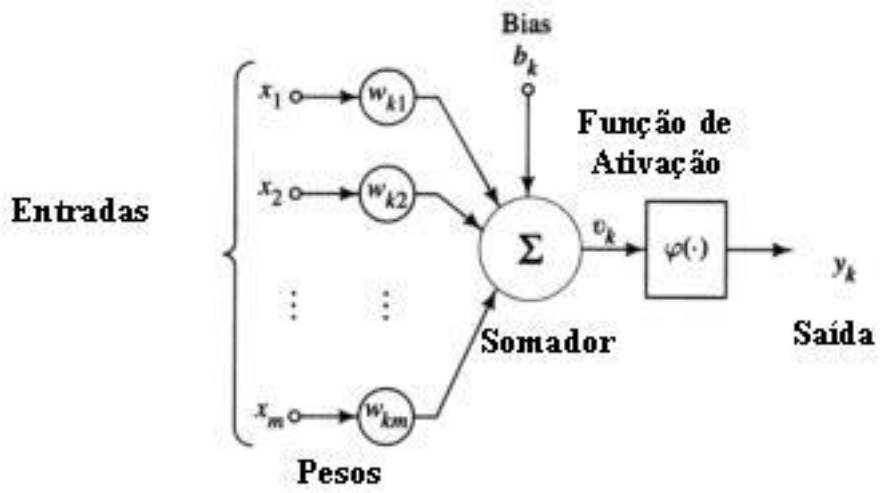

Figura 2.2: Modelo Não-Linear de um Neurônio

É possível ainda fazer uma analogia em termos de fisiologia entre o modelo do neurônio matemático (Figura 2.2) e o neurônio humano. Essa analogia é apresentada na Tabela 2.1.

Além disso, é necessário ressaltar a existência de um outro componente, o bias, similar a um excitador, ou inibidor quando conveniente, de neurônios controlado pela função de ativação. O bias é entendido ainda como uma entrada adicional de neurônio,

Quadro Comparativo Neurônio Humano x Neurônio Matemático		
Neurônio Humano	**Neurônio Matemático**	**Analogia**
Dendritos	Sinais de Entrada	Cada dendrito de um neurônio z é representado por um sinal de entrada.
Sinapse	Sinais de Entrada + Pesos Sinápticos	Cada sinal de entrada é associado a um peso sináptico diferente pertencente ao conjunto dos inteiros. Essa associação é equivalente a sinapse do neurônio humano.
Corpo Celular	Somador + Função de Ativação	O corpo celular é, matematicamente, representado pela união do somador a função de ativação.
Axônio	Saída	A saída de um neurônio matemático representa o axônio de um neurônio humano

Tabela 2.1: Quadro Comparativo Neurônio Humano x Neurônio Matemático

porém com um peso fixo, já que seus valores devem variar entre -1 e 1, para que funcione inibindo ou exitando os neurônios da rede.

Sobre a forma de aprendizado, ou seja, de ganho de conhecimento em redes neurais, o processo é realizado em duas etapas. A priori, é realizada uma fase de treinamento, ou aprendizagem, na qual uma parte do todo é inserida na rede para ser treinada, para que a rede possa extrair informações relevantes a respeito desses dados de entrada. Essas informações extraídas são armazenas na rede e a rede aprende sobre elas. Em suma, a redes aprendem com um conjunto de treinamento e são capazes de generalizar as informações em um conjunto de teste semelhante de resultados não conhecidos. Esse aprendizado é feito através de algoritmos específicos, chamados algoritmos de aprendizado.

As informações obtidas na fase de treinamento são testadas na parte restante da base de dados. Essa fase é chamada de fase de testes, e consiste na generalização dos resultados obtidos da primeira fase aplicados à segunda para geração de respostas do problema inicialmente proposto. As redes neurais tem a capacidade de conseguir produzir resultados corretos mesmo para aqueles dados que não participaram da fase de treinamento.

Em HAYKIN (1994) é possível derivar uma associação de estruturas de aprendizagem cerebrais humanas e de uma RNA. É nítida a observação de que tanto o cérebro quanto uma RNA, a partir de um processo de aprendizagem conseguem adquirir conhecimento do ambiente e, a seguir, todo esse conhecimento absorvido no cérebro é armazenado através de sinapses enquanto as RNAs fazem esse armazenamento nos pesos sinápticos.

Assim, a RNAs conseguem adquirir aprendizagem através de exemplos em um determinado ambiente para, posteriormente generalizar as informações absorvidas e aplicá-las em situações semelhantes mas ainda não conhecidas.

2.4 Arquiteturas de Redes Neurais Artificiais

Essa seção apresenta as diferentes formas de se estruturar a topologia de uma RNA e duas formas de classificação da mesma. No entanto, vale a pena ressaltar que existem várias formas diferentes de se classificar uma rede neural, serão apresentadas as que satisfazem o objetivo do trabalho. Após a classificação das RNAs, serão difinidos os conceitos relacionados a sistemas estáticos e sistemas dinâmicos. Posteriormente, são apresentadas também as principais regras de saída e ativação das RNAs.

2.4.1 Topologias de Redes

Quanto a topologia, existem uma infinidade de formas para se estruturar uma RNA. A estrutura definida está diretamente relacionada ao propósito que se deseja obter com a construção da rede. Será abordada a classificação quanto ao padrão de conexões entre as unidades da rede e propagação dos dados, além da classificação quanto ao número de camadas, fatores determinantes para aplicação de um algoritmo de aprendizado, adequado ao problema proposto.

Quanto a classificação relacionada ao padrão de conexões e propagação dos dados, em SMAGT e KROSE (1996) e HAYKIN (2009) as redes são classificadas em redes feedforward e redes recorrentes.

- Redes feedforward ou redes com alimentação adiante: são redes nas quais os dados fluem estritamente da unidade de entrada para a unidade de saída. O processamento de dados pode ser estendido através de múltiplas camadas, mas não existem conexões de feedback, isto é, conexões estendidas das unidades de saída das unidades de entrada de uma mesma camada ou de camadas anteriores. Como exemplo de redes feedforward, temos as TLFN e as TDNN, ambas estudadas no próximo capítulo.

- Redes Recorrentes: são redes que contém conexões de feedback, ou seja, as conexões

entre as unidades de entrada e processamento formam um ciclo direto. Ao contrário das redes feedforward, as propriedades dinâmicas da rede são de fato importantes, essas redes tem memória dinâmica e compõe os chamados sistemas dinâmicos. Como exemplo de redes recorrentes, temos duas redes abordadas posteriormente nesse trabalho, as Redes de Elman e as Redes de Jordan.

A Figura 2.3, apresenta um modelo de redes feedforward e redes recorrentes.

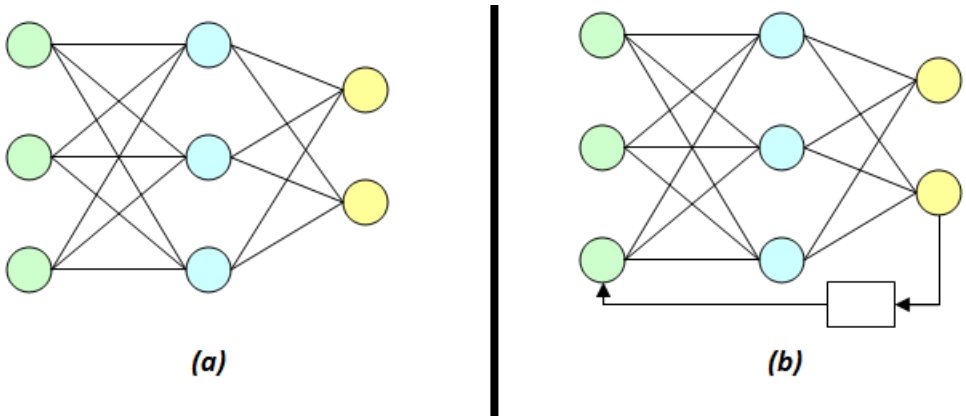

Figura 2.3: Exemplo de rede feedforward (a) e rede recorrente (b)

A classificação quanto ao número de camadas é dada de forma mais simples, como mostrado em SILVA (2003). Temos:

- Redes de camada única (Figura 2.4): são aquelas que possuem uma única camada de neurônios entre as unidades de entrada e saída da rede.

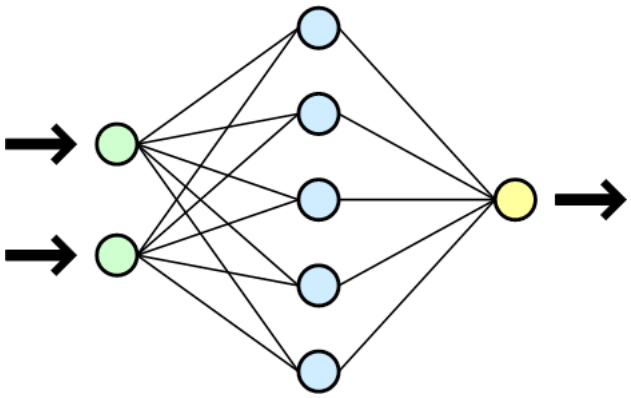

Figura 2.4: Exemplo de rede de camada única

- Redes de múltiplas camadas (Figura 2.5): são aquelas que possuem uma ou mais camadas intermediárias entre as unidades de entrada e saída da rede.

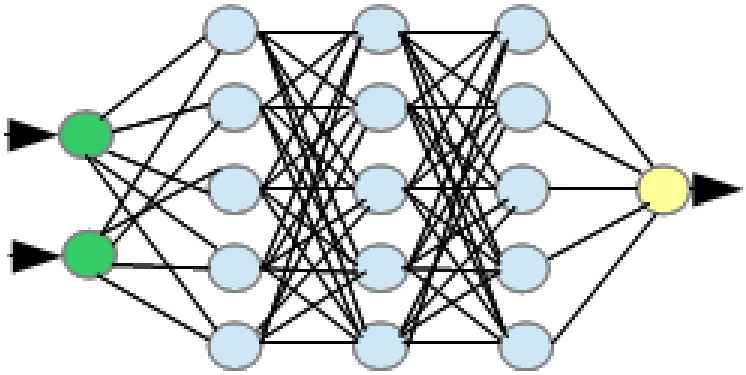

Figura 2.5: Exemplo de rede de múltiplas camadas

2.4.2 Funções de Ativação

Para o funcionamento correto da rede, é necessário uma regra que detenha o efeito do total de entradas sobre a unidade de ativação da rede, levando em consideração ainda o valor do bias. Para isso, temos a função de ativação que, como visto, funciona como um compressor para a saída do neurônio. Em todas as fórmulas apresentadas, temos que b_k é o valor do bias e u_k é dado pelo somatório da combinação das entradas x_j ponderadas pelos respectivos pesos sinápticos w_{kj} representado pela equação:

$$u_k = \sum_{j=1}^{n} w_{kj} x_j \tag{2.1}$$

O primeiro modelo de neurônio matemático proposto por McCulloch e Pitts (MCCULLOCH e PITTS, 1943) fazia uso de uma função de ativação denominada função sinal. A função sinal apresenta os seguintes valores:

- Função sinal:

$$y_k = \begin{cases} 1, \text{ se } u_k + b_k \geq \theta \\ -1, \text{ se } u_k + b_k < \theta \end{cases} \tag{2.2}$$

Entretanto, existem ainda outros tipos de função de ativação, como visto em (SMAGT e KROSE, 1996). Uma das mais usadas e mais simples é a função Degrau, apresetada pela equação 2.3. Vale a pena ressaltar a existência da função linear por

partes, ou semi-linear. A função semi-linear, apresenta valores entre -1 e 1 e a inclinação da reta é determinada pelo valor α. A equação 2.4 apresenta a equação da função linear por partes.

- Função Degrau:

$$y_k = \begin{cases} 1, \text{ se } u_k + b_k \geq \theta \\ 0, \text{ se } u_k + b_k < \theta \end{cases} \tag{2.3}$$

- Função Linear:

$$y_k = \begin{cases} 1, \text{ se } \mathrm{u_k} + b_k \geq \alpha \\ \mathrm{u_k} + b_k, \text{ se } \alpha > \mathrm{u_k} + b_k > -\alpha \\ -1, \text{ se } \mathrm{u_k} + b_k \leq -\alpha \end{cases} \tag{2.4}$$

Existem ainda as funções de ativação chamadas função sigmóide. Essas são caracterizadas por serem estritamente crescente e apresentarem um gráfico em formato de "s", como por exemplo a função logística e a função tangente hiperbólica. A função logística apresenta valores entre 0 e 1 enquanto a função tangente hiperbólica apresenta valores entre -1 e 1. Essa é a principal diferença entre essas funções de ativação e se dá devido a uma modificação da primeira por uma bias. A seguir são apresentadas as equações das duas funções sigmóides explicadas.

- Função Logística:

$$y_k = \frac{1}{1 + e^{-(u_k + b_k)}} \tag{2.5}$$

- Função Tangente Hiperbólica:

$$y_k = \tanh(u_k + b_k) \tag{2.6}$$

Graficamente, essas funções são apresentadas a seguir, na Figura 2.6.

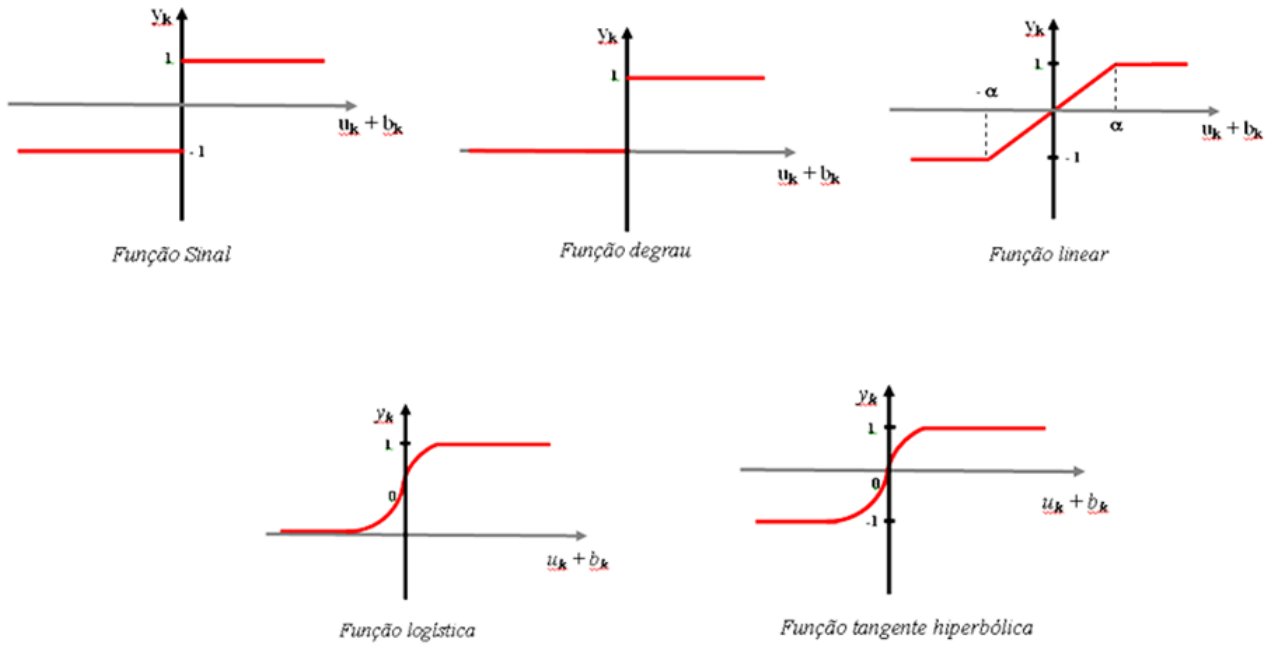

Figura 2.6: Funções de Ativação

Esses argumentos são capazes de definir a arquitetura e a topologia das Redes Neurais Artificiais que serão estudadas nos próximos capítulos desse trabalho.

2.5 Perceptron e Multilayer Perceptron

O perceptron começou a ser desenvolvido em 1957 por Frank Rosenblatt na Universidade de Cornell. Em 1958, foi apresentado à comunidade científica, através do trabalho intitulado "The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain" (ROSENBLATT, 1958). O trabalho de Rosenblatt apresentou contribuições significativas no campo inteligência artificial, apresentando um modelo de "sistema nervoso hipotético".

Esse modelo de classificador desenvolvido por Rosenblatt foi baseado no modelo de neurônio artificial desenvolvido por McCulloch e Pitts em 1943 (MCCULLOCH e PITTS, 1943). O classificador conta com a função degrau e sistema de aprendizado supervisionado. O perceptron no entanto, apresenta uma limitação em relação às aplicações possíveis, ele só pode ser aplicado para classificação de conjuntos linearmente separáveis.

O perceptron é um modelo de classificador binário, cuja saída y'_i está diretamente

relacionada a entrada do sistema x_i, ao valor computado para os pesos w_i e ao valor do bias b_i. A inclusão do termo bias se dá devido a necessidade de deslocamento do hiperplano para ajuste da classificação dos conjuntos. Esquematicamente, apresenta-se a equação 2.6 e a Figura 2.7. Esse classificador é composto apenas pela camada de entrada da rede, uma camada oculta e a camada de saída.

$$y_i' = x_i w_i + b_i \tag{2.7}$$

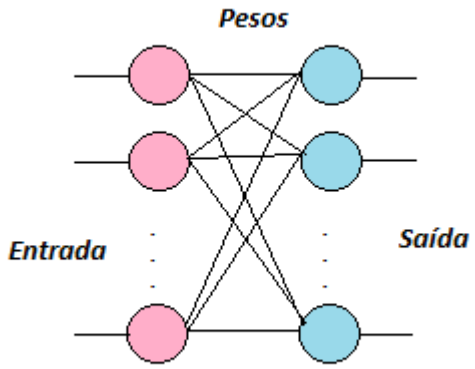

Figura 2.7: Modelo Perceptron

O perceptron multicamadas, mais conhecido como Multilayer Perceptron (MLP) é uma extensão do modelo de Rosenblatt com intuito de solucionar o problema da classificação de conjuntos linearmente separáveis. A inclusão de mais camadas ocultas na rede aumento o poder computacional do sistema, ao passo que dificulta a aplicação do algoritmo de treinamento.

