Campus de São Carlos

**CONTEXTO E MODULARIZAÇÃO EM REDES
NEURAIS RECORRENTES PARA APRENDIZAGEM
DE SEQUÊNCIAS TEMPORAIS**

Autor: André Santiago Henriques

Orientador: Prof. Dr Aluizio F. R. Araújo.

UNIVERSIDADE DE SÃO PAULO

**ESCOLA DE ENGENHARIA
DE SÃO CARLOS**

Serviço de Pós-Graduação EESC/USP

EXEMPLAR REVISADO

Data de entrada no Serviço: 06 / 08 / 01

Ass.: Guilherme

CONTEXTO E MODULARIZAÇÃO EM REDES NEURAIS RECORRENTES PARA APRENDIZAGEM DE SEQÜÊNCIAS TEMPORAIS

André Santiago Henriques

Dissertação apresentada à Escola de Engenharia de São Carlos, da Universidade de São Paulo, como parte dos requisitos para a obtenção do Título de Mestre em Engenharia Elétrica.

DEDALUS - Acervo - EESC

31100036708

Orientador: Prof. Dr. Aluizio Fausto Ribeiro Araújo

São Carlos
2001

Class.	TESE - EESC
Cott.	2739
Tombo	T0152/01

1197185

Ficha catalográfica preparada pela Seção de Tratamento
da Informação do Serviço de Biblioteca - EESC/USP

H519c Henriques, André Santiago
Contexto e modularização em redes neurais
recorrentes para aprendizagem de seqüências temporais
/ André Santiago Henriques. -- São Carlos, 2001.

Dissertação (Mestrado) -- Escola de Engenharia de
São Carlos-Universidade de São Paulo, 2001.
Área : Engenharia Elétrica.
Orientador: Prof. Dr. Aluizio Fausto Ribeiro
Araújo.

1. Sistemas neurais modulares. 2. Contexto.
3. Redes neurais recorrentes. 4. Seqüências temporais.
I. Título.

FOLHA DE JULGAMENTO

Candidato: Engenheiro **ANDRÉ SANTIAGO HENRIQUES**

Dissertação defendida e julgada em 29-06-2001 perante a Comissão Julgadora:

Prof. Dr. **ALUÍZIO FAUSTO R. ARAÚJO (Orientador)**
(Escola de Engenharia de São Carlos/USP)

APROVADO

Prof. Adj. **FELIPE MAIA GALVÃO FRANÇA**
(COPPE - UFRJ)

APROVADO

Prof. Tit. **MÁRCIO LUIZ DE ANDRADE NETTO**
(Universidade Estadual de Campinas - UNICAMP)

APROVADO

p/ Profa. Associada **VILMA ALVES DE OLIVEIRA**
Coordenadora do Programa de Pós-Graduação
em Engenharia Elétrica

JOSÉ CARLOS A. CINTRA
Presidente da Comissão de Pós-Graduação da EESC

**Aos meus pais, Fernando e Léa,
e à minha esposa, Beatrice, por
todo apoio e incentivo.**

AGRADECIMENTOS

Ao Prof. Dr. Aluizio Fausto Ribeiro Araújo, que soube ser um excelente orientador e um grande amigo.

A todos aqueles que me ajudaram, ou de alguma forma colaboraram comigo no desenvolvimento desta pesquisa. Em especial à Prof. Dra. Maria Creusa Bretas Salles pela disponibilidade e esclarecimentos sobre estatística e processos estocásticos, ao Prof. Dr. Marco Terra e aos companheiros Guilherme Barreto, Arthur, Renato Tinós, Tácio, Benedito e Karla.

Ao Departamento de Engenharia Elétrica pela utilização e fácil acesso aos seus laboratórios, bem como aos professores e funcionários que estiveram sempre dispostos a colaborar.

À Fundação Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), pelo auxílio financeiro fornecido no período da realização deste trabalho.

SUMÁRIO

LISTA DE FIGURAS.....	i
LISTA DE TABELAS.....	ix
LISTA DE ABREVIATURA E SIGLAS.....	x
LISTA DE SÍMBOLOS.....	xi
RESUMO.....	xiii
ABSTRACT	xiv
1. INTRODUÇÃO.....	1
2. REPRODUÇÃO DE SEQUÊNCIAS TEMPORAIS.....	8
2.1. Comentários sobre as Sequências Temporais a serem Aprendidas.....	10
2.2. O Planejamento de Trajetórias para o Braço Mecânico.....	13
3. CONTEXTO E REDES NEURAIS PARA CONTEXTO.....	15
3.1. Redes Neurais que Incluem Processamento de Contexto Temporal.....	16
4. SISTEMAS NEURAIS MODULARES.....	22
4.1. Motivações para a Utilização de Sistemas Modulares.....	22
4.2. Decomposição de Tarefas.....	28
4.3. Treinamento dos Módulos.....	30
4.4. Tomada de Decisão Multi-módulo	31
5. MODELOS PARA SEQUÊNCIAS TEMPORAIS.....	32
5.1. O Modelo de D'Arbo Junior.....	32
5.2. O Modelo TOTEM.....	36
5.3. O Modelo Markoviano Escondido IOHMM	42
6. PREMISSAS PARA IMPLEMENTAR O SISTEMA NEURAL MODULAR	46
6.1. Revisitando o Problema.....	46
6.2. O Algoritmo Supervisionado de Quantização Vetorial	50
6.3. Algumas Abordagens e Modelos Estocásticos	52
6.3.1. Uma breve introdução de Mecânica Estatística.....	52
6.3.2. A Máquina de Boltzmann	54
6.3.3. Têmpera Determinística para Aprendizagem por Quantização Vetorial	57

7. DESENVOLVIMENTO DO SISTEMA PROPOSTO.....	61
7.1. O Módulo Estimador de Contexto Espacial (MECE).....	61
7.1.1. Experimentos Preliminares com o MECE.....	67
7.2. O Módulo de Reprodução de Sequências Temporais.....	70
7.2.1. Acrescentando Unidades Extras de Contexto.....	70
7.2.2. Treinamento e Execução do MRST	72
7.3. O Sistema Neural Modular Completo	77
8. EXPERIMENTOS.....	80
8.1. Trajetórias para um Braço Mecânico.....	81
8.2. Trajetórias Bidimensionais	97
8.3. Experimentos com o MECE	100
8.4. Discussão dos Resultados.....	108
9. CONCLUSÕES	110
REFERÊNCIAS BIBLIOGRÁFICAS.....	117
APÊNDICE	

LISTA DE FIGURAS

FIGURA 1: QUATRO SITUAÇÕES DISTINTAS DE TRAJETÓRIAS ABERTAS QUE PODEM OCORRER NO CONJUNTO DE TREINAMENTO: (A) TRAJETÓRIAS QUE NÃO POSSUEM ESTADOS COMUNS; (B) TRAJETÓRIAS QUE POSSUEM ESTADOS FINAIS COMUNS; (C) TRAJETÓRIAS QUE PARTEM DO MESMO ESTADO INICIAL;(D) TRAJETÓRIAS QUE SE CRUZAM (POSSUEM ESTADOS INTERMEDIÁRIOS COMUNS).	12
FIGURA 2: EXEMPLOS DE TRAJETÓRIAS FECHADAS PARA AVALIAÇÃO DO MODELO: (A) TRAJETÓRIAS FECHADAS SEM PONTOS REPETIDOS; (B) TRAJETÓRIAS FECHADAS COM PONTOS REPETIDOS.	12
FIGURA 3: MECANISMO EXTERNO DE MEMÓRIA DE CURTO PRAZO NO QUAL UM PADRÃO TEMPORAL É TRANSFORMADO EM UM PADRÃO ESPACIAL ATRAVÉS DE UMA LINHA DE ATRASOS NA ENTRADA.	17
FIGURA 4: ARQUITETURA PROPOSTA POR ELMAN NA QUAL O ESTADO (CONTEXTO TEMPORAL) DAS UNIDADES ESCONDIDAS É COPIADO NAS UNIDADES DE CONTEXTO DA CAMADA DE ENTRADA E EM SEGUIDA REALIMENTADO NA CAMADA ESCONDIDA. AS SETAS LARGAS INDICAM CONEXÕES TODOS-PARA-TODOS, ENQUANTO A SETA FINA NA REALIMENTAÇÃO INDICA CONEXÕES UM-PARA-UM.	19
FIGURA 5: A) REDE RECORRENTE COM MEMÓRIA DE ORDEM 1; B) REDE RECORRENTE COM MEMÓRIA DE ORDEM 2; C) REDE LOCALMENTE RECORRENTE COM MEMÓRIA DE ORDEM 1; D) REDE LOCALMENTE RECORRENTE COM MEMÓRIA DE ORDEM 2.	20
FIGURA 6: A REDE NEURAL DO EXEMPLO (A) É MAIS SUSCEPTÍVEL À COMUNICAÇÃO CRUZADA ESPACIAL DO QUE A REDE NEURAL DO EXEMPLO (B).	27
FIGURA 7: MODELO DE REDE RECORRENTE QUE SERÁ UTILIZADO COMO PONTO DE PARTIDA PARA O DESENVOLVIMENTO DE UM SISTEMA DE PLANEJAMENTO DE TRAJETÓRIAS.	33
FIGURA 8: SISTEMA MODULAR TOTEM, FORMADO PELA CAMADA DE ENTRADA E QUATRO MÓDULOS SUCESSIVOS.	38

FIGURA 9: A UNIDADE VENCEDORA DO MÓDULO SOM EXCITA (VIA CONEXÃO UM PARA UM) A UNIDADE CORRESPONDENTE NO MÓDULO STM. O DETALHE MOSTRA A UNIDADE DO MÓDULO STM COM INIBIÇÕES LATERAIS E REALIMENTAÇÃO EXCITATÓRIA.	39
FIGURA 10: ARQUITETURA DO SISTEMA MODULAR HÍBRIDO, IOHMM, FORMADO POR MÓDULOS NEURAIIS COOPERATIVOS, E UMA ABORDAGEM DE MODELOS MARKOVIANOS ESCONDIDOS.	43
FIGURA 11: DIAGRAMA DO SISTEMA A SER PROPOSTO, FORMADO POR DOIS MÓDULOS. O PRIMEIRO MÓDULO RECEBE O PADRÃO DE ENTRADA, IDENTIFICA A SEQUÊNCIA A SER REPRODUZIDA E FORNECE UM PROTÓTIPO DO CONTEXTO ESPACIAL, E UMA PROBABILIDADE ASSOCIADA. O SEGUNDO MÓDULO UTILIZA O PROTÓTIPO DO CONTEXTO ESPACIAL FORNECIDO PARA REPRODUZIR A SEQUÊNCIA DESEJADA.	48
FIGURA 12: ARQUITETURA DO MODELO LVQ1, FORMADO POR DUAS CAMADAS DE UNIDADES CONECTADAS POR UM CONJUNTO DE PESOS DE PROPAGAÇÃO DIRETA.	50
FIGURA 13: ARQUITETURA DA MÁQUINA DE BOLTZMANN; O MODELO POSSUI UM GRUPO DE UNIDADES VISÍVEIS E UM GRUPO DE UNIDADES ESCONDIDAS.	55
FIGURA 14: ARQUITETURA PROPOSTA PARA REALIZAR O MECE.	62
FIGURA 15: ALGORITMO UTILIZADO PARA O TREINAMENTO DO MECE.	66
FIGURA 16: CONJUNTO FORMADO POR SEIS TRAJETÓRIAS E UTILIZADO EM EXPERIMENTOS PRELIMINARES COM O MÓDULO ESTIMADOR DE CONTEXTO ESPACIAL.	68
FIGURA 17: ESTUDO DO NÚMERO DE ERROS E DE DÚVIDAS EM FUNÇÃO DO PARÂMETRO β ($1/T$) NA CLASSIFICAÇÃO OBTIDA COM O MECE.	69
FIGURA 18: ADAPTAÇÃO PROPOSTA PARA O MODELO DE D'ARBO JUNIOR. FORAM INCLUÍDAS UNIDADES EXTRAS NA CAMADA DE CONTEXTO VISANDO MELHORAR O DESEMPENHO DO MODELO NA APRENDIZAGEM DE SEQUÊNCIAS CONTENDO PADRÕES COM POUCAS ENTRADAS.	71
FIGURA 19: ALGORITMO DE TREINAMENTO UTILIZADO PARA O MODELO MODIFICADO DE D'ARBO JUNIOR, E QUE REALIZA O PAPEL DO MRST.	76

FIGURA 20: SISTEMA MODULAR PROPOSTO E IMPLEMENTADO NESTA PESQUISA PARA REPRODUÇÃO DE SEQUÊNCIAS TEMPORAIS.....	77
FIGURA 21: DOIS CONJUNTOS DE TREINAMENTO UTILIZADOS POR D'ARBO JUNIOR, FORMADOS POR 3 TRAJETÓRIAS (DE 11 PONTOS CADA) CONSIDERANDO DUAS SITUAÇÕES DISTINTAS: A) TRAJETÓRIAS QUE NÃO COMPARTILHAM PONTOS COMUNS; E B) TRAJETÓRIAS QUE COMPARTILHAM PONTOS INTERMEDIÁRIOS.....	82
FIGURA 22: (A) CONJUNTO DE TRAJETÓRIAS UTILIZADO PARA DUAS SITUAÇÕES DISTINTAS: 1. COM O MESMO PONTO FINAL (TRAJETÓRIAS DE A PARA B); E 2. COM O MESMO PONTO INICIAL (TRAJETÓRIAS DE B PARA A). (B) CONJUNTO COM TRAJETÓRIAS FECHADAS.	83
FIGURA 23: RESULTADO OBTIDO COM A REDE PARCIALMENTE RECORRENTE REPRODUZINDO AS TRAJETÓRIAS SC, QUANDO OS PONTOS INICIAL E FINAL FORNECIDOS SÃO IGUAIS AOS TREINADOS.	84
FIGURA 24: RESULTADO OBTIDO COM O SISTEMA MODULAR PROPOSTO REPRODUZINDO AS TRAJETÓRIAS SC, QUANDO OS PONTOS INICIAL E FINAL FORNECIDOS SÃO IGUAIS AOS TREINADOS.	84
FIGURA 25: RESULTADO OBTIDO COM A REDE PARCIALMENTE RECORRENTE REPRODUZINDO OS ÂNGULOS DAS JUNTAS PARA AS TRAJETÓRIAS SC, QUANDO OS PONTOS INICIAL E FINAL FORNECIDOS SÃO IGUAIS AOS TREINADOS.	85
FIGURA 26: RESULTADO OBTIDO COM O SISTEMA MODULAR PROPOSTO REPRODUZINDO OS ÂNGULOS DAS JUNTAS PARA AS TRAJETÓRIAS SC, QUANDO OS PONTOS INICIAL E FINAL FORNECIDOS SÃO IGUAIS AOS TREINADOS.	86
FIGURA 27: RESULTADO OBTIDO COM A REDE PARCIALMENTE RECORRENTE REPRODUZINDO OS TORQUES NAS JUNTAS DO BRAÇO MECÂNICO PARA AS TRAJETÓRIAS SC, QUANDO OS PONTOS INICIAL E FINAL FORNECIDOS SÃO IGUAIS AOS TREINADOS.....	87
FIGURA 28: RESULTADO OBTIDO COM O SISTEMA MODULAR REPRODUZINDO OS TORQUES NAS JUNTAS DO BRAÇO MECÂNICO PARA AS TRAJETÓRIAS SC, QUANDO OS PONTOS INICIAL E FINAL FORNECIDOS SÃO IGUAIS AOS TREINADOS.....	88

FIGURA 29: RESULTADOS UTILIZANDO-SE A REDE RECORRENTE (RR) PARA REPRODUZIR A TRAJETÓRIA T1 DO CONJUNTO DE TRAJETÓRIAS DO TIPO M-1, QUANDO O PONTO INICIAL FORNECIDO É DIFERENTE DAQUELE VISTO NO TREINAMENTO. (A) PARTINDO DO 2° PONTO; (B) PARTINDO DO 4° PONTO; (C) PARTINDO DO 7° PONTO; (D) PARTINDO DO 9° PONTO.....	90
FIGURA 30: RESULTADOS COM A REDE RECORRENTE (RR) PARA REPRODUZIR A TRAJETÓRIA T1 DO CONJUNTO DE TRAJETÓRIAS DO TIPO CC, QUANDO O PONTO INICIAL FORNECIDO É DIFERENTE DAQUELE VISTO NO TREINAMENTO. (A) PARTINDO DO 2° PONTO; (B) PARTINDO DO 4° PONTO; (C) PARTINDO DO 7° PONTO; (D) PARTINDO DO 9° PONTO.	91
FIGURA 31: RESULTADOS UTILIZANDO-SE O SISTEMA MODULAR COMPLETO (SC) PARA REPRODUZIR A TRAJETÓRIA T1 DO CONJUNTO DE TRAJETÓRIAS DO TIPO M-1, QUANDO O PONTO INICIAL FORNECIDO É DIFERENTE DAQUELE VISTO NO TREINAMENTO. (A) PARTINDO DO 2° PONTO; (B) PARTINDO DO 4° PONTO; (C) PARTINDO DO 7° PONTO; (D) PARTINDO DO 9° PONTO.	92
FIGURA 32: RESULTADOS UTILIZANDO-SE O SISTEMA MODULAR COMPLETO (SC) PARA REPRODUZIR A TRAJETÓRIA T1 DO CONJUNTO DE TRAJETÓRIAS DO TIPO CC, QUANDO O PONTO INICIAL FORNECIDO É DIFERENTE DAQUELE VISTO NO TREINAMENTO. (A) PARTINDO DO 2° PONTO; (B) PARTINDO DO 4° PONTO; (C) PARTINDO DO 7° PONTO; (D) PARTINDO DO 9° PONTO.	93
FIGURA 33: RESULTADOS DA REPRODUÇÃO DA TRAJETÓRIA FECHADA (OITO), PELOS DOIS MODELOS IMPLEMENTADOS, PARTINDO DE DIFERENTES PONTOS INTERMEDIÁRIOS: (A) E (B) PARTINDO DO 10° PONTO; (C) E (D) PARTINDO DO 25° PONTO; (E) E (F) PARTINDO DO 40° PONTO. AS FIGURAS DA ESQUERDA SE REFEREM À REDE RECORRENTE, ENQUANTO AS FIGURAS DA DIREITA SE REFEREM AO SISTEMA MODULAR.	94
FIGURA 34: DOIS CASOS DE ERRO DE GENERALIZAÇÃO OBTIDOS COM O SISTEMA MODULAR PROPOSTO: (A) TENTANDO PARTIR DO 7° PONTO DA TRAJETÓRIA T2 DO CONJUNTO M-1, O MECE IDENTIFICOU ERRONEAMENTE A TRAJETÓRIA T1 PARA SER REPRODUZIDA; (B) VISANDO REPRODUZIR A TRAJETÓRIA T2 DO TIPO CC A PARTIR DO 6° PONTO, O MECE ERRONEAMENTE IDENTIFICOU A TRAJETÓRIA T3 COMO SENDO A DESEJADA PARA A REPRODUÇÃO.	95
FIGURA 35: TRAJETÓRIAS 2D UTILIZADAS NA SEGUNDA FASE DE EXPERIMENTOS COM O SISTEMA MODULAR PROPOSTO.	98

FIGURA 36: RESULTADOS OBTIDOS COM O SISTEMA NEURAL MODULAR RECUPERANDO AS TRAJETÓRIAS A PARTIR DE PONTOS INICIAL E FINAL TREINADOS.	99
FIGURA 37: RESULTADOS OBTIDOS COM O MECE IDENTIFICANDO PADRÕES CONTENDO PONTOS INICIAIS NÃO TREINADOS. ESTRELAS INDICAM PONTOS x_i , UTILIZADOS NO LUGAR DE x_0 , PARA OS QUAIS A REDE CONSEGUIU CLASSIFICAR CORRETAMENTE OS PADRÕES. CÍRCULOS REPRESENTAM OCORRÊNCIA DE DÚVIDAS. E TRIÂNGULOS REPRESENTAM PONTOS PARA OS QUAIS A IDENTIFICAÇÃO REALIZADA PELO MECE FOI ERRADA.....	103
FIGURA 38: TRAJETÓRIA T1 EM CONJUNTO COM PONTOS INICIAIS E MÉDIAS DE OUTRAS TRAJETÓRIAS. OS QUADRADOS INDICAM OS PONTOS INICIAIS DE CADA TRAJETÓRIA. OS ASTERISCOS INDICAM AS MÉDIAS DE CADA TRAJETÓRIA.....	105
FIGURA 39: TRAJETÓRIA T4 EM CONJUNTO COM PONTOS INICIAIS E MÉDIAS DE OUTRAS TRAJETÓRIAS. OS QUADRADOS E ASTERISCOS INDICAM RESPECTIVAMENTE O PONTO INICIAL E A MÉDIA DE CADA TRAJETÓRIA.....	105
FIGURA 40: RESULTADOS OBTIDOS COM O MECE IDENTIFICANDO PADRÕES CONTENDO PONTOS INICIAIS TREINADOS E PONTOS FINAIS NÃO TREINADOS. ESTRELAS INDICAM PONTOS x_i , UTILIZADOS NO LUGAR DE x_f , PARA OS QUAIS A REDE CONSEGUIU IDENTIFICAR CORRETAMENTE OS PADRÕES. CÍRCULOS REPRESENTAM OCORRÊNCIA DE DÚVIDAS. E TRIÂNGULOS REPRESENTAM PONTOS PARA OS QUAIS A IDENTIFICAÇÃO REALIZADA PELO MECE FOI ERRADA	107

FIGURAS DOS APÊNDICES

FIGURA 41: EXEMPLO DE CADEIA DE MARKOV COM 4 ESTADOS (S_1 , S_2 , S_3 E S_4) E PROBABILIDADES DE TRANSIÇÃO DO ESTADO i PARA O ESTADO j ζ_{ij}	IV
FIGURA 42: CADEIA DE MARKOV SIMPLES PARA O EXPERIMENTO DA MOEDA. ESSE MODELO CONSIDERA APENAS UMA MOEDA SENDO LANÇADA.....	VI
FIGURA 43: MODELO MARKOVIANO ESCONDIDO QUE CONSIDERA DUAS MOEDAS DIFERENTES SENDO LANÇADAS ALEATORIAMENTE NO EXPERIMENTO DA MOEDA.....	VII

FIGURA 44: RESULTADO OBTIDO COM A REDE PARCIALMENTE RECORRENTE REPRODUZINDO AS TRAJETÓRIAS CC, QUANDO OS PONTOS INICIAL E FINAL FORNECIDOS SÃO IGUAIS AOS TREINADOS.	IX
FIGURA 45: RESULTADO OBTIDO COM O SISTEMA MODULAR PROPOSTO REPRODUZINDO AS TRAJETÓRIAS CC, QUANDO OS PONTOS INICIAL E FINAL FORNECIDOS SÃO IGUAIS AOS TREINADOS.	IX
FIGURA 46: RESULTADO OBTIDO COM A REDE PARCIALMENTE RECORRENTE REPRODUZINDO OS ÂNGULOS DAS JUNTAS PARA AS TRAJETÓRIAS CC, QUANDO OS PONTOS INICIAL E FINAL FORNECIDOS SÃO IGUAIS AOS TREINADOS.	X
FIGURA 47: RESULTADO OBTIDO COM A REDE PARCIALMENTE RECORRENTE REPRODUZINDO OS TORQUES DAS JUNTAS PARA AS TRAJETÓRIAS CC, QUANDO OS PONTOS INICIAL E FINAL FORNECIDOS SÃO IGUAIS AOS TREINADOS.	XI
FIGURA 48: RESULTADO OBTIDO COM O SISTEMA MODULAR PROPOSTO REPRODUZINDO OS ÂNGULOS DAS JUNTAS PARA AS TRAJETÓRIAS CC, QUANDO OS PONTOS INICIAL E FINAL FORNECIDOS SÃO IGUAIS AOS TREINADOS.	XII
FIGURA 49: RESULTADO OBTIDO COM O SISTEMA MODULAR PROPOSTO REPRODUZINDO OS TORQUES DAS JUNTAS PARA AS TRAJETÓRIAS CC, QUANDO OS PONTOS INICIAL E FINAL FORNECIDOS SÃO IGUAIS AOS TREINADOS.	XIII
FIGURA 50: RESULTADO OBTIDO COM A REDE PARCIALMENTE RECORRENTE REPRODUZINDO AS TRAJETÓRIAS 1-M, QUANDO OS PONTOS INICIAL E FINAL FORNECIDOS SÃO IGUAIS AOS TREINADOS.	XIV
FIGURA 51: RESULTADO OBTIDO COM O SISTEMA MODULAR PROPOSTO REPRODUZINDO AS TRAJETÓRIAS 1-M, QUANDO OS PONTOS INICIAL E FINAL FORNECIDOS SÃO IGUAIS AOS TREINADOS.	XIV
FIGURA 52: RESULTADO OBTIDO COM A REDE PARCIALMENTE RECORRENTE REPRODUZINDO OS ÂNGULOS DAS JUNTAS PARA AS TRAJETÓRIAS 1-M, QUANDO OS PONTOS INICIAL E FINAL FORNECIDOS SÃO IGUAIS AOS TREINADOS.	XV

FIGURA 53: RESULTADO OBTIDO COM A REDE PARCIALMENTE RECORRENTE REPRODUZINDO OS TORQUES NAS JUNTAS PARA AS TRAJETÓRIAS 1-M, QUANDO OS PONTOS INICIAL E FINAL FORNECIDOS SÃO IGUAIS AOS TREINADOS.....	XVI
FIGURA 54: RESULTADO OBTIDO COM O SISTEMA MODULAR PROPOSTO REPRODUZINDO OS ÂNGULOS DAS JUNTAS PARA AS TRAJETÓRIAS 1-M, QUANDO OS PONTOS INICIAL E FINAL FORNECIDOS SÃO IGUAIS AOS TREINADOS.....	XVII
FIGURA 55: RESULTADO OBTIDO COM O SISTEMA MODULAR PROPOSTO REPRODUZINDO OS TORQUES NAS JUNTAS PARA AS TRAJETÓRIAS 1-M, QUANDO OS PONTOS INICIAL E FINAL FORNECIDOS SÃO IGUAIS AOS TREINADOS.....	XVIII
FIGURA 56: RESULTADO OBTIDO COM A REDE RECORRENTE REPRODUZINDO AS TRAJETÓRIAS M-1, QUANDO OS PONTOS INICIAL E FINAL FORNECIDOS SÃO IGUAIS AOS TREINADOS....	XIX
FIGURA 57: RESULTADO OBTIDO COM O SISTEMA MODULAR PROPOSTO REPRODUZINDO AS TRAJETÓRIAS M-1, QUANDO OS PONTOS INICIAL E FINAL FORNECIDOS SÃO IGUAIS AOS TREINADOS.	XIX
FIGURA 58: RESULTADO OBTIDO COM A REDE RECORRENTE REPRODUZINDO OS ÂNGULOS DAS JUNTAS PARA AS TRAJETÓRIAS M-1, QUANDO OS PONTOS INICIAL E FINAL FORNECIDOS SÃO IGUAIS AOS TREINADOS.	XX
FIGURA 59: RESULTADO OBTIDO COM A REDE RECORRENTE REPRODUZINDO OS TORQUES NAS JUNTAS PARA AS TRAJETÓRIAS M-1, QUANDO OS PONTOS INICIAL E FINAL FORNECIDOS SÃO IGUAIS AOS TREINADOS.	XXI
FIGURA 60: RESULTADO OBTIDO COM O SISTEMA MODULAR PROPOSTO REPRODUZINDO OS ÂNGULOS DAS JUNTAS PARA AS TRAJETÓRIAS M-1, QUANDO OS PONTOS INICIAL E FINAL FORNECIDOS SÃO IGUAIS AOS TREINADOS.....	XXII
FIGURA 61: RESULTADO OBTIDO COM O SISTEMA MODULAR PROPOSTO REPRODUZINDO OS TORQUES NAS JUNTAS PARA AS TRAJETÓRIAS 1-M, QUANDO OS PONTOS INICIAL E FINAL FORNECIDOS SÃO IGUAIS AOS TREINADOS.....	XXIII

FIGURA 62: RESULTADO OBTIDO COM O SISTEMA MODULAR PROPOSTO REPRODUZINDO AS TRAJETÓRIAS FECHADAS, QUANDO OS PONTOS INICIAL E FINAL FORNECIDOS SÃO IGUAIS AOS TREINADOS.	XXIV
FIGURA 63: RESULTADO OBTIDO COM O SISTEMA MODULAR PROPOSTO REPRODUZINDO OS ÂNGULOS DAS JUNTAS PARA AS TRAJETÓRIAS FECHADAS, QUANDO OS PONTOS INICIAL E FINAL FORNECIDOS SÃO IGUAIS AOS TREINADOS.....	XXV
FIGURA 64: RESULTADO OBTIDO COM O SISTEMA MODULAR PROPOSTO REPRODUZINDO OS TORQUES NAS JUNTAS PARA AS TRAJETÓRIAS FECHADAS, QUANDO OS PONTOS INICIAL E FINAL FORNECIDOS SÃO IGUAIS AOS TREINADOS.....	XXVI
FIGURA 65: RESULTADOS DA REPRODUÇÃO DA TRAJETÓRIA CIRCULAR FECHADA, OBTIDOS COM OS DOIS MODELOS IMPLEMENTADOS, PARTINDO DE DIFERENTES PONTOS INTERMEDIÁRIOS: (A) E (B) PARTINDO DO 10º PONTO; (C) E (D) PARTINDO DO 25º PONTO; (E) E (F) PARTINDO DO 40º PONTO. AS FIGURAS DA ESQUERDA SE REFEREM À REDE PARCIALMENTE RECORRENTE, ENQUANTO QUE AS FIGURAS DA DIREITA SE REFEREM AO SISTEMA MODULAR DESENVOLVIDO NESTA PESQUISA.	XXVII

LISTA DE TABELAS

TABELA 1: NÚMERO MÉDIO DE ÉPOCAS E TEMPO MÉDIO GASTO PARA O TREINAMENTO DE CADA UM DOS MODELOS IMPLEMENTADOS E PARA CADA CONJUNTO DE TRAJETÓRIAS.	89
TABELA 2: RESULTADOS OBTIDOS COM O SISTEMA MODULAR PARA OS CONJUNTOS DE TESTES CRIADOS PARA CADA TIPO DE TRAJETÓRIA, PARA AVALIAÇÃO DA CAPACIDADE DE GENERALIZAÇÃO DO SISTEMA.	97
TABELA 3: EQUAÇÕES DESCREVENDO AS 8 TRAJETÓRIAS 2D UTILIZADAS NO CONJUNTO DE TREINAMENTO DA SEGUNDA FASE DE EXPERIMENTOS COM O SISTEMA MODULAR PROPOSTO NESTA PESQUISA.	98
TABELA 4: RESULTADOS OBTIDOS PARA DIFERENTES VALORES DE L_{duv} : VALORES AJUSTADOS PARA β ; CUSTO TOTAL, L , NO CONJUNTO DE TESTES; DÚVIDAS E ERROS PARA AS SITUAÇÕES TREINADAS E NÃO TREINADAS.	101
TABELA 5: PROBABILIDADES DE IDENTIFICAÇÃO DE CADA UM DOS PONTOS PERTENCENTES A T1, E PARA OS QUAIS O SISTEMA APRESENTOU DÚVIDAS DE CLASSIFICAÇÃO.	104
TABELA 6: PROBABILIDADES DE IDENTIFICAÇÃO DE CADA UM DOS PONTOS PERTENCENTES À REGIÃO 1 DA TRAJETÓRIA T4, E PARA OS QUAIS O SISTEMA APRESENTOU DÚVIDAS E ERROS DE IDENTIFICAÇÃO.	106

LISTA DE ABREVIATURA E SIGLAS

BPTT	BackPropagation Through Time.
RTRL	Real Time Recurrent Learning.
HMM	Hidden Markov Model.
EM	Expectation-Maximization.
GEM	Generalized EM.
IOHMM	Input-Output Hidden Markov Model.
TOTEM	Topological and Temporal correlator network.
ART	Adaptive Resonance Theory.
fuzzy-ART	fuzzy Adaptive Resonance Theory.
STM	Short Term Memory.
MLP	Multilayer Perceptron.
RNMs	Redes Neurais Modulares.
SOM	Self-Organizing Map.
MECE	Módulo Estimador de Contexto Espacial.
MRST	Módulo Reprodutor de Sequências Temporais.
LVQ	Learning Vector Quantization
PUMA	Programable Universal Machine Assembly
SC	Trajetórias sem cruzamentos
CC	Trajetórias com cruzamentos em pontos intermediários
1-m	Trajetórias com o mesmo ponto de partida
m-1	Trajetórias com o mesmo ponto final

LISTA DE SÍMBOLOS

S	- Sequência Temporal
t	- Instante de tempo
x_0	- Ponto (ou estado) inicial de uma sequência
x_i	- Ponto (ou estado) intermediário de uma sequência
x_f	- Ponto (ou estado) final de uma sequência
x'_0	- Ponto inicial fornecido para a reprodução
x'_i	- Ponto intermediário fornecido para a reprodução
x'_f	- Ponto final fornecido para a reprodução
$\hat{x}_0$	- Ponto inicial estimado
$\hat{x}_f$	- Ponto final estimado
$\bar{x}_i$	- Média dos pontos intermediários
T	- Pseudotemperatura
β	- é o inverso de T
x	- Vetor representando um ponto de uma sequência ou o estado em uma camada de neurônios
α	- Taxa de aprendizagem
ψ	- Fator de momento
w_{ij}	- Conexão entre i e j
w_j	- Vetor de pesos que chegam em uma unidade i
c_i	- Classe identificada pela unidade i
$d(.,.)$	- Distância Euclidiana entre dois vetores
D	- Distorção total esperada
$i_j(t)$	- ativação da j -ésima unidade de entrada (ou de contexto) no instante t
$h_j(t)$	- ativação da j -ésima unidade escondida no instante t
$o_j(t)$	- ativação da j -ésima unidade de saída no instante t
k_{ij}	- conexão de realimentação da unidade j de saída para a unidade i de entrada (ou contexto)
q_i^e	- ativação que chega na entrada da unidade i de contexto (ou entrada).
q_i^h	- ativação que chega na entrada da unidade escondida i .
q_i^s	- ativação que chega na entrada da unidade i de saída.
n_e	- número de unidades na camada de contexto (ou entrada)
n_h	- número de unidades na camada escondida

n_s	- número de unidades na camada de saída
y	vetor de características espaciais e temporais correlacionadas
N_j	- sub-rede j de estado
O_j	- sub-rede j de saída
$\varphi_{j,t}$	- distribuição de probabilidade fornecida pela rede N_j no instante t
$\eta_{j,t}$	- esperança da saída, fornecida pela sub-rede O_j no instante t
η_t	- esperança da saída do sistema no instante t
ζ_t	- distribuição de probabilidades de estados no instante t
$\zeta_{j,t}$	- probabilidade do estado j fornecida pela sub-rede N_j no instante t
v	- padrão de entrada
$\hat{v}$	- Protótipo de contexto espacial fornecido pelo MECE
$\hat{v}_{MRST}$	- entrada do MRST, obtida a partir de $\hat{v}$
$P(A B)$	- probabilidade condicional de A dado B
$E[A B]$	- esperança condicional de A dado B
T_k	- temperatura em graus Kelvin
K_B	- constante de Boltzmann
E_i	- Energia do sistema quando este assume o estado i
H	- entropia de Shannon
Z	- função de partição
$L(.,.)$	- função custo de identificação de um padrão
L_{total}	- custo total de identificação em um conjunto
L_{div}	- limiar de dúvida
δ_i	- função delta calculada no algoritmo de retropropagação
N	- número de apresentações do conjunto durante o treinamento
N_{max}	- número máximo de apresentações do conjunto, permitido no treinamento
α_0	- taxa inicial de aprendizagem
$erro_{Max}$	- máximo erro permitido para cada variável do padrão de entrada

RESUMO

Henriques, A. S. (2001), *Contexto e modularização em redes neurais recorrentes para aprendizagem de seqüências temporais*. São Carlos, 2001. Dissertação (Mestrado) – Escola de Engenharia de São Carlos, Universidade de São Paulo.