As redes MLP constituem modelos relativamente simples de redes neurais feedforward. Nesse modelo, a função de ativação normalmente é não linear, em geral sigmóide. A saída do sistema é a mesma para o modelo do perceptron de camada única. Esquematicamente, o modelo MLP é apresentado na Figura 2.8.

O algoritmo de aprendizado para os modelos do perceptron e do MLP é o Backpropagation (BP) em sua forma padrão. As arquiteturas feedforward estudadas nesse

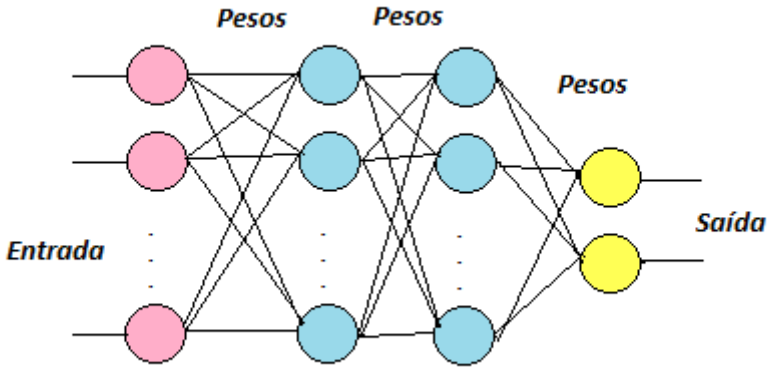

Figura 2.8: Modelo Perceptron de multicamadas

trabalho baseiam-se em redes do tipo MLP.

2.6 Sistemas Estáticos e Sistemas Dinâmicos

As redes neurais classificadas como feedforward são amplamente conhecidas por suas várias aplicações, entre elas, problemas de classificação e reconhecimento de padrões. Essas redes são os principais componentes dos chamados Sistemas Estáticos e, portanto, conhecidas como Redes Neurais Estáticas. Essas redes tem a capacidade de simular a característica de memória a longo prazo, assim como os seres humanos. A chamada memória a longo prazo consiste nas informações contidas nos PEs. Os PEs também representam memória sobre os dados treinados.

No entanto, existem alguns problemas que não podem ser modelados utilizando sistemas estáticos e redes feedforward, por exemplo, problemas que apresentam algum tipo de ordem no conjunto de dados de entrada, ou problemas que utilizam noção de temporalidade. Sistemas que solucionam esses problemas são chamados sistemas dinâmicos e são modelados através da utilização de redes recorrentes. Os sistemas dinâmicos, ou também chamadas redes dinâmicas, são topologias desenvolvidas para introduzir a inclusão das relações temporais no mapeamento de entradas e saídas de dados (PRINCIPE *et al.*, 2000). Nesses sistemas, os estados variam de acordo com o tempo.

Para tornar mais claro o entendimento dos conceitos aqui explorados, em PRIN-

CIPE *et al.* (2000), tem-se o seguinte exemplo. "Suponha-se que existam dois conjuntos com a mesma média e variância, mas pertencentes a classes diferentes. Treinar uma rede feedforward (MLP, RBF, etc.) para distinguir essas duas classes é obviamente impossível. No entanto, assume-se que essas duas classes tem históricos temporais diferentes, tendo sido criados por duas fontes de sinais sinusoidais de diferentes frequências. Fica claro que um combinador linear pode separar as duas classes, mas não é possível utilizar um MLP para que isso seja feito. Esse é um caso extremo que ilustra a diferença entre sistemas estáticos e dinâmicos. Na modelagem estática, são utilizadas propriedades estáticas do conjunto de dados de agrupamento para distingui-los entre uma classe e outra. Em sistemas dinâmicos, a noção temporal impõe uma estrutura do espaço de entrada que pode ser usada para separar o conjunto de dados com estatísticas de sobreposição, ou seja, a sequência na qual os dados são visitados são diferentes, e essa diferença pode ser usada para separação. Na prática, em muitos casos, não há informação sobre quando há ou não uma estrutura temporal subjacente ao conjunto de dados e então raramente fica claro quando o uso de uma modelagem dinâmica ajudaria a aumentar a performance do problema."

2.7 Aprendizado na rede

Baseado em HAYKIN (1994), BRAGA *et al.* (2000) e PRINCIPE *et al.* (2000), a seguir, são definidos quatro tipos de aprendizado nas RNAs.

- Aprendizado Supervisionado: Esse tipo de aprendizado consiste na apresentação de um conjunto de dados no qual os dados de entrada são acompanhados das respectivas classes desejadas e os dados de saída gerados pela rede são comparados aos dados das classes desejados. Esse modelo de aprendizado supervisionado é o mais comum e mais usado nas RNAs. "O método é chamado de aprendizado supervisionado porque a entrada e a saída desejadas para a rede são fornecidas por um 'supervisor' externo, também chamado de professor, com o objetivo de guiar os parâmetros da rede para que essa encontre as respectivas ligações entre os dados de entrada e saída correspondentes." BRAGA *et al.* (2000) A medida de desempenho do aprendizado

supervisionado é determinada pelo erro gerado pela rede, ou seja, dos pares de entrada e saída, quais a rede acertou, quais a rede errou para cada classe. Um exemplo de algoritmo de aprendizado supervisionado é o Backpropagation.

- Aprendizado Não Supervisionado: Esse método, como o próprio nome diz, consiste na ausência de um supervisor dentro do conjunto de dados, ou seja, não há o valor esperado de saída da rede previamente determinado, com base nos dados de entrada fornecidos. Nesse paradigma, não existe uma resposta desejada dentro dos dados da rede para posterior comparação com os resultados obtidos pela rede. As redes que trabalham com algoritmos de aprendizado não supervisionados trabalham buscando similaridades dentro do conjunto de dados de entrada, os grupos são criadas de acordo com as similaridades encontradas dentro dos dados de entrada. O ajuste dos pesos é feito da mesma forma, ou seja, conforme as semelhanças são encontradas, os pesos são ajustados para identificar dados com a mesma similaridade posteriormente. Em SMAGT e KROSE (1996) temos que o modelo de aprendizado não supervisionado é aquele no qual "a unidade de saída de dados é treinada para responder a grupos padronizados baseados nos dados de entrada. Nesse paradigma, o sistema deve descobrir características estatisticamente persistentes da população dos dados de entrada." Em KASKI e KOHONEN (1994) tem-se, como exemplo de algoritmo de aprendizado não supervisionado, o algoritmo Winner-take-all.

- Aprendizado por Reforço: Esse modelo é uma variação, caso particular, do aprendizado supervisionado. A diferença consiste no tipo de resposta à rede, no caso supervisionado, ocorre uma comparação entre a saída da rede e a resposta desejável presente na base de dados, enquanto no aprendizado por reforço só é informado se a saída gerada pela RNA é correta ou incorreta. Dessa maneira, dentro do paradigma de aprendizado por reforço não é fornecida para a rede a resposta correta ao dado fornecido na entrada do sistema. Além disso, "o paradigma de aprendizado por reforço pode ter três tipos de forma de aprendizado. Associativa, quando o meio fornece informações além do reforço, não associativo quando o sinal de reforço é a única entrada que o sistemas recebe do meio e classe III, quando os sinais de reforço e os padrões de entrada podem depender das saídas anteriores da rede" (BRAGA

et al., 2000). Como exemplo de algoritmo de aprendizado por reforço, tem-se o Q-learning (WATKINS e DAYAN, 1992).

Dentre as diversas formas de se classificar uma RNA mencionadas em PRINCIPE *et al.* (2000), será abordada nessa seção a classificação relacionada ao método de aprendizado de uma rede neural. Como dito anteriormente, o sistema de aprendizagem de uma RNA é similar ao sistema de aprendizagem do cérebro humano. O neurônio artificial pode aprender através de métodos de tentativas e erros, ou seja, através de exemplos. Contudo, o que mais chama atenção quando se trata de redes neurais é a capacidade que esse tipo de estrutura tem de generalização. Nesse trabalho, os algoritmos que aprendizado estudados são supervisionados.

Essa generalização se dá perante a configuração necessária da rede para que a partir dos valores de entrada sejam gerados valores de saída consistentes. A primeira etapa de configuração é a definição do algoritmo de aprendizagem a ser usado. Pelo menos dois conjuntos de dados são formados, o de treinamento e o de teste. Além disso, um conjunto de validação pode ser formado também. O conjunto de treinamento é responsável pelo aprendizado da rede. Um bom conjunto de treinamento aumenta a possibilidade de maior capacidade de generalização da rede. Já o conjunto de teste é responsável por testar a rede e verificar a eficiência da generalização. Os algoritmos de aprendizado são responsáveis por gerar atualização dos pesos sinápticos de uma rede durante a fase de treinamento. Essa atualização representa a capacidade que uma RNA vai ter de atender a situações diferentes quando aplicado o conjunto de teste, ou seja, dado o problema e novos dados de entrada presentes no conjunto de teste, ocorre a generalização. Durante o treinamento o ajuste dos pesos é feito de maneira a minimizar o erro de treinamento sem perder a capacidade de generalização, por isso, usa-se o conjunto de validação. Com isso encaminha-se os pesos ajustados para a solução do problema, caso essa exista.

O conjuntos de funções e procedimentos que tem como objetivo promover aprendizado em uma RNA é chamado de algoritmo de aprendizado. Existem vários modelos e algoritmos que visam promover aprendizado numa rede, os principais tipo de aprendizado são o supervisionado e o não supervisionado, segundo SMAGT e KROSE (1996) também conhecidos como aprendizado associativo e aprendizado auto ajustável, respec-

tivamente. Além desses dois tipos básicos, serão apresentados ainda o aprendizado semi-supervisionado e o aprendizado por reforço. Em BRAGA *et al.* (2000), são mencionados ainda outros diferentes mecanismos de aprendizado que não serão relatados nesse trabalho, são eles aprendizado hebbiano, modelo de Linsker, regra de Oja, regra de Yuille e modelo de Kohonen.

2.8 Medindo Desempenho

A questão relacionada a medição do desempenho de uma rede neural artificial envolve uma série de parâmetros e objetivos a serem alcançados, entre eles o tipo de algoritmo de aprendizado. Para cada tipo de aprendizado, a avaliação do desempenho é feito de uma forma diferente, no paradigma supervisionado, por exemplo, o desempenho da rede pode ser medido pela quantidade de amostras que a rede acerta em relação as classes inseridas como parâmetros de entrada. Existem no entanto, medidas que podem ser obtidas para todos os tipos de aprendizado, por exemplo, os tempos de treinamento, validação e teste.

Uma das medidas de desempenho mais usadas é a correção de erros. A correção de erros pode ser feita por vários métodos são aplicáveis a vários tipos de mecanismos de aprendizado e é usado como forma de avaliar a diferença entre um valor estimado e o verdadeiro valor da entidade. A visualização gráfica do cálculo do erro é apresentada abaixo, na Figura 2.9. Entre os métodos de correção de erros, tem-se o Erro médio Quadrado (MSE) e suas variações como a Raiz do Erro Médio Quadrado (RMSE), Erro Médio Percentual Absoluto (MAPE) e o Erro Médio Quadrado Normalizado (NMSE), entre outros, como apresentado em POMMERANZENBAUM (2009). O MSE é o principal mecanismo de correção de erros e é definido abaixo juntamente com o erro absoluto e o erro médio absoluto (MAD).

- Erro absoluto: Seja a um número exato e a' um número aproximado. O módulo da diferença entre os valores aproximado e exato é chamado de erro absoluto e dado pela equação 2.8.

$$\varepsilon = |a - a'| \tag{2.8}$$

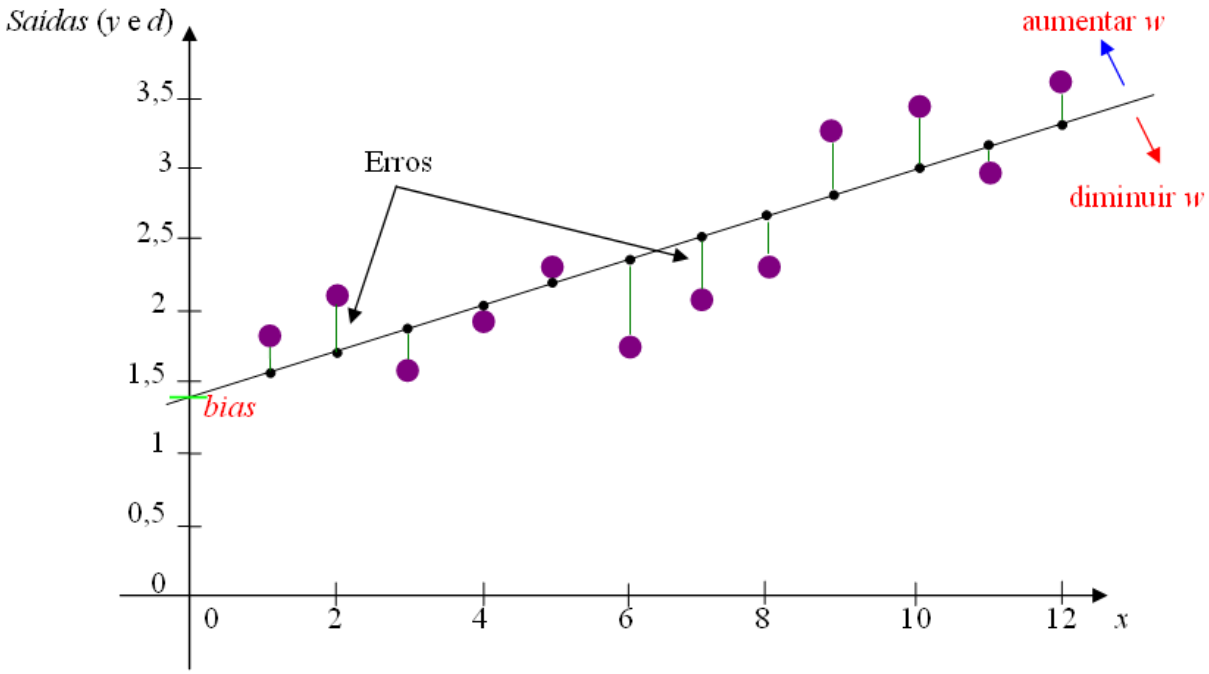

Figura 2.9: Cálculo do erro

- Erro Médio Quadrado (MSE): O objetivo para aumentar a eficiência da rede é minimizar o erro. A minimização aumenta a acurácia do método. Quando tem-se MSE = 0, implica em perfeita acurácia do método, no entanto, esse valor dificilmente é obtido em problemas reais. Seja ε_i o erro absoluto para uma saída i e n o total de saídas da rede. O MSE é ser expresso pela equação 2.9.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\varepsilon_i)^2 \tag{2.9}$$

- Erro Médio Absoluto ou Desvio Médio Absoluto (MAD): Este tipo de erro é muito usado para delimitar os erros cometidos quando trata-se de aproximação. O calculo do erro absoluto é dado pelo módulo da diferença entre o valor real e o valor estimado e é dado pela equação 2.10.

$$MAD = \frac{1}{n}\sum_{i=1}^{n}|\varepsilon_i| \tag{2.10}$$

Essas duas medidas serão usadas mais a frente no trabalho, durante a seção de experimentos.

3 Time-lagged Feedforward Network e Time Delay Neural Network

Esse capítulo visa abordar duas topologias que se mantém em um nível intermediário de complexidade, não sendo topologias voltadas para redes estáticas nem topologias para redes inteiramente recorrentes. Alguns conceitos iniciais são necessários antes da apresentação das topologias de rede.

3.1 Conceitos Iniciais

As TLFNs embora sejam redes parcialmente estáticas, são componentes de sistemas dinâmicos, e foram desenvolvidas para incluir relacionamentos temporais, explicitando essa relação para o mapeamento de dados temporais de entrada e dados de saída.

É importante ressaltar que em sistemas temporais o conjunto de medidas extraídas para formação dos dados de entrada deixam de ser um conjunto de amostras de entrada independentes e passam a ser funções de tempo. Sendo assim, em PRINCIPE *et al.* (2000) tem-se que a alteração da ordem das medidas representam uma distorção no sinal de tempo gerado pelo sensor que produz a sequência de amostras de entrada. Por isso, em sistemas temporais a ordem das amostras deve ser preservada, ao contrário do que acontece em sistemas estáticos.

A próxima seção aborda o paradigma adotado pela topologia TLFN e os conceitos necessários sobre esse padrão.

3.2 Paradigma para processamento de sinais temporais

Para que seja possível a exploração de sinais temporais é necessário que algumas alterações sejam feitas na máquina de aprendizado.

Em sistemas dinâmicos, é incluído um novo termo para considerar a informação temporal de passado disponível como parte dos dados de entrada dentro das topologias de RNA. Esse termo é chamado de memória curta, ou short-term memory.

O objetivo inicial da inclusão da memória curta no modelo é transformar a rede estática em uma rede dinâmica. Com a incorporação dessa memória na estrutura, a saída de rede passa a ser uma função temporal. A rede então passa a conter antiga parte estática associada agora a memória.

Uma forma de implementação dessa estrutura é apresentada em HAYKIN (1994), e consiste em uma forma simples de estruturar a rede neural com a utilização de atraso temporais (time delays), que podem ser implementados a niveis sinápticos dentro da rede ou através da camada de entrada.

"A estruturas de memória curta presentes nos sistemas de RNAs dinâmicos, ou também chamados de conexões recorrentes são sensíveis à sequência de informação apresentada. Quando comparados com sistemas estáticos, os pesos estáticos funcionam de maneira diferente dos pesos dinâmicos que são capazes de codificar informações sobre os dados de entrada de maneira aprimorada através da filtragem que ocorre dentro da janela de memória curta" (PRINCIPE *et al.*, 2000).

Portanto, tem-se a importância da modificação do paradigma de processamento para inclusão de sinais temporais processados pela memória curta. A representação temporal é criada então dentro da máquina de aprendizagem em oposição ao janelamento do sinal de entrada. O novo paradigma de processamento de sinais é apresentado a seguir, na Figura 3.1.

Arquiteturas de rede para processamento temporal podem tomar várias formas. A próxima seção detalha uma das primeiras topologias feedforward que fazem uso desse tipo de memória, a topologia Time-lagged Feedforward Network.

3.3 Time-lagged Feedforward Networks

O reconhecimento de padrões que envolvem temporalidade requer um processamento de dados que toma não somente a noção temporal,mas também da resposta a um determinado instante de tempo. Esse ultimo, por sua vez, depende não só do valor da entrada

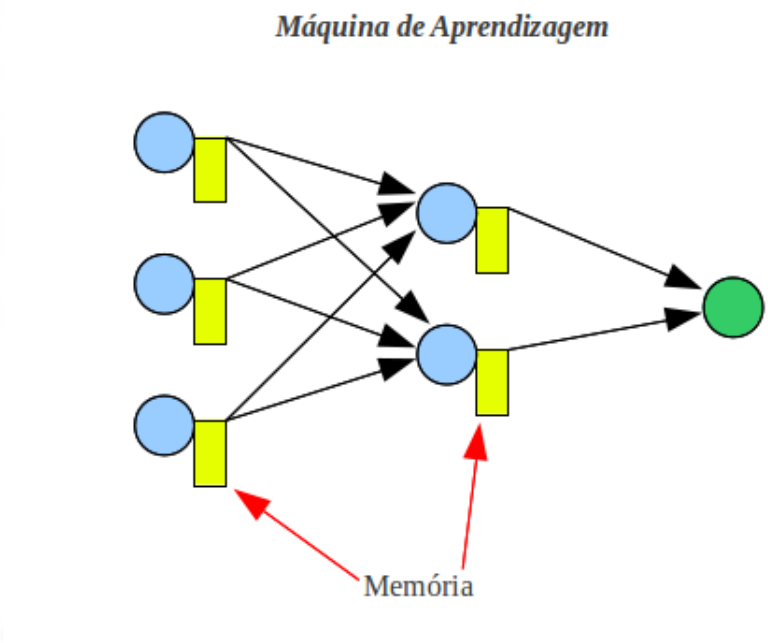

Figura 3.1: Paradigma de Processamento de Sinais Temporais

relacionada, como também dos dados do passado relacionados a essa entrada.

"Quando se utiliza uma rede neural para realizar processamento temporal deve-se sempre atentar para o problema da estabilidade. No caso de uma rede TLFN deve-se criar um mecanismo de memória de curto prazo estável, o que não é muito complexo. No caso de redes recorrentes é bem mais difícil este cálculo. Em função disto, o treinamento de uma rede TLFN é mais fácil que o treinamento de uma rede recorrente genérica. Entretanto, apesar da rede TLFN funcionar como um mapeador universal, certas funções exigem um tamanho muito grande da rede para atingir as características especificadas" (NIEVOLA, 2005).

Segundo PRINCIPE *et al.* (2000), para o caso das topologias TLFN, uma das vantagens mais significativas observadas ao fazer uso dessa arquitetura dá-se pelo fato de que como são redes com processamento adiante, ou seja, feedforward, elas carregam consigo algumas das propriedades desse tipo de rede que não envolve temporalidade, nesse caso, as TLFN apresentam já pelo tipo de processamento uma estabilidade básica intrínseca.

As redes neurais feedforward para processamento temporal existem em mais de uma forma, assim como o tipo de memória curta associada a elas. As redes chamadas de

Time-lagged Feedforward Networks (TLFN) foram um dos primeiros modelos feedforward com noção temporal implementados. Para o treinamento, o principal algoritmo usado é o Backpropagation estático.

De maneira genérica, a estrutura básica de uma TLFN é apresentada na Figura 3.2.

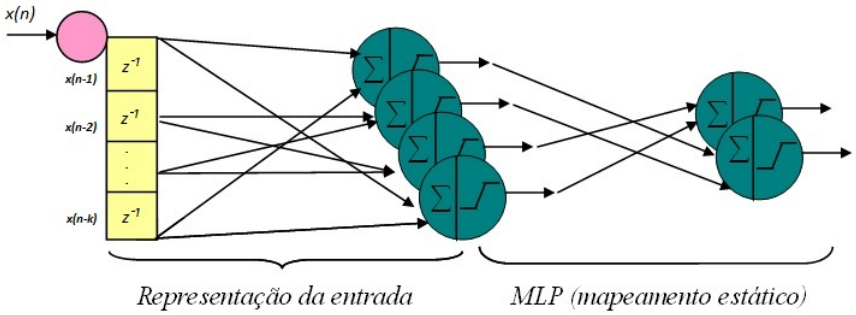

Figura 3.2: Time-lagged Feedforward Network

As TLFN apresentam duas arquiteturas de redes básicas. As TLFN simples, cuja arquitetura envolve memória curta em qualquer uma das camadas da rede, sejam elas camadas de entrada ou camadas implícitas na rede. E também as focadas, em NIEVOLA (2005), chamada Focussed TLFN, topologia abordada na próxima seção. Essa arquitetura apresenta a estrutura de memória somente na camada de entrada, nessas topologias primeiramente ocorre representação temporal linear, associada a memória, em seguida, ocorre o início do estágio de mapeamento usando uma rede feedforward, por exemplo, o Multilayer Perceptron (MLP) ou as redes Radial Base Function (RBF).

Como a diferença nas topologias consiste apenas na localização da memória curta aplicada, o estudo será aprofundado, a seguir, considerando apenas as TLFN focadas.

3.3.1 Time-lagged Feed Forward Networks Focadas

Segundo HAYKIN (1994), em estruturas do tipo TLFN Focadas, um MLP estático adquire a capacidade de processamento temporal. A serie temporal é vista pelo MLP como vários mapeamentos feitos a partir de um vetor de dados de entrada até o valor de saída gerado pela rede.

Em PRINCIPE *et al.* (2000) e HAYKIN (1994) é apresentada a técnica para

construção de uma TLFN. A estrutura apresentada leva em consideração a construção da rede usando um MLP.

Como mencionado anteriormente, as TLFN focadas compõem um tipo de rede feedforward que apresenta uma estrutura de memória curta contida nos elementos de processamento (PE) da camada de entrada exclusivamente e, ainda, uma série de PEs com características não lineares.

Para essa topologia, é construído um filtro não linear dentro da rede estática e mostrado na Figura 3.3, na qual $x(n)$ representa os dados de entrada da rede, $y(n)$ os dados de saída e $d(n)$ a resposta desejada. Esse tipo de rede pode ser implementado com somente um neurônio ou composto por uma rede de neurônios, ambas as estruturas são mostradas na Figura 3.4 e na Figura 3.5.

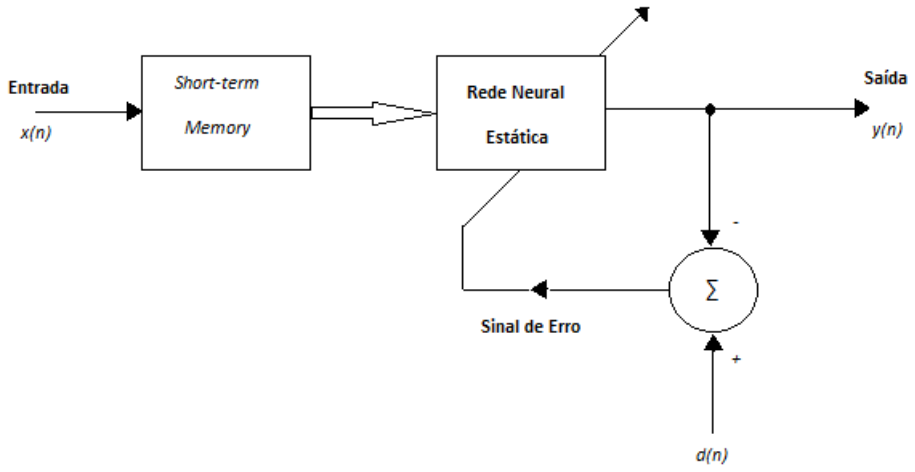

Figura 3.3: Filtro Não Linear

A Figura 3.5 refere-se a uma TLFN focada e esse sistema é apresentado através de um filtro não linear formado por dados de memória atrasada de ordem p , ou seja, presentes no passado do problema e componentes dos dados de entrada. Em HAYKIN (1994), tem-se a aplicação um exemplo supondo um treinamento com o Backpropagation estático e, assumindo que a MLP usada possui apenas uma camada oculta.

Ao tempo t, para $t = n$, tem-se um padrão temporal aplicado a camada de entrada da rede, vetor sinal, que pode ser apresentado como o estado do filtro linear, de ordem p, para um tempo $t = n$.

O vetor sinal é dado pela equação 3.1.

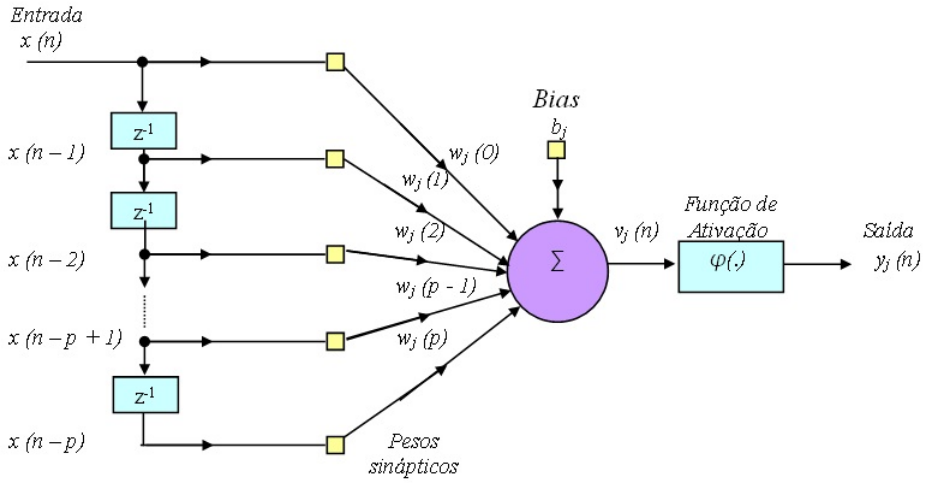

Figura 3.4: TLFN Focada com um neurônio

$$x(n) = [x(n), x(n-1), \cdots, x(n-p)]^T \tag{3.1}$$

A saída do filtro não linear, assumindo que o MLP apresenta uma única camada oculta, como mostrado na Figura 3.5, é dado pela equação 3.2, na qual admite-se que a saída do neurônio da TLFN focada é linear. Os pesos sinápticos de saída do neurônio são denotados por w_j, com j variando de 1 até m_1, e m_1 é o tamanho da camada oculta do MLP. O componente bias é denotado por b.

$$\begin{aligned} y(n) &= \sum_{j=1}^{m_j} w_j y_j(n) \\ &= \sum_{j=1}^{m_j} w_j \varphi(\sum_{l=0}^{p} w_j(l) x(n-1) + b_j) + b_0 \end{aligned} \tag{3.2}$$

Em ambos os casos, o mapeamento completo de todas as entradas e saídas é dividido em duas etapas subsequentes. Inicialmente, tem-se a camada de PEs da memória, que representa um estágio de representação temporal linear. Um mapeador, agora não linear, da camada de representação abstrata dos dados de saída do sistema. Esse estágio de representação procura a melhor projeção do sinal de entrada dentro de uma projeção intermediária aceitável, com intuito de minimizar o erro de retropropagação e, consequen-

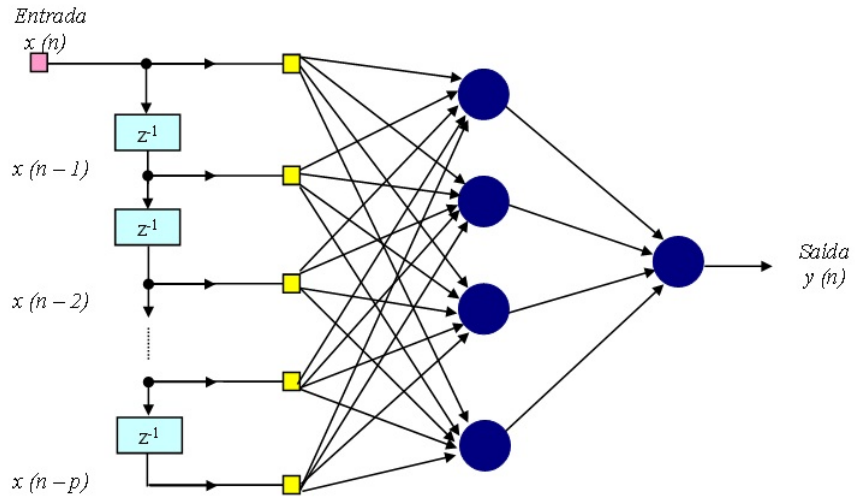

Figura 3.5: TLFN Focada composta por uma rede de neurônios

temente minimizar o erro de saída do sistema futuramente.

Ainda assim, de acordo com PRINCIPE *et al.* (2000), encontradas as medidas certas para que sejam obtidos os pesos ótimos, o estágio de representação da saída é equivalente à projeção ortogonal do sinal de entrada dentro do espaço de projeção.

A rede feedforward usada para a construção do modelo, nesse caso a MLP, é a responsável por todo esse mapeamento, uma vez que as entradas da MLP são as saídas dos filtros da memória curta implementada. Ainda de acordo com PRINCIPE *et al.* (2000), nessas estruturas a quantidade de filtros de memória e número de PEs da MLP são relacionados.

3.4 Time Delay Neural Networks

Primeiramente proposta por LANG e HINTON (1988) e WAIBEL *et al.* (1989) as Time Delay Neural Networks (TDNN) são um tipo específico de TLFN, nas quais as camadas intermediárias e a saída do sistema são duplicadas ao longo de um tempo t. Esse tipo de arquitetura é feedforward e composta por um MLP que tem sua primeira camada substituída por uma linha de atraso temporal. Como trata-se de uma especiação das TLFN, as TDNN também podem ser treinadas utilizando algoritmos de treinamento de redes estáticas. Segundo PRINCIPE *et al.* (2000) a razão para esse tipo de rede poder

ser treinada usando um algoritmo de treinamento estático reside no fato que a linha de atraso na camada de entrada não contém parâmetro livres, então apenas os parâmetros que sofrem adaptação na rede com o passar do tempo estão contidos no caminho feedforward estático.

"Uma vez treinada a rede, todos os pesos são fixados. A rede pode, então, ser utilizada para operar um sinal de entrada de tempo real, propagando o sinal através da rede, camada por camada" (BRAGA *et al.*, 2000).

A arquitetura TDNN é conhecida por ter um aumento de eficiência quando comparada as TLFN. No entanto, corresponde a um ponto de dificuldade de modelagem do problema a determinação do tamanho da linha de atraso. As TDNNs são de natureza focadas e a estrutura de memória é carregada para dentro da máquina de aprendizagem. Essa topologia de rede associa uma estrutura mais simples do que a de uma TLFN a uma capacidade de processamento notável.

Segundo HAYKIN (1994), as TDNNs tendem a apresentar uma maior eficiência em problemas de classificação de padrões temporais, que consistem em sequências de vetores de características dimensionais fixas, como fonemas (menor unidade sonora de uma palavra). No entanto, existe uma infinidade de outras aplicações, por exemplo para identificação de sistemas, no qual o principal objetivo é tornar a saída da TDNN o mais próximo possível da saída do sistema a ser modelado. Outra aplicação possível é na previsão de dados futuros baseados numa combinação não linear das amostras anteriores de entrada, como mencionado em NIEVOLA (2005).

A Figura 3.6 apresenta a arquitetura de uma TDNN contendo apenas uma camada oculta do MLP e uma linha de atraso com k + 1 estágios.

Embora as redes feedforwards que incorporam noções temporais à sua arquitetura representem uma boa solução para problemas temporais, elas ainda não apresentam um modelo ideal para esse tipo de problema. As redes ideais para problemas dinâmicos, inclusive os temporais, são as chamadas redes recorrentes, que serão estudadas no capítulo seguinte.

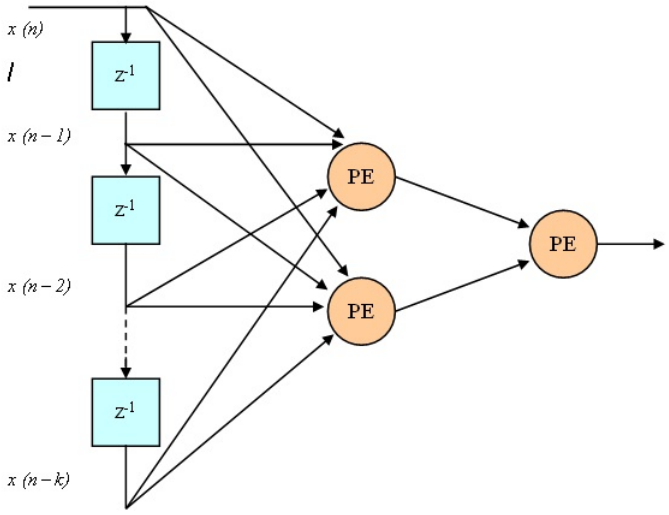

Figura 3.6: Time Delay Neural Network

3.5 Algoritmo Backpropagation

O algoritmo backpropagation foi primeiramente mencionado idealmente no final da década de 60, precisamente em 1969 por Arthur Earl Bryson e Yu Chi-Ho no trabalho intitulado "Applied Optimal Control: Optimization, Estimation and Control" (BRYSON e HO, 1975). Em 1974, Paul Werbos apresentou em sua tese de doutorado um modelo de treinamento de redes neurais artificiais através das propagações dos erros e, posteriormente, publicou o livro "The Roots of Backpropagation" (WERBOS, 1994).