Este trabalho apresenta um sistema neural modular, que processa separadamente informações de contexto espacial e temporal, para a tarefa de reprodução de seqüências temporais. Para o desenvolvimento do sistema neural foram considerados redes neurais recorrentes, modelos estocásticos, sistemas neurais modulares e processamento de informações de contexto. Em seguida, foram estudados três modelos com abordagens distintas para aprendizagem de seqüências temporais: uma rede neural parcialmente recorrente, um exemplo de sistema neural modular e um modelo estocástico utilizando a teoria de modelos markovianos escondidos. Com base nos estudos e modelos apresentados, esta pesquisa propõe um sistema formado por dois módulos sucessivos distintos. Uma rede de propagação direta (módulo estimador de contexto espacial) realiza o processamento de contexto espacial identificando a seqüência a ser reproduzida e fornecendo um protótipo do contexto espacial para o segundo módulo. Este é formado por uma rede parcialmente recorrente (módulo de reprodução de seqüências temporais) para aprender as informações de contexto temporal e reproduzir em suas saídas a seqüência identificada pelo módulo anterior. Para a finalidade mencionada, este mestrado utiliza a distribuição de Gibbs na saída do módulo para contexto espacial de forma que este forneça probabilidades associadas aos protótipos de contexto espacial, indicando o grau de certeza do módulo, e possibilitando a utilização de procedimentos especiais para os casos de dúvida. O sistema neural foi testado em conjuntos contendo trajetórias abertas, fechadas, e com diferentes situações de ambigüidade e complexidade. Duas situações distintas foram avaliadas: (a) capacidade do sistema em reproduzir trajetórias a partir de pontos iniciais treinados; e (b) capacidade de generalização do sistema reproduzindo trajetórias considerando pontos iniciais ou finais em situações não treinadas. A situação (b) é um problema de difícil solução em redes neurais recorrentes devido à falta de contexto temporal, essencial na reprodução de seqüências. Foram realizados experimentos comparando o desempenho do sistema modular proposto com o de uma rede parcialmente recorrente operando sozinha e um sistema modular neural (TOTEM). Os resultados sugerem que o sistema proposto apresentou uma capacidade de generalização significativamente melhor, sem que houvesse uma deterioração na capacidade de reproduzir seqüências treinadas. Estes resultados foram obtidos em sistema mais simples que o TOTEM.

Palavras-chave: sistemas neurais modulares, contexto, redes neurais recorrentes, seqüências temporais.

ABSTRACT

Henriques, A. S. (2001), *Context and modularization in recurrent neural networks for temporal sequences learning*. São Carlos, 2001. Dissertação (Mestrado) – Escola de Engenharia de São Carlos, Universidade de São Paulo.

This work presents a new modular neural system to deal separately with spatial and temporal context information, during temporal sequence processing. Given the initial and final states of the sequence, the neural system can reproduce the whole sequence linking these points. The proposed model involves concepts on recurrent neural networks, stochastic models, modular neural systems and context information processing. Three models based on distinct approaches to learn temporal sequences were particularly important in this work: a partially recurrent neural network, a modular neural system and a stochastic model based on the *Hidden Markov Models* theory. This master thesis presents a new modular neural system composed of two supervised neural networks. A feedforward neural network (spatial context estimator) to identify the desired sequence to be reproduced and to provide a spatial context prototype to the second module. This is a partially recurrent neural network to reproduce the sequence identified by the former module. Moreover, the first module employs the Gibbs distribution in the spatial context estimator outputs in such a way to obtain the uncertainty of the sequence identification task. Thus, with these probability values, special procedures may be used whenever a doubt occurs. The proposed system was evaluated in different domains containing open and closed sequences with different levels of complexity due to space dimension and level of ambiguity of the trained trajectories. The system was evaluated according to its ability to reproduce the sequence whenever versions of the initial and final points are provided. A version may be exactly the points seen during the training stage or points trained as intermediate states. The latter is considered a difficult task for recurrent neural networks due to the lack of temporal context information. Experiments were done comparing the performance of the proposed modular neural system with the performance of a recurrent neural network itself and a modular neural system (a model called TOTEM) for sequence reproduction. The results suggest that the proposed modular neural system presented ability to generalize significantly better than that of the recurrent neural network without deteriorating its ability to reproduce sequences starting from trained situations. The neural system may reproduce the results of the TOTEM with a simpler topology.

Keywords: modular neural systems, context information, recurrent neural networks, temporal sequences.

1. INTRODUÇÃO

O estudo de redes neurais artificiais se aplica a uma grande variedade de problemas da engenharia. Os modelos de redes neurais são capazes de se adaptarem para aprender através de dados apresentados durante um treinamento. A aprendizagem por experiência é uma atividade essencial em redes neurais na obtenção de informações a respeito do ambiente [41]. Essa característica torna os sistemas neurais bem mais flexíveis do que abordagens algorítmicas baseadas em rotinas pré-processadas [62].

Redes neurais artificiais são sistemas originalmente inspirados na observação dos sistemas nervosos biológicos que são capazes de anexar conhecimento através da extração de regularidades de um conjunto de dados [59]. Tais sistemas são formados por uma grande quantidade de unidades de processamento (neurônios) altamente interconectadas. Tais conexões (conhecidas como pesos, ou sinapses) ponderam os sinais transmitidos pelas unidades de processamento, e geralmente seus valores são adaptados através das chamadas regras de aprendizagem. De maneira geral, uma rede neural é um algoritmo de aprendizagem de máquinas desenvolvido para modelar a forma com que o cérebro executa tarefas particulares ou funções de interesse [33].

São os pesos sinápticos da rede que guardam o “conhecimento” adquirido por uma rede neural. O procedimento utilizado para realizar a aprendizagem de uma rede neural é conhecido como algoritmo de aprendizagem. Este procedimento contém regras de aprendizagem que definem como modificar, de maneira ordenada, os pesos sinápticos da rede, para atingir um determinado objetivo.

Algumas propriedades relevantes e capacidades observadas em redes neurais artificiais serão descritas a seguir [33]:

- **Não linearidade:** Geralmente, redes neurais artificiais são sistemas não lineares, pois são construídas a partir da interconexão de várias unidades (neurônios) que realizam um processamento não linear. A não linearidade é uma propriedade importante, por exemplo, quando se deseja gerar ou aprender sinais (ou funções) não lineares.
- **O mapeamento entrada-saída:** A aprendizagem de uma rede neural envolve a modificação de seus pesos sinápticos tendo como objetivo aprender um

mapeamento entrada-saída. Um exemplo típico é o caso da aprendizagem supervisionada, na qual vários pares entrada-saída são apresentados para a rede durante o treinamento. Nesse caso, o algoritmo de aprendizagem corrige os pesos da rede de forma a minimizar o erro na saída quando uma entrada é fornecida.

- **Adaptabilidade:** Redes neurais possuem a capacidade de se adaptar para se adequar a um determinado ambiente. Em particular, uma rede neural treinada para operar em um certo ambiente específico pode ser facilmente retreinada para se adaptar a pequenas mudanças ocorridas com este ambiente. Como regra geral, pode-se dizer que quanto mais adaptativo for um sistema (e garantindo certas condições para a estabilidade) mais robusto será este sistema para trabalhar em ambientes com características não estacionárias.
- **Informação Contextual:** O conhecimento é representado pela estrutura, e estados de ativação das unidades da rede neural. Todo neurônio de uma rede é potencialmente afetado pela atividade global dos neurônios restantes. Consequentemente, informações contextuais são tratadas naturalmente por uma rede neural.
- **Tolerância à falta:** Devido à natureza distribuída da informação armazenada na rede, a perda de algumas conexões e/ou unidades, não necessariamente acarretará em uma redução significativa do desempenho da rede. Geralmente, o que se observa é uma degradação paulatina à medida que uma maior quantidade de elementos da rede são danificados.

Os modelos de redes neurais com aprendizagem supervisionada¹ mais conhecidos são as redes perceptron multicamadas (MLP – *Multilayer Perceptron*) treinadas com um algoritmo de retropropagação do erro. Tais modelos têm sido amplamente utilizados em inúmeras aplicações. Para uma boa compreensão de tais modelos, [60] apresenta um estudo detalhado a respeito da teoria, arquitetura e aplicações destes.

Recentemente, tem-se aumentado muito o interesse em utilizar redes neurais artificiais em problemas tais como a classificação, identificação, controle, predição, diagnóstico de falhas, entre outros processos dinâmicos. Em muitos problemas de aprendizagem, além das informações espaciais dos padrões, a estrutura temporal dos dados também contém informações importantes que devem ser aprendidas pelo sistema. Nesse caso, o sistema precisa ter algum mecanismo de memória de curto prazo capaz de reter informações a respeito do passado dos dados de entrada e/ou dos estados do sistema (tais informações são conhecidas como

¹ Os modelos supervisionados são aqueles que, durante o treinamento, utilizam um “professor” para indicar ao modelo qual a saída desejada para cada padrão de entrada treinado.

informações de contexto temporal) [3],[6],[7],[10],[12]. Problemas desse tipo surgem em uma grande variedade de aplicações, tais como: processamento de linguagem escrita ou falada, reconhecimento de gestos, navegação de um robô, produção de sinais atuadores em tarefas de controle, e predição e/ou reprodução de seqüências temporais, entre outros.

Hoje em dia, é possível encontrar na literatura recente inúmeras abordagens distintas utilizando diferentes arquiteturas de redes neurais artificiais para aprendizagem de seqüências temporais, [3],[7],[12],[14],[22],[30],[62]. Tais arquiteturas se diferem principalmente no que se refere ao mecanismo adotado para obtenção de uma memória de curto prazo (ou mecanismo para obtenção de contexto temporal). Uma das abordagens mais simples consiste na utilização de redes de propagação direta com registradores de deslocamento na entrada da rede. Porém, esse tipo de abordagem geralmente apresenta deficiências para reter, de forma flexível, informações sobre o passado pois a duração das contingências temporais que podem ser capturadas é fixada *a priori* pela arquitetura, ao invés de ser inferida, durante o treinamento, a partir dos dados de entrada[12].

Uma abordagem um pouco mais complexa consiste na utilização de redes neurais recorrentes. Vários pesquisadores têm realizado trabalhos mostrando que redes neurais supervisionadas recorrentes podem servir de aproximadores universais de sistemas dinâmicos [29],[43]. Tais redes permitem a modelagem de sistemas dinâmicos arbitrários, e podem armazenar informações contextuais de forma flexível, ou seja, por durações não predeterminadas que podem variar de uma seqüência para outra [12]. Comparando-se redes neurais recorrentes e redes de propagação direta, é correto dizer que enquanto as redes recorrentes possuem uma dinâmica mais complexa, estas geralmente são menores que as redes de propagação direta quando ambas estão sendo utilizadas para a mesma tarefa [62].

D'Arbo Junior [22], em sua dissertação de mestrado, propõe uma arquitetura de rede neural parcialmente recorrente formada por três camadas de unidades do tipo perceptron. Neste modelo, as informações de contexto temporal são processadas via conexões de realimentação intracamada (na camada de entrada) e intercadas (da camada de saída para a camada de entrada). Esse modelo foi utilizado para reproduzir trajetórias para um braço mecânico. Nos experimentos realizados por D'Arbo Junior, seu modelo apresentou uma boa capacidade para aprender trajetórias com diferentes níveis de complexidade, e que possuem pontos em comum (também conhecidos como pontos recorrentes ou pontos ambíguos). Além disto, nos casos de menor complexidade, o sistema apresentou um bom desempenho mesmo na presença de ruídos. Devido aos bons resultados apresentados, o modelo proposto por D'Arbo Junior [22] será utilizado nessa pesquisa como uma referência para comparar e avaliar o desempenho dos modelos a serem propostos nesse trabalho.

Conforme será detalhado mais adiante no Capítulo 2, o problema tratado nesta pesquisa consiste em, dado um ponto inicial e um ponto final, reproduzir a seqüência temporal aprendida

que conecta tais pontos. Com relação a este problema, uma das principais limitações apresentadas pelo modelo de D'Arbo Junior consiste na capacidade de generalização ao reproduzir seqüências em situações não treinadas, como por exemplo, quando um ponto inicial fornecido na fase de execução é, na realidade, um ponto intermediário de uma seqüência treinada. Essa é uma deficiência comum em vários modelos de redes neurais para reprodução de trajetórias, pois como será visto mais adiante, esse tipo de generalização é uma tarefa complexa na qual o grande problema consiste na falta de informações de contexto. Sendo assim, uma das metas traçadas nesta pesquisa se refere à melhoria da capacidade de generalização sem que se deteriore a capacidade do sistema em reproduzir seqüências a partir de pontos treinados. Conforme será mostrado mais adiante, essa melhoria está diretamente relacionada com a melhoria do sistema com relação à forma de tratar as informações de contexto necessárias para a reprodução das seqüências.

Por outro lado, nos últimos anos, tem-se observado um aumento significativo tanto do estudo quanto das aplicações de sistemas neurais modulares em uma grande variedade de problemas. Em parte, isto pode ser creditado à melhoria na compreensão dos sistemas biológicos, que apresentam inúmeros exemplos de modularidade [25],[26],[31]. Um outro motivo, que explica o aumento da utilização de sistemas modulares em problemas práticos, está relacionado com o enorme aumento da capacidade de memória e de processamento dos computadores nos últimos tempos. Esse aumento das capacidades computacionais propicia a implementação de sistemas cada vez maiores, capazes de lidar com o processamento de uma grande quantidade de informações, e desta forma, capazes de realizar o processamento de vários módulos ao mesmo tempo. Em algumas abordagens mais recentes, sistemas neurais modulares têm sido utilizados para aprendizagem de seqüências temporais [18],[62]. Um dos pontos interessantes em tais abordagens consiste na utilização de módulos distintos para tratarem separadamente informações espaciais e informações temporais.

Seguindo essa tendência para modularização, Srinivasa e Ahuja [62] propõem um sistema neural modular conhecido como TOTEM (*TOpological and TEMporal correlator network*). Trata-se de uma rede formada por cinco camadas sendo uma camada de entrada e quatro módulos, cada um realizando uma função específica. Assim, enquanto o primeiro módulo processa a informação espacial (através de mapas auto-organizáveis), dois módulos seguintes trabalham a informação temporal, e um último módulo realiza um agrupamento das informações que caracterizam as seqüências aprendidas. A arquitetura deste sistema permite tanto o reconhecimento de seqüências quanto a reprodução delas, e será estudada com maiores detalhes mais adiante nesta pesquisa pois este modelo serve como inspiração (no que se refere à utilização de sistemas modulares para aprendizagem de trajetórias) além de ser uma boa referência para comparação dos resultados obtidos nesta pesquisa.

A partir de estudos sobre contexto, sistemas neurais modulares e modelos estocásticos, foi desenvolvido um sistema modular no qual a tarefa de aprendizagem e reprodução de seqüências temporais foi dividida em duas subtarefas distintas sucessivas. A primeira subtarefa envolve a utilização de uma rede neural de propagação direta para obtenção de informações espaciais (contexto espacial ou global) importantes para caracterizar ou identificar uma seqüência a partir de uma quantidade reduzida de informações. A outra subtarefa é realizada por uma rede neural recorrente que utiliza a informação de contexto espacial da primeira em conjunto com informações de contexto temporal (informações a respeito de estados e/ou saídas passadas do sistema) para aprender e reproduzir trajetórias. Este modelo permitiu um aumento significativo na capacidade de generalização da rede neural recorrente, sem que houvesse perda de desempenho em situações treinadas. Além disso, em uma comparação entre o custo computacional do sistema modular proposto e o custo computacional de uma rede neural recorrente operando sozinha, percebeu-se que praticamente não houve diferença no que diz respeito ao tempo gasto para treinamento e reprodução das seqüências.

O treinamento do sistema modular proposto baseia-se em dois algoritmos bem conhecidos na literatura: a) uma variação do algoritmo LVQ (*Learning Vector Quantization*) [41], utilizado para treinar o módulo de propagação direta; e b) uma variação do algoritmo de retropropagação, utilizada para treinar o módulo de rede neural recorrente. Quanto à variação do algoritmo de aprendizagem do tipo LVQ, utilizado no módulo para processamento de informações espaciais, algumas características novas foram introduzidas para que este pudesse atender melhor às necessidades do sistema. Mais especificamente, utilizou-se uma distribuição de Gibbs na saída do módulo de forma que, associada à informação de contexto espacial, este módulo também apresenta uma probabilidade indicando o grau de certeza de suas respostas. Com relação ao algoritmo de retropropagação, este é um algoritmo que utiliza a informação do gradiente de uma função do erro no procedimento de adaptação dos pesos. Ou seja, o algoritmo de retropropagação nada mais é do que uma forma encontrada para implementar o método de otimização conhecido como método do gradiente descendente. Apesar de ser bastante utilizado na literatura, um dos principais problemas desse método consiste no seu tempo de convergência, que geralmente é elevado [33],[50],[68],[69]. Sendo assim foi feito um estudo de heurísticas e métodos de otimização utilizados visando aumentar a velocidade de convergência do algoritmo de retropropagação. Em experimentos preliminares foram comparados diferentes métodos e heurísticas que utilizam taxa de aprendizagem variável. Aquele que apresentou convergência mais rápida foi então escolhido para ser utilizado em todos os experimentos seguintes.

Nos experimentos realizados, o sistema modular proposto foi colocado para aprender a reproduzir seqüências com diferentes níveis de complexidade. Foram utilizados diferentes conjuntos de trajetórias abertas e fechadas que possuem pontos comuns em diferentes situações

(pontos ambíguos no início, no meio e/ou no final das trajetórias). O sistema proposto foi testado em dois domínios diferentes, são eles: a) conjunto de trajetórias bidimensionais geradas a partir de equações propostas por Srinivasa & Ahuja [62]; e b) conjunto de trajetórias propostas por D'Arbo Junior [22] para o problema de controle de um braço mecânico. Este último é um problema complexo com inúmeras aplicações que vão desde a linha de produção de uma fábrica de automóveis, até a construção de robôs autônomos para exploração de ambientes hostis e/ou de difícil acesso para o ser humano.

Os resultados finais obtidos (nos dois domínios) com o sistema proposto nesta pesquisa foram comparados aos resultados apresentados por Srinivasa & Ahuja [62] (que utilizaram a rede TOTEM) e aos resultados obtidos com uma rede baseada no modelo inicialmente proposto por D'Arbo Junior. Com base nessas comparações, pode-se afirmar que o sistema proposto nesta pesquisa apresentou um bom desempenho para aprender e reproduzir trajetórias com diferentes níveis de complexidade, além de uma significativa melhoria do sistema (comparado aos outros modelos) com relação à capacidade de generalização do modelo, ou seja, a capacidade de seguir uma trajetória a partir de um ponto intermediário não treinado. Essa boa capacidade de generalização foi conseguida principalmente por dois motivos: a) a abordagem modular na qual a tarefa de reprodução de trajetórias é realizada em duas etapas; e b) a forma com que o sistema trata separadamente os dois tipos de informação de contexto (o contexto espacial, e o contexto temporal).

Nesta dissertação, primeiramente será apresentado e definido o problema proposto de reprodução de seqüências temporais (Capítulo 2). Em seguida, serão apresentados estudos a respeito de contexto (Capítulo 3) e modularidade (Capítulo 4), importantes para analisar os modelos existentes para reprodução de trajetórias, assim como para formalizar e justificar os conceitos e métodos utilizados na proposição e desenvolvimento do sistema neural modular para reprodução de seqüências temporais. Três abordagens distintas, e recentemente propostas na literatura, de modelos para aprendizagem de seqüências temporais serão então estudadas (Capítulo 5), avaliando-se as principais capacidades e limitações de cada uma. Tais modelos foram importantes pois serviram de inspiração, e de referência para avaliação e comparação dos resultados obtidos. Na seqüência, o problema proposto será revisitado, e os conceitos até então estudados nesta pesquisa serão utilizados para a proposição da abordagem modular adotada para solucionar o problema proposto (Capítulo 6). Antes de se escolher a arquitetura final utilizada em cada um dos módulos do sistema, foi necessário realizar um estudo de modelos que realizam tarefas desejadas para os módulos em questão (Capítulo 6). Então, finalmente, a arquitetura do sistema neural modular assim como seu algoritmo de treinamento são apresentados (Capítulo 7). O sistema será avaliado em diferentes experimentos (Capítulo 8), dando especial atenção à capacidade de generalização do sistema, quando comparado a outros modelos. Na última fase de experimentos, o módulo para contexto espacial é avaliado operando

sozinho na tarefa de identificação de seqüências a partir de uma quantidade pequena de informações. E por último são realizados comentários finais sobre o sistema implementado e sobre os resultados obtidos, assim como discussões a respeito de trabalhos futuros que poderiam dar continuidade ao trabalho realizado (Capítulo 9).

2. REPRODUÇÃO DE SEQUÊNCIAS TEMPORAIS

Inúmeras tarefas realizadas por seres humanos e/ou por vários animais envolvem a produção ou o planejamento de seqüências de ações para se atingir um objetivo, ou para responder a uma seqüência ordenada de estímulos. Sistemas de planejamento de seqüências podem ser de grande utilidade em problemas envolvendo o controle inteligente de processos, sistemas de tomada de decisões, planejamento de trajetórias de robôs (ou braços mecânicos) em uma linha de produção, entre outros. Esse capítulo visa descrever o problema de planejamento, generalização e reprodução de seqüências temporais que será tratado nesta pesquisa. Para tanto, inicialmente serão comentados alguns conceitos e definições preliminares.

Uma seqüência temporal pode ser definida como sendo um conjunto finito de vetores n -dimensionais (também conhecidos como vetores de características, elementos da seqüência, ou padrões de entrada) ordenados no tempo [7]. Sendo assim, uma seqüência temporal S contendo $f+1$ elementos será representada da seguinte forma:

$$S = \{x(t_0), x(t_1), x(t_2), \dots, x(t_f)\} \quad (2.1.)$$

Se o intervalo de tempo entre elementos sucessivos for constante e igual a Δt (ou seja, a taxa de amostragem é constante), então S poderá ser representada por:

$$S = \{x(t), x(t+\Delta t), x(t+2\Delta t), \dots, x(t+f\Delta t)\}. \quad (2.2.)$$

Neste caso, por uma questão de simplicidade de representação, omite-se o instante de tempo t_i escrevendo:

$$S = \{x_0, x_1, x_2, \dots, x_f\}. \quad (2.3.)$$

O processamento de dados temporais é uma tarefa bastante complexa para a qual não existe uma abordagem unificada [7]. Mesmo assim, algumas propriedades comuns são geralmente consideradas [7]:

- **Ordem temporal:** se refere à posição relativa de cada elemento da seqüência com relação a outros elementos;
- **Métrica:** se refere à duração relativa de cada padrão no tempo com relação aos outros padrões da seqüência;
- **Densidade ou amostragem:** esta propriedade está relacionada com o número de padrões contidos em uma seqüência.

Definindo-se uma trajetória como sendo o conjunto de pontos sequenciais formando um caminho a ser percorrido por um sistema no decorrer do tempo, pode-se dizer então que a aprendizagem de trajetórias é uma abordagem de aprendizagem de seqüências temporais.

De acordo com Hertz et al. [34], os problemas de aprendizagem de seqüências temporais podem ser divididos em três classes: Reconhecimento de seqüências, reprodução de seqüências e associação temporal. Araújo & Barreto [3] citam também o caso de geração de seqüências temporais. No reconhecimento de seqüências, o modelo deve apresentar uma saída específica indicando que uma dada seqüência conhecida S foi apresentada. Na reprodução de seqüências, o modelo deve gerar o restante de uma seqüência conhecida quando uma parte desta for apresentada. Para a associação temporal, o modelo deve ser capaz de gerar uma seqüência específica quando uma outra seqüência (diferente da gerada) for apresentada. A geração de seqüências envolve a criação de uma trajetória a partir de algumas informações sobre esta, tais como seus estados inicial e final. Geralmente, nesta última abordagem, o sistema precisa ser capaz de recolher informações do ambiente e, de forma autônoma, construir algum tipo de mapeamento inverso.

É importante notar que a reprodução de seqüências é uma tarefa mais complexa que o reconhecimento de seqüências [66]. Isto porque para reconhecer uma seqüência, esta é fornecida inteiramente, por vezes com ruído, ao sistema, que deve ser capaz de perceber as correlações espaciais e temporais entre os padrões e a partir do reconhecimento de tais correlações, identificar a seqüência. Já na reprodução de seqüências, não se fornece a seqüência inteira, mas apenas alguns elementos (ou as vezes apenas um ponto de partida e um ponto final) diminuindo significativamente a quantidade de informações contextuais fornecidas para o sistema. Além disso, enquanto na identificação de seqüências a saída do sistema é apenas um rótulo identificador, na reprodução de seqüências a saída tem que ser um conjunto de elementos com localização espacial e ordenados no tempo (formando a seqüência em questão).

O problema a ser tratado nesta pesquisa é o de reprodução de seqüências temporais. Para esse problema, o sistema receberá em sua entrada apenas as informações do ponto inicial e do ponto alvo de uma certa seqüência conhecida que se deseja reproduzir. O modelo deverá então ser capaz de recuperar os pontos intermediários dessa seqüência (esta tarefa também é conhecida como rastreamento – *tracking*) conectando o ponto inicial ao ponto alvo.

O problema de reprodução de seqüências temporais se torna ainda mais difícil quando se deseja reproduzir uma seqüência a partir de um ponto intermediário x_i , $i \neq 0$, sendo que esta situação não havia sido anteriormente treinada. A capacidade em resolver este último caso pode ser vista como um tipo de capacidade de generalização, e é de grande relevância em problemas de reprodução de seqüências temporais pois: (a) um sistema capaz de generalizar não precisa voltar sempre para o início de uma dada trajetória a ser percorrida, e desta forma poderá atingir o ponto alvo mais rapidamente; (b) inúmeros sistemas sofrem variações dos seus estados

iniciais devido a desgastes que ocorrem naturalmente, e nesses casos, um sistema capaz de generalizar poderá continuar atuando apesar do estado inicial ter sido alterado; (c) de uma forma geral, pode-se dizer que sistemas capazes de generalizar são mais robustos com relação a ambientes e/ou sinais de entrada ruidosos.

Conforme será mostrado mais adiante, a falta de informações de contexto (temporal e espacial) são as principais responsáveis pela deterioração da capacidade de generalização. Além disso, esta limitação é observada em vários modelos, principalmente aqueles formados por uma única rede neural recorrente [11],[22]. Sendo assim, uma das principais metas traçadas nesta pesquisa consiste na obtenção de um sistema capaz de estimar as informações de contexto quando estas não estão presentes, e desta forma, o sistema proposto poderá ser capaz de obter um bom desempenho, não apenas no que se refere à reprodução de trajetórias treinadas (comparando-se com o modelo de D'Arbo Junior), mas também apresentando um desempenho significativamente melhor (devido ao melhor processamento de informações contextuais) no que se refere à capacidade de generalização.

Para atingir os objetivos propostos, foram utilizados conceitos de redes neurais recorrentes, modularidade, tratamento de contexto e modelos estocásticos. Tais conceitos serão explicados mais adiante nesse trabalho. Na seção a seguir, serão feitos comentários, definições e restrições quanto aos conjuntos de seqüências temporais utilizados nos experimentos desta pesquisa.

2.1. Comentários sobre as Seqüências Temporais a serem Aprendidas

As seqüências temporais consideradas nesta pesquisa são todas formadas por um número finito de pontos ordenados, e podem ser abertas (aquelas para as quais o ponto final se difere de qualquer outro ponto anterior na seqüência em questão) ou fechadas (aquelas para as quais o ponto final é igual ao ponto inicial). A princípio, tanto as seqüências abertas quanto as fechadas podem possuir pontos intermediários repetidos. Além disto, um conjunto de seqüências a serem aprendidas pode conter trajetórias distintas que passam por pontos comuns (pontos compartilhados). Porém, não será permitido que duas trajetórias distintas compartilhem ao mesmo tempo os pontos inicial e final.

De maneira geral, ambos os pontos (compartilhados e repetidos) também são conhecidos como pontos recorrentes, e causam ambigüidades durante a reprodução de seqüências. Tais ambigüidades aumentam significativamente a complexidade do problema, e podem ser solucionadas através do processamento de informações de contexto.

O sistema foi avaliado em dois domínios diferentes. No primeiro, foram consideradas as trajetórias propostas por D'Arbo Junior para o problema de reprodução de trajetórias para um braço mecânico. Já no segundo domínio, foi considerado um conjunto de trajetórias no espaço bidimensional. Tais trajetórias foram propostas por Srinivasa & Ahuja [62] para testar o modelo TOTEM. Sendo assim, este domínio foi escolhido para que os resultados obtidos com o modelo proposto nesta pesquisa pudessem ser comparados não apenas com os resultados obtidos com o modelo de D'Arbo Junior mas também com os resultados apresentados por Srinivasa & Ahuja [62].

Além da diferença na natureza da informação codificada, uma diferença significativa entre as seqüências do primeiro e do segundo domínio está na dimensão dos vetores (estados) que formam as trajetórias. Ou seja, enquanto as trajetórias do primeiro domínio são formadas por vetores de dimensão igual a 15 (3 entradas espaciais, 6 entradas de torques e 6 entradas de ângulos), as trajetórias do segundo domínio contém pontos definidos no espaço bidimensional. Conforme será visto mais adiante neste trabalho, devido a esta diferença, bons resultados obtidos em um domínio não necessariamente se repetirão no outro. Na realidade, será mostrado que o modelo de D'Arbo Junior (diferente do que a princípio se imaginaria) se comporta melhor para as trajetórias do primeiro domínio (com mais dimensões) do que para as trajetórias do segundo domínio. Os motivos para essa diferença de comportamento também serão comentados mais adiante nesta pesquisa.

Para o primeiro domínio, o desempenho do sistema será avaliado para diferentes situações de ambigüidades e complexidades. Com relação às trajetórias abertas, as seguintes situações poderão ocorrer no conjunto de treinamento: a) trajetórias que não possuem pontos compartilhados (Figura 1a); b) trajetórias que possuem estados finais comuns (Figura 1b); c) trajetórias que partem do mesmo estado inicial (Figura 1c); e d) trajetórias que compartilham estados intermediários (Figura 1d). Já com relação às seqüências fechadas, serão utilizadas seqüências sem pontos repetidos como por exemplo a seqüência apresentada na Figura 2a. Ou seqüências com pontos repetidos, como por exemplo a seqüência apresentada na Figura 2b. Os resultados finais obtidos, para o primeiro domínio, com o sistema proposto nesta pesquisa serão comparados aos resultados obtidos com uma rede neural recorrente baseada no modelo proposto por D'Arbo Junior [22]. Conforme já mencionado anteriormente, a principal deficiência deste último modelo (e de inúmeros outros modelos formados por uma única rede neural recorrente) consiste na falta de informações de contexto, quando colocado para processar trajetórias em situações não treinadas, deteriorando a capacidade de generalização do sistema. Um estudo mais detalhado do modelo assim como de suas deficiências será realizado no Capítulo 5.

Figura 1: Quatro situações distintas de trajetórias abertas que podem ocorrer no conjunto de treinamento: (a) trajetórias que não possuem estados comuns; (b) trajetórias que possuem estados finais comuns; (c) trajetórias que partem do mesmo estado inicial; (d) trajetórias que se cruzam (possuem estados intermediários comuns).

Figura 2: Exemplos de trajetórias fechadas para avaliação do modelo: (a) trajetórias fechadas sem pontos repetidos; (b) trajetórias fechadas com pontos repetidos.

O conjunto de trajetórias do segundo domínio possui as seguintes características: a) contém tanto trajetórias abertas quanto trajetórias fechadas; b) as trajetórias apresentam diferentes níveis de complexidade (pontos não ambíguos, pontos de pequena ambigüidade, e pontos altamente ambíguos); e c) diferentes trajetórias podem possuir pontos compartilhados.

Na próxima seção, serão comentados alguns conceitos relevantes para o problema do primeiro domínio (reprodução de trajetórias para controle de um braço mecânico).