No entanto, consta de 1986 a apresentação do algoritmo como tem sido utilizado até hoje. Desenvolvido por David E. Rumelhart, Geoffrey E. Hinton, Ronald S. Willians, foi apresentado no trabalho intitulado "Learning Representations by Back-propagating Error" (RUMELHART *et al.*, 1986). O crédito pela criação do algoritmo é dado a esses autores pois, além de proporem sua utilização no aprendizado de máquina, apresentaram ainda uma aplicação prática do algoritmo.

Esse método é comumente utilizado para treinamento de redes neurais, mas principalmente voltado para redes do tipo MLP, constituídas por uma camada de entrada, um número finito de camadas intermediárias escondidas, ou ocultas, e uma camada de saída. As topologias estudadas nesse capítulo, as TLFN e as TDNN, constituem redes do tipo feedforward, cuja arquitetura é baseada nas redes MLP e, por isso, a forma padrão

do backpropagation é a mais indicada para treinamento dessas redes.

Ainda assim, a forma padrão desse algoritmo de treinamento é dividida em duas vertentes. Tem-se a linha estática e a linha dinâmica, como apresentado em BRAGA *et al.* (2000).

"Enquanto o algoritmo estático não altera a estrutura da rede, variando apenas os valores de seus pesos, o algoritmo dinâmico pode tanto reduzir quanto aumentar o tamanho da rede (número de camadas, número de nós nas camadas intermediárias e número de conexões)".

Vale ressaltar que quando se utiliza o algoritmo backpropagation estático, uma mesma regra de aprendizado é empregada, independentemente do tipo de rede MLP utilizada, seu tamanho ou seu formato.

O algoritmo foi baseado na regra delta, usada em redes que possuem função de ativação linear, e na generalização da mesma, para funções de ativação não lineares.

3.5.1 Regra delta

A regra delta foi proposta por Bernard Widrow e Marcian E. Hoff em 1960 e apresentada no trabalho "Adaptive Switching Circuits" (WIDROW e HOFF, 1960) e é também conhecida como método Least Mean Square, ou LMS.

Esse modelo é um dos mais conhecidos para regras de aprendizado e, consiste em uma regra de aprendizado sobre a descida do gradiente para atualização dos pesos da rede, com intuito de minimizar o erro de saída do sistema. Sendo assim, se a diferença entre o valor de saída gerado pela rede e o valor esperado é zero, tem-se que ocorreu aprendizado. Caso contrário, os pesos são novamente ajustados visando minimizar essa diferença.

Essa regra é aplicada em sistemas que consideram apenas uma camada de entrada e uma camada de saída, considerando ainda uma função de ativação linear. A saída desse sistema é dada pela fórmula descrita pela equação 3.3.

$$y' = \sum_j w_j x_j + b \tag{3.3}$$

Sendo y' a saída gerada pela rede composta por j neurônios, w_j corresponde ao peso de j-ésimo sinal de entrada x_j a ser ajustado, multiplicados entre si e acrescidos de um valor b correspondente ao valor da bias.

A regra delta para o caso mais simples descrito acima é dada, para um neurônio j e para o i-ésimo peso w_{ji} pela fórmula apresentada pela equação 3.4, na qual α é uma constante chamada de taxa de aprendizagem, y_i é o i-ésimo valor desejado de saída, y_i é o i-ésimo valor obtido pela rede e x_i é o i-ésimo valor de entrada da rede.

$$\Delta w_i = \eta(y_i - y_i')x_i \tag{3.4}$$

A derivação da regra delta parte do principal objetivo dessa regra de aprendizagem, a minimização do erro. O erro ligado a função de ativação é dado pelo LMS.

$$E = \sum_i E_i = \frac{1}{2}\sum_i (y_i - y_i') = \frac{1}{2}\sum_i \delta_i \tag{3.5}$$

O LMS encontra valores para todos os pesos de forma a minimizá-los. A minimização do erro segue a regra da descida do gradiente que, na verdade, tem como objetivo alterar o valor dos pesos proporcionalmente ao valor negativo da derivada do erro gerado por cada peso, dado por E_i, em relação ao valor desse peso, dado por w_i, como é mostrado na equação 3.6.

$$\Delta w_i = -\frac{\partial E_i}{\partial w_i}\eta \tag{3.6}$$

A derivada da sentença acima é dada por:

$$\frac{\partial E_i}{\partial w_i} = \frac{\partial E_i}{\partial y_i}\frac{\partial y_i'}{\partial w_i} = \frac{\partial E_i}{\partial y_i'}(x_i) \tag{3.7}$$

De 3.5 e 3.7 temos que:

$$\frac{\partial E_i}{\partial y_i'} = -(y_i - y_i') = -\delta_i \tag{3.8}$$

Substituindo em 3.6, tem-se a regra delta.

$$\Delta w_i = \eta(y_i - y_i')x_i = \eta(\delta_i)x_i \tag{3.9}$$

A generalização da regra delta, como mostrado pela equação 3.4, se dá através da inclusão do termo $\varphi(x)$ dado pela função de ativação usada no modelo. A generalização da regra é então definida a seguir.

$$\Delta w_i = \eta(\delta_i)x_i\varphi'(x) \tag{3.10}$$

Para o caso anteriormente descrito, no qual considera-se a função linear de ativição, temos que $\varphi'(x) = 1$.

3.5.2 Algoritmo Backpropagation

O processo aplicado pelo Backpropagation (BP) é bastante simples, e de maneira geral é composto por duas etapas. Cada etapa ocorre em um sentido de fluxo, uma "para frente" (forward) e a outra "para trás" (backward).

Conforme a rede é treinada, o erro tende a ser minimizado. Uma precisão deve ser determinada como critério de parada. Esse primeiro passo determina a fase forward e é chamado de passo de propagação. Esquematicamente, observa-se a Figura 3.7.

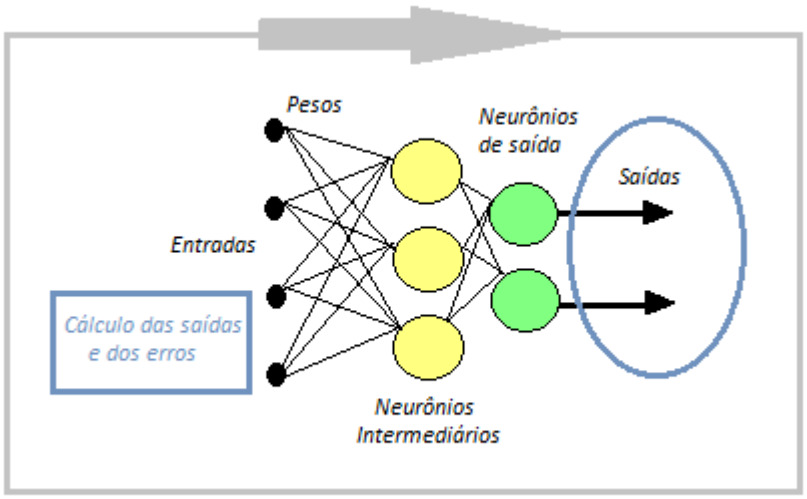

Figura 3.7: Esquema do Passo de Propagação

O segundo passo é o backward e é comumente chamado de retropropagação. Consiste em sucessivas alterações nos pesos sinápticos desde a camada de saída da rede até a camada de entrada. Essas alterações são definidas pela aplicação da regra delta. Ocorre uma correção de pesos no sentindo inverso, da camada de saída até a primeira camada oculta da rede, daí o nome backward. Esquematicamente, tem-se a Figura 3.8.

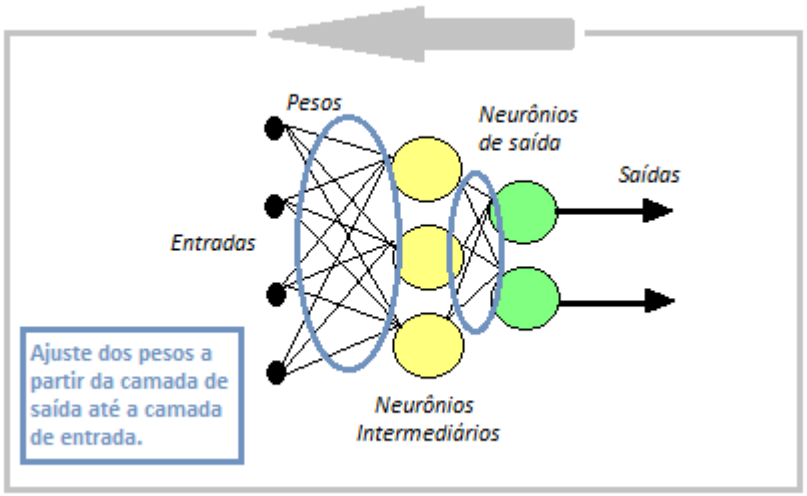

Figura 3.8: Passo Backward

De acordo com BRAGA *et al.* (2000), o algoritmo do BP é um algoritmo de aprendizado supervisionado que utiliza pares de entradas e saídas desejadas, para, por

um meio de correção de erros, ajustar os pesos da rede.

O termo backpropagation deriva de "backward propagation of error" . De maneira bem objetiva, segundo SMAGT e KROSE (1996), seu funcionamento consiste primeiramente na propagação dos valores de ativação para as unidades de saída, o que tende a gerar um erro para cada unidade de saída. O objetivo é fazer o erro convergir para zero. Em segundo lugar, deve-se aplicar a regra delta, com intuito de reduzir o erro.

A fase forward do algoritmo consiste no somatório ponderado das entradas, dada pela esquação 3.3. Obtendo o resultado do somatório, deve-se então aplicar a função de ativação. Para redes MLP as funções mais comuns são a logística e a tangente hiperbólica, ambas apresentadas no Capítulo 2. Aplicada a função de ativação, é gerada a saída da rede. Essa saída funciona como entrada da próxima camada e sucessivamente.

Tendo sido determinada a saída final da rede, calcula-se então o erro gerado para cada neurônio da rede. Seja $e_j(n)$ o erro para um dado neurônio j da rede, $y_j(n)$ a saída esperada da rede e $y'_j(n)$ a saída obtida. O erro é dado pela equação 3.12.

$$e_j(n) = y_j(n) - y'_j(n) \tag{3.11}$$

Os passos do algoritmo backpropagation são apresentados adiante (BRAGA *et al.*, 2000).

- Algoritmo Backpropagation

  1. Inicializar pesos e parâmetros.
  2. Repetir até o erro ser mínimo ou até a realização de um dado número de ciclos:

     2.1 Para cada padrão de treinamento X

     2.1.1 Definir a saída da rede através da fase *Forward*.

     2.1.2 Comparar as saídas produzidas com as saídas desejadas.

     2.1.3 Atualizar os pesos dos nodos através da fase *Backward*.

- Passo de Propagação ou Fase Forward

  1. A entrada é apresentada à primeira camada de rede (camada C^0).

2. Para cada camada C^i a partir da camada de entrada:

   2.1 Após os nodos da camada $C^i(i > 0)$ calcularem seus nodos da camada de saída, estes servem como entrada para a definição das saídas produzidas pelos nodos da camada C^{i+1}.

3. As saídas produzidas pelos nodos da última são comparadas às saídas desejadas. Calcula-se o erro para cada neurônio.

- Passo de Retropropagação ou Fase Backward

1. A partir da última camada, até chegar na camada de entrada:

   1.1 Os nodos da camada atual ajustam seus pesos de forma a reduzir seus erros. Aplica-se a regra delta.

   1.2 O erro de um nodo das camadas intermediárias é calculado utilizando os erros dos nodos da camada seguinte conectados a ele, ponderados pelos pesos das conexões entre eles.

4 Rede de Elman e Rede de Jordan

Este capítulo visa abordar duas das primeiras redes recorrentes desenvolvidas visando a abordagem temporal, são elas as Redes de Elman, desenvolvidas por Elman em 1990 e, as Redes de Jordan, introduzidas primeiramente por Jordan em 1986.

Embora essas redes sejam amplamente utilizadas até hoje, existem ainda outros modelos de redes recorrentes, não tão simples quanto os modelos de Elman e Jordan mas também eficientes. Entre esses modelos, tem-se a topologia de rede distribuída TLFN, as redes de Hopfield, o modelo de Groosberg e o modelo de Freeman (PRINCIPE *et al.*, 2000).

As redes recorrentes podem ser aplicadas a uma gama considerável de problemas, em diversas áreas de conhecimento. Além das aplicações citadas anteriormente nesse trabalho, que visam apenas a abordagem temporal, em JAIN (2001) e FETZ e SHAPE (2002) tem-se que esse tipo de rede dinâmica também pode ser aplicado em sistemas neurofisiológicos relacionados ao entendimento do mecanismo de comportamento cerebral.

Outra aplicação estabelecida para esse tipo de rede foi proposto por LIANG *et al.* (1999) e consiste em um método desenvolvido para síntese musical de instrumentos chineses de corda.

Jordan e Elman se propuseram a criar uma rede simples, baseada nos elementos de processamento e recorrência de redes que apresentassem um fácil treinamento e conseguissem realizar mapeamentos baseados em pequenas topologias. A seções 4.2 e 4.3 tratam especificamente dessas topologias.

4.1 Conceitos Iniciais

Uma rede recorrente é, basicamente, uma rede neural que apresenta correções de loops fechados, ou seja, ciclos.

A construção desse tipo de rede neural varia de acordo com a arquitetura de rede pretendida. Além de fixar a arquitetura, ainda assim, existem várias maneiras de se fazer

essa construção. Duas arquiteturas básicas são definidas e apresentadas nas Figura 4.1 e Figura 4.2 abaixo mostradas.

- Redes completamente recorrentes ou fully connected recurrents networks: As redes completamente recorrentes são caracterizadas por possuírem interconexões entre todos os nós das rede, ou seja, essas redes não tem sinais de entrada distintos entre os nós, todos os nós apresentam sinais de entrada compartilhados, de todos os outros nós da rede. Podem ainda existir sinais de entrada vindos do próprio nó.

- Rede recorrentes simples ou Simple recurrent networks: Nesse tipo de arquitetura, embora alguns nós façam parte de uma arquitetura essencialmente feedforward, outros nós dentro da mesma estrutura recebem sinais de entrada de outros nós da rede, consistindo em uma associação entre o modelo feedforward e o modelo de rede completamente recorrente.

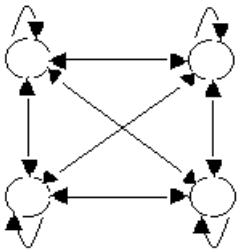

Figura 4.1: Rede Completamente Recorrentes

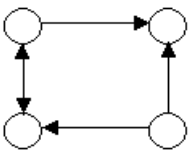

Figura 4.2: Rede Parcialmente Recorrente

Elman e Jordan introduziram o conceito de recorrência de maneiras diferentes, mas ambos utilizaram o modelo de redes recorrentes simples, ou seja, nem todos os nós da rede apresentam recorrência.

Vale ainda ressaltar que para as redes recorrentes, não é mais possível usar a implementação do backpropagation estático usadas nas topologias TLFN e TDNN para treinamento. Os algoritmos de treinamento de redes recorrentes são um pouco mais complexos. Entre esses algoritmos, tem-se uma variação do backpropagation chamada de backpropagation through time (BPTT) que será estudado no Capítulo 5.

Além da explicação relativa aos algoritmos de aprendizado para redes recorrentes, é necessário ainda uma definição formal dos modelos de Elman e Jordan, que foram utilizados durante os experimentos desse trabalho. As próximas seções se dedicam a essa formalização dos modelos.

4.2 Rede de Jordan

A partir de uma rede MLP simples, contendo três camadas, uma de entrada, uma camada escondida intermediária da rede e uma camada de valores de saída, Jordan, em 1986, apresentou o trabalho intitulado "Attractor dynamics and parallelism in a connectionist sequential machine" (JORDAN, 1986), no qual cria um modelo que acrescenta a essa rede simples, uma nova camada de nós intermediários chamados unidades de contexto. Essa nova camada é responsável por introduzir a noção temporal e, além disso, introduzir recorrência a rede.

Esse modelo, chamado de Redes de Jordan, apresenta recorrência parcial, foi um dos primeiros modelos de redes neurais recorrentes criadas e, esquematicamente, esse modelo é apresentado na Figura 4.3.

No modelo de Jordan, a recorrência apresenta-se nas ligações das unidades de contexto aos nós da camada interna oculta da rede que se ligam aos nós da camada de saída e essa última, por sua vez, volta a conectar-se aos nós da camada de unidades de contexto, criando um ciclo que determina a recorrência da rede. Além disso, os nós da camada de contexto apresentam ligações recorrentes internas, ou seja, os sinais saem de um nó da camada de contexto e retornam ao mesmo nó da camada, de maneira direta. Sendo assim, a saída da rede de Jordan é copiada para as unidades de contexto. Esse modelo é definido como localmente recorrentes.

A quantidade de nós da camada de unidades de contexto é diretamente propor-

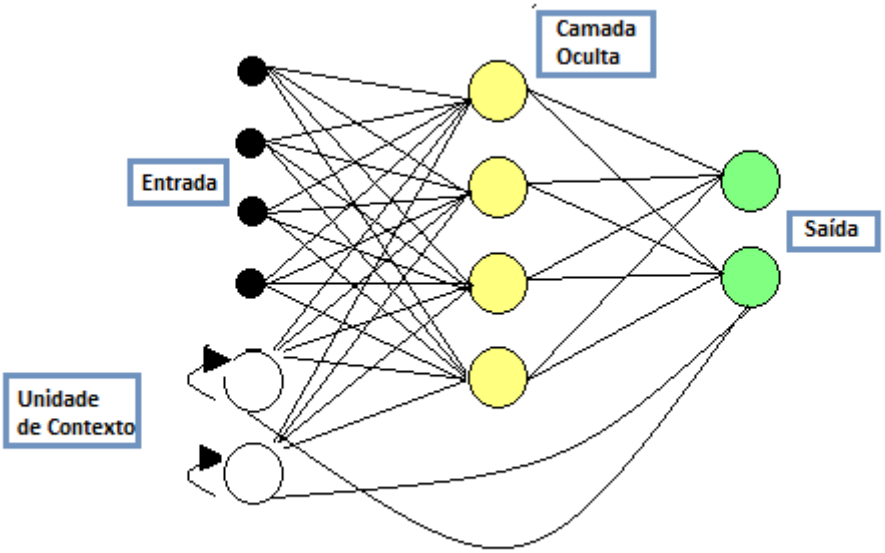

Figura 4.3: Rede de Jordan

cional à quantidade de unidades de saídas da rede.

Segundo SMAGT e KROSE (1996), as conexões entre a saída da rede e as unidades de contexto tem seus pesos fixados em +1 e o aprendizado se dá apenas entre as unidades da camada de entrada e a camada oculta e entre a camada oculta e as unidades da camada de contexto. Sendo assim, todas as regras de aprendizagem usadas para treinamento das redes MLP podem ser usadas para treinamento das redes de Jordan.

4.3 Rede de Elman

O modelo de Elman foi desenvolvido por Jeffrey L. Elman e apresentado à comunidade científica em 1990, através do trabalho intitulado “Finding Structure in Time” (ELMAN, 1990). Esse modelo é baseado no modelo de Jordan segundo dois aspectos.

Primeiramente, o modelo de Elman também foi criado a partir de uma rede MLP simples com 3 camadas, a de entrada, a de saída e uma camada intermediária escondida. Além disso, o modelo de Elman introduz a camada de unidades de contexto, introduzidas pelas redes desenvoldidas por Jordan.

Nesse modelo, as unidades da camada de entrada e da camada de saída são responsáveis pela interação com o ambiente externo. As unidades da camada de contexto

funcionam como unidades adicionais da camada de entrada.

A recorrência nas redes de Elman é estabelecida entre a camada intermediária e as unidades de contexto. A camada escondida fornece conexões de saída como entrada das unidades de contexto que, por sua vez, fornecem sinais de saída como sinais de entrada para os nós da camada oculta.

Segundo KOSKELA *et al.* (2000), os valores das unidades de contexto são completamente recorrentes aos valores da camada oculta. No entanto, o modelo da rede de Elman é um modelo parcialmente recorrente.

Também no modelo proposto por Elman, os nós da camada intermediária e das unidades de contexto são conectados com pesos fixados em +1 como no modelo de Jordan. Além disso, a quantidade de neurônios das unidades de contexto são diretamente proporcionais a quantidade de neurônio da camada de saída.

Apesar da semelhança, SMAGT e KROSE (1996) destaca duas diferenças substânciais entre os modelos de Elman e Jordan. Primeiro, as unidades da camada de contexto não apresentam ligações para as próprias unidades de contexto. Além disso, no modelo de Jordan, a recorrência é dada em conjunto com a camada de saída da rede, enquanto no modelo de Elman só as unidades de contexto e a camada oculta apresentam recorrência.

O modelo de Elman é apresentado na Figura 4.4.

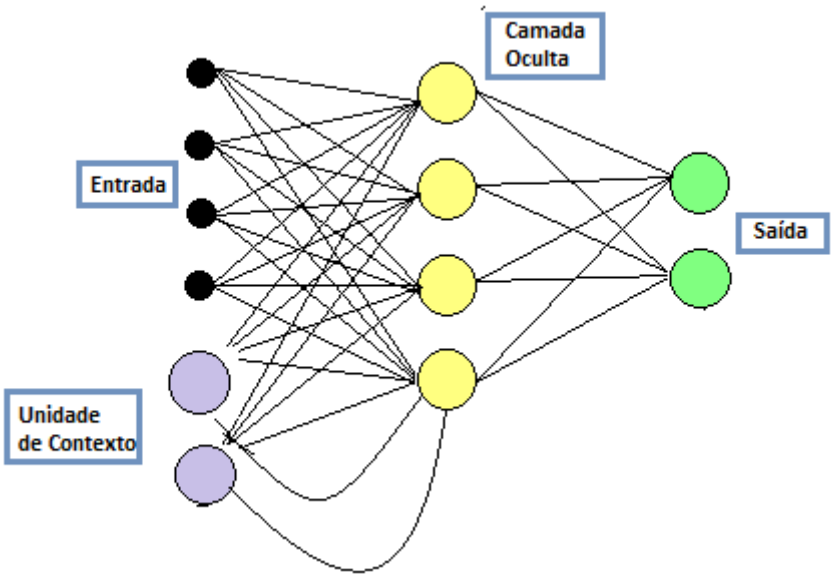

Figura 4.4: Rede de Elman

5 Backpropagation Through Time

O algoritmo do Backpropagation apresentado no Capítulo 3, funciona para redes estáticas, tem uma implementação relativamente fácil e tem sido aplicado a uma infinidade de problemas, desde que foi apresentado à comunidade científica. Ainda assim, esse algoritmo é aplicado apenas em casos de mapeamento estáticos, ou seja, nos quais uma rede feedforward é dada por um conjunto de vetores de dados de entrada.

Entretando, tendo em vista que o objetivo da inteligência artificial é promover algoritmos que imitem o funcionamento do cérebro humano e esse, por sua vez, não funciona a partir de um mapeamento estático de entradas e saídas, houve a necessidade de se criar modelos e algoritmos de treinamento mais eficientes. Como mencionado anteriormente, o cérebro humano é uma máquina de aprendizado altamente potente, multidimensional e não linear. Além disso, compõe um sistema dinâmico e, por isso, para desenvolver uma máquina que funciona da mesma maneira que o cérebro é preciso criar sistemas capazes de recriar esse processamento.

É por essa razão, que algoritmos para sistemas dinâmicos e redes recorrentes, que incluem conexões de feedback e atrasos temporais, são estudados.

O objetivo desse capítulo é apresentar os principais conceitos de um algoritmo de aprendizado supervisionado para treinamento de redes recorrentes, derivado do BP estático, o Backpropagation Trough Time (BPTT).

5.1 Conceitos iniciais

O processo de construção de um algoritmo de aprendizado supervisionado para redes recorrentes é bastante similar ao caso das redes feedforward. Uma função para o cálculo do erro é definida, bem como o gradiente da função é derivado em relação aos pesos da rede. Entretanto, existe uma diferença substancial. A entrada e a saída de uma rede feedforward são vetores estáticos, já as de uma rede recorrente são sequências temporais.

Essa diferença implica ainda no fato de que pequenas alterações em um nodo da

rede podem gerar mudanças em vários nós da camada seguinte ou, ainda, de uma mesma camada, o que torna o cálculo do erro para uma rede recorrente muito mais complexo do que para uma rede estática.

Além do cálculo do erro nessas redes se tornar mais complexo, a regra para calcular a descida do gradiente passa a ser diferente também. No modelo para redes recorrentes existem duas maneiras de se calcular com exatidão o gradiente da saída da rede. De acordo com DOYA (2001), esses dois métodos são chamados de Forward e Backward.

O método Forward estima os efeitos provocados na trajetória de uma rede por pequenas alterações nos pesos através de um sistema de equações dinâmicas. Esse método é útil em aprendizados on-line. Já o método Backward estima a causa dos erros retro-propagados na saída do problema ao longo do tempo. Esse método requer o cálculo de operações assíncronas relacionadas a evolução da trajetória antes do cálculo do gradiente.

Vale ressaltar que, ambos os métodos podem ser aplicados para modelagem de tempo discreto ou tempo contínuo.

Para construir o algoritmo do Backpropagation Through Time, formulado para modelagem de tempo discreto, são necessárias algumas alterações nos paradigmas do algoritmo estático, de forma a atender as novas necessidades da rede, bem como aos novos parâmetros de cálculo de erro e descida do gradiente.

Para que seja possível estender o BP estático a aplicações dinâmicas, dois fatores devem ser alterados nesse algoritmo. Primeiramente, o parâmetro de resposta da rede deve ser formulado de acordo com a noção temporal introduzida pelas redes recorrentes. Além disso, deve-se estabelecer um novo critério para o erro durante o treinamento. Em segundo lugar, é preciso que haja um desdobramento temporal do modelo de rede recorrente, de maneira a adaptar a lista ordenada de dependências da rede recorrente.

As alterações necessárias para possibilitar a extensão do BP em BPTT são descritas nas próximas seções, bem como a formulação do algoritmo.

5.2 Parâmetros de um sistema recorrente

O BPTT é desenvolvido para modelagem de tempo discreto e, por isso é aplicado em sistemas recorrentes discretos. Assim como o BP, o BPTT possui duas fazes, entretanto, um pouco mais complexas.

A adaptação de parâmetros vem da inclusão necessária de sinais gerados pela rede devido a recorrência do sistema, chamados coeficientes de feedback que, podem ter influência local, ou seja, apenas na amostra em questão, ou global, em todas as amostras da base. Além disso, nesses sistemas, a saída da rede para uma dada unidade de tempo n depende não apenas da entrada da rede, mas também da saída gerada pelo sistema no tempo $n-1$, modificado pelos coeficientes de feedback. Esquematicamente, tem-se a Figura 5.1.

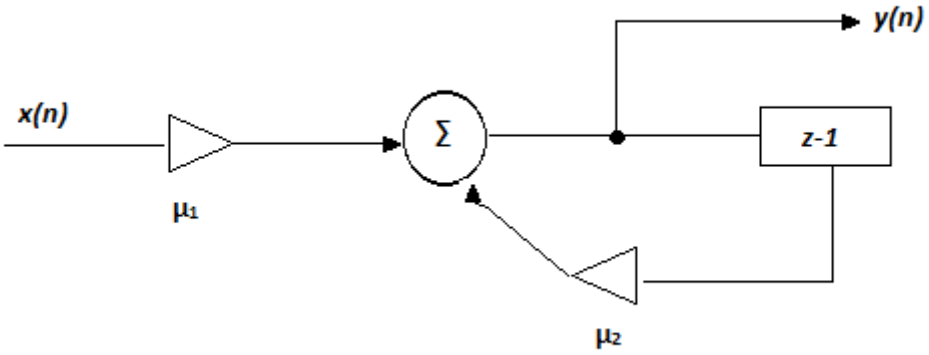

Figura 5.1: Sistema Recorrente Simplificado

Seja $y(n)$ a saída gerada pela rede, $x(n)$ a representação da entrada da rede para um dado tempo n, μ_1 o coeficiente de feedback recorrente (global) e μ_2 o coeficiente de feedback recorrente local. Segundo PRINCIPE *et al.* (2000), a saída de um sistema recorrente discreto é dado pela equação 5.1.

$$y(n) = \mu_1 y(n-1) + \mu_2 x(n) \tag{5.1}$$

Calculando a derivada de $y(n)$ em relação a μ_1 e μ_2, dada pelas equações 5.2 e

5.3, é possível observar, a relação global e local dos coeficientes de feedback.

$$\frac{\partial}{\partial \mu_2} y(n) = x(n) \tag{5.2}$$

$$\frac{\partial}{\partial \mu_1} y(n) = y(n-1)\frac{\partial}{\partial \mu_1}\mu_1 + \mu_1 \frac{\partial}{\partial \mu_1} y(n-1) \tag{5.3}$$

É possível inferir que, devido a natureza recursiva da equação 5.1, a equação 5.3 depende de μ_1, ao passo que o mesmo não ocorre com a equação 5.2. Essa dependência em relação aos coeficientes culmina com a principal diferença entre os casos estáticos e dinâmicos.

A equação 5.3 diz basicamente que o "efeito de qualquer mudança no parâmetro recorrente, μ_1, dura para sempre, enquanto o efeito de qualquer mudança no parâmetro feedforward, μ_2, representa alterações somente na amostra corrente" (PRINCIPE *et al.*, 2000).

O algoritmo estático do BP não apresenta conexões de feedback e, portanto, também não apresenta os parâmetros introduzidos para a implementação temporal do BPTT.

Além das alterações referente a inclusão dos coeficientes de feedback, é necessária, também, uma reformulação do cálculo do erro. Essa reformulação será feita na próxima seção.

5.3 Formulação do erro

Quando se trata da formulação do erro para o caso recorrente do BP, ficam claras as várias semelhanças existentes entre os modelos estático e dinâmico do algoritmo.

De acordo com PRINCIPE *et al.* (2000), o critério para o cálculo do erro em redes neurais dinâmicas mais comumente usado é a trajetória de aprendizado, no qual o custo, ou seja, o valor do erro total associado a rede, é um somatório em função de um tempo

n. Esse somatório parte de um tempo inicial tal que $n = 0$ até um tempo final tal que $n = T$.

Além disso, esse cálculo de erro, segundo DOYA (2001), pode ser aplicado em qualquer algoritmo de modelagem de tempo discreto, além do BPTT, como o Real-Time Recurrent Learning (RTRL), proposto por Robinson e Fallside em 1987 (ROBINSON e FALLSIDE, 1987).

Seja j um neurônio de uma das camadas de uma rede recorrente, $y'_j(n)$ a saída gerada pela rede em um dado tempo n e $y_j(n)$ o valor desejado de saída para o mesmo tempo n. O erro calculado para o neurônio j no tempo n é dado por $e_j(n)$ e é apresentado pela equação 5.4.

$$e_j(n) = y_j(n) - y'_j(n) \tag{5.4}$$

Para um tempo n, o somatório dos erros naquele instante produzidos por todos os neurônios da rede é chamado de erro instantâneo e é formulado de acordo com a equação 5.5.

$$E(n) = \frac{1}{j} \sum_{j} e_j^2(n) \tag{5.5}$$

Finalmente, considerando um intervalo de tempo tal que $0 \leq n \leq T$, temos que o erro total gerado pela rede é definido em função do número de neurônios e das unidades de tempo pertencentes ao intervalo. A equação 5.6 apresenta esse resultado.

$$E_{total} = \sum_{n=0}^{T} E(n) = \sum_{n} \sum_{m} e_m^2(n) \tag{5.6}$$

O objetivo a ser alcançado consiste, assim como na implementação estática do BP, em minimizar o erro gerado pela rede. O cálculo da descida do gradiente, regra delta, para a extensão temporal do BP, faz uso dos resultados obtidos para o cálculo do

erro mostrados nessa seção. A formulação da regra delta para o BPTT é apresentado na próxima seção.

5.4 O algoritmo Backpropagation Through Time

A aplicação do algoritmo do BP para redes recorrentes, foi desenvolvido por Paul J. Werbos e apresentado à comunidade científica em 1990 no trabalho intitulado "Backpropagation through time: what it does and how to do it" (WERBOS, 1990).

Além disso, como visto nas seções anteriores, uma série de parâmetros do modelo estático devem ser alterados, tendo em vista as novas relações de dependência de estados em relação ao tempo. O valor gerado pela rede num dado tempo n, $y(n)$, depende do valor obtido anteriormente, $y(n-1)$.

Segundo HAYKIN (1994), para o treinamento de uma rede recorrente é preciso um algoritmo de aprendizado supervisionado no qual, a resposta atual da rede para cada neurônio da camada de saída é comparado com a resposta desejada para cada instante de tempo pertencente ao intervalo.

O algoritmo temporal do BP é dividido basicamente em duas etapas, como mostrado esquematicamente na Figura 5.1. A primeira, chamada forward, consiste nas duas etapas do BP estático acrescidas de algumas alterações. A segunda, backward, apresenta fases não pertencentes ao modelo estático do algoritmo, mas que se correlacionam. As etapas do BPTT são descritas mais detalhadamente a seguir.

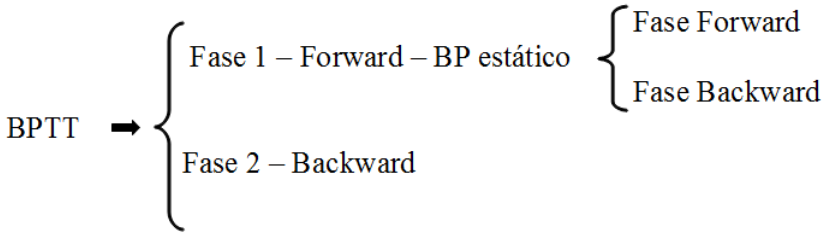

Figura 5.2: Fases do Backpropagation Through Time

A fase Forward correspondente ao cálculo da saída gerada pela rede, para cada neurônio j e para cada unidade de tempo pertencente ao intervalo pré estabelecido. Essa

formulação é definida na equação 5.1.

Para gerar atualização na rede e possibilitar a correção dos pesos foi desenvolvido um método de expansão da rede em função de um dado intervalo de tempo n. Essa etapa correponde a segunda etapa do BP estático. Essa fase consiste basicamente em uma técnica de transformação da rede em uma rede equivalente feedforward para cômputo da correção dos pesos em função do tempo.