2.2. O Planejamento de Trajetórias para o Braço Mecânico

O planejamento de trajetórias em sistemas não lineares é um problema complexo e de grande importância em inúmeras áreas da engenharia, principalmente na área de controle. O planejamento e controle de um braço mecânico é um problema que envolve três tarefas: planejamento de trajetória, a cinemática inversa, e a dinâmica inversa.

Um problema é dito *bem colocado* quando sua solução existe, é única e depende continuamente dos dados iniciais. Se qualquer um desses critérios falharem, então o sistema é dito *mal colocado* [22],[40].

Com relação ao controle de um braço mecânico manipulador, a cinemática é um estudo analítico da geometria do movimento, em função do tempo, do braço do robô com relação a um sistema de coordenadas fixas [21],[22]. Em um problema de cinemática direta, deseja-se obter a posição e a orientação do efetuador com respeito a um sistema de coordenadas, dados os ângulos das juntas do manipulador e os parâmetros geométricos dos elos. Já na cinemática inversa são dadas a orientação e a posição desejadas do efetuador, juntamente com os parâmetros geométricos dos elos com relação a um sistema de coordenadas e deseja-se obter os ângulos das juntas que atendam à posição especificada. Nesse caso, o problema será mal colocado se o manipulador for redundante, isto é, se possuir mais graus de liberdade do que o necessário para a execução da tarefa, e desta forma, existirem para uma mesma posição espacial, mais de uma combinação de ângulos das juntas.

A dinâmica dos manipuladores estuda o caminho pelo qual o manipulador se move uma vez aplicados forças externas ou torques através de atuadores [21],[22]. A dinâmica direta tem como objetivo descrever o comportamento dinâmico do sistema, uma vez que foram fornecidas as forças e momentos em função do tempo. Por outro lado, a dinâmica inversa tem como objetivo transformar ângulos e comprimento das juntas em ações de controle, calculando o torque, nas juntas, requeridos para um dado conjunto de ângulos, velocidade e aceleração dessas juntas. Novamente temos um caso de problema mal colocado pois existe um número infinito de formas de onda para o torque no atuador da junta, que alcança o ângulo desejado no mesmo período de tempo.

A reprodução de seqüências com pontos recorrentes entre outras resoluções de problemas mal colocados, são problemas que contêm ambigüidades nas soluções. Para resolver tais ambigüidades, pode-se utilizar sistemas capazes de estimar e/ou considerar informações de contexto. Por exemplo, se durante a reprodução de uma seqüência, o sistema atingir um ponto compartilhado por duas ou mais trajetórias distintas, então o conhecimento a respeito de estados passados (já percorridos pelo sistema) pode conter informações de contexto temporal suficientes para caracterizar a trajetória que está sendo seguida, e consequentemente reduzir a possibilidade do sistema seguir um caminho errado (ou não desejado).

No próximo capítulo serão expostos definições e conceitos básicos a respeito de contexto, e de modelos de redes neurais capazes de considerar informações de contexto.

3. CONTEXTO E REDES NEURAIS PARA CONTEXTO

O estudo dos diferentes tipos de informações de contexto, assim como o estudo da representação, aquisição, e processamento de tais informações utilizando-se redes neurais serão essenciais para a proposição do sistema neural modular para reprodução de trajetórias. Sendo assim, neste capítulo serão vistos definições e conceitos sobre contexto, e sobre redes neurais que trabalham com informações de contexto.

De acordo com o dicionário da língua portuguesa [42], o significado da palavra contexto pode ser definido como sendo: (a) conjunto do texto que precede ou sucede uma frase, um grupo de palavras, uma palavra; ou (b) conjunto de circunstâncias que acompanham um acontecimento.

Para o estudo realizado nesta pesquisa, a definição (b), é mais apropriada que a definição (a). Porém, em inteligência artificial, e particularmente, em estudos de redes neurais, torna-se necessária uma definição mais específica para o conceito de contexto. Para tal, serão considerados dois tipos de informações de contexto: o contexto temporal e o contexto espacial. O contexto temporal pode ser definido como um conjunto de informações a respeito de entradas, saídas e/ou estados passados de um sistema [3],[6],[10]. Ou seja, é um conjunto de informações a respeito de um histórico do sistema. Sendo assim, imaginando-se que no decorrer do tempo um dado sistema percorra um caminho de estados, as informações de contexto temporal podem ser úteis para que o sistema “saiba” a sua localização (no instante atual) no espaço de estados, e qual deverá ser o próximo estado a ser percorrido. Já o contexto espacial é um conjunto de informações sobre a posição espacial de um ou mais padrões com relação a outros padrões [3],[6]. Em outras palavras, o contexto espacial contém informações que identificam ou localizam um padrão (ou um conjunto de padrões) no espaço de padrões. Alguns exemplos de diferentes estudos e abordagens a respeito de contexto são dados a seguir:

- **Processamento de linguagem natural:** O contexto (temporal e espacial) é crucial na comunicação pois pode eliminar ambigüidades e/ou múltiplos sentidos da informação, além disto, ajuda a determinar dados ou parâmetros que estejam faltando em uma mensagem [1],[2].
- **Contexto Espacial na Categorização:** A categorização é um processo não supervisionado no qual se deseja separar padrões de entrada em diferentes categorias.

Este processo depende da perspectiva adotada e da posição de cada padrão com relação aos outros padrões, ou seja, depende das informações de *contexto espacial*. Em outras palavras, o contexto espacial de um padrão influencia a forma de interpretar e perceber este padrão com relação aos outros padrões [2].

- **Contexto temporal no planejamento de ações:** Inúmeros sistemas exigem que diferentes ações sejam realizadas dependendo da situação (estado atual do sistema) e da sequência de ações passadas (histórico dos estados do sistema) [7],[35]. O contexto temporal pode conter informações a respeito de dependências temporais existentes entre os estados passados. Sendo assim, é de fundamental importância que um sistema para planejamento de ações (ou de sequências de ações) seja capaz de armazenar e utilizar adequadamente as informações do contexto temporal.

O planejamento ou reprodução de sequências temporais pode ser entendido como um caso particular do problema de planejamento de ações. Nesse caso, uma ação significa caminhar para o próximo ponto da sequência. Para seguir corretamente uma trajetória, o sistema deve ser capaz de armazenar informações de contexto temporal e utilizar adequadamente tais informações. Da mesma forma como foi dito para o problema de processamento de linguagem natural, o contexto na reprodução de sequências temporais é crucial para solucionar pontos ambíguos (recorrentes em uma trajetória ou cruzamento de trajetórias) ou múltiplos caminhos a serem seguidos.

A idéia de contexto temporal está intimamente ligado ao conceito de memória de curto prazo (STM – *Short Term Memory*), pois a memória de curto prazo em um sistema tem como função armazenar as informações temporais (o passado próximo do sistema) e conseqüentemente, armazenar informações de contexto temporal. Qualquer sistema cuja resposta é obtida não apenas em função da entrada e do estado atual, mas também em função de entradas e/ou estados passados é um sistema de comportamento dinâmico contendo algum mecanismo de memória de curto prazo, e conseqüentemente, é um sistema que de alguma forma trabalha com contexto temporal. Na próxima seção serão estudados alguns conceitos básicos a respeito de redes neurais artificiais, assim como diferentes abordagens e modelos de redes neurais capazes de considerar informações de contexto temporal.

3.1. Redes Neurais que Incluem Processamento de Contexto Temporal

Na última década, tem-se percebido um interesse crescente por parte dos pesquisadores em desenvolver métodos que permitam a introdução de dinâmicas temporais em redes neurais.

A principal diferença entre esses métodos consiste na forma como uma memória de curto prazo é considerada no modelo. Os principais métodos para a obtenção de redes neurais que possuem um comportamento dinâmico consistem em [14]: (a) inserir uma memória externa (*buffer*) em algum lugar (geralmente na entrada) da rede neural estática; (b) utilizar conexões de realimentação. Nos dois casos, deseja-se que uma entrada qualquer $x(t)$ possa influenciar uma saída futura $y(t+h)$, de forma que $\partial y(t+h)/\partial x(t)$ seja diferente de zero para algum h . No caso de uma rede assintoticamente estável, esta derivada vai para zero a medida que h vai para o infinito. O valor de h a partir do qual a derivada apresentada passa a ser desprezível é conhecido como *profundidade temporal* [14],[24]. Em outras palavras, valores altos para a profundidade temporal implicam em dependências temporais de longo prazo (um estado presente depende de estados do passado distante).

A forma mais simples de se obter uma rede neural com comportamento dinâmico consiste na inserção de uma memória externa em uma rede perceptron multicamadas (MLP) treinada com algoritmo de retropropagação. A implementação desta memória pode ser feita de diversas maneiras. Uma possibilidade consiste na transformação de padrões temporais em padrões espaciais. Nesse caso o padrão a ser apresentado na entrada da rede é formado por uma linha de atrasos contendo $n+1$ elementos da sequência como mostra a Figura 3. Sendo assim, o valor de n está diretamente relacionado com o valor de profundidade temporal comentado anteriormente. Ou seja, para que o sistema possa trabalhar com padrões que contenham dependências de longo prazo, será preciso escolher valores altos para n .

Figura 3: Mecanismo externo de memória de curto prazo no qual um padrão temporal é transformado em um padrão espacial através de uma linha de atrasos na entrada.

A rede utilizada nesse caso é uma rede estática de propagação direta. Geralmente utiliza-se um algoritmo clássico de retropropagação. As saídas deste modelo são tipicamente *pontos fixos* no espaço de características, e dessa forma, podem ser bem utilizadas para a classificação de seqüências [62]. A principal desvantagem desta abordagem consiste no fato da janela conter um número fixo de elementos, acarretando em problemas para aprender padrões com dependências temporais de longo prazo em conjunto com padrões com dependências

temporais de curto prazo. Uma outra dificuldade relacionada ao problema de dependências temporais consiste em encontrar o tamanho ideal de janela para cada aplicação. Tais modelos não possuem a flexibilidade para variar a quantidade de informações passadas que precisam ser levadas em conta durante o treinamento. Para esse tipo de abordagem, quanto maior for a profundidade temporal desejada, maior será a janela (valor n da quantidade de instantes de tempo a serem considerados na entrada da rede) necessária, maior será a quantidade de pesos na entrada (aumentando o custo computacional) e provavelmente menor será a capacidade de generalização do modelo [14].

Já as redes recorrentes, possuem realimentações (mecanismo interno de memória de curto prazo) que possibilitam comportamentos dinâmicos (que não são possíveis em redes de propagação direta) tais como saídas cíclicas (ciclo limite) e caos [62]. Uma vantagem das redes recorrentes com relação às redes de propagação direta é que, geralmente, para a mesma tarefa de reconhecimento de seqüências, são necessárias redes recorrentes bem menores do que redes de propagação direta.

As redes neurais recorrentes podem ser totalmente recorrentes ou parcialmente recorrentes [34]. O primeiro caso é formado por uma ou mais camadas nas quais cada unidade se conecta a todas as outras. Porém, esse tipo de rede possui desvantagens bem conhecidas [14] tais como: estrutura complexa com grande quantidade de conexões entre unidades, e treinamento complexo e lento. De fato, tais arquiteturas possibilitam a modelagem de diferentes sistemas dinâmicos, porém, em inúmeros problemas específicos, é possível que redes neurais com estruturas mais simples e treinamento mais rápido possam ser utilizadas no lugar de redes totalmente conectadas.

Redes neurais parcialmente recorrentes são arquiteturas contendo conexões de alimentação direta e conexões de realimentação. Existem arquiteturas onde as realimentações são modeladas internamente nos neurônios. Tais arquiteturas formam uma classe especial de redes parcialmente recorrentes, são as redes localmente recorrentes [14],[28],[52],[63],[64]. Existem basicamente três formas de realimentação local:

- **Realimentação da soma ponderada:** nesse caso, a realimentação pode ser versões passadas da soma ponderada das entradas, e essas podem estar combinadas com versões passadas das variáveis de entrada.
- **Realimentação de sinapse:** nesse caso, cada conexão possui uma realimentação em sua sinapse. Alguns autores propõem estruturas de realimentação na forma de filtros de resposta de impulso finito ou infinito.
- **Realimentação de saída:** nesse caso, a saída anterior de uma unidade é fornecida à sua entrada.

Uma explicação mais detalhada a respeito de redes com realimentações locais, assim como variações de cada caso podem ser vistas em [14],[28],[64]. Geralmente, redes localmente recorrentes são adequadas para a implementação de memórias mais curtas, podem apresentar um desempenho aceitável na tarefa de reconhecimento de seqüências temporais, porém possuem limitações quando trabalhando com a reprodução de seqüências temporais ou com o reconhecimento de seqüências complexas em geral [12].

Entre as redes parcialmente recorrentes, um dos trabalhos primeiros é o de Elman [27]. Ele propôs uma rede neural (Figura 4) capaz de considerar o contexto temporal para a classificação de seqüências ou até mesmo para a reprodução de pequenas seqüências. Esta é uma arquitetura com três camadas, nas quais, as conexões de alimentação direta são treináveis, enquanto que as conexões de realimentação (da camada escondida para as unidades de contexto) são do tipo um-para-um, fixas e com valores iguais a 1. Perceba que nesse modelo, o estado da camada escondida é copiado nas unidades de contexto, e em seguida, realimentado na camada escondida. Dessa forma, as unidades de contexto fornecem o mecanismo de memória de curto prazo para a rede, que passa a ter um comportamento dinâmico, levando-se em conta dependências temporais dos padrões de entrada. De acordo com Kremer [44], o poder computacional desse modelo é tão grande quanto o de máquinas de estados finitos.

Figura 4: Arquitetura proposta por Elman na qual o estado (contexto temporal) das unidades escondidas é copiado nas unidades de contexto da camada de entrada e em seguida realimentado na camada escondida. As setas largas indicam conexões todos-para-todos, enquanto a seta fina na realimentação indica conexões um-para-um.

Inúmeras variações deste modelo têm sido estudadas por vários autores na literatura [22],[33],[34],[44]. É comum utilizar variações do algoritmo de retropropagação do erro para treinar modelos de redes neurais recorrentes [22],[33],[34],[55].

De acordo com Bengio et. al [11], qualquer sistema dinâmico paramétrico com uma recorrência não linear (como o que ocorre em redes neurais recorrentes) terá maior dificuldade

de treinamento com gradiente descendente, quanto maior for a duração das dependências temporais a serem aprendidas. Esse é um problema com o gradiente da função erro, e sendo assim, persiste independentemente do tipo de algoritmo de computação do gradiente (RTRL, BPTT, entre outros) que tenha sido utilizado [11],[12].

Lin et al. [49] mostram que aumentando a ordem da memória de uma rede recorrente é possível melhorar sua performance com relação às dependências de longo prazo. A ordem da memória está relacionada com a quantidade de atrasos que foram considerados no modelo. Alguns exemplos são ilustrados na Figura 5. A Figura 5a mostra um exemplo genérico de rede neural com realimentação na camada de entrada e com uma memória de ordem 1. A Figura 5b mostra a mesma rede da Figura 5a, porém acrescida de realimentações com um atraso maior, e dessa forma, caracterizando uma rede com memória de ordem 2. O mesmo acontece nas Figuras 5c e Figura 5d que ilustram respectivamente uma rede localmente recorrente com memória de ordem 1, e a mesma rede localmente recorrente, que acrescida de determinadas realimentações e atrasos, passa a ter uma memória de ordem 2. Porém, da mesma forma como ocorre com a rede de propagação direta da Figura 3 é de se esperar que, quanto maior a ordem da memória de um determinado modelo, maior será o custo computacional e pior será a capacidade generalização do modelo.

Figura 5: a) rede recorrente com memória de ordem 1; b) rede recorrente com memória de ordem 2; c) rede localmente recorrente com memória de ordem 1; d) rede localmente recorrente com memória de ordem 2.

Conforme será visto no Capítulo 6, o sistema proposto nesta pesquisa utiliza uma abordagem modular para processar separadamente informações de contexto espacial e temporal. Sendo assim, o próximo capítulo será dedicado ao estudo de Sistemas Neurais Modulares.

4. SISTEMAS NEURAIS MODULARES

Vários motivos existem para a utilização de uma arquitetura modular, entre outros podemos destacar: melhoria da capacidade de generalização; aumento da velocidade de aprendizagem e da eficiência; redução da complexidade do modelo, tornando o sistema mais facilmente compreendido e/ou modificável, e consideração de plausibilidade neurobiológica e/ou psicológica, pois acredita-se que seja sensato supor que em vários aspectos, o processamento de informações no cérebro envolve modularidade.

Catfolis & Meert [18] descrevem a modularização como sendo uma técnica para transformar tarefas complexas em pequenas subtarefas relacionadas e mais facilmente gerenciáveis. Tal estratégia permite reutilização de módulos em diferentes problemas, ou a alteração de uma determinada sub tarefa (ou módulo) sem a necessidade de alterar as demais subtarefas (ou módulos) do sistema.

Para Sharkey [61], os sistemas modulares podem ser caracterizados de acordo com: (a) as razões para decomposição de tarefas (a exploração específica de capacidades especialistas de módulos individuais, a plausibilidade biológica ou psicológica, a melhoria da performance); (b) o método utilizado para a decomposição de tarefas (decomposição automática, baseada na aplicação cega de algoritmos de partição de dados, e decomposição explícita, baseada em conhecimento a priori das tarefas e capacidades especialistas dos módulos); e (c) a forma com que os módulos se relacionam (sucessiva, cooperativa ou supervisória). Cada um desses itens serão explicados detalhadamente mais adiante neste capítulo. Na próxima seção serão descritas motivações justificando a utilização de sistemas modulares.

4.1. Motivações para a Utilização de Sistemas Modulares

Inicialmente serão comentadas motivações biológicas para a utilização de sistemas modulares. As redes neurais biológicas são grandes responsáveis por inspirações e motivações no desenvolvimento de redes neurais artificiais. Para muitos biólogos, o estudo de redes neurais artificiais pode ser um importante método que ajudará na compreensão dos sistemas neurais biológicos. Porém, muitas suposições em redes neurais artificiais a respeito do padrão de

conectividade, sinapses, ativação, entre outras, não são plausíveis em sistemas biológicos. A plausibilidade de uma rede neural deveria ser considerada de acordo com o objetivo desejado [5], que pode ser:

- (a) uma engenharia reversa do sistema biológico cujo propósito seria a compreensão de certas complexidades deste sistema.
- (b) utilização de modelos de redes neurais para solucionar problemas computacionais. Nesse caso, os modelos não precisam ser totalmente plausíveis do ponto de vista biológico.

Com relação ao projeto de redes neurais modulares, algumas idéias podem ser extraídas das redes neurais biológicas, são elas:

- **Modularidade e especialização funcional:** O cérebro é composto de várias áreas distinguíveis por suas anatomias e funcionalidades. Inúmeras pesquisas buscam a compreensão do funcionamento de determinadas estruturas em áreas distintas, e da interação entre diferentes estruturas cerebrais [26],[31]. Vários estudos sugerem que o cérebro seja formado por inúmeros módulos de processamento especializados e responsáveis em realizar determinadas tarefas. Porém, isto não significa que tais módulos sejam independentes, pois se assim fosse, cada módulo seria capaz de computar a sua tarefa específica independentemente de outros módulos, o que não é verdade [31]. Por exemplo, o córtex visual processa diferentes atributos tais como forma, cor e movimento, em regiões “anatomicamente distintas”. Porém, mudanças na percepção de forma e cor causam mudanças na percepção de movimento e vice-versa, e ainda, mudanças na percepção de brilho (intensidade luminosa) provocam mudanças na percepção de profundidade e vice-versa [31].
- **Competição/Cooperação entre módulos nos níveis micro e macrobiológicos:** As vezes, grupos de células estão sujeitos a uma exclusão competitiva. Módulos do córtex visual cooperam via certos padrões de conectividade para formar a visão global de objetos [5],[31]. A macrobiologia também fornece idéias interessantes a esse respeito. A cooperação entre trabalhadores no reino da formigas resulta em capacidades que vão muito além das capacidades de uma única formiga, ou mesmo da soma simples de todas elas. Essas observações motivam o projeto de um esquema de cooperação entre módulos que possa resultar em um desempenho superior à simples soma das capacidades individuais dos módulos [5].

- **Extensibilidade:** No córtex cerebral observa-se uma organização modular com estruturas formadas por colunas de neurônios [26]. Além disto, observa-se que cada coluna contém um número limitado de neurônios (geralmente, não mais que 100). Estudos a respeito de tais estruturas sugerem que o cérebro evita possíveis problemas de convergência mantendo as colunas relativamente pequenas. Além disso, novos módulos (estruturas colunares) poderiam ser adicionados (ou utilizados) sem que haja uma mudança significativa no comportamento dos outros módulos. Esta característica é uma motivação visando o uso de arquiteturas modulares nas quais a adição de novos módulos permita uma extensibilidade “ilimitada”.

Além das motivações biológicas, existem também motivações psicológicas para a utilização de sistemas neurais modulares. Algumas dessas motivações psicológicas são descritas a seguir:

- **Aprendizagem em estágios:** Existem várias evidências de que a aprendizagem nos seres humanos não é apenas uma aprendizagem em uma etapa (*one-pass-learning*). Ela pode ser dividida em estágios não interferentes distintos referentes à inteligência sensoriomotora, inteligência imaginativa, e operações formais [5]. Tais observações a respeito da aprendizagem nos seres humanos motivam o projeto de redes neurais modulares (RNMs) com vários estágios de aprendizagem.
- **Mistura de aprendizagens supervisionada e não supervisionadas:** Nos seres humanos, a aprendizagem supervisionada utiliza conceitos adquiridos pela aprendizagem não supervisionada para estabelecer associações desejadas. Esta idéia é aplicada em RNMs utilizando-se um pré-estágio não supervisionado de extração de características ou de quantização vetorial para apoiar um módulo de aprendizagem supervisionada [5],[26]. Pode-se utilizar módulos supervisionados para simulação dos efeitos do meio ambiente, e módulos não supervisionados para a simulação dos efeitos da auto-aprendizagem. Evidências anatômicas, fisiológicas e teóricas apoiam a hipótese de que o cerebelo é uma estrutura especializada para a aprendizagem supervisionada, os gânglios basais para a aprendizagem por reforço, e o córtex cerebral para a aprendizagem não supervisionada [26].
- **Ausência de interferência retroativa:** A interferência retroativa faz com que a aprendizagem de novos padrões de treinamento interfira no conhecimento já aprendido, decorrente de padrões anteriores [5],[53]. Para resolver este problema, algumas redes neurais modulares são capazes de desenvolver uma representação

interna capaz de discriminar padrões de treinamento subsequentes, desde que estes sejam suficientemente diferentes [32].

- **Decomposição de tarefas:** Os sistemas de processamento de informações nos seres humanos às vezes decompõem uma tarefa complexa em partes de tamanhos viáveis para conseguir resolvê-las. Esta também é uma motivação para a utilização de aprendizagem modular.

A seguir serão comentadas algumas motivações computacionais para a utilização de sistemas neurais modulares.

- **Complexidade da aprendizagem:** No treinamento de uma rede neural MLP treinada por um algoritmo de retropropagação, inicialmente, aprende-se o conhecimento decorrente dos padrões mais fáceis (não sobrepostos, facilmente separáveis). Ao fazer isto, a rede neural dedica boa parte de suas unidades escondidas para mapear tais padrões. Na sequência, para aprender os mapas mais difíceis (por exemplo, os que envolvem padrões sobrepostos), não haverá unidades escondidas suficientes para definir precisamente as regiões de fronteira. Dessa forma, se o procedimento de aprendizagem prosseguir visando obter uma performance satisfatória para padrões difíceis, a aprendizagem inicial (dos padrões mais fáceis) será prejudicada devido à construção de mapas decorrentes dos padrões difíceis. Em outras palavras, a medida em que o treinamento prossegue, a aprendizagem dos padrões mais fáceis começa a se degradar antes mesmo da rede conseguir aprender os mais difíceis. Este problema se deve ao acoplamento² das unidades escondidas [5], que é um sintoma típico de redes neurais não modulares. Uma solução para este problema seria utilizar valores mais elevados de taxa de aprendizagem para os padrões mais complexos. Porém, esta não é uma boa solução devido ao fenômeno chamado “interferência catastrófica” [53], ou seja, a aprendizagem de um dado padrão mais complexo certamente seria beneficiada, mas ao mesmo tempo, a informação que a rede já aprendeu se degradaria de tal forma que prejudicaria a performance global do processo. No caso da utilização de estruturas modulares, a tarefa poderia ser dividida entre os módulos, diminuindo o efeito de acoplamento. Além disso, cada módulo teria que realizar uma tarefa menos complexa, e o desempenho final do sistema poderia ser melhorado [5].

² Por acoplamento de unidades entende-se que mais de uma unidade está sendo utilizada pela rede neural para responder a uma mesma informação. Isto não é necessariamente indesejável, pois faz parte do conceito de processamento distribuído das informações, que é uma propriedade natural em redes neurais. Porém, conforme o exemplo dado no texto, às vezes esse acoplamento pode comprometer a aprendizagem.

- **Tamanho da Rede Neural:** Um problema complexo em redes perceptron multicamadas consiste na determinação do número de unidades escondidas a ser utilizado [37]. Não existe um critério geral para definir qual o tamanho (número de unidades escondidas) mais adequado para a rede [5]. Os métodos utilizados para resolver este problema geralmente introduzem muita complexidade ao procedimento de treinamento, e mesmo assim, ainda não são adequados como um método geral. Porém, sabe-se que a sensibilidade com relação a tamanhos impróprios de rede se reduz à medida que o tamanho dos módulos de aprendizagem diminui [37]. Pode-se dizer que esta é uma grande vantagem que uma rede modular pode oferecer.
- **Generalização:** É uma medida para avaliar o desempenho da rede neural depois de encerrado o treinamento. Geralmente, esta medida é realizada avaliando-se o desempenho da rede ao ser executada utilizando-se dados de entrada não vistos durante o treinamento. O número de padrões utilizados no treinamento é importante para que se garanta boa capacidade de generalização. Alguns autores costumam relacionar este número com o tamanho da rede neural. Mas, além do problema da quantidade de padrões a ser utilizada no treinamento, existe também a qualidade desses padrões e a posição destes no espaço com relação às bordas de diferentes decisões. De qualquer forma, decompondo a tarefa objetivo em pequenos módulos de menor complexidade implica em uma diminuição da taxa de conexões por nó, e conseqüentemente, dos limites práticos e teóricos a respeito da quantidade de padrões de treinamento. Isto é particularmente importante quando são tratados problemas nos quais o número de padrões disponíveis para o treinamento é bastante limitado [5].
- **Velocidade de treinamento:** Geralmente a velocidade de treinamento diminui quanto mais acurado for o desenvolvimento do procedimento de aprendizagem [37]. Técnicas utilizadas para aumentar a velocidade de treinamento costumam introduzir parâmetros adicionais que aumentam a complexidade do problema, e quando não estão bem ajustados, podem acarretar em diminuição da velocidade de convergência [5]. A necessidade de treinamentos em alta velocidade motiva a utilização de redes neurais modulares contendo pequenos módulos (treinados em paralelo) com uma boa velocidade de convergência. Uma estrutura modular também melhora a velocidade devido à redução do efeito das “informações conflitantes de treinamento” (ou comunicação cruzada).

As redes neurais MLPs sofrem de dois tipos de comunicação cruzada, a espacial e a temporal [5]. A comunicação cruzada espacial ocorre quando as saídas da rede provêm

informações de erros conflitantes para uma única unidade escondida em uma única interação. A Figura 6 mostra dois exemplos de rede MLP. A rede da Figura 6a é mais susceptível à comunicação cruzada espacial do que a rede da Figura 6b. Isto porque para a rede da Figura 6a, durante cada iteração no treinamento, cada uma das unidades escondidas recebe informações de erro de todas as unidades de saída. Sendo assim, se o erro para uma unidade de saída é positivo, e para outra unidade de saída é negativo, então estas duas informações conflitantes de erro serão retropropagadas em conjunto para todas as unidades escondidas, o que pode acarretar em um treinamento mais demorado. Já na rede da Figura 6b, não ocorre a comunicação cruzada espacial porque unidades diferentes na saída propagam erros para unidades diferentes na camada escondida. Neste caso, o treinamento será mais rápido não apenas devido à eliminação da comunicação cruzada espacial, mas também devido à redução da quantidade de conexões do modelo. Esse tipo de solução pode ser visto como uma abordagem modular na qual a saída do sistema é vista como sendo formada por vários módulos indicados pelas linhas tracejadas. Vale lembrar que se as saídas são altamente correlacionadas, pode acontecer uma perda no desempenho da rede em troca do ganho de velocidade.

Figura 6: A rede neural do exemplo (a) é mais susceptível à comunicação cruzada espacial do que a rede neural do exemplo (b).

Por outro lado, a comunicação cruzada temporal ocorre quando uma rede neural recebe informações conflitantes de treinamento no decorrer do tempo (ou das iterações). Geralmente isto ocorre quando o conjunto de treinamento contém padrões com informações conflitantes.

Normalmente, quando funções dissimilares são aprendidas por diferentes módulos, menores e mais simples que uma rede não modular, a velocidade de aprendizagem é melhorada. Ao modularizar uma estrutura não recorrente de uma rede de retropropagação, a velocidade de aprendizagem aumenta às custas de uma queda na precisão.

Auda & Kamel [5] sugerem três estágios comuns no projeto de redes neurais modulares: decomposição de tarefas, treinamento e tomada de decisão multi-módulo. Tais estágios serão explicados nas seções a seguir.

4.2. Decomposição de Tarefas

Existem diferentes formas de se decompor um problema. Em particular, a decomposição pode ser *explícita* ou *automática*.

- **Decomposição Explícita:** Depende do grau de compreensão da tarefa em questão e das capacidades dos componentes modulares. Provê uma forma de incorporação de conhecimento a priori e de compreensão da tarefa em questão. Por exemplo, uma decomposição pode ser imposta pela estrutura da tarefa quando trabalhando-se com dados de diferentes fontes e/ou formas. A decomposição também pode ser feita baseando-se em teorias ou evidências a respeito das estruturas contidas no cérebro humano, ou de seu sistema de processamento de informações.
- **Decomposição automática:** É a decomposição através de uma aplicação cega de algum algoritmo de partição de dados. É particularmente útil quando não se tem um conhecimento profundo a respeito da tarefa a ser realizada. Por exemplo temos os métodos: *Mistura de Especialistas* [38] e *Mistura Hierárquica de Especialistas* [39] que são abordagens nas quais os dados de entradas são particionados em vários subespaços, e módulos simples são treinados para um ajuste local dos dados. Tal partição de dados é, geralmente, mais efetiva do que o treinamento utilizando-se todos os dados do espaço de entrada.

As formas com que os módulos se relacionam podem ser caracterizadas como sendo *sucessiva*, *cooperativa* ou *supervisória* [5],[61]:

- **Sucessiva:** Envolve a decomposição de uma tarefa global em subtarefas sucessivas, cada uma das quais, sendo executada por um módulo especialista.
- **Cooperativa:** Neste caso, o processamento deixa de ser sequencial como no caso anterior. Os dados de entrada poderiam ser entregues a vários módulos especialistas, e a saída do sistema poderia ser dada pela saída de um determinado módulo selecionado, ou pela composição da saída de vários módulos. De forma alternativa, algum mecanismo poderia ser utilizado para selecionar qual o módulo especialista mais indicado para processar um conjunto particular de dados de entrada.
- **Supervisória:** Uma rede neural pode ser treinada para selecionar parâmetros de uma segunda rede, com base nas observações do efeito de vários valores de parâmetros na performance da rede. Uma outra forma de relacionamento supervisório consiste no treinamento de uma determinada rede para prever

possíveis erros em uma outra rede neural principal, tendo como base as entradas, saídas, e/ou estados desta última.

Em alguns casos, os módulos de redes neurais irão realizar tarefas naturalmente separadas (diferentes), o que sugere que cada módulo irá realizar uma dessas tarefas. No entanto, normalmente são necessárias técnicas de decomposição de tarefas mais sistemáticas. Dois métodos mais sistemáticos para a modularização são: a aprendizagem modularizada e a estrutura modularizada.

A modularização da aprendizagem implica na aprendizagem em estágios (seja para uma rede neural modular ou não modular) [5]. Além de apresentar mais facilidades para a avaliação e correção, a modularização também torna a representação dos dados de entrada mais facilmente compreendida. Algumas abordagens de aprendizagem modularizada serão descritas a seguir [5]:

- ***Melhorando a aprendizagem supervisionada pela extração de características:*** Introduz-se um estágio de aprendizagem não supervisionada antes do estágio de aprendizagem supervisionada com o objetivo de obter uma representação dos dados mais adequada ao problema. Com a extração de características não supervisionada, é possível reduzir a dimensionalidade dos dados de entrada, e consequentemente diminuir os custos computacionais devido ao menor número de pesos na aprendizagem supervisionada.
- ***Melhorando a aprendizagem supervisionada através da quantização:*** Ao invés de reduzir o tamanho da tarefa de aprendizagem supervisionada, esta técnica objetiva diminuir a complexidade, por exemplo, utilizando um módulo de quantização para transformar uma tarefa de classificação de padrões não linearmente separáveis em uma tarefa de classificação de padrões linearmente separáveis. Dessa forma, um classificador supervisionado de apenas uma camada poderá ser utilizado.
- ***Decompondo o conjunto de dados de aprendizagem:*** Os dados serão decompostos em conjuntos de difícil e fácil aprendizagem de acordo com o erro obtido no treinamento. Após a aprendizagem dos padrões fáceis, estes são removidos do treinamento, e o algoritmo é aplicado iterativamente.