Para mapear todas essas relações, considerando ainda a evolução do sistema para um intervalo de tempo limitado, existe um procedimento, de acordo com HAYKIN (1994), PRINCIPE *et al.* (2000) e DOYA (2001), chamado de desdobramento em função do tempo.

Esse procedimento consiste na expansão da rede recorrente em uma rede feedforward a partir do alargamento da primeira e posterior repetição dos coeficientes de feedback e pesos da rede, como pode ser observado na Figura 5.3.

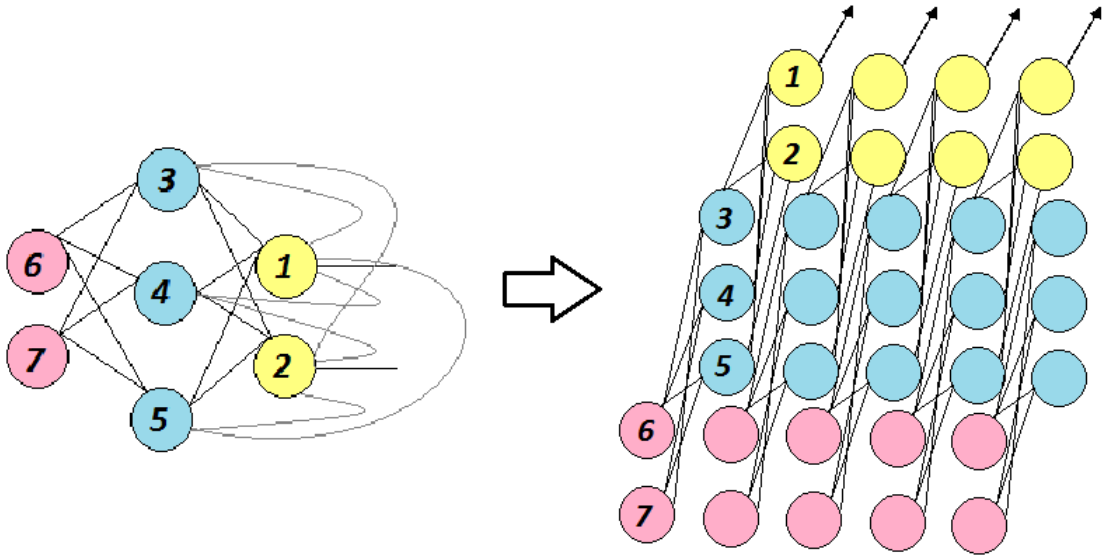

Figura 5.3: Desdobramento de uma rede recorrente em uma rede feedforward

Para o caso discreto, de acordo com DOYA (2001), esse método é Backward e estima a causa do erro de saída da rede através de um processo contrário a evolução do tempo. Ele procede transformando a evolução dos vários estados da rede em uma rede feedforward de múltiplas camadas e, posteriormente, é aplicado o algoritmo do Backpropagation estático.

Outra definição importante para esse método é apresentado em HAYKIN (1994),

e determina que a estratégia para o desdobramento da rede em relação ao tempo, consiste na tentativa de remoção de todos os atrasos temporais presentes na rede através da expansão da mesma em uma rede estática equivalente, porém, significativamente maior. A aplicação do BP estático ocorre de maneira subsequente para o cômputo do gradiente de erro instantâneo.

A cada instante de tempo é criado um novo estado na rede desdobrada, ou seja, para um intervalo contendo n unidades de tempo, a rede é replicada n vezes e, com isso, é possível obter a rede equivalente, porém feedforward. Segundo GRAU *et al.* (2013), cada conexão replicada da rede é acompanhada de valores de pesos dados por w_{ji} em todo o intervalo de tempo. Ao final desse intervalo limitado e estabelecido previamente, uma rede feedforward é gerada, com os pesos já atualizados.

A segunda etapa do BPTT, a backward, consiste no reestabelecimento da rede recorrente, pelo processo inverso ao de desdobramento da rede. Essa etapa é chamada de folding process. Nesse estágio, a rede percorre o caminho inverso do desdobramento obtendo a rede recorrente original, porém os pesos da rede são agora atualizados através de uma agregação dos pesos equivalentes na rede feedforward para cada passo do desdobramento.

5.4.1 Atualização dos pesos da rede

A fórmula apresentada pela equação 5.7 define o valor do peso atualizado para a próxima unidade de tempo, $(n+1)$, em função do valor do peso obtido para a unidade de tempo anterior, n. De acordo com HAYKIN (1994), para se chegar a equação final do cômputo do valor de atualização dos pesos dado pela equação 5.7 é preciso seguir o seguinte caminho e considerar algumas análises da rede, como mostrado adiante.

$$
\begin{aligned}
& w_{ij}(n+1) = w_{ij}(n) + \eta\delta_j(n)x_i(n) \\
& \delta_j(n) \begin{cases} e_j(n)\varphi'(v_j(n)) \\ \varphi'(v_j(n)) \sum\limits_{r\in A} \Delta_r(n) w_{rj} \end{cases}
\end{aligned}
\tag{5.7}
$$

Inicialmente, deve-se partir da equação para o cálculo do erro total da rede,

equação 5.6. O erro total do gradiente (E_{total}) pode ser visto como um somatório dos erros instantâneos. A derivada parcial do erro total em relação ao vetor de pesos w_{ji} para dois neurônios j e i tais que a saída do neurônio i é conectada ao neurônio j é dada pela equação 5.8.

$$\frac{\partial E_{total}}{\partial w_{ji}} = \sum_{n} \frac{\partial E_{total}}{\partial v_j(n)} \frac{\partial v_j(n)}{\partial w_{ji}} \tag{5.8}$$

O termo $v_j(n)$ denota o campo local induzido na rede, é definido pela equação 5.9, sendo b_j o valor do bias para o neurônio j e $x_i(n)$ o sinal de entrada da rede para o i-ésimo neurônio no tempo n.

$$v_j(n) = \sum_{i} w_{ji}^T x_i(n) + b_j \tag{5.9}$$

Assim, é possível estabelecer uma fórmula recursiva para o ajuste dos pesos baseados na equação 5.8, na qual α é a taxa de aprendizado. Tem-se que a fórmula recursiva para ajuste dos pesos é dada pela equação 5.10.

$$w_{ji}(n+1) = w_{ji}(n) + \eta \frac{\partial E_{total}}{\partial v_j(n)} \frac{\partial v_j(n)}{\partial w_{ji}} \tag{5.10}$$

Da equação 5.8 e 5.9, é possível inferir que para qualquer neurônio j, a derivada parcial do campo local induzido em relação ao vetor de pesos é daada pela equação 5.11

$$\frac{\partial v_j(n)}{\partial w_{ji}} = x_i(n) \tag{5.11}$$

A partir disso, é possível determinar o gradiente local do neurônio j para um tempo n, denotado por $\delta_j(n)$, como mostrado na equação 5.12.

$$\delta_j(n) = -\frac{\partial E_{total}}{\partial v_j(n)} \tag{5.12}$$

Substituindo os valores encontrados nas equações 5.11 e 5.12, na fórmula recursiva para o ajuste dos pesos dado pela equação 5.10, tem-se uma nova equação dada pela fórmula 5.13.

$$w_{ji}(n+1) = w_{ji}(n) + \eta\delta_j(n)x_i(n) \tag{5.13}$$

Ainda de acordo com HAYKIN (1994), a forma explícita para o cálculo do gradiente local só pode ser feito tendo com base a localização exata do neurônio j na rede. Nesse caso, há duas abordagens possíveis. O neurônio j pode pertencer a uma camada oculta da rede ou a camada de saída da mesma.

O caso mais simples e, portanto, o primeiro abordado, determina que o neurônio j pertence a camada de saída da rede. Sendo assim, a fórmula para o cálculo exato do gradiente local é dado pela equação 5.14, na qual, $\varphi'(v_j(n))$ é a derivada da função de ativação em relação ao campo local induzido.

$$\delta_j(n) = \frac{\partial E_{total}}{\partial v_j(n)} = -\frac{\partial E(n)}{\partial v_j(n)} = e_j(n)\varphi'(v_j(n)) \tag{5.14}$$

O segundo caso determina que o neurônio j pertence a uma camada oculta da rede. Nesse caso, o cálculo do gradiente local é consideravelmente mais complexo e a derivação envolve uma série de passos que, por simplicidade, serão omitidos nesse trabalho.

Seja A, "um conjunto formado por todos os neurônios cujas entradas são alimentadas pelo neurônio j de maneira forward" (HAYKIN, 1994) e, $\Delta_r(n)$ é o vetor de gradientes locais para um dado neurônio r. A fórmula para o cálculo do gradiente de um neurônio j quando esse pertence a uma camada oculta da rede é apresentada pela equação 5.15.

$$\delta_j(n) = \varphi'(v_j(n)) \sum_{r \in A} \Delta_r(n) w_{rj} \tag{5.15}$$

A partir das equações 5.13, 5.14 e 5.15, é formulada a forma recursiva de atualização dos pesos da rede, dado conforme a equação 5.7.

Tendo percorrido esses passos, a implementação do BPTT está formulada.

5.4.2 Passos do Backpropagation Through Time

Tendo sido determidadas todas as fases do BPTT, é possível obter os passos do algoritmo de maneira sucinta. Como observado em HAYKIN (1994), os passos do algoritmo temporal do BP são apresentados abaixo.

- Passos do Backpropagation Through Time

    1. Propagar o sinal de entrada através da rede na direção forward, camada por camada. Determinar o sinal de erro $e_j(n)$ para o neurônio j da camada de saída subtraindo o valor obtido do correspondente valor desejado.
    2. Para um neurônio j pertencente a camada de saída da rede, compute o gradiente (Equação 5.14) e atualize os pesos referentes a esse neurônio (Equação 5.7).
    3. Para cada neurônio pertencente a uma camada intermediária da rede, compute o seu gradiente (Equação 5.15) e atualiza os pesos referentes a esse neurônio (Equação 5.7).

Formulado o algoritmo de aprendizado supervisionado para o treinamento de redes recorrentes, a seguir, são feitas algumas aplicações e análise dos resultados obtidos.

6 Aplicações e Análise de Resultados

Esse capítulo trata das diferentes aplicações possíveis tanto para redes estáticas que envolvem noções temporais (TLFN e TDNN) quanto para redes recorrentes estudadas nesse trabalho (Jordan e Elman).

Para a execução dessas aplicações, será usado o software Neurosolutions. Ele foi desenvolvido por uma série de autores, entre eles, José Principe, Curt Lefebvre e Neil Euliano, também autores de uma das obras mais importantes para o desenvolvimento desse trabalho, o livro intitulado "Neural and Adaptive Systems: Fundamentals Through Simulations". A versão usada é a 4.23 de 2003 que foi fornecida como material auxiliar do livro, e é de responsabilidade da NeuroDimension.

Com as aplicações propostas pelo software, é possível fazer uma comparação da eficiência das redes TDNN e TLFN aplicadas a um mesmo problema, utilizando o algoritmo estático do BP, por exemplo.

Além disso, aplicações com redes recorrentes também são propostas, como por exemplo, treinamento do coeficiente de feedback para ajuste de uma rede de Elman, utilizando o BPTT.

Inicialmente, estabelecendo uma ordem para execução das aplicações, serão analisadas as estruturas estáticas e as comparações pertinentes e, posteriormente, as redes recorrentes.

6.1 Aplicações com TLFN e TDNN

Essa seção consiste na execução e comparação de resultados obtidos para 3 problemas diferentes executados pelas redes TLFN e TDNN focadas. Cada um dos três problemas são executados por cada uma das topologias e os problemas, bem como os resultados obtidos e a análise desses resultados são apresentados adiante.

6.1.1 Classificação de Fonemas

Pela definição que consta no Dicionário da língua portuguesa, fonena é "a menor unidade sonora do sistema fonológico da língua. Cada fonema tem a função de estabelecer uma diferença de significado entre uma palavra e outra."

O problema da classificação de fonemas, de acordo com PRINCIPE *et al.* (2000), é que ele se faz em função do tempo. Muito embora, muita das vezes os fonemas representam letras isoladas do alfabeto, os fonemas são produzidos por jatos de ar que saem dos pulmões e fazem as cordas vocais vibrarem. Essas últimas por sua vez, são moldadas pelas ressonâncias do trato vocal.

Quando coletados os sons produzidos pela voz por um microfone, eles são transformados em tensões elétricas que podem ser observadas através de um osciloscópio.

O objetivo desse problema é usar as redes TLFN focadas e TDNN focadas para reconhecimento das ondas formadas pelos fonemas e diferenciá-los. Para isso, é necessário realizar processamento temporal.

Para o treinamento dessas redes, ainda de acordo com PRINCIPE *et al.* (2000), tem-se que quando o som a ser classificado aparece na série temporal, é definido que a resposta desejada para a rede tem valor 1. Quando o fonema desejado está ausente, é atribuída como saída desejada o valor 0. Os diferentes sinais desejados de saída da rede devem ser criados em função do tempo para o treinamento das redes. A resposta desejada para a saída da rede é também uma função temporal.

Para ambas as topologias, é usado o algoritmo do BP estático para o treinamento. Além disso, para as redes TDNN focadas, o termo Tap significa alteração da noção de temporalidade da rede e, o termo $TapDelay$ o número de amostras do sinal de entrada atrasadas.

TLFN Focadas

Para o treinamento da topologia TLFN focada, o único parâmetro que se permite alguma alteração é o número de épocas, no entanto, a alteração desse parâmetro não gerou diferença para o cálculo do erro médio da rede.

Os resultados obtidos para essa rede são apresentados na Tabela 6.1. Grafica-

mente, a comparação entre a saída deseja e a obtida após o treinamento são mostradas na Figura 6.1.

Reconhecimento de Padrões - TLFN focada - Erro médio	
Número de elementos de Processamento	11
Erro médio	0,013781

Tabela 6.1: Tabela de valores obtidos para a rede TLFN focada

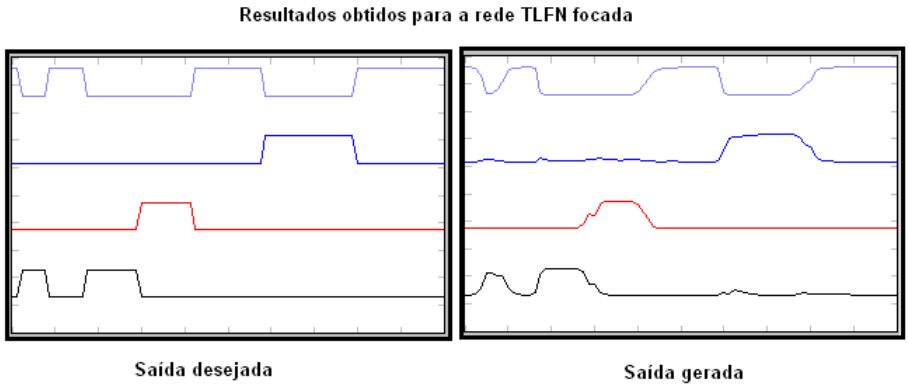

Figura 6.1: Resultados obtidos para a rede TLFN focada

TDNN Focadas

Os resultados obtidos para a rede TDNN focada são mostrados abaixo. A Tabela 6.1 apresenta os valores médios do erro encontrado para cada diferente execução da aplicação. A Figura 6.2 apresenta os diferentes padrões encontrados, de acordo com as variações dos parâmetros.

A partir da análise dos resultados, é nítida a diferença obtida quando é retirado a noção temporal da rede. Isso acontece quando o parâmetro Tap é alterado para 1. É possível observar a influência significativa no cálculo do erro que, agora, passa a ser 10 vezes maior, se comparado ao erro obtido com os valores padrões da rede, para $Tap = 3$.

Análise dos resultados

Embora as redes TDNN sejam uma especiação das redes TLFN, é possível observar a diferença entre os parâmetros da rede. A flexibilidade obtida durante a alteração dos

Reconhecimento de Padrões - TDNN Focada - Erro Médio		
	Tap Delay	
Tap	1	2
1	0,030943	-
3	0,004973	0,001823
5	0,001176	0,003319

Tabela 6.2: Tabela de valores obtidos para a rede TDNN focada

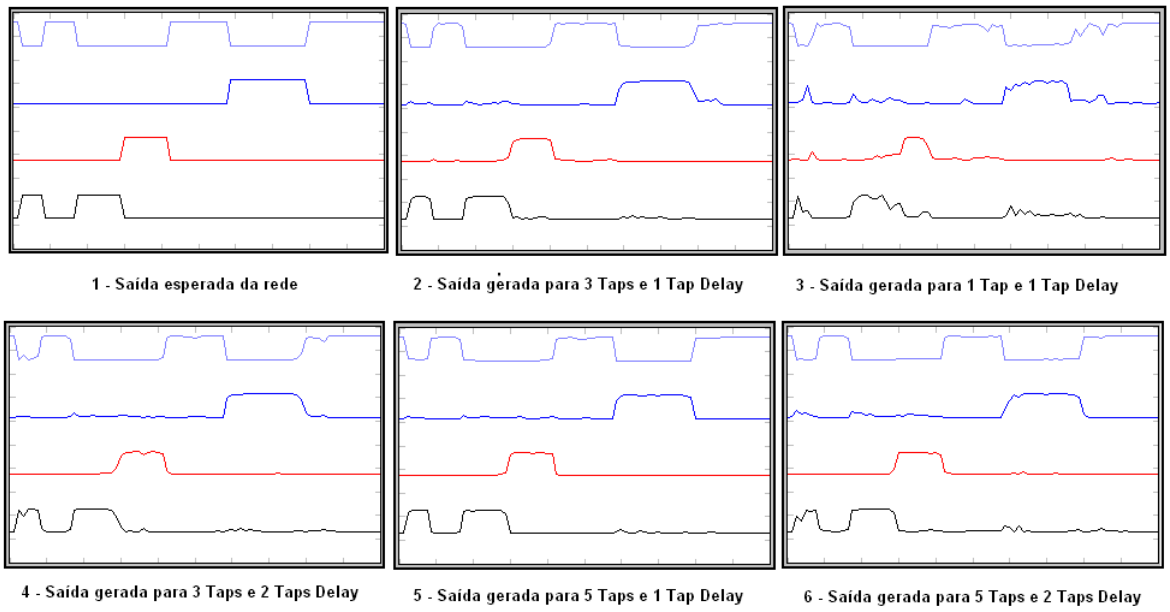

Figura 6.2: Resultados obtidos para uma rede TDNN focada

parâmetros da rede TDNN fazem diferença no resultado final e na minimização do erro. Em todos os casos, exceto no qual a rede TDNN tem sua noção temporal removida, a rede TLFN tem um desempenho inferior ao da rede TDNN. Assim, concluí-se que a inclusão da variável temporal é benéfica para a solução do problema e minimização do erro no sistema.

6.1.2 Predição não linear de série caótica

Predição é o ato de dizer antecipadamente resultados futuros baseados em dados subjetivos, regras ou conjecturas. No contexto dessa aplicação, predição, de acordo com PRINCIPE *et al.* (2000) é uma forma de identificação de sistemas nos quais tanto a entrada de dados quanto a saída tem origem na mesma fonte de dados, porém com diferentes

intervalos de tempo, a entrada tende a ser atrasada por uma amostra.

A série temporal determinada para essa aplicação é produzida por um sistema dinâmico não linear e é chamada de Sistema Mackey-Glass (GLASS e MACKEY, 1988), é mostrada na Figura 6.3. O objetivo do sistema é tentar predizer os valores dessa série.

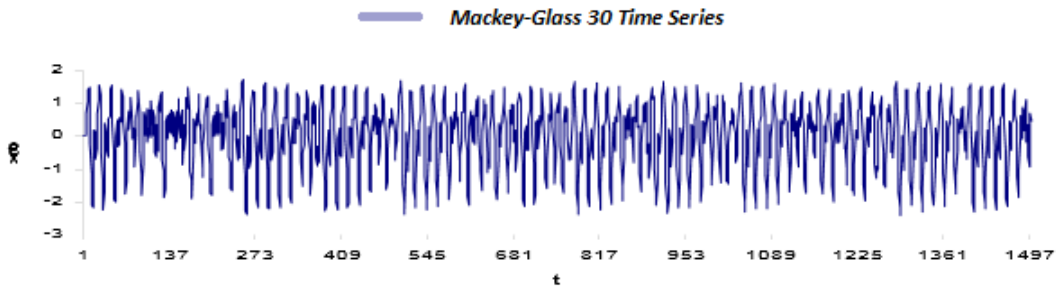

Figura 6.3: Mackey-Glass 30 Time series

O sistema proposto por Mackey-Glass, é uma série temporal caótica, tendo em vista que os sinais são sempre diferentes e, por isso, a longo prazo, é impossível de ser adivinhada. A importância desssa aplicação consiste no fato de que na maioria dos problemas do mundo real, os sinais de entrada tendem a ser caóticos e assim, vê-se que é possível modelá-los.

Análise dos resultados

A análise dos resultados dessa etapa mostra que, de acordo com a Tabela 6.3 e a Tabela 6.4, em geral, as redes TDNN focadas apresentam um melhor desempenho quando comparadas com as redes TLFN focadas.

Para o caso da rede TLFN focada, mostrada na Figura 6.4, a alteração na quantidade de elementos de processamento, quando saem do padrão pré estabelecido pela rede, $PEs = 5$, gera alteração nos resultados obtidos para o erro médio. Tanto a diminuição para $PEs = 2$ quanto o aumento para $PEs = 15$ fornecem mudanças significativas sobres os valores encontrados.

Outra observação é feita, quando se consideram os gráficos obtidos tanto para as redes TLFN quando para as redes TDNN. Embora os valores para os erros da rede TLFN focada sejam maiores do que os da rede TDNN focada (Figura 6.5), a redes TLFN

se adequam melhor a saída desejada do que as redes TDNN que, claramente, apresentam distorções em relação a saída desejada nos picos da série.

Predição Não Linear de Série Temporal Caótica - TLFN Focada - Erro Médio		
Número de Elementos de Processamento		
2	5	15
0,013434	0,006422	0,019740

Tabela 6.3: Tabela de valores obtidos para a rede TLFN focada

Predição Não Linear de Série Temporal Caótica - TDNN Focada - Erro Médio		
	Tap Delay	
Tap	1	2
1	0,013580	-
3	0,007185	-
5	0,005949	0,008402

Tabela 6.4: Tabela de valores obtidos para a rede TDNN focada

6.2 Redes de Jordan e Elman

Essa seção consiste em avaliar resultados obtidos de aplicações dos modelos recorrentes de Jordan e Elman com os algoritmos do BP estático e do BPTT. Inicialmente, as redes de Jordan e Elman são testadas com a implementação estática do Backpropagation. Como aprensentam topologias bastante similares, a aplicação seguinte utiliza apenas a rede de Elman para treinamento do parâmetro de feedback da rede, usando o BPTT.

6.2.1 Rede de Elman e Jordan aplicadas ao BP estático

Essa aplicação consiste num conjunto de dados com dependência temporal. Cada item pertencente ao conjunto de dados, tem-se que esse item permanece em foco por uma quantidade predeterminada de unidades de tempo. Por exemplo, "Supõe-se que 9 itens sejam codificados como 0,1, 0,2 ... 0,9 e que esses itens aparecem de forma aleatória, no entanto, a instância $0,1$ aparece por uma unidade de tempo, a instância 0,2 aparece durante 2 unidades de tempo e assim sucessivamente" (PRINCIPE *et al.*, 2000).

Na prática, Elman associou esse problema gerado com valores aleatórios a de sons de consoantes seguidas por um número prederterminado de sons de vogais.

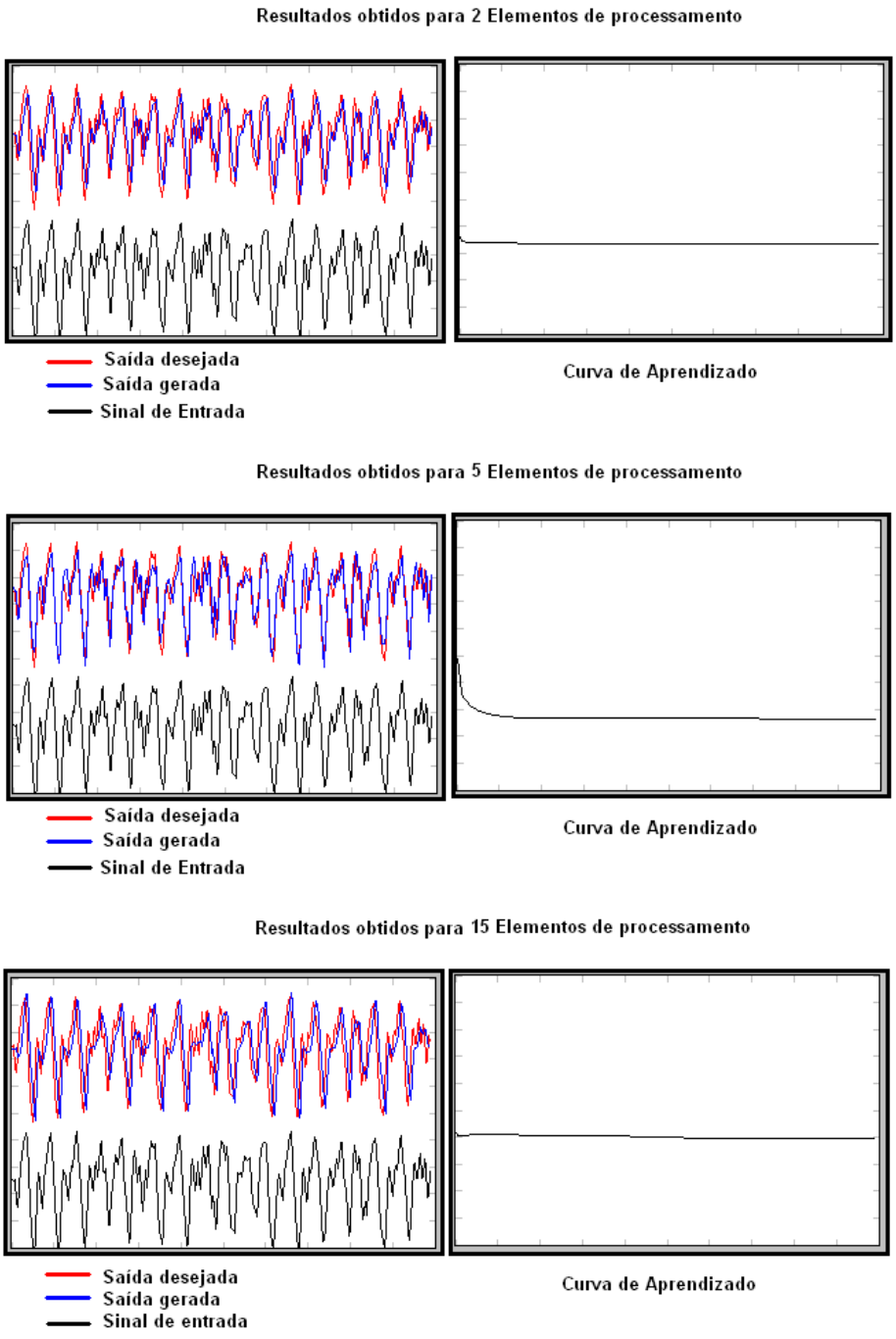

Figura 6.4: Resultados obtidos para a TLFN focada

A resposta desejada para esse sistema é o mesmo sinal obtida para o sinal de entrada, exceto que saída deve ser avançada em uma unidade de tempo.

Como os modelos aqui testados são de redes recorrentes, a utilização do algoritmo estático do BP produz apenas resultados aproximados, tendo em vista que esse não é o melhor algoritmo para treinamento desse tipo de rede. Por isso, os parâmetros de feedback devem ser mantidos fixos.

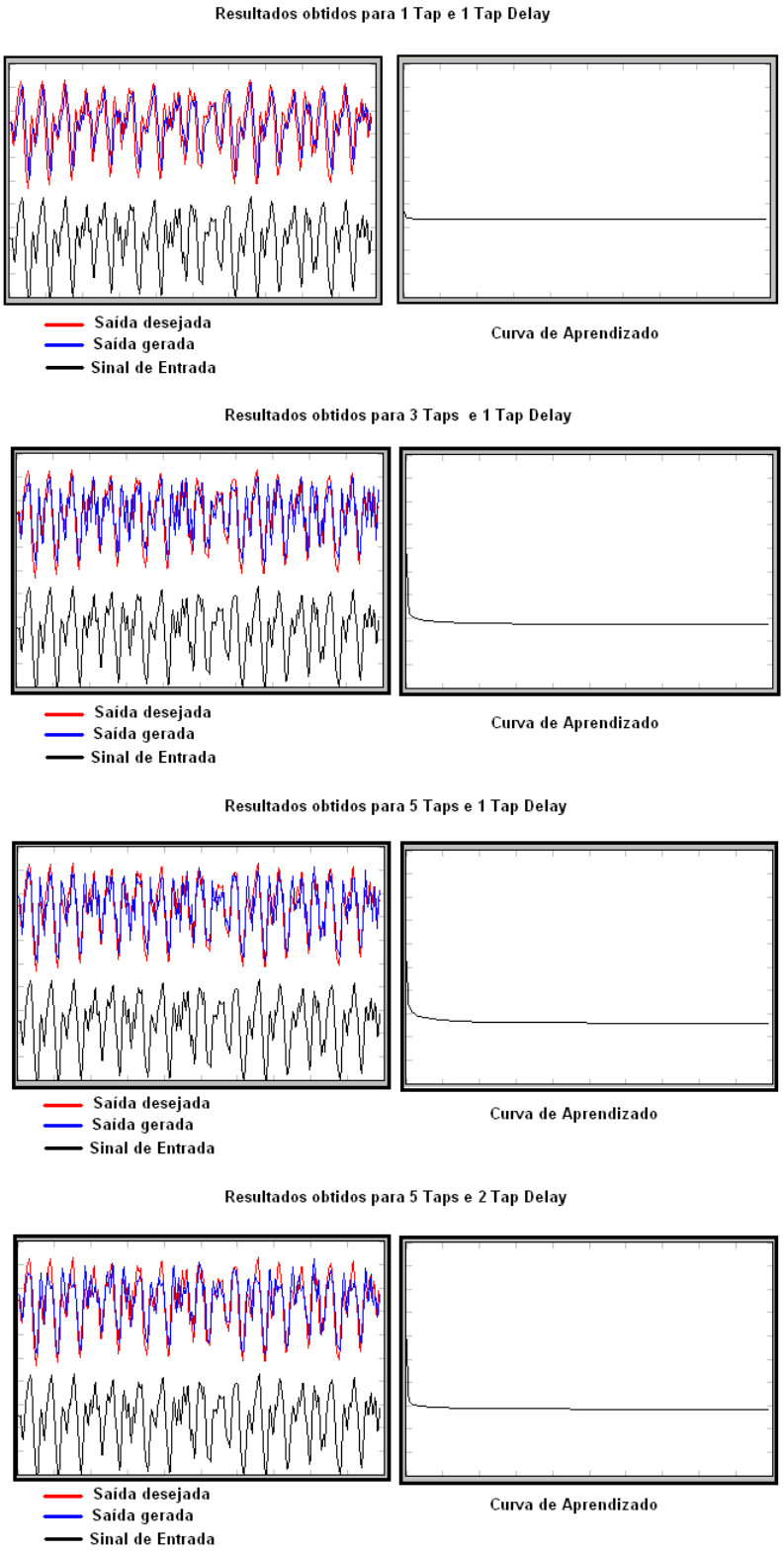

Figura 6.5: Resultados obtidos para a TDNN focada

Análise dos Resultados

Como é de se esperar, os resultados obtidos para as redes de Elman (Figura 6.7) e Jordan (Figura 6.6) treinadas com o BP estático, não apresentam valores eficientes relacionados

ao erro médio.

Essa aplicação mostra que é possível usar o BP sem noção temporal para treinamento dessas topologias simples recorrentes, mas que devido a razões óbvias, ou seja, a não adequação do algoritmo de aprendizado ao tipo de rede, os resultados obtidos não são os desejados.

Além disso, na Tabela 6.5 e Tabela 6.6 é mostrada que a alteração em relação aos elementos de processamento das camada de contexto e camada oculta da rede não apresentam redução significativa dos valores do erro médio.

A alteração no número de épocas não altera os resultados obtidos, o que sugere, uma estabilidade relativa das redes .

Rede de Jordan e Backpropagation Estático - Erro Médio			
Número de Elementos de Processamento	Camada Oculta		
Camada de Contexto	2	5	10
2	-	-	0,023997
5	-	0,022831	-
10	0,023938	-	0,024290

Tabela 6.5: Tabela de valores obtidos para a rede de Jordan

Rede de Elman e Backpropagation Estático - Erro Médio			
Número de Elementos de Processamento	Camada Oculta		
Camada de Contexto	2	5	10
2	-	-	0,024739
5	-	0,024012	-
10	-	-	0,024421

Tabela 6.6: Tabela de valores obtidos para a rede de Elman

6.2.2 Treinamento do parâmetro de feedback com o BPTT

Essa aplicação consiste na adaptação dos parâmetros do algoritmo do BPTT através de uma rede de Elman. Embora seja uma aplicação simples, é possível inferir uma série de resultados sobre ela.