No caso da estrutura modularizada, a tarefa é decomposta de tal forma que seja distribuída em vários módulos separados estruturalmente. Na realidade, existem inúmeras possibilidades para se obter uma estrutura modularizada. Uma possibilidade seria a utilização

de módulos competitivos executando (em paralelo) a mesma tarefa, de tal forma que, no final, algum mecanismo de tomada de decisão escolha qual é a melhor resposta. Uma outra possibilidade é utilizar módulos cooperativos que, assim como os competitivos, trabalham em paralelo na mesma tarefa, a diferença é que a resposta final será dada pela média (ou por algum tipo de composição) das respostas dos módulos. Também é possível quebrar o problema em diferentes tarefas, e utilizar módulos especialistas para cada tarefa. A dificuldade nesse caso é que, geralmente, este tipo de modularização exige um bom conhecimento do problema para se obter uma divisão adequada de tarefas. Enfim, essas são apenas algumas das inúmeras possibilidades para se obter uma estrutura modularizada.

4.3. Treinamento dos Módulos

As estruturas modulares geralmente treinam seus módulos de forma independente um dos outros. Desta forma, as equações de aprendizagem são completamente desacopladas, possibilitando uma convergência mais rápida decorrente da utilização de paralelismo [8],[9].

As subtarefas (e as vezes os módulos) podem ser definidas de forma adaptativa durante o treinamento [5], alguns exemplos desse tipo de comportamento são dados a seguir:

- Os mapas auto-organizáveis podem ser utilizados para assinalar cada amostra nova de treinamento a uma sub-rede, dependendo do resultado do agrupamento.
- A mistura de especialistas [33],[38],[39] pode ser utilizada, entre outras possibilidades, treinando-se cada rede especialista para cada padrão usado durante a aprendizagem. O módulo que conseguir produzir a saída mais próxima da desejada será o módulo vencedor. Enquanto isso, módulos de decisão aprendem a decompor as tarefas assinalando o padrão de entrada ao especialista vencedor.

Cooperação entre os módulos pode ser necessária para melhor atingir os objetivos na aprendizagem. Algumas formas de cooperação são fusão de pesos, de saídas ou de nós de diferentes módulos. Alguns exemplos são descritos a seguir [5]:

- Um certo número de módulos idênticos são treinados para uma dada tarefa, e médias de seus pesos são obtidas após a ocorrência de um dado número de épocas para evitar mínimos locais (o conjunto de treinamento não precisa ser necessariamente o mesmo para todos os módulos).
- Múltiplos módulos idênticos são utilizados e a média de suas saídas é obtida para aumentar a tolerância a faltas.
- Waibel [65] propõe vários módulos de três camadas sendo treinados separadamente. Quando esses atingem os níveis de desempenho desejados, os pesos

de suas camadas escondidas são então fixados. Em seguida, todos os nós das camadas escondidas dos diferentes módulos são concatenados e algumas unidades escondidas são adicionadas com o objetivo de “colar” os módulos, ou seja, descobrir características adicionais em uma segunda fase de treinamento. Este procedimento também é conhecido como redes *merge-and-glue*.

Quando o sistema modular é formado por vários módulos paralelos (competitivos ou cooperativos), torna-se necessário um procedimento para tomada de decisão, ou seja, escolher qual (ou quais) módulos serão considerados para fornecer a resposta do sistema. Esse assunto será discutido na seção a seguir.

4.4. Tomada de Decisão Multi-módulo

O relacionamento entre módulos paralelos pode variar desde o completo desacoplamento até a cooperação ou a competição. Quando os módulos são desacoplados, a decisão geralmente é tomada de acordo com o valor máximo de ativação dos módulos. Cada sub-rede não possui qualquer informação a respeito das outras sub-redes, e não há inibição entre os módulos. Uma forma de competição entre módulos pode ser feita utilizando-se redes não supervisionadas (normalmente de duas camadas). A primeira camada recebe o vetor de entrada e propaga os sinais para a camada de saída. Nesta última, cada nó se refere a um dos módulos. Quando o padrão de entrada é apresentado, o módulo correspondente ao nó vencedor da rede não supervisionada tomará a decisão final. Para módulos cooperativos, todas as saídas das diferentes sub-redes são combinadas ao invés de utilizar apenas uma sub-rede vencedora. O grau de dependência dos módulos difere de uma técnica para outra. Uma forma de cooperação define um bit extra na saída de cada módulo. Dessa forma, o módulo utiliza esse bit para indicar se o padrão de entrada pertence ao seu grupo de decisões. Em uma outra abordagem, vários módulos idênticos são treinados para uma mesma tarefa. O processo de tomada de decisão é feito a partir de uma votação, ou seja, a decisão a ser tomada é indicada pelo voto da maioria. Vale lembrar que, como os módulos são idênticos (se diferem apenas na inicialização e talvez, em pequenas diferenças em seus parâmetros), estes podem tender a cometer os mesmos erros em determinados problemas, e assim, a melhoria de performance com relação a uma rede não modular pode não ser significativa. Uma melhoria no esquema de votação pode ser conseguida levando-se em conta não apenas a preferência de cada módulo, mas também a intensidade de suas respostas.

No próximo capítulo serão apresentados alguns modelos de redes neurais e processos estocásticos encontrados na literatura recente para aprendizagem de seqüências temporais.

5. MODELOS PARA SEQUÊNCIAS TEMPORAIS

Este Capítulo estuda três modelos com abordagens bem distintas, capazes de aprender sequências temporais, e que contribuam na fundamentação e/ou concepção do novo modelo a ser proposto nesta pesquisa. O primeiro é o modelo de D'Arbo Junior [22], que é utilizado como ponto de partida para implementação do sistema a ser proposto mais adiante, além de servir como referência para a análise e comparação dos resultados obtidos. Já o segundo modelo a ser apresentado, o TOTEM, é um sistema composto por 4 módulos sucessivos, onde cada um destes realiza uma tarefa específica predeterminada. A abordagem modular, assim como a forma com que este modelo trata as informações de contexto temporal e espacial serviram de inspiração para a proposição do sistema apresentado no Capítulo 7. O terceiro modelo é um tipo de modelo Markoviano escondido (HMM – *Hidden Markov Model*) proposto por Bengio e Frasconi [12], e conhecido como IOHMM (*Input-Output Hidden Markov Model*). Este modelo utiliza uma abordagem estocástica para aprender sequências temporais, e por isso apresenta características e um comportamento bem distintos quando comparado aos modelos anteriores.

5.1. O Modelo de D'Arbo Junior

D'Arbo Junior [22] propõe e compara o desempenho de seis modelos neurais recorrentes na tarefa de reprodução de sequências temporais. O modelo apontado por D'Arbo Junior como sendo o de melhor desempenho nesse tipo de tarefa foi o ponto de partida do nosso trabalho, além de ser utilizado como referência para as avaliações e comparações dos resultados. Esta Seção apresenta a rede de D'Arbo Junior, e comenta a respeito de seu comportamento, suas características, propriedades desejadas, e algumas limitações.

D'Arbo Junior propôs um modelo formado por três camadas de unidades, sendo a camada de entrada totalmente conectada (todas as unidades da camada estão conectadas entre si via conexões de realimentação intracamada) e conectada via conexões diretas (no sentido entrada-saída) à camada escondida. Esta se conecta via conexões diretas à camada de saída, que por sua vez é totalmente realimentada à entrada (Figura 7). Nesse modelo, a camada de entrada

recebe todas as realimentações do sistema, e sendo assim, é esta camada que processa todas as informações de contexto temporal presentes no sistema. Por isto, a camada de entrada também pode ser chamada de camada de contexto.

As conexões de realimentação intracamada utilizadas na camada de contexto colaboram para a manutenção de informações a respeito dos estados que esta camada assumiu no passado. Já as conexões de realimentação intercamadas (da saída para a entrada) são responsáveis pela manutenção das informações sobre os elementos de uma seqüência já percorridos pela rede durante a reprodução desta seqüência. D'Arbo Junior sugere, a partir dos resultados obtidos em sua pesquisa de mestrado, que o tratamento conjunto dessas duas informações de contexto temporal (sobre o histórico dos estados da camada, e sobre o histórico dos pontos percorridos da seqüência) em uma mesma camada (a camada de contexto) proporciona para a rede capacidade de reprodução de seqüências superior àquela verificada se tais informações fossem processadas em camadas distintas [4],[22].

Figura 7: Modelo de rede recorrente que será utilizado como ponto de partida para o desenvolvimento de um sistema de planejamento de trajetórias.

O estado de ativação da i -ésima unidade de entrada é dado por:

$$i_i(t+1) = f \left(\sum_{j=1}^{n_i} k_{ij} o_j(t) + \sum_{j=1}^{n_e} w_{ij} i_j(t) \right) \quad (5.1.1)$$

sendo n_e o número de unidades de entrada;

n_s o número de unidades na saída;

k_{ij} o peso da conexão de realimentação da unidade j da saída para a unidade i da entrada;

w_{ij} o peso da conexão da unidade j para a unidade i , ambas da camada de entrada;

$o_j(t)$ o estado de ativação da unidade j de saída no instante de tempo t .

A ativação da i -ésima unidade da camada escondida é dada por:

$$h_i(t) = f \left(\sum_{j=1}^{n_e} w_{ij} i_j(t) \right) \quad (5.1.2.)$$

sendo w_{ij} o peso da conexão da unidade j de entrada para a unidade i da camada escondida;

$i_j(t)$ o estado de ativação da unidade j da camada de entrada no instante de tempo t .

E finalmente, o estado de ativação da i -ésima unidade de saída é calculado da seguinte forma:

$$o_i(t) = f \left(\sum_{j=1}^{n_h} w_{ij} h_j(t) \right) \quad (5.1.3.)$$

sendo w_{ij} o peso da conexão da unidade escondida j para a unidade i de saída;

n_h o número de unidades escondidas;

$h_j(t)$ o estado de ativação da unidade escondida j no instante de tempo t .

Para o treinamento desse modelo, utiliza-se uma variação do algoritmo clássico de retropropagação do erro. Nessa variação, as conexões de realimentação são tratadas como se fossem conexões vindas de unidades virtuais. E tais unidades são, na realidade, cópias das unidades reais, porém, atrasadas um instante de tempo.

Para o problema de reprodução de trajetórias, o número de unidades de entrada, n_e , e o número de unidades de saída, n_s , são iguais à dimensão dos padrões (ou vetores) de entrada. O procedimento proposto por D'Arbo Junior para reproduzir uma trajetória, pode ser descrito da seguinte forma:

- a) Inicialmente são fornecidos x_0 (ponto inicial) e x_f (ponto final);
- b) Para o instante inicial $t=0$, o estado de ativação das unidades da camada de entrada é inicializado com os valores do ponto final x_f (indicando para a rede a meta a ser atingida), enquanto o estado de ativação das unidades da camada de saída é inicializado com os valores do ponto inicial x_0 (indicando o ponto em que a rede se encontra no início do processo). As unidades da camada escondida são inicializadas com ativação igual a zero.

- c) Para cada instante de tempo t , calculam-se os valores de ativações das unidades de entrada, escondidas, e de saída, através das equações (5.1.1.), (5.1.2.) e (5.1.3.) respectivamente. Os valores de ativação encontrados nas unidades de saída expressam o próximo ponto na seqüência temporal (ou na trajetória a ser seguida), e são realimentados na camada de entrada. No próximo instante de tempo, as ativações são novamente calculadas, e o processo se repete até que toda a seqüência tenha sido reproduzida.

Perceba que, no procedimento de inicialização descrito em b, os padrões x_i (ponto inicial) e x_f (ponto final) consistem nas únicas informações dadas para o modelo. Conforme os conceitos estudados no Capítulo 3, pode-se dizer que os padrões x_i e x_f contém informações de contexto espacial, pois fornecem pistas a respeito do posicionamento espacial da trajetória a ser seguida, e a partir de tais informações, o modelo deve ser capaz de identificar e seguir corretamente esta trajetória. Sendo assim, se os pontos x_i e x_f fornecidos para o processo de execução (reprodução) forem diferentes daqueles vistos durante o treinamento, poderá ocorrer ausência de (e/ou ruído em) informações de contexto espacial, e conseqüentemente, a rede poderá apresentar dificuldades para “localizar no espaço” a trajetória correta a ser seguida.

Um outro aspecto importante a ser notado se refere à falta de informações de contexto temporal. Ou seja, no instante $t = 0$, o sistema não contém nenhum tipo de informação sobre estados passados. Isso significa dizer que, se o ponto inicial x_i , fornecido para a execução, for um ponto intermediário da trajetória a ser seguida, a rede poderá apresentar dificuldades em se “localizar dentro da trajetória”. Este problema fica ainda mais complexo se o ponto x_i fornecido durante a execução for um ponto ambíguo (visitado mais de uma vez em uma mesma trajetória ou compartilhado por diferentes trajetórias).

Vale lembrar que o problema de falta de informações de contexto temporal é comum em inúmeros modelos de redes neurais recorrentes, e não apenas no modelo de D’Arbo Junior.

D’Arbo Junior aplica este modelo em um problema de robótica no qual deseja-se reproduzir trajetórias para guiar um braço mecânico. Nesse caso específico, D’Arbo Junior mostra que uma contribuição relevante deste modelo é o fato deste ser capaz de produzir trajetórias tendo como saída, não apenas os pontos espaciais, mas também os ângulos e os torques nas juntas para cada ponto da trajetória. Ou seja, um único modelo executa em uma única etapa o equivalente aos problemas de geração de trajetórias, cinemática inversa e dinâmica inversa.

A seguir, serão feitos alguns comentários a respeito da utilização do modelo de D’Arbo Junior para a aprendizagem de seqüências temporais:

- D'Arbo Junior mostra que seu modelo, comparado a outros modelos de redes neurais recorrentes, apresenta um bom desempenho para aprendizagem de seqüências temporais em situações treinadas;
- Para situações não treinadas, fornecendo-se pontos iniciais e/ou finais diferentes daqueles vistos durante o treinamento, o modelo de D'Arbo Junior, na maioria das vezes, não é capaz de recuperar corretamente a trajetória desejada. Basicamente, isso ocorre devido a dois motivos: ausência de contexto temporal, e ausência ou ruído nas informações de contexto espacial.
- O modelo de D'Arbo Junior trata todas as informações de contexto temporal em uma mesma camada, a camada de entrada (ou de contexto). O autor sugere que dessa forma, a rede apresenta uma maior capacidade para aprendizagem e reprodução. Porém, conforme será visto mais adiante no Capítulo 7, tal arquitetura pode apresentar problemas quando a dimensão dos padrões de entrada for pequena (por exemplo: trajetórias bidimensionais). Nesse caso, haverá um número reduzido de unidades de entrada (para o exemplo dado haveriam apenas duas unidades de entrada) e, conseqüentemente haverão poucas conexões de realimentação, o que pode ser insuficiente para o tratamento adequado das informações de contexto temporal.
- Uma outra limitação observada no modelo de D'Arbo Junior consiste na longa duração do treinamento. O principal motivo para essa demora se deve à utilização do algoritmo de retropropagação, que é naturalmente um algoritmo lento. Mais adiante nesta pesquisa serão mencionadas heurísticas, assim como estudos recentes para acelerar o algoritmo de retropropagação.

Na Seção a seguir será estudado um modelo com uma abordagem modular para realizar as tarefas de identificação e reprodução de seqüências temporais.

5.2. O Modelo TOTEM

Srinivasa e Ahuja [62] propõem uma rede neural para aprendizagem de seqüências temporais conhecida como rede TOTEM. Trata-se de um exemplo de sistema neural modular, formado por 4 módulos sucessivos, e com uma divisão explícita de tarefas. Este modelo desperta interesse devido à sua capacidade em aprender uma grande quantidade de seqüências, e à sua robustez a ruídos. Um outro aspecto interessante neste modelo reside na forma não

supervisionada³ com que as informações contextuais são extraídas e armazenadas. As informações de contexto espacial dos componentes de uma sequência são extraídas em um módulo contendo uma rede neural do tipo mapa auto-organizável [41], enquanto as informações de contexto temporal são obtidas em outro módulo formado por um tipo particular de rede neural recorrente.

A rede TOTEM é capaz de aprender sequências que atendam às seguintes restrições:

- a) Os sinais espaço-temporais são pré-segmentados, ou seja, o começo e o fim de uma sequência são marcados ou fornecidos;
- b) Os sinais espaço-temporais que pertencem a uma mesma categoria possuem aproximadamente a mesma duração e a mesma velocidade;
- c) Os padrões espaço-temporais pertencentes a uma mesma categoria possuem aproximadamente o mesmo número de elementos. Sob tais restrições, a rede TOTEM é capaz de reconhecer e/ou reproduzir um grande número de sequências mesmo quando existirem ruídos na sequência a ser reconhecida [62].

A rede TOTEM é formada por 5 camadas, sendo uma camada de entrada, e 4 módulos sucessivos (a resposta de um módulo é alimentada na entrada do módulo seguinte) de arquiteturas distintas (Figura 8). Este é um exemplo de decomposição explícita de tarefas no qual cada módulo é especializado em realizar uma subtarefa específica. Tais subtarefas, determinadas a partir de um estudo prévio das etapas que poderiam estar envolvidas no problema, são descritas a seguir: (a) extração das informações espaciais dos padrões; (b) extração da ordem temporal; (c) construção de um padrão correlacionando as características espaciais e temporais dos elementos de uma sequência; (d) e agrupamento dos padrões construídos na etapa anterior.

Conforme será visto mais adiante, durante o treinamento, assim como para a tarefa de identificação de trajetórias, o sistema realiza um processamento direto (no sentido entrada-saída), enquanto que, para a tarefa de reprodução de trajetórias, o sistema realiza um processamento inverso (no sentido saída-entrada). Por isso, as setas representando “links” entre módulos na Figura 8 são bidirecionais.

O treinamento do modelo TOTEM é realizado em diferentes etapas, e possui procedimentos específicos para cada módulo do sistema. A primeira camada está totalmente conectada à camada SOM, e serve apenas para propagar o sinal de entrada.

O treinamento do sistema começa no módulo SOM, baseado no modelo de mapa auto-organizável proposto por Kohonen [41]. O módulo SOM é formado por um arranjo 2-D de

³ Modelos não supervisionados não recebem um sinal externo (professor) indicando a resposta desejada. Tais modelos se adaptam para encontrar características e/ou regularidades nos dados de entrada [34].

neurônios que recebem os padrões de entrada via o vetor de pesos w_{ij} (onde i e j indicam a posição espacial do neurônio no arranjo 2-D). Cada vetor de pesos funciona como um protótipo da entrada. Uma vez fornecido um padrão de entrada, a unidade que possuir o protótipo mais próximo (de menor distância Euclidiana) deste padrão é dita “vencedora”, e os pesos desta unidade e dos seus vizinhos são adaptados de forma a diminuir essa distância (Apêndice A.1.). Após o treinamento, a projeção do vetor de pesos do primeiro módulo no espaço de entrada conterá as relações de vizinhança naquele espaço. Ou seja, a rede SOM terá aprendido informações sobre a topologia do espaço de características de entrada.

Figura 8: Sistema modular TOTEM, formado pela camada de entrada e quatro módulos sucessivos.

Já o 2º módulo é um tipo de memória de curto prazo (STM) responsável pela extração das informações temporais. Assim como no 1º módulo, as unidades do 2º módulo também estão dispostas em um arranjo 2-D. Inicialmente, cada unidade do 1º módulo estará conectada por uma conexão excitatória a uma unidade correspondente no 2º módulo. Além disso, as unidades do 2º módulo possuem realimentações locais excitatórias, e estão totalmente conectadas entre si via conexões inibitórias (Figura 9).

Após o treinamento do 1º módulo, cada padrão do conjunto de treinamento é apresentado (desta vez na ordem correta da sequência a que pertence) à entrada do sistema. Para cada padrão apresentado, somente a unidade vencedora do módulo SOM será ativada (apresentará uma saída igual a 1) enquanto todas as outras permanecerão com suas saídas iguais

a zero. Para cada unidade (i, j) do módulo SOM, existe uma variável f_{ij} (frequência do vencedor) associada. Esta variável, que no início da apresentação de cada seqüência vale zero, é incrementada cada vez que a unidade a ela associada vencer durante a apresentação de uma dada seqüência. Essa informação de frequência é útil para detectar pontos que ocorrem mais de uma vez durante uma mesma seqüência, e será utilizada mais adiante no processo.

Figura 9: A unidade vencedora do módulo SOM excita (via conexão um para um) a unidade correspondente no módulo STM. O detalhe mostra a unidade do módulo STM com inibições laterais e realimentação excitatória.

Não é permitido que uma mesma unidade do módulo STM seja excitada mais de uma vez durante a apresentação de uma dada seqüência. Sendo assim, cada vez que ocorrer uma repetição da unidade vencedora na camada SOM, uma nova unidade (associada à vencedora) será criada e excitada no módulo STM. Ou seja, ao final da aprendizagem, cada unidade (i, j) do módulo SOM estará conectada a f_{ij} unidades no módulo STM. Essa foi a solução adotada pelos autores do modelo TOTEM para resolver o problema da ocorrência de padrões repetidos dentro de uma mesma seqüência temporal.

A ativação de cada unidade do módulo STM é uma função monotonicamente decrescente no tempo. E assim sendo, a ordem dos padrões entrados será dada pela intensidade das ativações no módulo STM. Alguns detalhes sobre esse módulo podem ser vistos no Apêndice A.2.

O 3º módulo do sistema utiliza as ativações do módulo STM para construir um padrão de características espaciais e temporais correlacionadas. Este módulo é ativado apenas quando um sinal externo de controle, G_1 , indicar que uma seqüência já foi completamente fornecida para os módulos anteriores. Quando isso acontecer, este módulo faz uma leitura seqüencial das M unidades ativas do módulo STM, seguindo uma ordem de ativação decrescente. Para cada unidade i lida (sendo i variando de 1 até M) obtém-se dois valores y_{i1} e y_{i2} . Sendo assim, após a leitura de todas as M unidades ativas do módulo STM, os valores y 's obtidos formarão o padrão

y (vetor de características espaciais e temporais correlacionadas cuja dimensão é igual a $2M$). A construção de y pode ser vista com mais detalhes no Apêndice A.3.

A função do 4º módulo consiste então em agrupar os vetores y 's (referentes às seqüências aprendidas) apresentados à sua entrada. Para um melhor entendimento sobre o esse módulo, é preciso comentar algumas características do modelo ART clássico. Trata-se de uma rede neural capaz de agrupar, de forma não supervisionada, padrões com entradas binárias. Uma das características importantes deste modelo, é a sua capacidade de recrutar novas unidades de saída (e conseqüentemente criar novos grupos ou classes de padrões) durante o processo de aprendizagem. Isso ocorre graças a um princípio de auto-organização conhecido como *Teoria da Ressonância Adaptativa* [15],[16]. Basicamente, o que ocorre é uma propagação do padrão de entrada para a saída, seguida de uma propagação de volta (ou regeneração) da saída para a entrada. Dentro desse processo, tanto o padrão de entrada quanto o padrão regenerado são utilizados no chamado *critério de vigilância*. Pode-se dizer que esse critério verifica se o agrupamento indicado pela unidade vencedora da saída realmente “representa bem” o padrão fornecido na entrada. Se, após a apresentação de um padrão de entrada, nenhum dos nós de saída atenderem ao critério de vigilância, então uma nova unidade de saída é criada para responder ao padrão entrado.

A rede fuzzy-ART [17], quarto módulo do modelo TOTEM, contém o mesmo mecanismo explicado a respeito do modelo ART. Porém, a rede fuzzy-ART utiliza operações de lógica nebulosa em suas equações, o que possibilita a este modelo a capacidade de realizar agrupamentos de padrões com entradas contínuas, como é o caso do padrão y . Alguns detalhes a mais sobre o módulo fuzzy-ART podem ser vistos no Apêndice A.4.

Para reconhecer uma seqüência, é preciso apresentar todos os padrões desta seqüência na entrada do sistema, e realizar sucessivamente as tarefas de cada um dos módulos. Ao final do processo, o rótulo indicado pela unidade de saída da rede fuzzy-ART identificará a seqüência apresentada. Para reproduzir uma seqüência, informa-se o rótulo da seqüência desejada, e efetua-se o processo inverso da seguinte forma: (a) a partir do rótulo da seqüência, obtém-se, do modelo fuzzy-ART, o padrão de características espaciais e temporais correlacionadas; (b) a leitura deste padrão em ordem de índices crescentes indicará a seqüência das unidades a serem ativadas no 2º módulo; (c) cada unidade do 2º módulo excita uma unidade correspondente no 1º módulo; (d) e finalmente, cada unidade excitada no primeiro módulo fornecerá o seu vetor de pesos que na realidade são protótipos dos padrões das seqüências aprendidas, e assim a seqüência é recuperada.

Alguns comentários a respeito de habilitações e limitações do modelo TOTEM serão feitos a seguir:

- De acordo com Srinivasa e Ahuja [62], este modelo é capaz de aprender uma grande quantidade de seqüências, inclusive seqüências ruidosas, desde que tais

seqüências atendam às restrições a respeito do tamanho das seqüências, da duração e da velocidade dos sinais discutidos no início desta seção.

- É correto afirmar que a habilidade do modelo TOTEM para reconhecer seqüências ruidosas se deve, em parte, ao fato deste modelo trabalhar separadamente as informações de contexto espacial e temporal. Ou seja, o mapeamento realizado pelo módulo SOM para uma seqüência sem ruídos deverá ser muito parecido com o mapeamento realizado para a mesma seqüência com padrões ruidosos (desde que esse ruído não seja muito elevado). Além disso, o modelo fuzzy-ART utilizado na saída também é robusto com relação a entradas ruidosas.
- A quantidade de unidades em cada camada é diretamente proporcional ao tamanho do espaço de entrada (ou espaço de características). Sendo assim, para um espaço de entrada grande (contendo uma grande quantidade de estados possíveis), a quantidade de unidades usada pelo sistema será muito grande, e o custo computacional deverá ser elevado.
- O sistema agrupa e rotula as seqüências aprendidas, de forma que, a reprodução de uma seqüência é feita a partir do rótulo que a identifica, e não a partir de um ponto da trajetória. Esse tipo de estratégia não se enquadra para resolver o problema proposto nesta pesquisa, pois conforme dito anteriormente (Capítulo 2), a única informação que deverá ser fornecida para o sistema consiste no padrão formado a partir dos pontos inicial e final. Ou seja, o sistema precisa descobrir por si só, qual a seqüência que conecta os pontos inicial e final fornecidos.
- O processo de aprendizagem e reprodução de seqüências utilizado pelo modelo TOTEM, contém algumas características e procedimentos que não são desejáveis para o sistema proposto nesta pesquisa, e são citados a seguir: (a) precisa de sinais externos para controlar o processo, (b) realiza determinadas tarefas (como a construção do padrão y observada no terceiro módulo) através de procedimentos computacionais pouco relacionados à teoria de redes neurais.

O modelo TOTEM, desperta interesse devido à capacidade de aprender uma grande quantidade de seqüências. Porém apresenta limitações quanto a: (a) o método utilizado para reproduzir seqüências (a partir de rótulos dados às seqüências, e não a partir de pontos destas seqüências); (b) o custo computacional que se torna elevado quando o espaço considerado contiver uma grande quantidade de estados possíveis.

A seguir será estudado o modelo conhecido como IOHMM. Esse modelo utiliza uma abordagem estocástica completamente diferente do que foi visto até o momento.

5.3. O Modelo Markoviano Escondido IOHMM

A partir de meados da década de 80 houve um significativo aumento na quantidade de estudos e aplicações envolvendo sistemas do tipo modelos Markovianos escondidos - HMM (Hidden Markov Model). Hoje em dia, tais modelos são utilizados nos mais variados problemas envolvendo análise, modelagem, reprodução e reconhecimento de estruturas sequenciais, tais como: reconhecimento de fala, identificação de gestos, caracterização de famílias de proteínas, modelagem e identificação de seqüências genéticas, entre outros [12],[36],[48],[56],[67]. Um outro aspecto interessante a ser observado reside no fato de que diversos autores vêm estudando formas de combinar técnicas de redes neurais e modelos Markovianos escondidos em sistemas para aprendizagem de seqüências [12],[13],[45],[46],[48]. O IOHMM é um exemplo de sistema modular que mistura a abordagem conexionista de redes neurais com a abordagem estocástica de HMMs.

Atualmente, é possível encontrar inúmeros estudos e variações de HMMs que se diferem principalmente no que diz respeito a:

- Diferenças quanto aos métodos de otimização utilizados para encontrar parâmetros do modelo;
- Utilização conjunta de HMMs e outros tipos de modelos (como por exemplo redes neurais);
- Procedimentos adotados para modelagem e implementação dos HMMs.

Os HMMs são modelos que englobam dois processos estocásticos: (a) uma cadeia de Markov de primeira ordem, e (b) para cada estado desta cadeia, tem-se a probabilidade de observação de um símbolo (ou probabilidade de saída), pertencente a um conjunto finito de símbolos. Apenas os símbolos são observáveis na saída do modelo, mas não os estados (os estados são ditos “escondidos”), por isso o nome Modelo Markoviano Escondido. Uma breve explicação a respeito de modelos Markovianos e em particular os HMMs pode ser encontrada no Apêndice B.

O IOHMM é um tipo de HMM no qual as distribuições de probabilidade de transição de estado assim como as distribuições de probabilidade de saída (ou de observação) são condicionadas à entrada [12]. Além disso, o IOHMM permite a modelagem de sistemas com entradas e saídas contínuas. Nesse caso, no lugar de uma probabilidade de observação (ou probabilidade de saída), o sistema apresenta em sua saída a esperança da saída condicionada à entrada. O método a ser comentado a seguir descreve o IOHMM para o caso em que entradas e saídas são contínuas.

No procedimento adotado por Bengio & Frasconi [12] para implementar o IOHMM, as distribuições de probabilidade de estado assim como a esperança da saída são modeladas utilizando-se respectivamente dois conjuntos de redes neurais: as sub-redes de estado e as sub-redes de saída.

A função de cada sub-rede de estado, N_j , é fornecer, no instante t , a distribuição de probabilidade condicional do estado x_t dado que o estado x_{t-1} seja igual a j e a entrada atual seja $\mathbf{u}_t$:

$$\varphi_{j,t} = P(x_t | x_{t-1} = j, \mathbf{u}_t) \quad (5.3.1.)$$

Já a sub-rede O_j de saída fornece um valor $\eta_{j,t}$ que é uma esperança da saída dado que a entrada no instante t seja $\mathbf{u}_t$ e o estado seja $x_t = j$. Isto é,

$$\eta_{j,t} = E[y_t | x_t = j, \mathbf{u}_t]. \quad (5.3.2.)$$

A arquitetura do modelo IOHMM pode ser vista na Figura 10. Os dois conjuntos de sub-redes possuem o mesmo tamanho pois cada uma das sub-redes de estado e de saída estão unicamente associadas a um dos estados do conjunto S contendo todos os n estados possíveis. Além disto, todas as sub-redes de estado e de saída recebem a mesma entrada $\mathbf{u}_t$.

Figura 10: Arquitetura do sistema modular híbrido, IOHMM, formado por módulos neurais cooperativos, e uma abordagem de modelos markovianos escondidos.

As saídas das sub-redes de estado são utilizadas recursivamente para calcular no instante de tempo t o vetor ζ_t , que representa a distribuição atual de estados dada uma seqüência de entradas passadas. Essa variável é obtida pela combinação linear das saídas das sub-redes de estado e de seu valor no tempo anterior

$$\zeta_t = \sum_{j=1}^n \zeta_{j,t-1} \varphi_{j,t} \quad (5.3.3.)$$

sendo

$$\zeta_{j,t} = P(x_t = j | \mathbf{u}_1^t), \quad (5.3.4.)$$

e $\mathbf{u}_1^t$ é a subsequência de entradas a partir do instante 1 até o instante t , inclusive.

ζ_t é um vetor de números positivos, e a soma de seus elementos, para cada instante t , é igual a um.

E finalmente, a saída global do sistema, denotada por $\eta_t \in R^r$ é calculada a partir da combinação linear das saídas de cada uma das sub-redes de saída. Ou seja,

$$\eta_t = \sum_{j=1}^n \zeta_{j,t} \eta_{j,t} \quad (5.3.5.)$$

sendo $\eta_{j,t}$ é a saída da sub-rede O_j .

Sobre o IOHMM pode-se dizer que:

- De acordo com as definições do *Capítulo 4*, o modelo IOHMM é um sistema modular, no qual é utilizada decomposição explícita de tarefas uma vez que os módulos (sub-redes de entrada e saída) realizam tarefas previamente determinadas (prever distribuição de estados, e prever a saída esperada do sistema). A relação entre os módulos é predominantemente cooperativa (diferentemente do caso anterior, o modelo TOTEM, quando a relação entre módulos era sucessiva), pois as sub-redes trabalham em paralelo de forma que: (a) o estado do sistema é obtido a partir da combinação das respostas de cada sub-rede de estado; e (b) a saída do sistema é obtida a partir da combinação das respostas de cada sub-rede de saída.
- Para o modelo IOHMM (assim como para qualquer outro modelo do tipo HMM) os estados não são observáveis (apenas os símbolos) e nem precisam ter significado físico. Além disso, não existe um critério único (e bem definido) para se saber a quantidade de estados necessária e suficiente para que o sistema apresente o melhor desempenho.
- O custo computacional do modelo IOHMM pode ser consideravelmente elevado quando a quantidade de estados possíveis para o sistema for grande, pois conforme mostrado na Figura 10, o número de sub-redes de estado assim como o número de sub-redes de saída é igual ao número de estados possíveis para o sistema.
- A forma com que esse modelo trata as informações de contexto temporal é completamente diferente da forma adotada nos modelos apresentados anteriormente. Para o IOHMM, tais informações são aprendidas durante o

treinamento através de probabilidades de transição de estado e probabilidades de observação na saída, dado o estado atual.

O capítulo a seguir pode ser dividido em duas partes. Na primeira, os principais conceitos vistos até o momento são utilizados na proposição de uma abordagem modular dentro da qual foi desenvolvido o sistema para reprodução de trajetórias. Dentro dessa abordagem, dois módulos distintos processam separadamente as informações de contexto espacial e temporal. Para se escolher a arquitetura do módulo para tratamento das informações de contexto espacial, primeiramente foi necessário estudar alguns modelos de redes neurais que possuem características e/ou comportamentos interessantes. Tais modelos serão apresentados na segunda parte do capítulo a seguir.