Inicialmente, o tamanho da trajetória é definido com 15 amostras, isso significa que “15 amostras são introduzidas como sinais de entrada e cada elemento de processamento da rede guarda 15 ativações da rede, a saída da rede vai ser comparada a 15

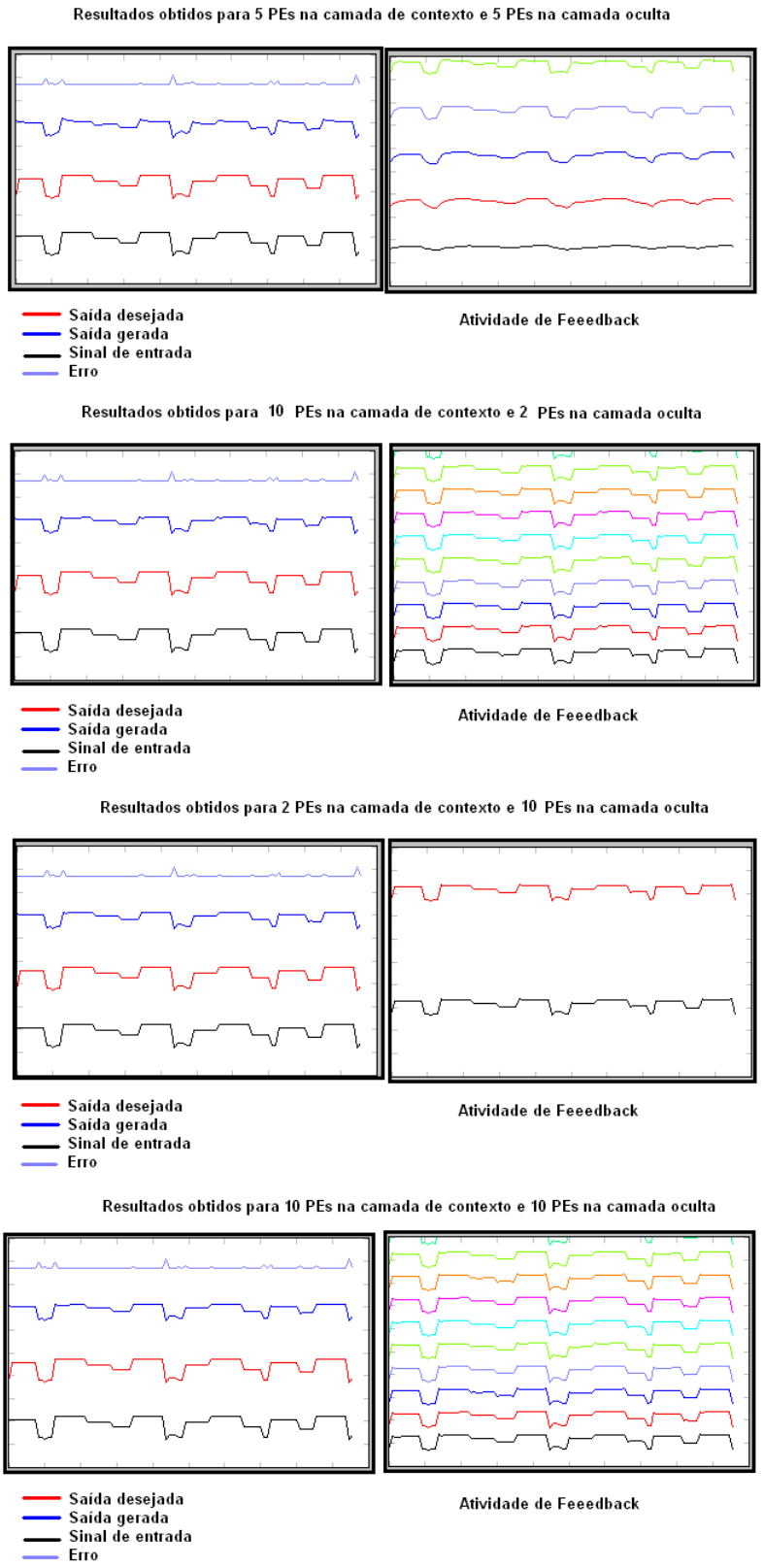

Figura 6.6: Resultados obtidos para rede de Jordan

amostras da resposta desejada e 15 vetores de erro são criados. Ao final da trajetória, cada elemento de processamento da rede contém 15 amostras de ativação e vetores de

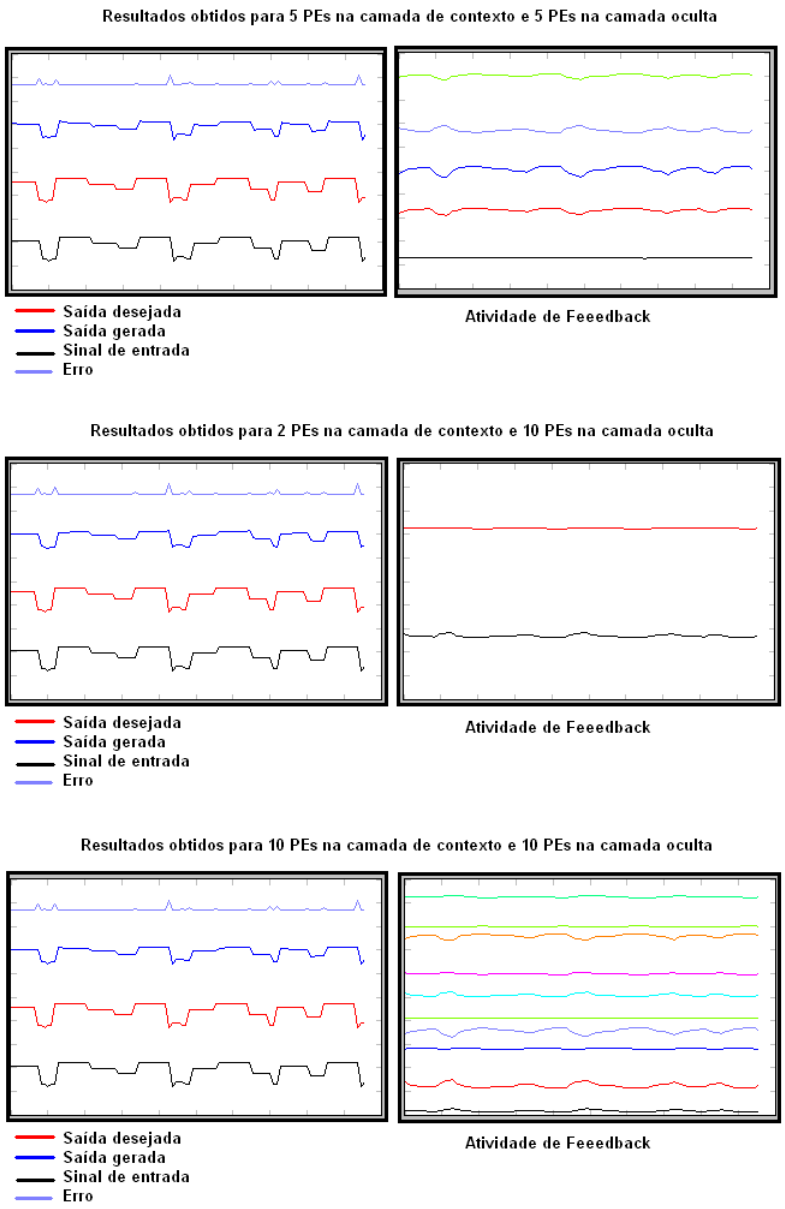

Figura 6.7: Resultados obtidos para rede de Elman

erros e, então, esses valores podem ser usados para atualização dos pesos" (PRINCIPE *et al.*, 2000).

Análise dos resultados

Essa aplicação promove a atualização dos parâmetros da rede utilizando o BPTT. Analisando o resultados é possível concluir que as alterações nos parâmetros de pesos da camada de contexto e da camada oculta não representam diferenças significativas em relação ao erro médio, essa relação é apresentada pela Tabela 6.7.

No entanto, é possível observar que os valores das constantes temporais são alterados, além dos acréscimo das mesmas, como mostrado na Figura 6.8.

Conclui-se portanto, que a alteração das constante temporais não faz sentido, uma vez que não melhora o erro de forma significativa, porém aumenta o tempo de execução da aplicação.

Rede de Elman, BPTT e Atualização dos Parâmetros da rede - Erro Médio		
Número de Elementos de Processamento	Camada Oculta	
Camada de Contexto	5	15
5	0,018729	-
15	-	0,017703

Tabela 6.7: Tabela de valores obtidos para a atualização de parâmetros na rede

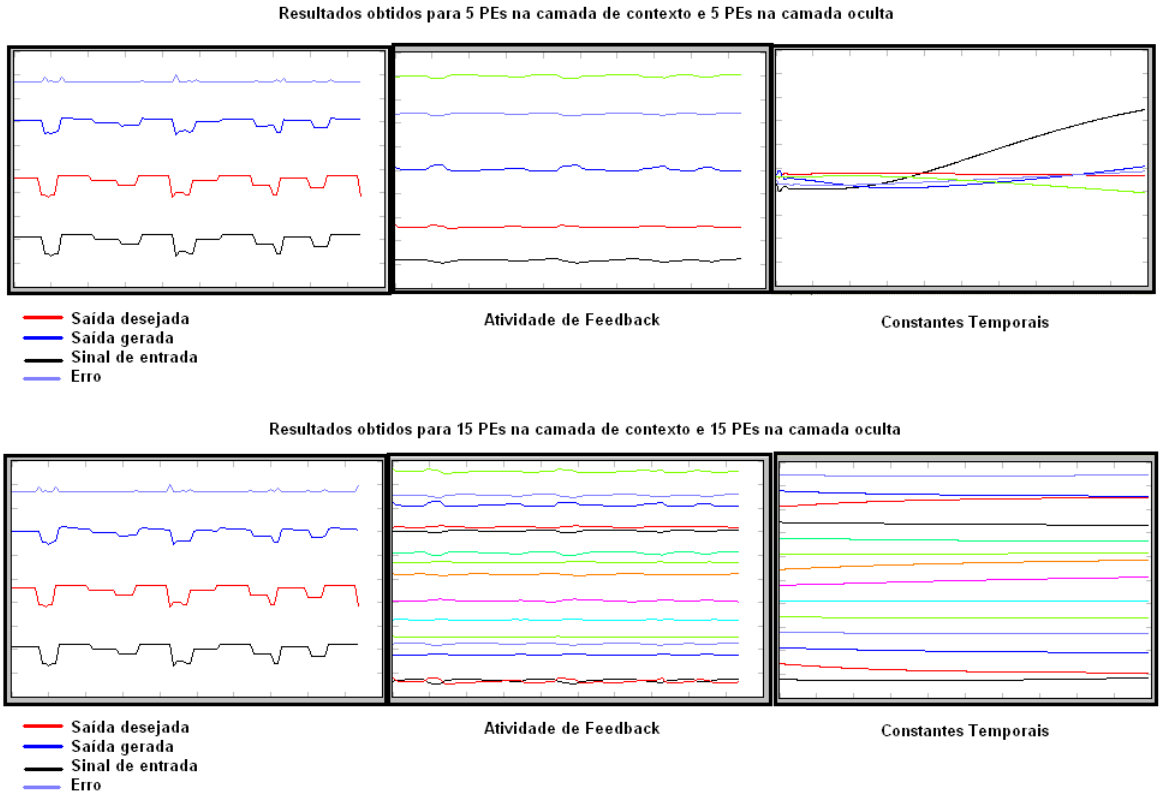

Figura 6.8: Resultados obtidos para rede a atualização de parâmetros na rede

7 Considerações Finais

Esse trabalho formulou um estudo sobre 4 topologias de redes às quais pode-se aplicar noções temporais. Dois algoritmos de aprendizado para essas topologias e aplicações dessas topoligias em problemas com intuito de analisar o comportamento dos sistemas em diversas situações, com o objetivo de analisar melhor o comportante de estruturas que contém informação temporal interna.

Algumas considerações importantes podem ser feitas. Primeiramente, as topologias TLFN e TDNN introduziram o conceito de memória curta para tratamento de problemas temporais, sem alterar seu padrão estático de processamento. Isso culminou, com uma maior facilidade de treinamento, tendo em vista que esse pode ser feito com o algoritmo do Backpropagation estático.

As topologias TDNN apresentam ainda a introdução do conceito de atraso temporal (time delay) com o qual se pode aumentar ou até mesmo retirar a temporalidade da rede, importante para avaliação dos resultados de algumas aplicações proposta nesse trabalho. No entanto, mantém os elementos de processamento da rede fixados, não permitindo qualquer alteração.

A topologia TLFN, por sua vez, embora permita a alteração do número de PEs da rede, não conta com a inclusão do atraso temporal. Essa topologia, como observado nas aplicações, apresenta uma menor eficiência de resultados em relação as TDNN, como também foi constatado no capítulo de aplicações.

Outras conclusões são obtidas das topologias recorrentes estudadas. As redes de Jordan foram precursoras na inclusão das estruturas chamada Unidades de Contexto. A seguir, Elman cria uma topologia semelhante a de Jordan que faz uso dessa mesma estrutura. No entanto, são topologias recorrentes, e a aplicação do BP estático, embora possível, não apresenta qualquer resultado significativo durante o processo. Criou-se então a necessidade de um algoritmo específico para tratar de redes recorrentes, o Backpropagation Through Time.

A utilização dessas estruturas recorrentes requer uma implementação considera-

velmente mais complexa que deve levar em conta não só o novo modelo de algoritmo de treinamento, mas também a maior complexidade das redes recorrentes. Mesmo embora, os modelos de Elman e Jordan sejam modelos mais simples de redes recorrentes, ainda assim requer uma maior atenção.

Por fim, considera-se que os objetivos foram alcançados, esse trabalho permite ponderar sobre qual topologia e respectivo algoritmo de treinamento usar, de acordo com a necessidade do problema.

Como trabalhos futuros, sugere-se o aprofundamento do estudo, tendo em vista a escassez de trabalhos em português apronfudados nessa área. Ainda em relação ao aprofundamento do estudo, outras estruturas recorrentes podem ser estudadas e comparadas as topologias estáticas temporais.

Além disso, sugere-se um estudo aprimorado do algoritmo BPTT em comparação a outros algoritmos para treinamento de redes recorrentes. Devido a existência de poucos trabalhos relacionados é possível ainda o aprofundamento também teórico dos estudos relacionados a esse algoritmo.

Ainda assim, é possível propor uma análise aprofundada de aplicações exaltanto as vantagens e desvantagens das diferentes topologias e algoritmos de aprendizado estudadas, nas mais diversas situações.

Referências Bibliográficas

BITTENCOURT, G. **Breve História da Inteligência Artificial.** Dissertação (Mestrado) — Universidade Federal de Santa Catarina, 1996.

BRAGA, A. de P.; LUDERMIR, T. B.; CARVALHO, A. C. P. de L. F. **Redes Neurais Artificiais: Teoria e Aplicações**, 2000.

BRYSON, A. E.; HO, Y.-C. **Applied Optimal Control: Optimization, Estimation and Control**, 1975.

DOYA, K. Recurrent networks: Learning algorithms. In: **The Handbook of Brain Theory and Neural Networks**, 2001.

ELMAN, J. L. Finding structure in time. **Cognitive Science**, 1990.

FETZ, E. E.; SHAPE, L. E. Recurrent networks: Neurophysiological modeling. **The Handbook of Brain Theory and Neural Networks**, 2002.

GLASS, L.; MACKEY, M. C. **From Clocks to Chaos: The Rhythms of Life**, 1988.

GRAU, I.; NáPOLES, G.; BONET, I.; GARCíA1, M. M. Backpropagation through time algorithm for training recurrent neural networks using variable length instances. **Computación y Sistemas**, 2013.

GUYTON, A. C.; HALL, J. E. **Tratado de Fisiologia Médica**, 2012.

HAYKIN, S. **Neural networks: A comprehensive foundation**, 1994.

HAYKIN, S. **Neural Networks and Learning Machines**, 2009.

JAIN, L. C. Recurrent neural networks: Design and applications. In: **The Handbook of Brain Theory and Neural Networks**, 2001.

JORDAN, M. I. Attractor dynamics and parallelism in a connectionist sequential. **Cognitive Science**, 1986.

KASKI, S.; KOHONEN, T. Winner-take-all networks for physiological models of competitive learning. **Neural Networks**, 1994.

KOSKELA, T.; LEHTOKANGAS, M.; SAARINEN, J.; KASKI, K. **Time Series Prediction with Multilayer Perceptron, FIR and Elman Neural Networks .** 2000.

LANG, K. J.; HINTON, G. E. A time delay neural network architecture for speech recognition. **Technical Report**, 1988.

LIANG, S. F.; SU, A. W. Y.; LIN, C. T. A new recurrent-network-based music synthesis method for chinese plucked-string instrument. **Proceedings of the International Joint Conference on Neural Networks**, 1999.

MCCULLOCH, W.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. **Bulletin of Mathematical Biophysics**, 1943.

MIGUEL, M. L. F. **Utilização de Redes Neurais para Previsão de Longo Prazo de Tráfego Internet a partir de Informações de Fluxo de Dados**. Dissertação (Mestrado) — Pontifícia Universidade Católica do Paraná, 2011.

MOSELEY, N. Modeling economic time series using a focussed time lagged feedforward neural network. **Proceedings of Student Research Day**, 2003.

NIEVOLA, J. C. Redes neurais artificiais. Pontifícia Universidade Católica do Paraná. 2005.

POMMERANZENBAUM, I. R. **Redes Neurais Artificiais e sua Aplicação na Predição de Séries Temporais Financeiras**, 2009.

PRIMO, A. **Conhecimento e interação: fronteiras entre o agir humano e inteligência artificial**, 2003.

PRINCIPE, J. C.; EULIANO, N. R.; LEFEBVRE, W. C. **Neural and Adaptive Systems: Fundamentals Through Simulations**, 2000.

ROBINSON, A. J.; FALLSIDE, F. The utility driven dynamic error propagation network. **Technical Report CUED/F-INFENG/TR.1**, 1987.

ROCHA, D. M.; JUNIOR, D. L. K.; EBECKEN, N. F. F.; CALôBA, L. P. Redes neurais para modelagem de sistemas estruturais offshore dinâmicos lineares com histerese. In: _____, 2007.

ROSENBLATT, F. The perceptron: A probabilistic model for information storage and organization in the brain. **Psychological Review**, 1958.

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Learning representations by back-propagating erros. **Letters to Nature**, 1986.

RUSSEL, S.; NORVIG, P. **Artificial Intelligence: A Modern Approach**, 2002.

SILVA, L. F. C. **Modelo de Rede Neural Artificial Treinada com o Algoritmo Backpropagation.** Dissertação (Mestrado) — Universidade Federal de Juiz de Fora, 2003.

SMAGT, P. V. D.; KROSE, B. **An Introduction to Neural Networks**, 1996.

TEIXEIRA, J. D. F. **Mentes e máquinas: uma introdução à ciência cognitiva.**, 1998.

WAIBEL, A.; HANAZAWA, T.; HINTON, G.; SHIKANO, K.; LANG, K. Phoneme recognition using time delay neural networks. **IEEE Transactions on Acoustics, Speech and Signal Processing**, 1989.

WATKINS, C. J. C. H.; DAYAN, P. Q-learning. **Machine Learning**, 1992.

WERBOS, P. J. Backpropagation through time: what it does and how to do it. **Proceedings of the IEEE**, 1990.

WERBOS, P. J. **The Roots of Backpropagation: From Ordered Derivates to Neural Networks and Political Forecasting (Adaptive and Learning Systems for Signal Processing, Communications and Control Series)**, 1994.

WIDROW, B.; HOFF, M. E. Adaptive switching circuits. 1960.