6. PREMISSAS PARA IMPLEMENTAR O SISTEMA NEURAL MODULAR

Inicialmente, neste Capítulo, o problema proposto será revisitado. Em seguida, será discutida a estratégia adotada para atingir os objetivos traçados. Utilizando-se os estudos realizados anteriormente a respeito de sistemas modulares, contexto, e dos modelos estudados no capítulo anterior, foi possível elaborar conceitualmente uma nova abordagem a ser adotada na construção de sistemas neurais para aprendizagem de seqüências temporais. Dentro dessa nova abordagem, o sistema deverá processar separadamente as informações de contexto espacial e temporal em dois módulos sucessivos com arquiteturas distintas. Conforme será visto mais adiante, uma rede parcialmente recorrente baseada no modelo de D'Arbo Junior foi utilizada para realizar a função do módulo para reprodução de seqüências temporais. Porém, para escolher a arquitetura do módulo para processamento de contexto espacial, foi necessário um estudo preliminar de alguns modelos de redes neurais e da mecânica estatística que apresentavam características e/ou comportamentos correlatos com o problema em questão.

6.1. Revisitando o Problema

O problema proposto nesse trabalho consiste em reproduzir um conjunto de seqüências temporais $S = \{S_0, S_1, S_2, \dots, S_m\}$ que foram previamente ensinadas ao sistema. Uma vez fornecido para o sistema um padrão de entrada contendo (a princípio) o ponto inicial, x_0 , assim como o ponto alvo, x_f , o sistema deverá ser capaz de reproduzir a seqüência S_i que conecta tais pontos. A capacidade de generalização pode ser definida de duas formas:

- a) Habilidade do sistema em reproduzir a trajetória correta que conecte os pontos $\{x'_0, x'_f\}$ sendo x'_0 e x'_f versões ruidosas dos pontos x_0 e x_f (generalização de situação treinada);
- b) Habilidade do sistema em reproduzir a trajetória correta conectando os pontos $\{x'_0, x'_f\}$ sendo que x'_0 e x'_f podem ser pontos intermediários da trajetória (generalização de situação não treinada).

À primeira vista, pode parecer correto dizer que a capacidade de generalização descrita em b, pode ser entendida como um caso particular de a no qual o padrão intermediário x'_0 (ou x'_f) é visto como uma versão ruidosa do padrão x_0 (ou x_f). Porém, na realidade a questão não é tão simples pois, no caso b ocorre também a falta de informações de contexto temporal. Para partir de um ponto intermediário, o sistema precisaria conhecer não apenas o ponto em si, mas também um histórico de pontos anteriores (contexto temporal). Essa é uma das principais dificuldades no que diz respeito à generalização na tarefa de reprodução de seqüências temporais. Além do mais, é correto afirmar que esta dificuldade é encontrada em diferentes arquiteturas de redes neurais recorrentes, pois, independentemente da arquitetura em questão, se o modelo não possui nenhuma informação a respeito dos seus estados passados, dificilmente ele conseguirá seguir corretamente uma seqüência temporal contendo pontos ambíguos.

Conforme as definições dadas no Capítulo 3, a informação a respeito do ponto inicial e final da trajetória é um tipo de informação de contexto espacial. A restrição imposta ao conjunto de treinamento (Capítulo 2) que exclui o caso de duas trajetórias contendo os mesmos pontos inicial e final garante que a informação de ponto inicial e final é uma informação de contexto espacial suficiente para identificar unicamente a trajetória que se deseja reproduzir. Surge então, uma primeira premissa para se chegar na abordagem que será adotada. A idéia inicial consiste em utilizar um módulo que seja capaz de identificar a seqüência a ser reproduzida, uma vez fornecidos os pontos $\{x_0, x_f\}$. Esse seria o primeiro passo para o sistema no processo de reprodução de trajetórias. Em seguida, um segundo módulo seria responsável pela tarefa de reproduzir a seqüência identificada pelo módulo anterior.

O modelo TOTEM também utiliza módulos para realizar tarefas de agrupamento e identificação de trajetórias no problema de reprodução de seqüências. Nesse caso, o módulo fuzzy-ART é utilizado para agrupar padrões contendo informações de contexto temporal e espacial correlacionadas (referentes às seqüências aprendidas). Além do mais, no processo de reprodução de trajetórias, o modelo TOTEM utiliza os protótipos que foram aprendidos pelo módulo fuzzy-ART para reproduzir as trajetórias desejadas propagando tais protótipos para as camadas anteriores. Um outro aspecto interessante com relação ao modelo TOTEM é o fato deste utilizar módulos distintos na aquisição de informações espaciais e temporais. Ou seja, enquanto o módulo SOM identifica as informações espacial, o módulo STM identifica as informações temporais. Porém conforme já foi comentado no capítulo anterior, a estratégia adotada pelo modelo TOTEM possui alguns pontos que não são desejados para o sistema nesta pesquisa, os principais são: a) processamento complexo envolvendo uma grande quantidade de módulos, além da necessidade de sinais e informações de controle, b) a necessidade de informar um rótulo que identifica a seqüência que se deseja reproduzir, c) a necessidade de inverter o processamento para realizar a tarefa de reprodução de seqüências.

Perceba que o rótulo de uma sequência é um tipo de informação que identifica unicamente a sequência a ele associada. No caso do modelo TOTEM, informar o rótulo significa dizer que o sistema precisa receber uma identificação exata a respeito da sequência desejada, para que ele possa reproduzi-la. Já na abordagem adotada nesta dissertação, o sistema deverá descobrir por si só, qual a sequência que conecta os pontos inicial e final fornecidos.

Com base nos estudos realizados e apresentados até o momento, esta dissertação propõe que a tarefa de reprodução de sequências temporais seja dividida em pelo menos três subtarefas:

- a) Identificação da sequência a ser reproduzida a partir dos pontos fornecidos na entrada do sistema;
- b) Obtenção de um protótipo de contexto espacial referente à sequência identificada;
- c) Reprodução da trajetória identificada a partir do seu protótipo de contexto espacial obtido anteriormente.

As subtarefas a e b podem ser realizadas utilizando-se um único módulo. Sendo assim, a Figura 11 mostra um diagrama com um sistema proposto formado por dois módulos sucessivos distintos. O módulo estimador de contexto espacial recebe uma entrada $\{x'_0, x'_t\}$ (sendo que x'_0 e x'_t podem não ser os pontos treinados como inicial e final, ou seja, podem ser pontos intermediários), identifica a sequência que conecta tais pontos e em seguida fornece um protótipo de contexto espacial correspondente a esta sequência. O módulo de reprodução de sequência temporal é capaz de considerar contexto temporal, e reproduzir a sequência correspondente ao protótipo de contexto espacial recebido.

Figura 11: Diagrama do sistema a ser proposto, formado por dois módulos. O primeiro módulo recebe o padrão de entrada, identifica a sequência a ser reproduzida e fornece um protótipo do contexto espacial, e uma probabilidade associada. O segundo módulo utiliza o protótipo do contexto espacial fornecido para reproduzir a sequência desejada.

Dentro da abordagem em elaboração, o primeiro módulo é o principal responsável quanto à capacidade de generalização do sistema pois entradas ruidosas e/ou parcialmente desconhecidas deverão ser tratadas e/ou corrigidas neste módulo. O protótipo de contexto espacial fornecido pelo módulo estimador de contexto espacial será, na realidade, o padrão $\{x_0,$

x_f (contendo pontos vistos durante o treinamento) recuperado a partir do padrão de entrada fornecido, $\{x'_0, x'_f\}$.

Desta forma, o primeiro módulo deve realizar as seguintes tarefas:

- a) **Identificação** – a seqüência que se deseja reproduzir deve ser identificada uma vez fornecidos dois pontos, x'_0 e x'_f . Lembrando que o ponto inicial fornecido, x'_0 ou o ponto final fornecido x'_f podem ser, na realidade pontos vistos como intermediários durante o treinamento;
- b) **Prototipação** – além de identificar a seqüência, o módulo estimador de contexto espacial deve ser capaz de fornecer protótipos do contexto espacial referente à seqüência identificada. Esta tarefa também está relacionada com a tarefa de preenchimento ou complementação de padrões de entrada parcialmente desconhecidos;
- c) **Informação de probabilidade ou da “incerteza” quanto ao contexto estimado** – esse tipo de informação pode ser útil para melhorar o desempenho do sistema, uma vez que, em situações de dúvida quanto à identificação da seqüência, procedimentos alternativos podem ser adotados;

Da forma como está sendo proposto, é correto afirmar que o tratamento de ruído (ou de entradas ruidosas) será feito, a princípio, no primeiro módulo. Ou seja, a princípio, o módulo para reprodução de seqüências temporais não precisa apresentar uma boa capacidade de generalização pois receberá informações de contexto espacial já tratadas pelo primeiro módulo. Porém, uma consequência desta abordagem consiste no fato de que, se o primeiro módulo erra na identificação da seqüência, o sistema todo erra pois o módulo para reprodução de seqüências temporais receberá informações totalmente incorretas (ou seja, referentes à seqüência errada). Em outras palavras, um erro do módulo estimador de contexto espacial deverá implicar na reprodução de uma trajetória diferente da que se deseja seguir. Esse tipo de limitação pode ser amenizado (ou desaparecer) com a introdução de probabilidades que possam exprimir, de alguma forma, o grau de certeza na identificação realizada pelo primeiro módulo, pois em situações de dúvida, procedimentos alternativos poderiam ser utilizados.

Uma outra característica importante de ser comentada com relação à abordagem que está sendo proposta consiste no fato do sistema não ser capaz de realizar interpolação de pontos ou trajetórias. Em outras palavras, da forma como está sendo proposto, o sistema sempre seguirá exatamente os pontos de uma das trajetórias aprendidas durante o treinamento.

A princípio, existem inúmeras possibilidades quanto à escolha dos modelos de redes neurais que poderiam desempenhar os papéis dos módulos que compõem o sistema da Figura 11. Sendo assim, serão apresentados a seguir, alguns estudos de modelos que possuem características desejadas para o papel do módulo de contexto espacial. Com relação ao módulo

para reprodução de seqüências temporais, será utilizado o modelo de D'Arbo Junior para desempenhar esta função. Os motivos para esta escolha, assim como estudos e modificações realizadas neste modelo serão apresentadas no Capítulo 7.

6.2. O Algoritmo Supervisionado de Quantização Vetorial

A seguir será descrito um modelo de rede neural de propagação direta treinado por um algoritmo conhecido como LVQ (Learning Vector Quantization) [41]. O nome LVQ se refere a uma família de métodos supervisionados que realiza quantização vetorial. Um dos principais objetivos de tais métodos consiste em definir regiões no espaço de entradas que caracterizam classes. A idéia central dos métodos LVQ pode ser descrita da seguinte forma: protótipos (ou vetores de pesos) são associados para cada classe (ou região) do espaço de entrada; e desta forma, uma vez fornecido um padrão de entrada v , o protótipo w_i mais próximo de v (menor distância Euclidiana) indicará a classe a que pertence v . Sendo assim, a aprendizagem nos algoritmos LVQ consiste basicamente em um procedimento para posicionar os protótipos de cada classe, de forma a minimizar o erro médio esperado quando for utilizada a regra de classificação pelo vizinho mais próximo.

O método a ser descrito a seguir é conhecido como LVQ1 [41]. Uma forma de se visualizar este modelo é apresentada na Figura 12. Trata-se de uma rede neural de propagação direta formada por apenas duas camadas (a camada de entrada e a camada de saída). O protótipo w_i é representado pelo conjunto de pesos que chegam na unidade de saída i . Sendo assim, o número de unidades na camada de saída é dado pelo número total de protótipos, enquanto o número de unidades de entrada é dado pela dimensão do espaço de entrada.

Figura 12: Arquitetura do modelo LVQ1, formado por duas camadas de unidades conectadas por um conjunto de pesos de propagação direta.

Uma vez assinalados um ou mais protótipos para cada região (ou classe) do espaço de entrada, define-se o índice do vizinho mais próximo de um padrão de entrada, $\mathbf{v}$, da seguinte forma:

$$c = \arg \min_i \{\|\mathbf{v} - \mathbf{w}_i\|\} \quad (6.2.1)$$

Ou seja, c também pode ser visto como a unidade “vencedora” na camada de saída, e consequentemente indica a classe a qual pertence o padrão de entrada $\mathbf{v}$.

O procedimento de aprendizagem (posicionamento dos protótipos) utiliza as seguintes equações:

$$\begin{aligned} \mathbf{w}_c(t+1) &= \mathbf{w}_c(t) + \alpha(t)[\mathbf{v}(t) - \mathbf{w}_c(t)] && \text{se } \mathbf{v} \text{ e } \mathbf{w}_c \text{ pertencerem à mesma classe} \\ \mathbf{w}_c(t+1) &= \mathbf{w}_c(t) - \alpha(t)[\mathbf{v}(t) - \mathbf{w}_c(t)] && \text{se } \mathbf{v} \text{ e } \mathbf{w}_c \text{ pertencerem a classes diferentes} \end{aligned} \quad (6.2.2)$$

Ou seja, o protótipo vencedor $\mathbf{w}_c$ deve ser modificado em direção ao padrão de entrada, $\mathbf{v}$, se ambos forem da mesma classe, e deve se afastar de $\mathbf{v}$ se forem de classes diferentes. É comum fazer a taxa de aprendizagem $\alpha(t)$ decrescer monotonicamente com o tempo para acelerar a convergência do processo de aprendizagem.

O algoritmo LVQ descrito não é utilizado para aproximar funções densidade de probabilidade, mas sim para definir diretamente as classes de um conjunto de amostras e suas respectivas bordas, utilizando-se a regra do vizinho mais próximo. O desempenho, a precisão e o tempo de convergência do treinamento dependem basicamente dos seguintes fatores [41]:

- **Escolha de um número ótimo de protótipos para cada classe** – esse número geralmente depende da complexidade e da forma das distribuições de cada classe existente;
- **Inicialização dos protótipos** – existem várias estratégias diferentes para inicialização dos protótipos, algumas são citadas a seguir: a) escolha aleatória de valores iniciais; b) inicializar protótipos a partir de padrões escolhidos, aleatoriamente, de cada uma das classes; c) utilizar uma rede SOM para realizar um mapeamento inicial do espaço de entrada, rotular as classes encontradas e utilizar os parâmetros da rede SOM como inicialização do modelo LVQ;
- **A escolha da taxa de aprendizagem** – geralmente estratégias que utilizam taxas monotonicamente decrescente são mais apropriadas e podem acelerar significativamente a convergência do algoritmo de aprendizagem;
- **O critério de parada** – o momento da interrupção do treinamento é importante para evitar um sobreajuste dos parâmetros. Alguns critérios de parada possíveis são descritos a seguir: a) interromper o treinamento após um número predeterminado de ciclos; b) interromper o treinamento a partir do momento em que o erro obtido em um conjunto de validação parar de decrescer.

O método LVQ1 descrito nesta Seção é um bom candidato para realizar a tarefa do primeiro módulo (estimador de contexto espacial) do sistema proposto (Figura 11) pois é capaz de realizar as tarefas de identificação e de prototipação (que são tarefas desejadas para o módulo estimador de contexto espacial). Porém, a princípio, esse método não informa a probabilidade ou alguma medida que possa exprimir a “incerteza” quanto à identificação realizada. Sendo assim, modelos com abordagens mais estatísticas foram estudados e serão apresentados a seguir.

6.3. Algumas Abordagens e Modelos Estocásticos

Neste capítulo serão estudados alguns modelos com abordagens estatísticas, inspirados em estudos da mecânica estatística, e que podem realizar pelo menos uma das seguintes tarefas: agrupamento, identificação, prototipação, e complementação de padrões (*pattern completion*). Essas tarefas estão todas relacionadas com a tarefa desejada para o módulo estimador de contexto temporal. Conforme já foi comentado anteriormente, o motivo de se estudar abordagens estatísticas consiste na necessidade de se obter, além da identificação da sequência e da prototipação do contexto espacial, informações de probabilidade indicando o grau de “incerteza” da resposta do módulo estimador de contexto no sistema proposto. Esse tipo de informação pode colaborar para melhoria do desempenho do sistema, principalmente no que diz respeito à capacidade de generalização.

6.3.1. Uma breve introdução de Mecânica Estatística

O estudo de mecânica estatística envolve o estudo de propriedades de equilíbrio macroscópico em sistemas com grande quantidade de elementos sujeitos a leis microscópicas da mecânica [33]. Um dos principais objetivos da mecânica estatística consiste em encontrar propriedades termodinâmicas de corpos macroscópicos a partir do movimento de elementos microscópicos tais como átomos e elétrons [33].

Um resultado fundamental da mecânica estatística diz que, quando um sistema está em equilíbrio térmico com o meio à sua volta, um estado i deste sistema ocorrerá com probabilidade

$$P_i = \frac{1}{Z} \exp\left(\frac{-E_i}{k_B T_K}\right) \quad (6.3.1.1.)$$

sendo: T_K a temperatura em graus Kelvin;

k_b a constante de Boltzmann;

Z é um fator de normalização, também conhecido como função de partição;
 E_i é a energia do sistema quando este assume o estado i .

Atendendo à condição de normalização das probabilidades, $\sum_i P_i = 1$, obtém-se

$$Z = \sum_i \exp\left(\frac{-E_i}{k_B T_K}\right) \quad (6.3.1.2.)$$

A distribuição descrita pelas equações (6.3.1.1.) e (6.3.1.2.) é conhecida como distribuição canônica ou distribuição de Gibbs. O fator exponencial $\exp(-E_i/k_B T_K)$ é comumente chamado de fator de Boltzmann. Dois aspectos importantes sobre a distribuição de Gibbs são descritos a seguir [33]:

- Estados de menor energia possuem maior probabilidade de ocorrência do que estados de maior energia;
- Para temperaturas altas, a distribuição de probabilidade de ocorrência dos estados tende a ser uniforme. À medida em que a temperatura diminui, a probabilidade de ocorrência se concentra em subconjuntos de estados cada vez menores contendo os estados de menor energia.

Quando a teoria da mecânica estatística é utilizada em algoritmos não necessariamente aplicados à mecânica, é comum substituir o produto $k_B T_K$ por uma única variável T , conhecida como pseudotemperatura. No estudo de redes neurais, o parâmetro T pode ser entendido como a variável que controla flutuações térmicas representando o efeito de “ruídos sinápticos” nas conexões entre neurônios [33].

A teoria da mecânica estatística serviu de inspiração para a criação de vários métodos estocásticos de otimização [33]. Um exemplo típico é o método conhecido como Têmpera Simulada, e que pode ser entendido como sendo uma técnica estocástica de relaxação que realiza movimentos aleatórios em uma superfície de energia visando encontrar um mínimo global [33],[58]. Este método foi inspirado nos processos químicos (ou físicos) nos quais, um dado sistema parte de uma condição inicial com altas temperaturas, e então, é lentamente esfriado (mantendo-se o equilíbrio térmico), visando atingir um estado de mínima energia no final do processo.

Nas próximas duas subseções serão estudadas duas técnicas distintas que têm como base estudos provenientes da mecânica estatística, e podem ser utilizadas em problemas envolvendo identificação e complementação de padrões. A primeira consiste em uma rede neural conhecida como Máquina de Boltzmann [33],[34], enquanto a segunda é uma abordagem para desenvolver classificadores de padrões utilizando-se um procedimento conhecido como Têmpera Determinística [33],[51],[58].

6.3.2. A Máquina de Boltzmann

O nome Máquina de Boltzmann foi dado em reconhecimento à equivalência formal entre o trabalho original de Boltzmann na mecânica estatística e o comportamento dinâmico da rede em questão [33]. A dinâmica do modelo é, de certa forma, semelhante à dinâmica de uma memória auto-associativa (como por exemplo a rede de Hopfield), porém com uma diferença significativa: a máquina de Boltzmann utiliza unidades escondidas [34]. Tais unidades permitem que o modelo aprenda a realizar correlações de ordem superior, e conseqüentemente, seja capaz de aprender, por exemplo, a solucionar o problema clássico da porta lógica XOR.

Este modelo também pode ser visto como sendo um procedimento para modelar distribuições de probabilidade existentes em um conjunto de dados. A Máquina de Boltzmann é capaz de aprender distribuições de probabilidades condicionais de forma que, durante a execução, se for fornecido um padrão incompleto, este modelo pode recuperar as informações desconhecidas. Da mesma forma, é possível realizar tarefas como o reconhecimento ou classificação de padrões.

Uma característica da máquina de Boltzmann é a utilização de conexões simétricas. Geralmente utilizam-se redes com unidades totalmente conectadas, porém, isso não é uma regra, e a única exigência consiste na simetria, ou seja, dado que w_{ij} é uma conexão sináptica da unidade j para a unidade i , tem-se necessariamente que $w_{ij} = w_{ji}$ e $w_{ii} = 0$ (não existem conexões de auto-realimentação). Além disso, os neurônios desse modelo podem ser divididos em dois grupos: os visíveis e os escondidos (Figura 13). Os neurônios visíveis provêm uma interface (entrada e saída) entre a rede e o meio no qual ela opera. Ou seja, são os neurônios visíveis que: a) recebem os padrões de entrada do sistema e b) fornecem os valores da saída do sistema. Já os neurônios escondidos não se comunicam com o meio externo à rede.

A Máquina de Boltzmann é uma rede neural formada por neurônios estocásticos. Tais neurônios possuem dois estados possíveis para a saída (geralmente utiliza-se a notação bipolar -1 e $+1$) e assumem um ou outro estado com uma certa probabilidade. Ou seja, uma dada unidade i apresentará em sua saída o valor $x_i = +1$ com probabilidade $p(h_i)$, e o valor $x_i = -1$ com probabilidade $1 - p(h_i)$:

$$h_i = \sum_j w_{ij} x_j \quad \text{e} \quad p(h_i) = \frac{1}{1 + \exp\left(\frac{-2h_i}{T}\right)} \quad (6.3.2.1.)$$

sendo T um parâmetro conhecido como pseudotemperatura.

Para a aprendizagem desse modelo, deseja-se encontrar valores para as conexões w_{ij} de forma que os estados das unidades visíveis passem a apresentar uma certa distribuição de probabilidade desejada [34].

Seja x o vetor que caracteriza o estado de todas as unidades da rede, define-se x_α como sendo a parte de x que se refere apenas às unidades visíveis, enquanto x_β é a parte de x que se refere apenas às unidades escondidas.

Figura 13: Arquitetura da máquina de Boltzmann; o modelo possui um grupo de unidades visíveis e um grupo de unidades escondidas.

Durante o treinamento, a máquina de Boltzmann opera, para cada padrão de treinamento fornecido, em duas fases distintas, são elas:

- *Fase positiva:* Nesta fase primeira fase, a rede opera com os estados das unidades visíveis fixados a partir dos padrões de entrada. Ou seja, durante essa fase, x_α assume os valores dos padrões de treinamento fornecidos.
- *Fase negativa:* Nesta fase, todas as unidades da rede estão livres para assumir quaisquer estados. Ou seja, x_α é inicializado com os valores de entrada, porém nos instantes seguintes os valores de x (ambos x_α e x_β) são obtidos a partir de (6.3.2.1).

Vale lembrar que tanto na fase positiva quanto na fase negativa, as unidades escondidas sempre operam livremente. Em outras palavras, independentemente da fase de operação da rede, os valores de x_β sempre são obtidos a partir de (6.3.2.1).

A idéia central da adaptação dos pesos durante o treinamento visa modificar os pesos na direção que minimiza a diferença entre as duas correlações médias indicadas a seguir:

- $\langle \rho_{ji}^+ \rangle$: correlação média ($\langle x_i x_j \rangle$) entre os estados dos neurônios i e j quando a rede opera na fase positiva
- $\langle \rho_{ji}^- \rangle$: correlação média ($\langle x_i x_j \rangle$) entre os estados dos neurônios i e j quando a rede opera na fase negativa

Ou seja, a adaptação dos pesos é dada por

$$\Delta w_{ij} = \varepsilon (\langle \rho_{ji}^+ \rangle - \langle \rho_{ji}^- \rangle) / T \quad (6.3.2.2.),$$

sendo i e j neurônios quaisquer independentemente de serem visíveis ou escondidos;

ε é um parâmetro de aprendizagem;

e T é a pseudotemperatura.

O processo da têmpera simulada é utilizado no treinamento de forma que, inicialmente, o valor utilizado para a pseudotemperatura T é alto, e a ordem com que as unidades são ativadas é aleatória. Cada unidade pode assumir um valor -1 ou $+1$ com uma certa probabilidade $p_i(h_i)$, expressa por uma função sigmoideal (6.3.2.1) de forma que a probabilidade da rede assumir um estado x é dada pela distribuição de Gibbs. À medida que o treinamento vai sendo realizado, o valor de T é lentamente diminuído, e os estados de menor energia da rede aos poucos vão sendo favorecidos.

Analisando-se a regra de aprendizagem (6.3.2.2.) da máquina de Boltzmann, percebe-se que esta possui dois termos. O primeiro se refere à fase positiva (com os estados das unidades visíveis determinados pelos padrões de treinamento), e pode ser entendido como uma aprendizagem Hebbiana. Já o segundo termo, referente à fase negativa, pode ser visto como um termo de “esquecimento Hebbiano” [34]. A utilização desses dois termos é crucial para a aprendizagem do modelo, e consiste numa forma de mostrar às unidades escondidas a diferença entre a atividade da rede operando livremente e operando sob a influencia de estímulos externos [33],[34]. Se apenas o termo de aprendizagem Hebbiana ($\langle \rho_{ji}^+ \rangle$) fosse utilizado, então flutuações de sinais geradas internamente na rede seriam aprendidas pelo modelo, e consequentemente degradariam a aprendizagem dos sinais externos (padrões de treinamento) aplicados ao modelo [34].

A dinâmica da máquina de Boltzmann envolve um custo computacional elevado pois, sempre que um padrão é apresentado à rede (seja no treinamento ou na execução), é preciso esperar um tempo até que o modelo entre em equilíbrio térmico. Alguns comentários sobre a máquina de Boltzmann são feitos a seguir:

- Um aspecto interessante reside no fato do modelo ser capaz de aprender distribuições de probabilidade desejadas.

- Um outro ponto interessante é o fato de ser possível garantir que o mínimo global da função de energia da rede foi atingido. Porém, o procedimento utilizando a idéia da tmpera simulada consideravelmente lento [33].
- O custo computacional do modelo elevado principalmente por dois motivos: (a) a necessidade do modelo operar em duas fases distintas (positiva e negativa); (b) toda vez que um padro fornecido ao modelo, um custo computacional relativamente elevado gasto at que o modelo atinja um equilbrio trmico.
- Uma das limitaes deste modelo consiste no fato de que os neurnios estocsticos utilizados podem assumir apenas valores binrios (ou bipolares), o que no interessante para o nosso trabalho.

A mquina de Boltzmann surgiu na dcada de 80, e foi o primeiro modelo de rede neural estocstica a ser amplamente estudado na literatura. Desde ento, diferentes variaes deste modelo tm sido propostas com o objetivo de acelerar a aprendizagem e execuo do modelo, como por exemplo: a mquina de Helmholtz [23], e a mquina de Boltzmann determinstica [33],[34]. Porm, com relao aos objetivos desta pesquisa, todos esses modelos apresentam uma mesma limitao indesejada nesta pesquisa e citada para a mquina de Boltzmann: o fato de trabalharem com padres de entrada binrios ou bipolares.

O prximo modelo a ser apresentados consiste em uma abordagem estatstica que aceita padres contnuos na entrada, e utiliza um procedimento de otimizao conhecido como tmpera determinstica.

6.3.3. Tmpera Determinstica para Aprendizagem por Quantizao Vetorial

Enquanto a tmpera simulada uma tcnica estocstica de relaxao que realiza movimentos aleatrios em uma superfcie de energia visando encontrar um mnimo global, a tmpera determinstica uma tcnica que incorpora a aleatoriedade do processo na funo de energia ou no custo desta funo, que ento otimizado de forma determinstica [33],[58].

Esta Seo descreve uma abordagem para projeto de classificadores de padres, dentro da qual o procedimento da tmpera determinstica [51],[58] pode ser utilizado no algoritmo para aprendizagem por quantizao vetorial (ou simplesmente aprendizagem VQ)⁴.

⁴ O termo VQ normalmente uma denominao genrica para os procedimentos estatsticos de quantizao vetorial. J o termo LVQ costuma ser utilizado especificamente para denominar modelos como o que foi descrito na Seo 6.2.

Seja $\mathbf{v}_i$ o i -ésimo vetor (padrão) de entrada e $\mathbf{w}_j$ um protótipo representando um grupo j de padrões, define-se uma medida de distorção, $d(\mathbf{v}_i, \mathbf{w}_j)$ como sendo a distância Euclidiana entre $\mathbf{v}_i$ e $\mathbf{w}_j$:

$$d(\mathbf{v}_i, \mathbf{w}_j) = \|\mathbf{v}_i - \mathbf{w}_j\| \quad (6.3.3.1.)$$

A distorção esperada, D , será dada por:

$$\begin{aligned} D &= \sum_i \sum_j P(\mathbf{v} = \mathbf{v}_i, \mathbf{w} = \mathbf{w}_j) \cdot d(\mathbf{v}_i, \mathbf{w}_j) \\ &= \sum_i P(\mathbf{v} = \mathbf{v}_i) \sum_j P(\mathbf{w} = \mathbf{w}_j | \mathbf{v} = \mathbf{v}_i) \cdot d(\mathbf{v}_i, \mathbf{w}_j) \end{aligned} \quad (6.3.3.2.)$$

A probabilidade condicional $P(\mathbf{w} = \mathbf{w}_j | \mathbf{v} = \mathbf{v}_i)$ é conhecida como probabilidade de associação e informa qual a probabilidade do protótipo $\mathbf{w}_j$ e o padrão de entrada $\mathbf{v}_i$ estarem associados (ou pertencerem à mesma classe).

O treinamento deste modelo utiliza a idéia da têmpera determinística e consiste em um problema de otimização no qual se deseja encontrar os parâmetros do modelo (ou protótipos) e a distribuição de probabilidade que minimiza a distorção esperada sujeita a um determinado nível de aleatoriedade. Uma boa medida para a aleatoriedade a ser utilizada é a entropia de Shannon descrita a seguir [33],[51],[58]:

$$H(\mathbf{x}, \mathbf{w}) = \sum_i \sum_j P(\mathbf{v} = \mathbf{v}_i, \mathbf{w} = \mathbf{w}_j) \log P(\mathbf{v} = \mathbf{v}_i, \mathbf{w} = \mathbf{w}_j) \quad (6.3.3.3.)$$

O problema de otimização descrito pode ser expresso da seguinte forma:

$$\text{minimizar o Lagrangiano } F = D - TH \quad (6.3.3.4.)$$

sendo T o multiplicador de Lagrange.

Para qualquer H positivo, a otimização de (6.3.3.4.) com relação às probabilidades de associação resulta na distribuição de Gibbs [33],[51],[58]:

$$P(\mathbf{w} = \mathbf{w}_j | \mathbf{v} = \mathbf{v}_i) = \frac{1}{Z_i} \exp\left(-\frac{d(\mathbf{v}_i, \mathbf{w}_j)}{T}\right) \quad (6.3.3.5.)$$

sendo Z_i a função de partição dada por:

$$Z_i = \sum_j \exp\left(-\frac{d(\mathbf{v}_i, \mathbf{w}_j)}{T}\right) \quad (6.3.3.6.)$$

Finalmente, o algoritmo da têmpera determinística para aprendizagem VQ pode ser descrito da seguinte forma:

- 1) Inicializar a pseudotemperatura T com um valor elevado;

- 2) Fixar os valores dos protótipos e calcular as probabilidades de associação através da distribuição de Gibbs;
- 3) Fixar as probabilidades de associação e minimizar $F = D - TH$ com relação aos protótipos;
- 4) Reduzir a pseudotemperatura T e voltar ao passo 2

Um possível critério de parada consiste na avaliação da entropia. Nesse caso, o algoritmo se repete até que a entropia H seja suficientemente pequena. Em outras palavras, para um valor de H_f predeterminado, o algoritmo se repete até que a condição $H < H_f$ seja verdadeira.

Algumas observações relevantes sobre o algoritmo descrito nesta seção serão feitas a seguir:

- O algoritmo da têmpera determinística converge em uma velocidade significativamente maior do que o algoritmo da têmpera simulada. Porém, é importante salientar que, diferentemente do procedimento da têmpera simulada, não se têm garantias de que o procedimento da têmpera determinística encontrará um mínimo global [33].
- Rose [58] mostra que essa abordagem pode ser utilizada tanto de forma supervisionada quanto não supervisionada.
- Durante a aprendizagem, à medida em que a temperatura diminui, as probabilidades de associação vão ficando mais “rígidas”. Ou seja, com a diminuição da temperatura, tais probabilidades adquirem valores cada vez mais próximos de 1 para os protótipos mais próximos dos padrões de entrada, e valores cada vez mais próximos de 0 para os protótipos distantes dos padrões de entrada. Em outras palavras, se o procedimento de treinamento é repetido até que a temperatura seja igual ou tendente a zero, então o resultado final será um classificador “rígido” do tipo classificador do vizinho mais próximo.
- Por outro lado, se o procedimento for interrompido a uma temperatura significativamente maior do que zero, então o resultado final será um classificador estocástico permitindo que protótipos mais distantes tenham valores significativos para as probabilidades de associação. Este último caso pode ser interessante em tarefas de classificação nas quais o custo do erro é bastante elevado (um exemplo típico é o diagnóstico médico, no qual, em determinadas situações, um erro pode acarretar custo para a saúde do paciente muito mais elevado do que a dúvida).
- Conforme dito anteriormente, deseja-se que o módulo estimador de contexto espacial seja capaz de identificar a sequência, fornecer um protótipo e fornecer uma

probabilidade de associação. Sendo assim, a abordagem descrita nesta subseção consiste, dentre os modelos estudados, a que mais se aproxima da opção desejada para fazer o papel do módulo estimador de contexto espacial.

- As principais limitações desta abordagem com relação ao que se deseja para a função de módulo estimador de contexto espacial são descritas a seguir:
 - a) o tempo de convergência do procedimento da tmpera determinstica, apesar de ser menor do que o da tmpera simulada, ainda significativamente maior do que o algoritmo LVQ pois, envolve, para cada iterao do algoritmo, avaliaes das probabilidades de associao assim como uma otimizao (minimizao) de $F = D - TH$ com relao aos prottipos.
 - b) Da forma como foi proposto por Rose [58] e Miller et al. [51], o procedimento de treinamento segue at que a temperatura seja zero ou bem prxima de zero. Ou seja, o modelo final encontrado consiste em um classificador “rgido” do vizinho mais prximo. No caso do mdulo estimador de contexto espacial, deseja-se encontrar um classificador mais estocstico. A princpio, uma possibilidade para se obter, como resultado final do treinamento, classificadores mais estocsticos, seria utilizar um critrio de parada que interrompa o treinamento antes da temperatura ser zero ou tendente a zero, visando diminuir a quantidade de classificaes erradas em troca do surgimento de situaes de dvidas (probabilidades de associao elevadas para mais de um prottipo).

Finalmente, tomando como base os estudos sobre contexto e modularidade, assim como os modelos apresentados nos Captulos 5 e 6, foi possvel desenvolver um sistema modular com tratamento de contexto espacial e temporal para reproduo de trajetrias, e com boa capacidade de generalizao. Esse sistema ser apresentado no captulo a seguir.

7. DESENVOLVIMENTO DO SISTEMA PROPOSTO

Neste capítulo será apresentado o sistema modular neural que foi implementado para reproduzir seqüências temporais. Inicialmente, serão apresentados os modelos utilizados como módulo estimador de contexto espacial e como módulo de reprodução de seqüências temporais. Após a definição dos módulos separadamente, serão apresentados a arquitetura e algoritmo de treinamento do sistema modular completo. Por último, serão apresentados alguns experimentos preliminares utilizados para que fossem feitos ajustes e pequenas modificações no sistema inicialmente proposto de forma a encontrar a configuração final para o sistema implementado.

7.1. O Módulo Estimador de Contexto Espacial (MECE)

Conforme mencionado anteriormente, a tarefa do MECE consiste em estimar uma informação de contexto espacial que identifique a seqüência que conecta os pontos (inicial e alvo) fornecidos na entrada. Ou seja, a função deste módulo consiste em, dada a entrada $\{x'_0, x'_t\}$, fornecer um protótipo de contexto espacial $\{x_0, x_t\}$ que identifica a seqüência que conecta os pontos entrados. A princípio, essa tarefa poderia ser realizada por diferentes modelos de redes neurais. Contudo, além da identificação da seqüência e do fornecimento do protótipo de contexto espacial, também se deseja que o módulo estimador de contexto espacial se caracterize por:

- Fornecimento de uma probabilidade associada à identificação da seqüência, de forma que seja possível indicar o grau de certeza do módulo, e conseqüentemente, desenvolver procedimentos apropriados para os casos de dúvida.
- Treinamento e execução rápidos.

Dos modelos estudados no capítulo anterior, o algoritmo LVQ1 é o que apresenta treinamento mais rápido, além de ser capaz de fornecer protótipos dos padrões de entrada, porém, não fornece probabilidades associadas à identificação das seqüências. Já o algoritmo da têmpera determinística para quantização vetorial é um algoritmo que além de identificar a seqüência e fornecer os protótipos, também considera probabilidades de associação, porém

apresenta um treinamento significativamente mais demorado do que o LVQ1. Sendo assim, o modelo escolhido para fazer o papel do módulo estimador de contexto espacial se baseia no treinamento LVQ1, porém acrescido de algumas modificações inspiradas na mecânica estatística e em classificadores estatísticos. Tais modificações possibilitaram trabalhar com probabilidades de associação na identificação das seqüências, conforme era desejado.

As idéias básicas para a concepção do modelo são dadas a seguir:

- O modelo terá uma arquitetura semelhante àquela apresentada no capítulo anterior para a rede LVQ1;
- O algoritmo utilizado para aprender os protótipos será baseado naquele adotado pela rede LVQ1 do capítulo anterior. Porém, conforme será visto mais adiante, o procedimento de aprendizagem foi modificado de forma que o algoritmo pudesse construir adequadamente os protótipos de contexto espacial das seqüências;
- Após o treinamento tipo LVQ1, uma segunda etapa é realizada. Nessa segunda etapa, a distribuição de Gibbs assim como alguns procedimentos encontrados em classificadores estatísticos são utilizados para que a rede possa apresentar em sua saída, probabilidades de associação em função da distância entre o padrão de entrada e o protótipo vencedor.

Sendo assim, o ponto de partida para o desenvolvimento do módulo estimador de contexto espacial será a rede LVQ1. Porém, para que o MECE não apresente uma classificação muito rígida do tipo classificador do vizinho mais próximo, serão utilizadas ferramentas estatísticas para incorporar probabilidades de associação. A Figura 14 apresenta o modelo proposto para realizar o MECE.

Figura 14: Arquitetura proposta para realizar o MECE.

Assim como a rede LVQ1 apresentada no capítulo anterior, esse modelo é uma rede de propagação direta formada por duas camadas de unidades. Todas as unidades da camada de entrada estão conectadas a todas as unidades da camada de saída através dos pesos, que, na realidade, são protótipos de contexto espacial das classes (ou seqüências) aprendidos durante o treinamento. A camada de entrada está dividida em três conjuntos de unidades apenas para ressaltar que os padrões de entrada utilizados para treinar a rede são formados por três informações espaciais: o ponto inicial x_0 , um ponto intermediário x_i ($i=0,2,...,f-1$), e o ponto final x_f de uma dada seqüência. Dessa forma, para cada ponto x_i , $i = 0,1,2,...,f-1$, pertencente a uma seqüência S_j , é possível construir um padrão de treinamento $v = [x_0, x_i, x_f]$. Tal padrão é, na realidade, um conjunto resumido de informações a respeito de S_j . Sendo assim, pode-se dizer que, após o treinamento, os protótipos associados a cada uma das seqüências aprendidas conterão dois tipos de informações espaciais: a) as informações sobre os extremos $[x_0, x_f]$, e b) uma informação $\bar{x}_i$ que corresponde à média dos pontos da seqüência. Conforme será visto mais adiante, a aprendizagem da informação b deverá ser útil para melhorar a capacidade de generalização do modelo. Porém, é importante frisar que, durante a execução, serão necessárias apenas as informações referentes aos pontos inicial e alvo fornecidos, pois do contrário, estaria havendo uma mudança na formulação do problema proposto. Essa abordagem ficará mais clara quando o procedimento de execução do módulo for apresentado, mais adiante neste capítulo.

O treinamento do MECE envolve duas etapas:

- 1.) *Posicionamento dos protótipos*: Os protótipos (ou pesos) são adaptados através de um procedimento supervisionado parecido com aquele apresentado na Seção 6.2 para a rede LVQ1, porém, ao invés de se utilizar as equações descritas em (6.2.1.) e (6.2.2.), utiliza-se apenas

$$w_j(t+1) = w_j(t) + \alpha(t)[v(t) - w_j(t)] \quad (7.1.1)$$

sendo w_j o protótipo referente à seqüência S_j ;

$v = [x_0, x_i, x_f]$ o padrão de treinamento contendo os pontos inicial, i -ésimo intermediário, e o final, da seqüência S_j .

- 2.) *Ajuste das probabilidades*: Um procedimento é utilizado para obtenção de probabilidades de associação. Pois, diferentemente do LVQ1 (que é um classificador rígido), o MECE deverá se comportar como um classificador estatístico apresentando probabilidades de associação, e conseqüentemente informando o grau de “incerteza” da resposta. Antes de explicar esta segunda etapa, é necessário fazer algumas definições e comentários.

A equação (7.1.1.) é igual à primeira equação de (6.2.2.) e visa aproximar o protótipo w_j do padrão v . Dessa forma, após o treinamento, o MECE terá aprendido as informações de início e fim de seqüência, $[x_0, x_f]$, assim como a informação a respeito da média dos pontos intermediários da seqüência, $\bar{x}_i$.

Por ser um classificador estatístico, o MECE poderá apresentar em sua saída uma resposta indicando dúvida. A resposta do MECE será da seguinte forma:

- Seja l_{duv} o valor predefinido do limiar de dúvida.
- Se a probabilidade de associação do vencedor é maior que o limiar l_{duv} , então a identificação assim como o protótipo são dados pelo vencedor.
- Se a probabilidade de associação do vencedor for menor que o limiar l_{duv} , então a resposta do classificador indica dúvida. Neste caso, um procedimento adicional pode ser utilizado visando diminuir o erro esperado na classificação. Três possibilidades de procedimentos são dadas a seguir:
 - a) Utilizar conhecimentos anteriores que poderiam solucionar alguns casos de dúvidas;
 - b) Permitir que mais de uma resposta possa ser considerada pelo sistema. Por exemplo, o sistema modular completo poderia reproduzir as duas seqüências mais prováveis, e em seguida, decidir qual das duas seria a solução correta;
 - c) Apresentar um comportamento estocástico, no qual todas as saídas podem vencer (todas as seqüências aprendidas têm chances de serem reproduzidas), porém aquelas que apresentam valores maiores para as probabilidades de associação, devem vencer com mais freqüência.

Nessa pesquisa, estudou-se apenas as possibilidades b e c. A forma escolhida para que o módulo possa apresentar probabilidades de associação, foi utilizar a distribuição de Gibbs na saída da rede. Sendo assim, dado um padrão de entrada v , a probabilidade de este padrão pertencer à classe c_i será dada por:

$$P(c_i|v) = \frac{\exp(-\beta \cdot d_i)}{\sum_j \exp(-\beta \cdot d_j)} \quad (7.1.2.)$$

sendo β igual ao inverso da pseudotemperatura: $\beta = 1/T$.

c_i a classe (seqüência) identificada pela unidade i da camada de saída.

d_i a distância euclidiana entre o padrão de entrada v e o protótipo w_i da unidade i de saída:

$$d_i = \|x - w_i\| \quad (7.1.3.)$$

Essa escolha foi feita devido às características da distribuição de Gibbs:

- Essa distribuição favorece os estados de menor energia. No caso do MECE, os protótipos mais próximos do padrão de entrada deverão apresentar maiores probabilidades de associação;
- Pode ser ajustada pelo parâmetro T (ou pelo parâmetro β). Sendo assim, após a aprendizagem dos protótipos, é necessária uma segunda etapa do treinamento na qual o parâmetro β deve ser ajustado conforme algum critério.

A princípio, a distribuição de Gibbs poderia ser implementada utilizando-se o parâmetro T . Porém, existem basicamente dois motivos que justificam a implementação de (7.1.1.) utilizando-se a variável β no lugar da pseudotemperatura T , são eles:

- A distribuição uniforme pode ser conseguida utilizando-se $\beta = 0$. No caso da pseudotemperatura, seria necessário fazer $T \rightarrow \infty$;
- Evita a ocorrência de divisão por zero na implementação.

O procedimento proposto e adotado para ajustar o valor de β visa minimizar uma função “custo” definida da seguinte forma:

Seja c_v a classificação sugerida (vencedora) pela rede, e c_s a classe (seqüência) correta correspondente ao padrão de entrada. A função “custo” L , ao identificar um padrão de entrada, é dada por:

$$L(c_v, c_s) = \begin{cases} 0 & \text{se } c_v = c_s \text{ e } P(c_v | \mathbf{v}) > l_{div} \text{ (classificação correta)} \\ 1 & \text{se } c_v \neq c_s \text{ e } P(c_v | \mathbf{v}) > l_{div} \text{ (classificação errada)} \\ (1 - P(c_s | \mathbf{v})) & \text{se } P(c_v | \mathbf{v}) < l_{div} \end{cases} \quad (7.1.4.)$$

Ou seja, para os casos em que não há dúvidas (a probabilidade de associação do vencedor é maior que o limiar de dúvida) o custo será igual a 0 se a classificação for correta e igual a 1 se a classificação for errada. No caso de haver dúvida (a probabilidade de associação do vencedor é menor que o limiar de dúvida) então o custo será igual a 1 menos a probabilidade de associação da classificação correta (que equivale a dizer que o custo será igual à probabilidade de erro).

O custo total esperado, L_{total} , é calculado para um certo conjunto de testes previamente construído. L_{total} é calculado pela soma dos custos obtidos para cada padrão do conjunto de testes. Sendo assim, o valor de β deve ser escolhido de forma a minimizar o valor de L_{total} .

O conjunto de testes é construído da seguinte forma:

- Escolhe-se aleatoriamente uma seqüência;
- Escolhe-se aleatoriamente um ponto intermediário x_i ($i = 0, 1, 2, \dots, f-1$) qualquer desta seqüência;

- c) Utilizando-se o ponto escolhido, em conjunto com os pontos inicial (x_0) e final (x_f) da sequência escolhida, cria-se 3 padrões distintos para o conjunto de testes:
 - $[x_0, x_i, x_f]$, $[x_i, x_i, x_f]$ e $[x_0, x_i, x_i]$;
- d) repetir as etapas a, b e c até que o conjunto de testes atinja o tamanho desejado.

Após minimizar L_{total} , o treinamento do módulo estimador de contexto espacial estará completo, e os protótipos aprendidos por esse módulo serão utilizados para treinar o módulo de reprodução de sequência temporal que será explicado na Seção 7.2.

A Figura 15 apresenta o algoritmo completo de treinamento do MECE.

- b) Inicializar $N = 0$ (número de apresentações dos padrões de treinamento)
 N_{max} (número de máximo de apresentações dos padrões de treinamento)
 α (taxa de aprendizagem)
 W (os protótipos são inicializados com valores iguais a zero)
- c) Enquanto $N < N_{max}$
 - Escolher uma sequência S_j aleatoriamente do conjunto de treinamento
 - Fazer x_0 = ponto inicial de S_j
 x_f = ponto final de S_j
 x_i = ponto intermediário de S_j escolhido aleatoriamente, $i=0,1,2,...,f-1$.
 Montar o padrão de entrada $v(t)=[x_0, x_i, x_f]$
 - Aplicar a regra de aprendizagem:

$$w_j(t+1) = w_j(t) + \alpha(t)[v(t) - w_j(t)]$$
 - Incrementar N
- d) Construir o conjunto de testes.
- e) Encontrar o valor β que minimiza o custo total L_{total} para o conjunto de testes criado.

Figura 15: Algoritmo utilizado para o treinamento do MECE.

Durante a execução do MECE, é fornecido o padrão de entrada $[x'_0, x'_f]$ contendo as informações a respeito dos pontos inicial e ponto final da sequência a ser reproduzida. Com relação ao ponto x'_i , este ponto será determinado da seguinte forma:

- 1) Para as situações em que se deseja generalizar, partindo de pontos x'_0 diferentes do x_0 visto durante o treinamento, então o padrão de entrada do MECE será $[x'_0, x'_0, x'_f]$. Ou seja, faz-se $x'_i = x'_0$;
- 2) Para as situações em que se deseja generalizar, informando um ponto alvo x'_f diferente do x_f visto durante o treinamento, então o padrão de entrada do MECE será $[x'_0, x'_f, x'_f]$. Ou seja, faz-se $x'_i = x'_f$;

- 3) Se ambos os pontos $\mathbf{x}'_o$ e $\mathbf{x}'_f$ são diferentes dos pontos $\mathbf{x}_o$ e $\mathbf{x}_f$ vistos durante o treinamento, ou se não se tem esse tipo de informação, ou ainda, se o padrão de entrada $[\mathbf{x}'_o, \mathbf{x}'_f]$ é inteiramente igual ao padrão visto durante o treinamento $[\mathbf{x}_o, \mathbf{x}_f]$, então, existem diferentes possibilidades:
- a) Escolher aleatoriamente um ponto ($\mathbf{x}'_o$ ou $\mathbf{x}'_f$) e fazer $\mathbf{x}'_i$ igual a este ponto escolhido;
 - b) Fazer $\mathbf{x}'_i = (\mathbf{x}'_o + \mathbf{x}'_f)/2$ (média dos pontos fornecidos);
 - c) Fazer $\mathbf{x}'_i = 0$;

Nesta pesquisa, foram consideradas apenas as situações 1, 2 e 3a. A seguir serão apresentados alguns experimentos preliminares realizados como o MECE.

7.1.1. Experimentos Preliminares com o MECE

Esta Seção apresenta alguns experimentos preliminares realizados com o modelo proposto para desempenhar o papel de MECE. Tais experimentos foram realizados com os seguintes objetivos:

- Estudar o comportamento do sistema em função do parâmetro β ;
- Realizar uma primeira avaliação quanto ao desempenho e adequação do modelo;

Utilizou-se um conjunto contendo seis trajetórias abertas, definidas no espaço tridimensional (Figura 16). Conforme foi proposto no Capítulo 2, diferentes situações de ambigüidades podem ser observadas. Para este conjunto, tem-se que:

- Todas as trajetórias são formadas por 11 pontos
- As trajetórias 1 e 2 partem de pontos distintos e atingem o mesmo alvo;
- A trajetória 3 não possui pontos compartilhados com nenhuma outra trajetória;
- As trajetórias 4 e 5 partem do mesmo ponto inicial e atingem alvos diferentes;
- As trajetórias 5 e 6 compartilham um ponto intermediário;

Foi feito treinamento do MECE da seguinte forma:

- Executou-se apenas os passos a, b e c do algoritmo apresentado na Figura 15;
- Foi utilizada uma taxa de aprendizagem constante, $\alpha = 0,05$;
- O critério de parada foi dado pelo número máximo de iterações, $N_{\max} = 200$;
- O conjunto de testes construído no passo c do algoritmo de treinamento continha 300 padrões;

Em seguida, foi feito um estudo visando avaliar o comportamento do sistema (observando-se a quantidade de erros e dúvidas) em função do parâmetro β . Sendo assim, variou-se o valor de β (de 0 a 10 com incrementos de 0,5) e, para cada valor, foram calculadas as probabilidades de associação na identificação de cada padrão pertencente ao conjunto e foram contabilizados os números de erros e dúvidas. O resultado obtido pode ser observado na Figura 17. O procedimento utilizado para ajustar o parâmetro β será comentado mais adiante.

Figura 16: Conjunto formado por seis trajetórias e utilizado em experimentos preliminares com o módulo estimador de contexto espacial.

Inicialmente (para $\beta = 0$) o número de dúvidas é igual ao tamanho do conjunto de testes pois inicialmente a distribuição é uniforme. A medida que o valor de β aumenta, o número de dúvidas decresce enquanto o número de erros aumenta. Para o caso de uma classificação rígida (fazendo β suficientemente grande para que não haja dúvidas na classificação) o número final de erros obtidos para esse conjunto de testes foi igual a 26.

Figura 17: Estudo do número de erros e de dúvidas em função do parâmetro β ($1/T$) na classificação obtida com o MECE.

Por último, um segundo conjunto de testes foi construído com o objetivo de avaliar o desempenho do modelo para identificar os padrões em situações treinadas. Ou seja, este conjunto foi formado por todos os 66 padrões $[x_0, x_i, x_f]$ possíveis, sendo x_0 e x_f os verdadeiros pontos inicial e final de cada seqüência utilizados durante o treinamento, e x_i um ponto qualquer da seqüência em questão. Para este segundo conjunto de testes, o modelo proposto foi capaz de identificar corretamente 100% dos padrões de entrada.

Os resultados obtidos com esses experimentos preliminares sugerem que:

- O modelo proposto pode ser utilizado para realizar a tarefa de estimativa de contexto espacial pois apresentou um bom desempenho na identificação das seqüências da Figura 16:
 - Sendo 91,3% de acertos no primeiro conjunto de testes;
 - e 100% de acertos no segundo conjunto de testes;
- A abordagem estatística utilizando-se a distribuição de Gibbs possibilita a obtenção de um classificador estatístico que pode ser “ajustado” pelo parâmetro β . Vale lembrar que este ajuste é dependente do problema em questão.

É correto afirmar que a avaliação destes resultados preliminares é positiva. Sendo assim, ficou definido que o modelo proposto nesta Seção, será utilizado para realizar o papel do MECE. Na próxima Seção será proposto o modelo estudado o modelo para ser utilizado como módulo de reprodução de seqüências temporais.

7.2. O Módulo de Reprodução de Seqüências Temporais

O Módulo de Reprodução de Seqüências Temporais (MRST) deverá ser uma rede capaz de reproduzir as seqüências temporais identificadas pelo MECE. A princípio, diferentes arquiteturas poderiam ser utilizadas para realizar o papel do MRST, porém, conforme já foi mencionado anteriormente, será utilizada uma rede neural parcialmente recorrente baseada na arquitetura proposta por D'Arbo Junior para desempenhar esta função. Alguns comentários a respeito dessa escolha são dados a seguir:

- Nos estudos de D'Arbo Junior seu modelo apresentou bom desempenho (comparado a outros modelos de redes parcialmente recorrentes) para aprendizagem de seqüências temporais em situações treinadas.
- Com a utilização deste modelo, será possível ter uma idéia bem clara no que diz respeito à melhoria em generalização conseguida através da abordagem modular com tratamento separado de informações de contexto espacial e temporal proposta nesta pesquisa, pois, os resultados serão comparados àqueles obtidos com o modelo de D'Arbo Junior operando sozinho, no qual a generalização⁵ foi muito pobre.
- O fato do modelo apresentar pobre capacidade de generalização não acarreta em grandes problemas pois, conforme dito anteriormente, dentro da abordagem proposta nesta pesquisa, o principal responsável pela capacidade de generalização é o MECE.

O algoritmo de treinamento assim como algumas modificações que foram realizadas no modelo serão apresentados nas sub-seções a seguir.

7.2.1. Acrescentando Unidades Extras de Contexto

Em todos os experimentos realizados por D'Arbo Junior [22] os conjuntos de treinamento utilizados eram todos formados por trajetórias de pontos definidos em um espaço de muitas dimensões. Por exemplo, as seqüências aprendidas para controlar a trajetória de um braço mecânico, eram formadas por padrões contendo 15 dimensões (3 entradas referentes à posição espacial, 6 entradas referentes aos ângulos e 6 entradas referentes aos torques). Porém,

⁵ Nos estudos de D'Arbo Junior, duas situações de generalização foram testadas: (a) para entradas ruidosas; e (b) quando, um ponto intermediário é fornecido no lugar do ponto inicial ou do ponto final.

alguns estudos preliminares realizados nesta pesquisa mostraram que a utilização do modelo de D'Arbo Junior para a aprendizagem de seqüências formadas por padrões contendo poucas entradas (por exemplo, padrões com apenas três entradas) pode apresentar alguns problemas.

Basicamente, o problema que ocorre na aprendizagem de padrões com poucas entradas é a falta de unidades e de conexões de contexto para processar informações temporais. Ou seja, da forma como foi proposto por D'Arbo Junior, para aprender padrões definidos, por exemplo, no espaço tridimensional, a arquitetura da rede seria definida com apenas 3 unidades na entrada e 3 unidades na saída. O que implica dizer que haveriam apenas 9 conexões de realimentação intercadas (da saída para a entrada) e outras nove intracamadas (dentro da camada de entrada). Neste caso, a quantidade de informações temporais processadas pela rede seria insuficiente para aprender seqüências grandes.

A solução proposta nesta pesquisa para resolver o problema descrito consiste em utilizar unidades extras na camada de contexto (Figura 18).

Figura 18: Adaptação proposta para o modelo de D'Arbo Junior. Foram incluídas unidades extras na camada de contexto visando melhorar o desempenho do modelo na aprendizagem de seqüências contendo padrões com poucas entradas.

É importante frisar que, assim como no modelo original, as seguintes regras para construção do modelo continuam valendo:

- Todas as unidades (tanto as de entrada quanto as extras) da camada de contexto continuam sendo totalmente interconectadas;
- Além disso, todas as unidades da camada de contexto estão totalmente conectadas à camada escondida;
- E por último, a camada de saída está totalmente interconectada (via conexões de realimentação) a todas as unidades (de entrada e extras) da camada de contexto.

Em síntese, as unidades extras são conectadas de maneira idêntica às unidades de entrada no modelo de D'Arbo Junior. A diferença é que as informações de entrada externas ao MRST são apresentadas apenas às unidades de entrada. Com esta modificação, é possível aumentar a quantidade de conexões de realimentação para processamento de informações temporais, simplesmente aumentando-se a quantidade de unidades extras na camada escondida.

A próxima subseção apresenta as equações, assim como o algoritmo de treinamento e o procedimento de execução do modelo apresentado, e que será utilizado para realizar o papel do MRST.

7.2.2. Treinamento e Execução do MRST

Para o modelo da Figura 18, cada característica do padrão de entrada é fornecida para uma única unidade de entrada da camada de contexto. Além disto, diferentemente da forma como foi proposto por D'Arbo Junior, o padrão de entrada (contexto espacial) apresentado ao modelo permanece fixado na entrada da rede durante toda a reprodução (ou aprendizagem) da seqüência.

O estado de ativação da i -ésima unidade da camada de contexto é dado por:

$$i_i(t+1) = f(q_i^c(t)) \quad (7.2.1.)$$

sendo
$$q_i^c(t) = v_i + \sum_{j=1}^{n_s} k_{ij} o_j(t) + \sum_{j=1}^{n_c} w_{ij}^c i_j(t)$$

v_i a i -ésima característica do padrão de entrada v se i é uma unidade de entrada,

$x_i = 0$ se i é uma unidade extra de contexto.

n_c o número de unidades da camada de contexto;

n_s o número de unidades da camada de saída;

k_{ij} o peso da conexão de realimentação da unidade j da saída para a unidade i da entrada;

w_{ij}^c o peso da conexão da unidade j para a unidade i , ambas da camada de contexto;

$o_j(t)$ o estado de ativação da unidade j de saída no instante de tempo t .

A função $f(.)$ é uma função sigmoideal definida da seguinte forma:

$$f(x) = a.tgh(b.v) \quad (7.2.2.)$$

sendo a um parâmetro que ajusta os limites superior e inferior da função,

b um parâmetro que ajusta a inclinação da função,

v é o valor na entrada da unidade.

A ativação da i -ésima unidade da camada escondida é dada por:

$$h_i(t) = f(q_i^h(t)) \quad (7.2.3.)$$

sendo $q_i^h(t) = \sum_{j=1}^{n_c} w_{ij}^h i_j(t)$

w_{ij}^h o peso da conexão da unidade j da camada de contexto para a unidade i da camada escondida;

$i_j(t)$ o estado de ativação da unidade j da camada de contexto no instante de tempo t .

E finalmente, o estado de ativação da i -ésima unidade de saída é calculado da seguinte forma:

$$o_i(t) = f(q_i^s(t)) \quad (7.2.4.)$$

sendo $q_i^s(t) = \sum_{j=1}^{n_h} w_{ij}^s h_j(t)$

w_{ij}^s o peso da conexão da unidade escondida j para a unidade i de saída;

n_h o número de unidades escondidas;

$h_j(t)$ o estado de ativação da unidade escondida j no instante de tempo t .

Para o treinamento do modelo, utilizou-se o algoritmo de retropropagação do erro. Este é, basicamente, um algoritmo de gradiente descendente que visa reduzir, através de um processo iterativo, a diferença entre as saídas desejada e obtida. Porém, um dos grandes problemas deste tipo de algoritmo consiste no tempo de convergência (que para muitos problemas é bastante elevado). Existem várias abordagens na literatura visando a obtenção de um processo mais rápido de convergência durante a aprendizagem [33],[34],[47],[50],[68],[69]. Dois possíveis caminhos para atingir este objetivo são:

- **Métodos de Segunda ordem.** Tais métodos geralmente conseguem reduzir drasticamente o número de épocas necessárias para a convergência. Em contrapartida, surgem algumas dificuldades tais como a necessidade de armazenamento de grande quantidades de dados (para cálculos de Jacobianos, Hessianas, etc...) assim como dificuldades matemáticas para a inversão de matrizes que em vários casos não são bem condicionadas.
- **Taxa de aprendizagem e momento variáveis.** Um valor alto para a taxa de aprendizagem pode acelerar o processo quando a busca ainda está longe de qualquer ótimo, ou quando o processo está atravessando uma região plana da função do erro (ou seja, região onde o gradiente possui um módulo muito pequeno). Por outro lado, quando o processo está próximo da solução desejada, um valor alto para a taxa de aprendizagem pode

provocar oscilações na função do erro, piorando a convergência, ou até mesmo afastando o problema da solução desejada. Sendo assim, para que um algoritmo seja eficiente e rápido, é preciso que este seja capaz de variar dinamicamente a taxa de aprendizagem e o fator de momento dependendo da região em que se encontra durante o processo de aprendizagem [69].

Nesta pesquisa, foram estudados e testados alguns métodos de treinamento utilizando taxa de aprendizagem variável para aceleração do algoritmo de retropropagação. Estes serão comentados a seguir:

- a) Um método simples que utiliza a idéia de taxa de aprendizagem variável consiste em aumentar ou diminuir por um fator fixo, os valores da taxa de aprendizagem e do momento baseando-se em simples observações dos sinais de erro na saída da rede [33],[34]. Um dos problemas dessa abordagem é que, apesar de conseguir reduzir o tempo para convergência em vários casos, outras vezes é comum de acontecer divergências drásticas durante o processo [69].
- b) De fato, Yu et al. [68] observa que os valores ótimos para a taxa de aprendizagem e para o momento variam quase que aleatoriamente de uma iteração para outra durante o processo, e sendo assim, sugerem um método que tenta encontrar uma taxa de aprendizagem ótima a cada iteração do treinamento, avaliando-se a derivada da função erro com relação à taxa de aprendizagem.
- c) Um outro método estudado consiste no procedimento estocástico conhecido como *busque-então-convirja* (*search-then-converge*). Este método utiliza a idéia de que, no início do treinamento é necessário fazer uma busca estocástica com passos grandes (utilizando valores altos para a taxa de aprendizagem) [33]. Porém, os passos devem ser reduzidos ao longo do treinamento, ou seja, a cada nova iteração no treinamento, a taxa de aprendizagem é atualizada de acordo com uma função monotonicamente decrescente.

Depois de treinar o modelo proposto utilizando-se cada um desses métodos comentados, e para os conjuntos de trajetórias estudados nesta dissertação, o procedimento *busque-então-convirja* foi o que apresentou os melhores resultados (menor tempo de treinamento) e sendo assim, foi o método adotado para o algoritmo de treinamento em todos os experimentos apresentados no Capítulo 8.

As conexões de realimentação são tratadas como se fossem conexões diretas vindas de unidades virtuais. E tais unidades são, na realidade, cópias das unidades reais, porém, atrasadas um instante de tempo. Dessa forma, é possível utilizar o algoritmo de retropropagação para treinar a rede recorrente do MRST. Além disto, o algoritmo de treinamento do MRST, utiliza

uma técnica conhecida como professor forçante [34]. A utilização dessa técnica é comum em redes neurais recorrentes, e consiste basicamente em, após a correção dos pesos, forçar para que as unidades de saída da rede assumam valores corretos de ativação. Um procedimento utilizando o professor forçante será detalhado e comentado a seguir:

- a) durante a apresentação de uma seqüência, no instante t , o padrão x_t é esperado na saída da rede;
- b) Propagar as ativações pela rede;
- c) Verificar o erro e corrigir os pesos;
- d) Forçar para que a saída assumo o valor do padrão x_t ; (professor forçante)
- e) no instante seguinte ($t+1$) o padrão x_{t+1} será esperado na saída
- f) voltar para o passo b.

Para o caso em que as unidades de saída contenham conexões de realimentação, o passo d é importante para acelerar a convergência do algoritmo pois no instante $t+1$, os valores propagados pelas conexões de realimentação serão valores corretos, e consequentemente a correção dos pesos no instante $t+1$ tenderá a ser mais precisa.

Conforme dito anteriormente, o algoritmo de treinamento do MRST utiliza uma taxa de aprendizagem variável, monotonicamente decrescente, através do método busque-então-convirja (*search-then-converge*). Nesse caso, após cada apresentação de uma seqüência completa, a taxa de aprendizagem, α , é atualizada da seguinte forma:

$$\alpha(t) = \frac{\alpha_0}{1 + \frac{N}{\tau}} \quad (7.2.5.)$$

sendo α_0 o valor inicial da taxa de aprendizagem

N o número de vezes que foram apresentadas seqüências durante o treinamento

τ um parâmetro que ajusta velocidade com que a taxa de aprendizagem decresce.

A correção do peso da unidade j para a unidade i é dada pela seguinte equação:

$$\Delta w_{ij}(t) = \alpha(t) \delta_i(t) V_j(t) + \psi \Delta w_{ij}(t-1) \quad (7.2.6.)$$

sendo $V_j(t) = i_j(t)$ se j é uma unidade da camada de contexto;

$V_j(t) = h_j(t)$, se j é uma unidade escondida;

$V_j(t) = o_j(t)$, se j é uma unidade da camada de saída;

ψ é o fator de momento;

$\delta_i(t)$ é calculado da seguinte forma:

$$\begin{cases} \delta_i(t) = f'(q_i^s(t))(\zeta_i - o_i) & \text{se } i \text{ é uma unidade da camada de saída} \\ \delta_i(t) = f'(q_i^h(t)) \sum_j w_{ij}^s \delta_j & \text{se } i \text{ é uma unidade da camada escondida} \\ \delta_i(t) = f'(q_i^c(t)) \sum_j w_{ij}^h \delta_j & \text{se } i \text{ é uma unidade da camada de contexto} \end{cases} \quad (7.2.7)$$

sendo ζ o valor desejado (alvo) para a i -ésima unidade de saída.

A Figura 19 apresenta o algoritmo de treinamento do MRST.

- a) Inicializar $N, N_{\max}, \alpha_0, \tau, \psi, \text{erro}_{\max}$
 todos os pesos são inicializados com valores próximos de zero

b) Enquanto todas as seqüências não forem aprendidas e $N < N_{\max}$

i) Enquanto o conjunto de treinamento não foi inteiramente apresentado

1) escolher aleatoriamente uma seqüência S_j ainda não apresentada

2) apresentar o padrão de entrada (contexto espacial) referente à seqüência S_j

3) enquanto a seqüência S_j não foi completamente apresentada

I) o próximo ponto da seqüência é tido como alvo

II) propagar as ativações utilizando-se as equações (7.2.1.) a (7.2.4.);

III) verificar o erro na saída e retropropaga-lo utilizando (7.2.7.);

IV) corrigir os pesos utilizando (7.2.6.)

V) corrigir a taxa de aprendizagem utilizando (7.2.5.)

VI) utilizar o professor forçante na saída

VII) voltar ao para I)

ii) Verificar se, para todos os pontos de todas as seqüências, $\text{erro} < \text{erro}_{\max}$ é verdadeiro

Figura 19: Algoritmo de Treinamento utilizado para o modelo modificado de D'Arbo Junior, e que realiza o papel do MRST.

Após o treinamento, o procedimento para reprodução de uma seqüência é realizado da seguinte forma:

- Apresentar o padrão de entrada (contexto espacial) à entrada da rede. Esse padrão fica fixado na entrada durante todo o processo de reprodução;
- Para cada instante de tempo t , obter o ponto x_t na saída da rede, até que toda a seqüência tenha sido reproduzida;

Utilizando-se os dois modelos apresentados neste Capítulo, a próxima Seção irá apresentar o sistema modular completo para reprodução de trajetórias.

7.3. O Sistema Neural Modular Completo

A Figura 20 apresenta o sistema modular com tratamento de informações de contexto espacial e temporal em módulos distintos e que foi proposto e implementado nesta pesquisa.

Figura 20: Sistema Modular proposto e implementado nesta pesquisa para reprodução de seqüências temporais.

O treinamento de cada um dos módulos é realizado separadamente, da seguinte forma:

- Inicialmente utiliza-se o algoritmo da Figura 15 para treinar o MECE;
- Terminado o treinamento do MECE, os protótipos aprendidos neste módulo serão utilizados como entradas para treinar o MRST utilizando o algoritmo da Figura 19.

O procedimento para reprodução de seqüências temporais utilizando-se o sistema neural modular da Figura 20 é realizado da seguinte forma:

- a) Informa-se um padrão de entrada $[x'_0, x'_f]$;
- b) A partir do padrão dado, é construído o padrão $v = [x'_0, x'_i, x'_f]$;
- c) O padrão v é fornecido para o módulo estimador de contexto espacial;
- d) O MECE identifica a seqüência S_j associada a x , além de apresentar uma probabilidade de associação;
- e) Se a probabilidade de associação for inferior a um certo limiar, então comunicar a dúvida e executar um procedimento auxiliar (caso exista algum).
- f) Se a probabilidade de associação for maior que o limiar de dúvida, então o protótipo de contexto espacial, $\hat{v} = [\hat{x}_0, \bar{x}_i, \hat{x}_f]$ da seqüência identificada é utilizado para alimentar a entrada do MRST com as informações de ponto inicial e final estimadas, $\hat{v}_{MRST} = [\hat{x}_0, \hat{x}_f]$;
- g) A informação de contexto espacial, $\hat{v}_{MRST}$, é mantida na entrada do MRST, enquanto a saída é comparada com a informação x'_0 (ponto de partida fornecido no início do procedimento de reprodução);
 - Caso um número elevado de passos tenha ocorrido e nenhuma saída apresentada no MRST seja igual ou próxima a x'_0 , então acusar erro de identificação e interromper o processo;
- h) A partir do momento em que a saída do MRST é igual ou muito próxima a x'_0 , a seqüência observada na saída do MRST passa a ser considerada, e a saída do MRST passa a ser comparada com x'_f (ponto final fornecido no início do procedimento de reprodução);
 - Caso um número elevado de passos tenha ocorrido e nenhuma saída apresentada no MRST seja igual ou próxima a x'_f , então acusar erro de identificação e interromper o processo;
- i) A partir do momento em que a saída do MRST é igual ou muito próxima a x'_f , indicar fim da reprodução da seqüência desejada;

Nos experimentos realizados, a construção do padrão $x = [x'_0, x'_i, x'_f]$ foi feita da seguinte forma:

- 1) Para as situações em que se deseja generalizar, partindo de pontos x'_0 diferentes do x_0 visto durante o treinamento, então o padrão de entrada do MECE será $[x'_0, x'_0, x'_f]$. Ou seja, faz-se $x'_i = x'_0$;

- 2) Para as situações em que se deseja generalizar, informando um ponto alvo $\mathbf{x}'_f$ diferente do $\mathbf{x}_f$ visto durante o treinamento, então o padrão de entrada do MECE será $[\mathbf{x}'_o, \mathbf{x}'_f, \mathbf{x}'_f]$. Ou seja, faz-se $\mathbf{x}'_i = \mathbf{x}'_f$;
- 3) Se ambos os pontos $\mathbf{x}'_o$ e $\mathbf{x}'_f$ são diferentes dos pontos $\mathbf{x}_o$ e $\mathbf{x}_f$ vistos durante o treinamento, ou se não se tem esse tipo de informação, ou ainda, se o padrão de entrada $[\mathbf{x}'_o, \mathbf{x}'_f]$ é inteiramente igual ao padrão visto durante o treinamento $[\mathbf{x}_o, \mathbf{x}_f]$, então escolher aleatoriamente um ponto ($\mathbf{x}'_o$ ou $\mathbf{x}'_f$) e fazer $\mathbf{x}'_i$ igual a este ponto escolhido;

Com relação ao passo e do procedimento de execução, o procedimento adotado nos experimentos do Capítulo 8 simplesmente considerou a unidade de saída com maior probabilidade.

E por último, com relação aos passos g e h, ao invés de interromper o processo em caso de problemas de identificação, uma alternativa seria procurar reproduzir a segunda seqüência mais provável (de acordo com as probabilidades de associação fornecidas pelo MECE).

A seguir serão feitos (ou revisados) alguns comentários a respeito do sistema neural modular para reprodução de seqüências temporais apresentado nesta pesquisa:

- O sistema trata separadamente as informações de contexto espacial e temporal;
- O principal responsável pela capacidade de generalização e robustez a ruídos é o MECE, pois o MRST é treinado e executado exatamente com as mesmas informações de entrada, ou seja, com os protótipos aprendidos no MECE;
- Um erro do MECE implica na reprodução, pelo MRST, de uma seqüência que foi aprendida durante o treinamento, porém não é a seqüência desejada;
- O sistema proposto não é adequado para realizar interpolações de pontos ou de seqüências, uma vez que o MRST sempre reproduzirá exatamente os pontos da seqüência aprendida e identificada pelo MECE.

No próximo Capítulo, serão apresentados os experimentos realizados em diferentes domínios, assim como os resultados obtidos em tais experimentos.

8. EXPERIMENTOS

Experimentos em dois domínios diferentes foram realizados para se avaliar o desempenho do sistema neural modular proposto neste mestrado. Os principais objetivos dos experimentos a serem apresentados neste capítulo são: (a) comparar o desempenho do sistema neural modular proposto nesta pesquisa com o da rede neural parcialmente recorrente descrita na Seção 7.2., cuja arquitetura foi baseada na rede de D'Arbo Junior [22]; (b) avaliar o desempenho do sistema proposto para trajetórias com diferentes níveis de complexidade, e que foram utilizadas por Srinivasa e Ahuja [62] para testar a rede TOTEM; (c) avaliar o comportamento do módulo estimador de contexto espacial e suas capacidades na tarefa de identificação de seqüências e fornecimento de probabilidades indicando a incerteza na identificação.

A rede neural parcialmente recorrente e o sistema neural modular completo foram implementados em C++ e executados em um computador Pentium III de 800 MHz.

Para analisar o desempenho dos modelos (rede neural parcialmente recorrente operando sozinha, e sistema neural modular completo), os resultados foram observados principalmente com relação à:

- Capacidade de recuperação de seqüências quando fornecidos os mesmos pontos inicial e final vistos durante o treinamento. Nesse caso, observou-se apenas se os modelos utilizados foram capazes de recuperar corretamente cada um dos pontos pertencentes à trajetória que conecta os pontos inicial e final fornecidos.
- Capacidade de generalização visando reproduzir as trajetórias treinadas. Verificou-se a capacidade dos modelos para seguir corretamente cada um dos pontos pertencentes à seqüência desejada (conectando os pontos inicial e final) quando o ponto inicial ou final fornecido é um ponto intermediário das seqüências treinadas, e este ponto não foi treinado como sendo ponto inicial ou final.

Para todos os experimentos realizados, os dados de entrada passaram por um processo de normalização descrito a seguir:

- Para cada variável v_i dos padrões de entrada, encontram-se no conjunto de treinamento os respectivos valores máximo (v_{max_i}) e mínimo (v_{min_i});

- Para cada variável v_i , o seu valor normalizado $vnorm_i$ deverá ser encontrado da seguinte forma:

$$vnorm_i = (v_i - v_{min_i}) / (v_{max_i} - v_{min_i}). \quad (8.1)$$

Embora os dados tenham sido normalizados, a notação, por simplicidade, a notação $vnorm_i$ não foi utilizada.

É importante salientar que os resultados apresentados neste Capítulo são resultados típicos e não são resultados exaustivos e nem melhores resultados. A Seção 8.1. apresentará os resultados obtido com trajetórias geradas para controlar um braço mecânico do tipo PUMA 560. Em seguida, a Seção 8.2 apresentará resultados obtidos com trajetórias bidimensionais propostas por Srinivasa e Ahuja [62]. A Seção 8.3. apresenta resultados obtidos em experimentos para avaliação do MECE atuando sozinho como identificador de seqüências. E finalmente, a Seção 8.4. fará uma análise dos resultados obtidos neste Capítulo.

8.1. Trajetórias para um Braço Mecânico

Neste domínio, as seqüências temporais são formadas por padrões (vetores) de 15 variáveis, sendo 3 coordenadas cartesianas do espaço tridimensional, seis ângulos e seis torques para as juntas do braço mecânico. Todas as trajetórias do domínio em questão foram obtidas utilizando-se a ferramenta *Robotics* do Matlab. Com este software, é possível gerar trajetórias para o modelo de braço mecânico com 6 graus de liberdade do PUMA 560 [19],[20].

Todos os experimentos no domínio em questão foram realizados para dois casos: (a) utilizando-se unicamente a rede neural parcialmente recorrente proposta na Seção 7.2. e cuja arquitetura foi baseada no modelo de D'Arbo Junior; e (b) utilizando-se o sistema neural modular completo proposto nesta pesquisa. Os resultados obtidos foram comparados com o intuito de avaliar o ganho de desempenho quando o módulo estimador de contexto espacial é acoplado na entrada do MRST.

Para os experimentos apresentados nesta seção, foram utilizadas os mesmos conjuntos de treinamento propostos por D'Arbo Junior [22], visando repetir seus experimentos e utilizá-los como referências para as comparações de resultados. Além disso, como as trajetórias propostas por D'Arbo Junior eram todas abertas, incluiu-se um conjunto de treinamento considerando-se trajetórias fechadas.

Os conjuntos propostos por D'Arbo Junior e utilizados nesta pesquisa para avaliação dos modelos implementados consideram quatro situações distintas no que diz respeito à existência e localização de pontos ambíguos. Tais conjuntos são todos formados por 3 trajetórias abertas, cada uma contendo 11 pontos.

As primeiras duas situações são apresentadas na Figura 21. As letras I e F indicam os pontos iniciais e finais de cada trajetória. A Figura 21a, apresenta a situação em que as trajetórias não possuem pontos compartilhados. Tais trajetórias serão chamadas de trajetórias SC (sem cruzamentos). A Figura 21b apresenta o caso em que as trajetórias compartilham pontos intermediários da seguinte forma: o quinto ponto da trajetória T1 é igual ao sexto ponto da trajetória T2, e o sétimo ponto da trajetória T2 é igual ao quinto ponto da trajetória T3. Tais trajetórias serão chamadas de trajetórias CC (com cruzamentos).

A Figura 22a, apresenta três trajetórias utilizadas para duas situações distintas: 1. trajetórias com o mesmo ponto final (chamadas de trajetórias m-1), caso em que os pontos iniciais considerados são indicados com a letra A e o ponto final é indicado por B; e 2. trajetórias com o mesmo ponto inicial (chamadas de trajetórias 1-m), partindo do ponto B e chegando nos pontos indicados por A. E finalmente, a Figura 22b apresenta um conjunto contendo duas trajetórias fechadas, cada uma formada por 50 pontos. Esse último conjunto não foi considerado por D'Arbo Junior em sua pesquisa, e foi incluído para que os modelos pudessem ser testados em situações envolvendo trajetórias fechadas e com maior número de pontos.

Figura 21: Dois conjuntos de treinamento utilizados por D'Arbo Junior, formados por 3 trajetórias (de 11 pontos cada) considerando duas situações distintas: a) trajetórias que não compartilham pontos comuns; e b) trajetórias que compartilham pontos intermediários.

Para todos os cinco conjuntos considerados, comparando-se treinamentos realizados pelos dois modelos e com aproximadamente o mesmo número de épocas⁶, o tempo gasto pelo sistema completo foi ligeiramente maior que o tempo gasto pela rede recorrente, sendo essa diferença nunca superior a 5%. Esse resultado sugere que a contribuição do MECE para o custo computacional final do sistema é pouco significativa, sendo a rede recorrente a principal responsável por este custo.

Quando os pontos inicial e final fornecidos para a reprodução de uma trajetória desejada eram iguais àqueles vistos durante o treinamento, tanto a rede parcialmente recorrente operando sozinha quanto o sistema modular proposto nesta pesquisa foram capazes de aprender, para todos os quatro casos considerados nas Figuras 22 e 23, todas as trajetórias, com um erro máximo (medido para cada ponto de cada trajetória) inferior a 1%. Cada um desses casos serão comentados a seguir.

Figura 22: (a) Conjunto de trajetórias utilizado para duas situações distintas: 1. com o mesmo ponto final (trajetórias de A para B); e 2. com o mesmo ponto inicial (trajetórias de B para A). (b) Conjunto com trajetórias fechadas.

A Figura 23 apresenta as coordenadas cartesianas das trajetórias do tipo SC recuperadas utilizando-se a rede recorrente sozinha. O mesmo caso, porém utilizando-se o sistema modular, é apresentado na Figura 24.

⁶ Como o backpropagation é o algoritmo muito sensível às condições iniciais, o número de épocas varia muito de um treinamento para outro. Sendo assim, uma das formas utilizadas para comparar o tempo gasto pelos dois modelos foi procurar treinamentos com números de épocas muito próximos e comparar os tempos.

Ainda neste mesmo experimento, os ângulos das juntas do braço para cada ponto das trajetórias SC, recuperados pela rede recorrente podem ser vistos na Figura 25, enquanto aqueles reproduzidos pelo sistema modular podem ser vistos na Figura 26. E por último, os torques para cada ponto das trajetórias SC reproduzidos pela rede recorrente e pelo sistema modular proposto podem ser vistos respectivamente nas Figuras 27 e 28.

Figura 23: Resultado obtido com a rede parcialmente recorrente reproduzindo as trajetórias SC, quando os pontos inicial e final fornecidos são iguais aos treinados.

Figura 24: Resultado obtido com o sistema modular proposto reproduzindo as trajetórias SC, quando os pontos inicial e final fornecidos são iguais aos treinados.

Perceba que em todos os casos citados (reproduções das coordenadas, ângulos e torques, realizadas pelos dois modelos implementados) os pontos recuperados estão praticamente sobrepostos aos pontos desejados, tendo os dois modelos apresentado resultados praticamente iguais. Na realidade, esse resultado já era esperado pois a arquitetura utilizada para a rede recorrente e para o MRST do sistema modular foi baseada na arquitetura proposta por D'Arbo Junior. Sendo assim, pelo menos para as situações treinadas, é de se esperar que resultados semelhantes àqueles obtidos por D'Arbo Junior também sejam encontrados aqui.

Figura 25: Resultado obtido com a rede parcialmente recorrente reproduzindo os ângulos das juntas para as trajetórias SC, quando os pontos inicial e final fornecidos são iguais aos treinados.

Por esses mesmos motivos apresentados, os resultados obtidos com os dois modelos na recuperação das trajetórias do tipo CC, 1- m, m-1 e trajetórias fechadas a partir de pontos inicial e final vistos durante o treinamento, foram similares àqueles apresentados nas Figuras 23-28 para trajetórias do tipo SC. Sendo assim, tais resultados estão apresentados no Apêndice C.

Figura 26: Resultado obtido com o sistema modular proposto reproduzindo os ângulos das juntas para as trajetórias SC, quando os pontos inicial e final fornecidos são iguais aos treinados.

Figura 27: Resultado obtido com a rede parcialmente recorrente reproduzindo os torques nas juntas do braço mecânico para as trajetórias SC, quando os pontos inicial e final fornecidos são iguais aos treinados.

Figura 28: Resultado obtido com o sistema modular reproduzindo os torques nas juntas do braço mecânico para as trajetórias SC, quando os pontos inicial e final fornecidos são iguais aos treinados.

O tempo médio e o número médio de épocas necessárias para o treinamento de cada um dos modelos implementados, na aprendizagem de cada um dos quatro conjuntos de trajetórias considerados até o momento são apresentados na Tabela 1.

Vale lembrar que, para o sistema neural modular proposto nesta pesquisa, o treinamento é realizado em duas etapas, sendo o MECE treinado para um número de épocas predeterminado, e em seguida, o MRST é treinado até que todas as trajetórias sejam aprendidas (ou até que um certo limite de épocas tenha sido atingido). Sendo assim, o número de épocas referente ao sistema modular apresentado na Tabela 1 se refere apenas ao número de épocas do MRST. Já o tempo, se refere ao tempo total de treinamento (MECE + MRST). Além disso, é importante salientar que o tempo gasto apenas com o treinamento do MECE não chegou a 1 segundo para qualquer dos casos considerados.

Tabela 1: Número médio de épocas e tempo médio gasto para o treinamento de cada um dos modelos implementados e para cada conjunto de trajetórias.

Trajetórias	Rede Parcialmente Recorrente		Sistema Modular	
	Nº médio de épocas	Tempo (s)	Nº médio de épocas	Tempo (s)
SC	1146,6	7,8	1148,0	8,0
CC	2069,0	13,9	2124,0	14,6
1-m	1269,5	8,6	1640,6	11,4
m-1	849,7	5,7	840,3	5,9
Fechadas*	18340,5	2763	25676,7	3840

* para as trajetórias fechadas foram realizados apenas 4 treinamentos, e sendo assim, tais valores podem não representar bem as médias reais.

Perceba que os dois modelos apresentaram desempenhos muito semelhantes, o que mais uma vez sugere que o acréscimo do módulo estimador de contexto espacial para trabalhar em conjunto com uma rede neural recorrente não implica em um acréscimo significativo no custo computacional final, pois o grande responsável por este custo é o módulo para reprodução de seqüências temporais (rede recorrente).

Os resultados apresentados até o momento sugerem que, para os conjuntos de treinamento propostos por D'Arbo Junior, e quando os pontos inicial e final fornecidos são iguais àqueles vistos durante o treinamento, o sistema neural modular proposto nesta pesquisa e a rede recorrente baseada na rede de D'Arbo Junior apresentam resultados semelhantes. Porém, diferenças significativas de desempenho surgem em situações nas quais é necessária a capacidade de generalização, ou seja, quando um ponto extremo fornecido para reprodução de uma trajetória não é igual àquele visto durante o treinamento.

Para todos os testes de generalização apresentados nesta Seção, o MECE foi considerado como um classificador rígido (não ocorre dúvidas pois a maior probabilidade sempre é escolhida como vencedora, independentemente do seu valor absoluto). A Figura 29 mostra os resultados obtidos com a rede recorrente operando sozinha, tentando recuperar a trajetória T1 do conjunto de trajetórias do tipo m-1 quando o ponto inicial fornecido foi, na realidade, um ponto intermediário da trajetória em questão.

Figura 29: Resultados utilizando-se a rede recorrente (RR) para reproduzir a trajetória T1 do conjunto de trajetórias do tipo m-1, quando o ponto inicial fornecido é diferente daquele visto no treinamento. (a) partindo do 2º ponto; (b) partindo do 4º ponto; (c) partindo do 7º ponto; (d) partindo do 9º ponto.

As letras I e F indicam respectivamente os pontos iniciais e finais que foram fornecidos para a rede. Perceba que, quanto maior a distância entre o ponto inicial fornecido e o primeiro ponto da trajetória, maior é a dificuldade da rede para reproduzir a trajetória desejada. Tentativas de reprodução das trajetórias T2 e T3 apresentaram resultados semelhantes a estes obtidos com a Trajetória T1.

Figura 30: Resultados com a rede recorrente (RR) para reproduzir a trajetória T1 do conjunto de trajetórias do tipo CC, quando o ponto inicial fornecido é diferente daquele visto no treinamento. (a) partindo do 2º ponto; (b) partindo do 4º ponto; (c) partindo do 7º ponto; (d) partindo do 9º ponto.

De um modo geral, pode-se dizer que, para o conjunto de trajetórias do tipo m-1, apesar de não conseguir recuperar corretamente todos os pontos intermediários conectando I e F fornecidos, a rede recorrente ainda conseguiu atingir (ou pelo menos se aproximar de) o ponto F na maioria das tentativas. Porém, o conjunto de trajetórias do tipo m-1 foi na realidade aquele para o qual a rede apresentou o melhor desempenho no que diz respeito à capacidade de

generalização, principalmente quando se deseja apenas atingir o ponto final. Uma possível explicação para esse resultado consiste no fato de que, sendo o ponto final F, o mesmo aprendido para todas as trajetórias, então este ponto acaba se tornando um forte atrator, facilitando para que a rede convirja para ele. Para os conjuntos restantes, o desempenho da rede recorrente piorou significativamente. A Figura 30 apresenta os resultados obtidos quando a rede recorrente foi colocada para reproduzir, a partir de pontos intermediários, a trajetória T1 do conjunto de trajetórias do tipo CC.

Figura 31: Resultados utilizando-se o sistema modular completo (SC) para reproduzir a trajetória T1 do conjunto de trajetórias do tipo m-1, quando o ponto inicial fornecido é diferente daquele visto no treinamento. (a) partindo do 2º ponto; (b) partindo do 4º ponto; (c) partindo do 7º ponto; (d) partindo do 9º ponto.

Percebe-se que, para o conjunto de trajetórias do tipo CC, a rede recorrente não foi capaz de generalizar, ou seja, mesmo quando partindo de pontos próximos aos iniciais treinados, a rede não foi capaz de atingir o ponto final, e muito menos de seguir corretamente

os pontos intermediários. Resultados semelhantes a estes obtidos para a Trajetória T1 também foram obtidos para as trajetórias T2 e T3 do tipo CC.

Os mesmos experimentos foram repetidos para o sistema modular. Para todas as situações apresentadas nas Figuras 31 e 32, o sistema neural modular foi capaz de atingir o ponto F, e mais do que isto, recuperar corretamente todos os pontos intermediários conectando I e F). Resultados semelhantes a estes apresentados também foram observados para as demais trajetórias abertas.

Figura 32: Resultados utilizando-se o sistema modular completo (SC) para reproduzir a trajetória T1 do conjunto de trajetórias do tipo CC, quando o ponto inicial fornecido é diferente daquele visto no treinamento. (a) partindo do 2º ponto; (b) partindo do 4º ponto; (c) partindo do 7º ponto; (d) partindo do 9º ponto.

A Figura 33 apresenta resultados obtidos com os dois modelos implementados tentando reproduzir a trajetória fechada em forma de oito, partindo de diferentes pontos intermediários.

Figura 33: Resultados da reprodução da trajetória fechada (oito), pelos dois modelos implementados, partindo de diferentes pontos intermediários: (a) e (b) partindo do 10° ponto; (c) e (d) partindo do 25° ponto; (e) e (f) partindo do 40° ponto. As figuras da esquerda se referem à rede recorrente, enquanto as figuras da direita se referem ao sistema modular.

Os resultados com a trajetória circular estão apresentados no Apêndice C.5. Percebe-se que também para as trajetórias fechadas, o sistema modular proposto nesta pesquisa apresentou boa capacidade para generalização enquanto a rede recorrente operando sozinha não apresentou resultados satisfatórios. Na realidade, até o momento foram apresentados apenas os casos para os quais o MECE foi capaz de identificar corretamente a seqüência desejada (aquela que conecta I e F). Sendo assim, o MECE forneceu, corretamente, protótipos de contexto espacial para o MRST. Esse último, recuperou a seqüência desejada partindo do ponto inicial real (treinado), e dessa forma, gerou o contexto temporal desconhecido (pois percorreu todos os estados anteriores ao ponto I fornecido). Além disso, o sistema somente passou a considerar a saída como válida quando encontrou I, e por último, encerrou o processo com sucesso quando encontrou o ponto final F. Dessa forma, o sistema foi capaz de generalizar reproduzindo os pontos com a mesma precisão apresentada para as situações treinadas.

Quando o MECE identifica erroneamente a seqüência a ser reproduzida, e nenhum procedimento alternativo é utilizado para verificar (testar) e/ou corrigir o erro, então, o sistema tentará reproduzir uma trajetória que foi aprendida, porém, erroneamente identificada.

Figura 34: Dois casos de erro de generalização obtidos com o sistema modular proposto: (a) tentando partir do 7º ponto da trajetória T2 do conjunto m-1, o MECE identificou erroneamente a trajetória T1 para ser reproduzida; (b) visando reproduzir a trajetória T2 do tipo CC a partir do 6º ponto, o MECE erroneamente identificou a trajetória T3 como sendo a desejada para a reprodução.

Se o ponto inicial fornecido, I, durante a fase de execução é muito diferente do ponto inicial treinado, o MECE ainda pode reconhecer corretamente a trajetória devido às informações a respeito da média e do ponto final. Porém, em determinados casos, essas

informações não são suficientes para se resolver determinados pontos ambíguos. Na Figura 34a, além do ponto I fornecido estar consideravelmente distante do ponto inicial treinado, a informação do ponto final é ambígua (pois é a mesma para todas as seqüências do tipo m-1). Sendo assim, MECE identificou a seqüência apenas pela informação da média (qual média está mais próxima de I). Essa informação é bastante imprecisa, e sozinha pode não ser suficiente para o reconhecimento (como foi o caso mostrado).

Com relação à Figura 34b, se apenas o ponto final F fosse avaliado, o sistema saberia reconhecer corretamente a trajetória, porém, o fato do ponto I fornecido estar mais próximo do ponto inicial de T3 e também mais próximo da média de T3, quando comparado com T2, induziu o MECE a erroneamente identificar a trajetória desejada como sendo T3 (e não T2). Casos semelhantes aos apresentados na Figura 34 aconteceram para todos os conjuntos de treinamento utilizados. Vale lembrar que para esses casos, o sistema deverá reconhecer o erro a partir do momento em que, após um número elevado de pontos percorridos, os pontos fornecidos I e/ou F não tiverem sido encontrados. Uma vez reconhecido o erro, é possível que o sistema recomece o processo de reprodução, porém, visando recuperar a segunda trajetória mais provável. Ou seja, o protótipo de contexto espacial fornecido pelo MECE será aquele associado à segunda maior probabilidade. Caso erre novamente, o processo poderá se repetir até que todas as trajetórias identificadas como possíveis tenham sido reproduzidas, e assim sendo, o sistema sempre irá conseguir reproduzir a trajetória desejada, a menos que essa não tenha sido treinada.

Para uma avaliação final a respeito da capacidade de generalização do sistema modular proposto nesta pesquisa, realizou-se o seguinte experimento:

- para cada tipo de trajetória (SC, CC, 1-m, m-1 e fechadas), criou-se um conjunto de testes contendo:
 - todas as possibilidades para partir de pontos intermediários, e chegar no ponto final treinado;
 - e todas as possibilidades para partir do ponto inicial treinado e chegar em pontos intermediários;

A Tabela 2 mostra a porcentagem de seqüências recuperados corretamente (na primeira tentativa) com o sistema modular proposto nesta pesquisa. Esses resultados são bastante expressivos uma vez que a rede recorrente quando operando sozinha, e para nenhum caso, foi capaz de reproduzir corretamente mais do que 10 % das tentativas possíveis no conjunto de testes. Vale lembrar que, para ser considerada uma reprodução correta, o modelo precisava reproduzir todos os pontos intermediários pertencentes à trajetória desejada, conectando I e F fornecidos, com um erro inferior a 1%.

Tabela 2: Resultados obtidos com o sistema modular para os conjuntos de testes criados para cada tipo de trajetória, para avaliação da capacidade de generalização do sistema.

Tipo de Trajetória	Porcentagem de reproduções corretas
m-1	87,4 %
1-m	84,6 %
SC	93,7 %
CC	86,8 %
fechadas	100 %

Para o caso das trajetórias fechadas, o sistema modular proposto conseguiu recuperar todas as situações possíveis (quando o ponto inicial fornecido é um ponto intermediário, ou quando o ponto final fornecido é um ponto intermediário).

A próxima seção apresenta o segundo domínio considerado para avaliar o desempenho do modelo proposto.

8.2. Trajetórias Bidimensionais

Nesta segunda fase de experimentos, o sistema neural modular proposto nesta pesquisa foi testado para um conjunto formado por 8 trajetórias 2D apresentando situações com diferentes níveis de complexidade. As equações que descrevem cada uma das trajetórias estão apresentadas na Tabela 3.

Para gerar as trajetórias do conjunto de treinamento, as equações apresentadas da Tabela 3 foram amostradas em 50 pontos com intervalos de tempo iguais (Figura 35). Perceba que o conjunto de treinamento contém 1 trajetória aberta (T2) e 7 trajetórias fechadas (T1, T3-T8). Algumas características apresentadas por este conjunto de treinamento são:

- Cada trajetória se cruza com pelo menos duas outras;
- As trajetórias T5 e T6 são consideradas bastante complexas (principalmente T6) por apresentarem regiões centrais com grande quantidade de cruzamentos;
- Devido à grande quantidade de pontos ambíguos e/ou recorrentes, pode-se considerar que a tarefa de aprendizagem e reprodução de tais trajetórias é de grande dificuldade.

Tabela 3: Equações descrevendo as 8 trajetórias 2D utilizadas no conjunto de treinamento da segunda fase de experimentos com o sistema modular proposto nesta pesquisa.

Trajelórias	$x(t)$	$y(t)$
T1	$0,8 + 0,5[\cos(t)]^3$	$-0,5 + 0,5[\sin(t)]^3$
T2	$-1,2 + 0,5[t - \sin(t)]$	$-0,2 + 0,5[1 - \cos(t)]$
T3	$0,5 + 0,5\cos(t)[1 - \cos(t)]$	$0,5\sin(t)[1 - \sin(t)]$
T4	$0,5\cos(t)[1 + \cos(t)]$	$0,5\sin(t)[1 + \sin(t)]$
T5	$-0,8 + 0,5\sin(3t)\cos(t)$	$-0,3 + 0,5\sin(3t)\sin(t)$
T6	$1 + 0,5\sin(5t)\cos(t)$	$0,5 + 0,5\sin(5t)\sin(t)$
T7	$1,73 + 0,75\cos(t)$	$-0,45 + 0,75\sin(t)$
T8	$2,3 + 1,25\cos(t)$	$-0,45 + 1,25\sin(t)$

Figura 35: Trajetórias 2D utilizadas na segunda fase de experimentos com o sistema modular proposto.

O sistema modular proposto nesta pesquisa foi capaz de aprender o conjunto de treinamento proposto, apresentando um erro máximo de 5% medido em cada ponto de cada trajetória. Devido à grande complexidade do conjunto, após 5 treinamentos, o número médio de épocas, assim como o tempo médio gasto para cada o treinamento foram bastante elevados, ficando respectivamente em torno de 202.000 épocas em 6 horas e 15 minutos. É importante

citar que alguns experimentos mostraram que quando as trajetórias T5 e T6 são excluídas do conjunto de treinamento, o número de épocas assim como o tempo gasto no treinamento reduzem em mais de 90%.

As 8 trajetórias reproduzidas, a partir de pontos iniciais e finais treinados, são apresentadas na Figura 36. Da mesma forma como foi feito para as trajetórias da seção anterior, um conjunto de testes (contendo todas as situações com pontos iniciais ou finais não treinados) foi criado para avaliar a capacidade de generalização do sistema. O sistema modular conseguiu reproduzir corretamente a trajetória desejada para 69% dos padrões de entrada existentes no conjunto de testes.

Figura 36: Resultados obtidos com o sistema neural modular recuperando as trajetórias a partir de pontos inicial e final treinados.

Conforme já foi comentado anteriormente, a capacidade de generalização do sistema está diretamente relacionada com a capacidade do MECE em identificar corretamente a sequência a ser reproduzida. Sendo assim, a seção a seguir apresentará resultados de experimentos realizados apenas com o MECE na tarefa de identificação de sequências.

8.3. Experimentos com o MECE

Da forma como foi modelado, o MECE sozinho pode ser definido como um identificador estocástico de seqüências capaz de realizar as seguintes tarefas:

- Identificar uma seqüência tendo como entrada apenas $[x_0, x_i, x_f]$ (lembrando que tais valores podem estar ruidosos ou incompletos);
- Para o caso de entradas ruidosas ou incompletas, fornecer um protótipo contendo as informações de entrada corrigidas $[x_0, x_i, x_f]$;
- E ainda, fornecer uma medida de incerteza na forma de probabilidades;

Dito isso, esta Seção apresentará experimentos e resultados para avaliação do comportamento e do desempenho do MECE na realização das tarefas descritas no parágrafo anterior. O conjunto de trajetórias escolhido para avaliar o MECE é o conjunto utilizado na seção anterior, contendo 8 trajetórias bidimensionais (Figura 35). A escolha desse conjunto se deve aos seguintes motivos:

- Com relação aos experimentos anteriores, este conjunto foi o domínio para o qual a sistema teve o seu pior desempenho no que diz respeito à capacidade de generalização. E conforme já foi comentado outras vezes, esta capacidade esta diretamente relacionada com a capacidade de identificação do MECE;
- O fato das trajetórias estarem definidas no plano facilita a análise e visualização de distâncias entre pontos, que será realizada mais adiante nesta seção;
- E por último, o fato desse conjunto conter uma maior quantidade de trajetórias, em diferentes situações (1 trajetória aberta e 7 fechadas, com muitos cruzamentos e regiões de grande ambigüidade) também o torna mais interessante.

Recordando o procedimento para ajuste das probabilidades de associação proposto no Capítulo 7, tem-se que, para um dado valor pré-determinado do limiar de dúvida l_{duv} , é preciso encontrar o valor do parâmetro β (que é o inverso da pseudotemperatura na distribuição de Gibbs) que minimize o custo total L na tarefa de identificação de padrões no conjunto de testes. Uma vez encontrado β , a distribuição de Gibbs poderá ser utilizada na saída do módulo para que esta possa expressar probabilidades de associação.

A principal dificuldade desse tipo de procedimento reside no fato de que, tanto a escolha do parâmetro l_{duv} , como também a própria função definida para o cálculo do custo, são

dependentes do problema em questão. Sendo assim, primeiramente foi realizado um experimento para avaliar o comportamento do MECE quando a função de custo proposta nesta pesquisa é utilizada com diferentes valores para o limiar de dúvida, l_{duv} .

O conjunto de testes utilizado no procedimento de ajuste de β foi construído da mesma forma como foi feito nos experimentos anteriores, contendo todas as situações (x_0 e x_f são iguais àqueles vistos no treinamento) e não treinadas (x_0 ou x_f são pontos intermediários).

A Tabela 4 apresenta os valores de β encontrados para diferentes l_{duv} utilizados. Além disso, são apresentados, para cada β , os seguintes resultados: (a) o custo total no conjunto de testes; (b) a quantidade de erros e de dúvidas para o subconjunto contendo somente os padrões de entrada iguais àqueles visto durante o treinamento; (c) a quantidade de erros e dúvidas para o subconjunto de padrões de entrada contendo pontos iniciais diferentes daqueles vistos durante o treinamento; e (d) a quantidade de erros e dúvidas para o subconjunto de padrões de entrada contendo pontos finais diferentes daqueles vistos durante o treinamento.

Tabela 4: Resultados obtidos para diferentes valores de l_{duv} : valores ajustados para β ; custo total, L , no conjunto de testes; dúvidas e erros para as situações treinadas e não treinadas.

l_{duv}	β	L	Parões [x_0, x_i, x_f] Situação treinada		Padrões [x_i, x_j, x_f] Ponto inicial não treinado		Padrões [x_0, x_i, x_j] Ponto final não treinado	
			dúvidas (%)	erros (%)	dúvidas (%)	erros (%)	dúvidas (%)	erros (%)
0,5	8,5	226,41	0,75	0	12,00	22,25	11,50	22,75
0,6	47	230,54	6,50	0	21,25	18,00	21,25	19,25
0,7	98,1	231,86	10,25	0	35,50	11,50	33,25	14,00
0,8	160,5	232,37	14,25	0	50,50	2,75	48,50	6,00
0,9	192,5	233,05	23,50	0	59,75	0,75	58,00	3,50

Os resultados apresentados sugerem que:

- Para a função de custo utilizada, a configuração de parâmetros que apresentou um menor custo no conjunto de testes foi $l_{duv} = 0,5$ e $\beta = 8,5$. Para tais valores, o sistema apresentou:
 - A taxa de acertos (que equivale a um menos a soma das porcentagens de dúvidas e erros) no subconjunto de padrões treinados igual a 99,25 %, ou seja, dado que este subconjunto contém 400 pontos (50 para cada uma das oito trajetórias), o

sistema identificou corretamente 397 pontos, apresentou dúvida em apenas 3 pontos, e não realizou nenhuma identificação errada.

- Para os dois outros subconjunto de padrões (contendo os padrões de entrada com pontos iniciais diferentes daqueles treinados, e os padrões de entrada com pontos finais diferentes daqueles treinados) o sistema apresentou a mesma taxa de acertos, 65,75% que em valores absolutos são 263 padrões corretamente classificados para 137 apresentando dúvidas ou erros.
- Para $l_{duv} = 0,9$ (pior caso no que diz respeito ao valor do custo total) o sistema apresentou erros baixíssimos em todos os subconjuntos considerados, porém às custas de uma grande quantidade de identificações tidas como duvidosas;
- Quanto maior o limiar de dúvida, maior é o β que minimiza o custo, e consequentemente, maiores serão as probabilidades associadas aos protótipos mais próximos. Esse crescimento de β pode ser explicado da seguinte forma: no cálculo do custo total, existem custos associados às dúvidas e aos erros, enquanto que para os acertos o custo associado é zero. Sendo assim, para limiares de dúvida maiores, a distribuição de Gibbs precisa ser mais seletiva (maiores valores para β) para que os acertos possam acontecer, e consequentemente, para que o custo total seja minimizado.
- Por outro lado, se o limiar de dúvida é mais elevado, um dado protótipo precisará estar bem próximo do padrão de entrada (relativamente aos demais protótipos) para que a sua probabilidade de associação consiga superar o limiar de dúvida. Essa explicação justifica o fato de que maiores valores para l_{duv} implicam em menor quantidade de erros e maior quantidade de dúvidas;
- O resultado para os padrões treinados mostra que o módulo é capaz de aprender a identificar as trajetórias tendo como informação apenas os pontos inicial e final treinados, e um ponto intermediários qualquer;
- Os resultados obtidos com os padrões contendo pontos inicial ou final diferentes daqueles treinados mostram que o modelo possui capacidade de generalização.
- Com relação aos dois últimos comentários, é importante frisar que o MECE utiliza quantidade bastante reduzida de informações para identificar a sequência, enquanto inúmeros outros modelos utilizados para identificação de sequências precisam que uma grande quantidade de pontos (ou às vezes toda a sequência) seja apresentada para conseguir realizar a identificação.

A Figura 37 apresenta o resultado, para $l_{duv} = 0.5$, da identificação realizada para o subconjunto contendo padrões do tipo $[x_i, x_i, x_f]$, $i=1,...,f-1$, ou seja, formados por pontos iniciais não treinados e pontos finais treinados. Em outras palavras, para cada uma das

seqüências do conjunto, cada padrão intermediário x_i foi utilizado no lugar de x_0 para compor um padrão de entrada do subconjunto em questão.

As estrelas do gráfico indicam os pontos x_i , para os quais a rede conseguiu classificar corretamente os padrões. Os círculos apresentam os pontos para os quais ocorreram dúvidas (a probabilidade de associação do protótipo vencedor foi menor do que l_{duv}). E por último, os triângulos representam pontos para os quais a identificação realizada pelo MECE foi errada.

Perceba que o sistema conseguiu identificar as trajetórias T1, T5 e T6, para a grande maioria de seus pontos intermediários no lugar do ponto inicial. Com relação à trajetória T2, o sistema é capaz de identificá-la quando utilizados pontos intermediários próximos ao seu início. A medida em que os pontos intermediários utilizados no lugar do ponto inicial se distanciam do início da trajetória, o sistema começa a apresentar dúvidas, e mais diante ainda, passa a apresentar erros.

Figura 37: Resultados obtidos com o MECE identificando padrões contendo pontos iniciais não treinados. Estrelas indicam pontos x_i , utilizados no lugar de x_0 , para os quais a rede conseguiu classificar corretamente os padrões. Círculos representam ocorrência de dúvidas. E triângulos representam pontos para os quais a identificação realizada pelo MECE foi errada.

Dois casos serão avaliados mais detalhadamente para ilustrar o comportamento do sistema nos resultados apresentados, sendo uma trajetória para qual o sistema apresentou boa

capacidade de generalização, e outra para qual o sistema apresentou maior dificuldade em generalizar.

A Tabela 5 apresenta as probabilidades de identificação dos pontos intermediários pertencentes a T1, e para os quais o sistema apresentou dúvidas. Tanto para o ponto x_{10} quanto para o ponto x_{15} , a maior probabilidade de associação identificaria corretamente a trajetória T1, porém, como essas probabilidades não ultrapassaram $l_{duv} = 0,5$, o sistema indicou dúvida. Já com relação aos pontos x_{11} a x_{15} , esses não seriam identificados corretamente pelas maiores probabilidades, uma vez que estas se referem à trajetória T4. Porém, para todos esses pontos, a segunda maior probabilidade os identificaria corretamente. Continuando a análise desses pontos, a Figura 38 apresenta a Trajetória 1 em conjunto com informações a respeito de pontos iniciais e de médias de outras trajetórias. Perceba que os pontos duvidosos x_{10} a x_{15} estão muito próximos dos pontos iniciais (indicados por asteriscos) das trajetórias T3 e T4. Além disso, as médias (indicadas por quadrados) das trajetórias T3 e T4 estão mais próximas da região de pontos duvidosos do que a média de T1 com relação à mesma região. Essa análise justifica as probabilidades de associação avaliadas pelo MECE e indicadas na Tabela 5.

Tabela 5: Probabilidades de identificação de cada um dos pontos pertencentes a T1, e para os quais o sistema apresentou dúvidas de classificação.

ponto	Prob T1	Prob T2	Prob T3	Prob T4	Prob T5	Prob T6	Prob T7	Prob T8
x_{10}	0,407	0,000	0,227	0,351	0,001	0,007	0,008	0,000
x_{11}	0,300	0,000	0,238	0,445	0,001	0,009	0,007	0,000
x_{12}	0,252	0,000	0,237	0,493	0,001	0,011	0,006	0,000
x_{13}	0,250	0,000	0,237	0,496	0,001	0,011	0,006	0,000
x_{14}	0,288	0,000	0,240	0,455	0,001	0,010	0,007	0,000
x_{15}	0,373	0,000	0,241	0,372	0,001	0,007	0,007	0,000

É interessante comentar que casos como este discutido são exemplos típicos de situações em que, heurísticas e procedimentos alternativos poderiam solucionar ou corrigir a identificação. Por exemplo, se de alguma forma o sistema soubesse que o ponto final dado está correto, seria possível solucionar a dúvida simplesmente avaliando-se apenas a distância entre o ponto final dado no padrão de entrada e o ponto final aprendido no protótipo de cada sequência.

A Figura 39 apresenta a trajetória para qual o sistema apresentou a maior quantidade de identificações erradas. Tais erros ocorreram em duas regiões distintas (delimitadas por retângulos).

Figura 38: Trajetória T1 em conjunto com pontos iniciais e médias de outras trajetórias. Os quadrados indicam os pontos iniciais de cada trajetória. Os asteriscos indicam as médias de cada trajetória.

Figura 39: Trajetória T4 em conjunto com pontos iniciais e médias de outras trajetórias. Os quadrados e asteriscos indicam respectivamente o ponto inicial e a média de cada trajetória.

A Tabela 6 apresenta as probabilidades de identificação apontadas pelo MECE para cada um dos pontos da Região 1 da Trajetória T4, e que apresentaram erros e dúvidas na identificação. Perceba que para todos os pontos, a probabilidade de associação foi mais elevada para T6 do que para T4 (sendo que para o ponto x_{18} o sistema apresentou uma dúvida, com ambas as probabilidades de T6 e T4 ficando abaixo de l_{div}). Para entender o que ocorreu, é preciso voltar à análise da Figura 39. Os pontos da Região 1 estão muito mais próximos tanto do início quanto da média de T6 (que na realidade estão sobrepostos) do que do início e da média de T4, levando o sistema a uma identificação errada.

Tabela 6: Probabilidades de identificação de cada um dos pontos pertencentes à Região 1 da trajetória T4, e para os quais o sistema apresentou dúvidas e erros de identificação.

ponto	Prob T1	Prob T2	Prob T3	Prob T4	Prob T5	Prob T6	Prob T7	Prob T8
x_5	0,002	0,000	0,048	0,430	0,000	0,521	0,000	0
x_6	0,001	0,000	0,035	0,306	0,000	0,658	0,000	0
x_7	0,001	0,000	0,029	0,237	0,000	0,733	0,000	0
x_8	0,001	0,000	0,026	0,203	0,000	0,770	0,000	0
x_9	0,000	0,000	0,025	0,187	0,000	0,787	0,000	0
x_{10}	0,000	0,000	0,026	0,181	0,001	0,792	0,000	0
x_{11}	0,000	0,000	0,028	0,182	0,001	0,788	0,000	0
x_{12}	0,000	0,000	0,030	0,189	0,001	0,779	0,000	0
x_{13}	0,000	0,000	0,034	0,201	0,001	0,762	0,000	0
x_{14}	0,001	0,000	0,039	0,220	0,002	0,738	0,000	0
x_{15}	0,001	0,000	0,047	0,249	0,002	0,702	0,000	0
x_{16}	0,001	0,000	0,058	0,289	0,003	0,649	0,000	0
x_{17}	0,001	0,000	0,075	0,346	0,003	0,575	0,000	0
x_{18}	0,001	0,000	0,100	0,421	0,004	0,474	0,000	0

A análise da Região 2 é análoga à essa realizada para a Região 1, porém desta vez, são os pontos do início e da média da trajetória T3 que estão muito próximos da Região 2, levando novamente o MECE aos erros de identificação. É importante dizer que o MECE foi sensível o suficiente para indicar que T4 é a segunda trajetória mais provável para todos os pontos erroneamente identificados (tanto na Região 1 quanto na Região 2).

Analisando-se as probabilidades para todos os pontos de todas as 8 trajetórias do subconjunto em estudo, percebeu-se que em 72 dos 89 (de um total de 400 padrões compondo o subconjunto) pontos incorretamente identificados, a segunda maior probabilidade

identificaria corretamente a sequência. Além disso, verificou-se que em 16 dos 48 pontos para os quais o sistema teve dúvida, a maior probabilidade indica corretamente a sequência. E por último verificou-se que em 22 dos 48 pontos para os quais o sistema apresentou dúvida, a segunda maior probabilidade indicaria corretamente a sequência desejada. O que reforça a idéia de que procedimentos e heurísticas adicionais poderiam melhorar ainda mais a resposta do MECE. Além disso, em situações nas quais o sistema tenha mais de uma chance para tentar a identificação, este acertará logo na segunda chance na grande maioria das vezes.

A Figura 40 apresenta o resultado, para $l_{duv} = 0.5$, da identificação realizada para o subconjunto contendo padrões do tipo $[x_0, x_i, x_i]$, $i=1, \dots, f-1$, ou seja, formados por pontos iniciais treinados e pontos finais não treinados. Em outras palavras, o ponto final considerado no padrão de entrada é na realidade o ponto intermediário x_i .

Figura 40: Resultados obtidos com o MECE identificando padrões contendo pontos iniciais treinados e pontos finais não treinados. Estrelas indicam pontos x_i , utilizados no lugar de x_0 , para os quais a rede conseguiu identificar corretamente os padrões. Círculos representam ocorrência de dúvidas. E triângulos representam pontos para os quais a identificação realizada pelo MECE foi errada

Para esse conjunto de testes, como os pontos iniciais são iguais àqueles vistos durante o treinamento, então análises como aquelas realizadas para a Figura 38 e para a Figura 39, dariam maior importância para a observação dos pontos finais e das médias (e não dos pontos iniciais e das médias como havia sido feito). Porém, nota-se claramente que os resultados da Figura 40 são muito parecidos com aqueles obtidos no subconjunto anterior (Figura 37). O principal motivo dessa semelhança se deve ao fato de que, sendo fechadas a maioria das trajetórias, seus pontos inicial e final treinados são numericamente iguais, o que torna as análises e resultados ainda mais parecidos com aqueles anteriores. Aliás, a principal diferença ocorreu na identificação dos padrões contendo pontos da trajetória T2, pois para esta trajetória o ponto inicial não só é diferente do final como também está bem longe do mesmo. Percebe-se que desta vez, o sistema apresentou erros para os pontos x_i no início da trajetória e melhora a sua identificação na medida em que o ponto x_i em questão esteja mais próximo do final, o que faz pleno sentido uma vez que x_i está sendo utilizado no lugar do ponto final x_f no padrão de entrada.

Da mesma forma que no conjunto anterior, para a maioria dos casos em que o sistema errou, a segunda maior probabilidade indicaria corretamente a trajetória desejada. Ou seja, dos 91 padrões identificados erradamente, 74 seriam identificados corretamente se a segunda maior probabilidade fosse levada em consideração. Além disto, dos 46 padrões para o qual o sistema apresentou dúvida, 23 seriam identificados corretamente com a maior probabilidade e 17 seriam identificados corretamente se a segunda maior probabilidade fosse utilizada.

Como as análises dos resultados apresentados na Figura 40 são muito parecidas com as análises dos resultados do caso anterior (Figura 37) fica desnecessária a repetição destas. E assim termina a fase de experimentos desta pesquisa. A próxima seção fará uma breve discussão a respeito dos resultados alcançados.

8.4. Discussão dos Resultados

Os resultados apresentados demonstraram um bom desempenho do sistema modular neural proposto nesta pesquisa para a tarefa de reprodução de seqüências temporais. O sistema foi capaz de lidar com diferentes situações de ambigüidades e complexidades. Mais do que isto, o sistema apresentou capacidade de generalização (quando pontos iniciais ou finais fornecidos são diferentes daqueles vistos durante o treinamento) significativamente melhor do que o desempenho do modelo de rede parcialmente recorrente operando sozinho.

Ficou comprovado que, dentro da abordagem modular adotada, a capacidade de generalização do sistema é dada pela capacidade de identificação do MECE. Se por um lado esta abordagem permite que o sistema reproduza uma trajetória partindo de pontos

intermediários e seguindo a trajetória em questão com exatamente o mesmo erro obtido para a situação treinada, por outro lado esta abordagem não é adequada para interpolação de pontos ou de trajetórias. Além disso, sempre que o MECE erra na identificação da entrada, o sistema reproduzirá na saída uma trajetória que foi treinada porém não é a desejada.

As análises realizadas com o MECE sugerem que este apresenta capacidade para identificação de trajetórias treinadas e/ou generalizadas. O MECE também fornece probabilidades de associação que indicam o grau de certeza em sua identificação. Tais probabilidades de associação encontradas foram coerentes dentro das análises realizadas na Seção 8.3., sugerindo que o procedimento proposto, utilizando-se a distribuição de Gibbs, e uma função custo para ajusta-la, pode ser uma boa ferramenta para introduzir respostas probabilísticas em classificadores rígidos (como é o caso do LVQ).

E por último, mostrou-se que, é possível utilizar procedimentos de forma que o sistema neural modular sempre encontrará a seqüência desejada, desde que esta tenha sido treinada.

9. CONCLUSÕES

Esta dissertação apresentou estudos a respeito de modularização, contexto e redes neurais aplicados no desenvolvimento de um sistema neural (MECE-MRST) para reprodução de seqüências temporais. Na modularização, uma tarefa mais complexa pode ser dividida em subtarefas mais simples. Frequentemente, no caso de uma decomposição explícita de tarefas (utilizando-se algum tipo de conhecimento a priori), é possível designar para tais subtarefas, módulos com arquiteturas distintas, cada um destes especializado em realizar a subtarefa em questão. Dois tipos de informações de contexto foram definidos para desfazer ambigüidades no processamento de seqüências temporais. O contexto espacial é caracterizado por informações sobre a posição espacial de um ou mais padrões com relação a outros padrões. O contexto temporal foi definido como um conjunto de informações a respeito de entradas, saídas e/ou estados passados de um sistema.

Foi estudado um sistema modular conhecido como TOTEM que processa separadamente as informações de contexto espacial e temporal em módulos com arquiteturas distintas. Esse modelo serviu de inspiração para a abordagem proposta nesta pesquisa. Sendo assim, análise comparativa entre o TOTEM e o modelo proposto será realizada a seguir:

- Conforme definições do Capítulo 4, ambos os modelos (o TOTEM e o sistema modular desenvolvido nesta pesquisa) são formados por módulos sucessivos obtidos por decomposição explícita da tarefa envolvendo aprendizagem e reprodução de seqüências temporais;
- Enquanto o modelo TOTEM é formado por quatro módulos, e precisa de sinais externos para sincronizar e/ou controlar o processamento das informações, o sistema proposto nesta pesquisa é formado por dois módulos, com algoritmos para treinamento e execução significativamente mais simples. Além disto, a quantidade de parâmetros que precisam ser ajustados é significativamente menor para o sistema modular proposto do que para o TOTEM;
- Para a tarefa de reconhecimento de seqüências, o modelo TOTEM precisa que toda a seqüência a ser reproduzida seja apresentada em sua entrada. Já o MECE é uma rede de propagação direta com uma arquitetura de duas camadas e algoritmo significativamente mais simples, capaz de realizar a tarefa de identificação da

seqüência tendo como entrada uma quantidade bastante resumida de informações, ou seja, apenas os pontos inicial e final. Além disso, o MECE apresenta uma boa capacidade de generalização (quando pontos inicial ou final são diferentes daqueles vistos durante o treinamento), provê um protótipo de contexto espacial (recuperando os pontos inicial e final treinados) e informa uma probabilidade de associação indicando a incerteza de sua classificação.

- A reprodução de uma seqüência utilizando-se o modelo TOTEM envolve o processamento inverso do sistema, aumentando assim a complexidade do seu algoritmo de execução. Além disto, o modelo TOTEM precisa necessariamente da informação do rótulo da seqüência (na saída do módulo fuzzy-ART) que se deseja reproduzir. Em outras palavras, a seqüência desejada precisa ser conhecida de antemão. Isso não ocorre com o MECE-MRST pois este identifica a seqüência desejada a partir dos pontos inicial e final.
- Pelos mesmos motivos apresentados no item anterior, o TOTEM não é capaz de generalizar, para uma seqüência temporal partindo de pontos iniciais (ou chegando em pontos finais) diferentes daqueles vistos durante o período de treinamento.
- Para o caso de trajetórias formadas por pontos definidos em um espaço com dimensão maior (como por exemplo as trajetórias para o robô no espaço 3D, utilizadas na Seção 8.1.) é possível que o modelo TOTEM apresente problemas com relação à quantidade de unidades e conexões utilizadas nos vários módulos, assim como com a redução significativa da dimensionalidade, que ocorre no módulo SOM. Por sua vez, o MECE-MRST é capaz de trabalhar em espaços de dimensões maiores sem mudanças significativas em sua operação.

Um outro sistema estudado para a tarefa de reprodução de trajetórias foi o IOHMM. Esse modelo também é um exemplo de arquitetura modular, porém estocástico e com uma abordagem completamente diferente daquela adotada pelo TOTEM. Alguns comentários a respeito do IOHMM serão dados a seguir:

- Este modelo é capaz de realizar tarefas como reconhecimento e reprodução de seqüências temporais, assim como a tarefa de associação temporal, utilizando-se a teoria de HMMs (dentro da abordagem conexionista), para modelar através de processos markovianos escondidos, dados que tenham características espaciais e temporais;
- O IOHMM é um sistema modular formado por dois conjuntos de redes neurais (as sub-redes de estado e as sub-redes de saída) treinadas, não pelo algoritmo de

retropropagação, mas sim por algoritmos estatísticos de otimização tais como o EM e o GEM;

- Diferentemente do que ocorre com os módulos do TOTEM e com o MECE-MRST, no IOHMM os módulos estão dispostos paralelamente e se relacionam de forma cooperativa;
- Uma outra diferença significativa entre o IOHMM e os demais modelos apresentados reside na forma com que o IOHMM trata as informações de contexto temporal. Ou seja, para este modelo, tais informações são aprendidas durante o treinamento através de probabilidades de transição de estado e probabilidades de observação na saída, dado o estado atual;
- A topologia do IOHMM sugere que a quantidade de módulos (sub-redes de estado e de saída) necessários pode se constituir em um problema pois, para cada estado escondido necessário no processo, são utilizadas uma sub-rede de estado e outra de saída. Sendo assim, o custo computacional do modelo pode se tornar significativamente elevado.

Nesta pesquisa foi proposta uma abordagem dentro da qual informações de contexto espacial e temporal são processadas separadamente em módulos neurais com arquiteturas distintas visando obter um sistema com boa capacidade de generalização. Dentro desta abordagem, foi desenvolvido um sistema modular neural formado por duas redes neurais supervisionadas. A primeira é uma rede de propagação direta utilizada para aprender informações de contexto espacial. Tais informações alimentam uma rede neural recorrente, utilizada para aprender e reproduzir seqüências temporais. O principal ganho obtido com a abordagem proposta consiste em uma significativa melhoria no comportamento da rede recorrente quando reproduzindo seqüências partindo de situações não treinadas. Vale lembrar, que a melhoria da capacidade de generalização foi alcançada sem que houvesse um comprometimento com relação à capacidade da rede recorrente em reproduzir as seqüências em situações treinadas. Além disto, o custo computacional para treinar e executar o sistema modular não foi significativamente maior do que o custo de treinar e executar a rede recorrente operando sozinha pois, conforme foi mostrado, o grande responsável pelo custo do sistema modular desenvolvido é a própria rede recorrente.

A princípio, para partir de um ponto intermediário, o módulo para reprodução de trajetórias precisaria conhecer não apenas o ponto em si, mas também um histórico de pontos anteriores (contexto temporal). Essa é uma das principais dificuldades no que diz respeito à generalização na tarefa de reprodução de seqüências temporais. Além do mais, é correto afirmar que esta dificuldade é encontrada em outras arquiteturas diferentes, pois, se o modelo

não possui nenhuma informação a respeito dos seus estados passados, este deverá encontrar dificuldades para seguir corretamente uma seqüência temporal, principalmente se esta contiver pontos ambíguos.

Foram realizados experimentos comparando-se uma rede neural recorrente e o MECE-MRST, que possui a mesma rede recorrente, porém, alimentada por um módulo estimador de contexto espacial. Para as situações em que ambos os pontos inicial e final fornecidos para a reprodução eram iguais àqueles fornecidos durante o treinamento, os dois modelos apresentaram resultados idênticos. Esse resultado já era esperado pois, tanto a arquitetura utilizada para a rede recorrente quanto aquela utilizada para o MRST foram baseadas na arquitetura proposta por D'Arbo Junior, e sendo assim, é de se esperar que, para as situações treinadas, sejam obtidos resultados semelhantes em ambos os casos. Porém, diferenças significativas de desempenho foram apresentadas em situações nas quais é necessária a capacidade de generalização, ou seja, quando o ponto inicial ou o ponto final fornecido para reprodução de uma trajetória não é igual àquele visto durante o treinamento. Neste caso, os resultados mostraram que quanto maior a distância entre o ponto inicial (ou final) fornecido e o primeiro (ou último) ponto da trajetória treinada, maior a dificuldade da rede recorrente operando sozinha para reproduzir a trajetória desejada. Esta dificuldade ocorre devido à falta de informações de contexto.

De um modo geral, pode se dizer que, para o conjunto de trajetórias do tipo m-1, a rede recorrente operando sozinha ainda conseguiu atingir (ou pelo menos se aproximar de) o ponto final na maioria das tentativas. Isso ocorreu provavelmente porque, sendo o ponto final, o mesmo aprendido para todas as trajetórias, então este ponto acaba se tornando um forte atrator, facilitando a convergência da dinâmica da rede para ele. Para os conjuntos restantes, o desempenho da rede recorrente no que diz respeito à capacidade de generalização piorou significativamente. Também para as trajetórias fechadas, o MECE-MRST apresentou boa capacidade para generalização enquanto a rede recorrente operando sozinha não apresentou resultados satisfatórios.

É importante lembrar que, para o algoritmo de reprodução proposto nesta pesquisa, o MECE realiza a identificação da seqüência desejada (aquela que conecta os pontos inicial e final fornecidos), e em seguida, fornece protótipos de contexto espacial para o MRST. Esse último, recupera a seqüência desejada partindo do ponto inicial real (treinado), e dessa forma, gera o contexto temporal desconhecido (pois percorre todos os estados anteriores ao ponto inicial fornecido). Além disso, o sistema somente considera a saída como válida quando encontra o ponto inicial fornecido, e encerra o processo com sucesso quando encontra o ponto final fornecido. Conforme os resultados mostraram, utilizando-se esse procedimento, o sistema é capaz de generalizar reproduzindo os pontos da seqüência desejada com a mesma precisão apresentada para as situações treinadas.

A representação adotada para o padrão de entrada também tem a sua influência no desempenho do sistema. Em alguns casos, quando o ponto inicial (ou o ponto final) fornecido durante a execução é muito diferente daquele visto durante o treinamento, ou quando o ponto inicial (ou final) coincide com um ponto ambíguo, a informação a respeito da média dos pontos de uma sequência passa a ser de grande relevância na identificação da sequência. Porém, essa informação é bastante imprecisa, e sozinha pode não ser suficiente para o reconhecimento.

No domínio das trajetórias para controlar um braço mecânico, e para todos os conjuntos de trajetórias considerados neste domínio, o MECE-MRST apresentou uma boa capacidade de generalização, sendo capaz de recuperar corretamente as sequências desejadas, para qualquer um dos conjuntos considerados, em mais de 84% das tentativas.

Com relação ao segundo domínio considerado (trajetórias bidimensionais), os resultados sugerem que o sistema modular proposto nesta pesquisa é capaz de aprender conjuntos com trajetórias complexas, contendo pontos de alta ambigüidade. Devido à grande complexidade do conjunto, o número médio de épocas, assim como o tempo médio gasto para cada treinamento foram bastante elevados, porém, alguns experimentos mostraram que quando as duas trajetórias de maior complexidade são excluídas do conjunto de treinamento, o número de épocas assim como o tempo gasto no treinamento se reduzem em mais de 90%. Além disso, na avaliação da capacidade de generalização, o sistema modular conseguiu reproduzir corretamente a trajetória desejada para 65% dos padrões de entrada existentes no conjunto de testes.

Foi mostrado que, dentro da abordagem proposta, o MECE é o principal responsável pela capacidade de generalização do sistema. Sendo assim, esta dissertação também apresentou um estudo a respeito do MECE operando sozinho como sendo um identificador de sequências capaz de:

- Identificar uma sequência tendo como entrada apenas $[x_0, x_i, x_f]$ (lembrando que tais valores podem estar ruidosos ou incompletos);
- Para o caso de entradas ruidosas ou incompletas, fornecer um protótipo contendo as informações de entrada corrigidas $[x_0, x_i, x_f]$;
- E ainda, fornecer uma medida de incerteza na forma de probabilidades.

As análises realizadas com o MECE, para o conjunto de trajetórias bidimensionais, sugerem que este apresenta boa capacidade para identificação, apresentando resultados significativos no que diz respeito à capacidade de generalização. Além disso, os resultados desse estudo mostraram que o ajuste de uma distribuição de Gibbs na saída do modelo para obtenção de probabilidades que indiquem uma medida de incerteza na identificação é um procedimento capaz de encontrar valores coerentes dentro das análises realizadas, sugerindo

que o procedimento proposto, utilizando-se a distribuição de Gibbs, e uma função custo para ajustá-la, pode ser uma boa ferramenta para introduzir respostas probabilísticas em classificadores rígidos, como é o caso do LVQ.

Analisando-se as probabilidades para todos os pontos de todas as 8 trajetórias do subconjunto em estudo, percebeu-se que em 72 dos 89 (de um total de 400 padrões compondo o subconjunto) pontos incorretamente identificados, a segunda maior probabilidade identificaria corretamente a seqüência. Além disto, verificou-se que em 16 dos 48 pontos para os quais o sistema teve dúvida, a maior probabilidade indica corretamente a seqüência. Verificou-se também que em 22 dos 48 pontos para os quais o sistema apresentou dúvida, a segunda maior probabilidade indicaria corretamente a seqüência desejada. O que reforça a idéia de que procedimentos e heurísticas adicionais poderiam melhorar ainda mais a resposta do MECE. Além disto, em situações nas quais o sistema tenha mais de uma chance para tentar a identificação, este acertará logo na segunda chance na grande maioria das vezes.

Dentro da abordagem modular adotada, a capacidade de generalização do sistema é associada à capacidade de identificação do MECE. Se por um lado esta abordagem permite que o sistema reproduza uma trajetória partindo de pontos intermediários e seguindo a trajetória em questão com o mesmo erro obtido para a situação treinada, por outro lado esta abordagem não é adequada para interpolação de pontos ou de trajetórias. Além disto, sempre que o MECE erra na identificação da seqüência, o sistema reproduz na saída uma trajetória que foi treinada porém não é a desejada.

Esta dissertação também mostrou que é possível utilizar procedimentos de execução de forma que o sistema teste (em ordem de probabilidades) cada uma das seqüências aprendidas, e sendo assim, sempre encontrará a seqüência (caso esta exista) aprendida que conecta os pontos inicial e alvo fornecidos.

Uma das principais limitações do sistema apresentado diz respeito ao tempo de treinamento, principalmente do MRST, que utiliza o algoritmo de retropropagação, que é lento para aprender muitas trajetórias, ou trajetórias com alto grau de complexidade. Embora, é importante salientar que o sistema opera rapidamente, sendo portanto aplicável a execução de tarefas em tempo real.

E por último, algumas possibilidades que poderiam ser exploradas em trabalhos futuros dizem respeito à melhoria do módulo estimador de contexto espacial visando, principalmente, aprimorar a capacidade de identificação do módulo, e conseqüentemente, a capacidade de generalização do sistema como um todo. Alguns caminhos que poderiam ser adotados são:

- Ao invés de utilizar uma representação do contexto espacial contendo um ponto inicial x_0 , um ponto final x_f e um ponto qualquer x_i conforme foi feito nesta pesquisa, talvez seja possível uma escolha ótima de pontos que melhor representem o contexto espacial da seqüência. Ou seja, o sistema poderia se encarregar de

escolher os pontos que resultariam na melhor capacidade de identificação e generalização;

- Utilizar outras variações do algoritmo LVQ, ou mesmo outras arquiteturas e algoritmos diferentes;
- Estudar outras funções de custo, assim como outros procedimentos para ajustar as probabilidades obtidas com a distribuição de Gibbs calculada na saída do modelo.

REFERÊNCIAS BIBLIOGRÁFICAS

- [1] Akman, V. (2000). "Rethinking context as a social construct", *Journal of Pragmatics*, vol.32, no.6, pp.743-759.
- [2] Akman, V., & Surav, M. (1996). "Steps toward formalizing context", *AI Magazine*, vol.17, no.3, pp.55-72.
- [3] Araújo, A. F. R., & Barreto, G. de A. (2002). "A self-organizing context-based approach to the tracking of multiple robot trajectories", *Applied Intelligence*, vol.16 no.1 (artigo aceito).
- [4] Araújo, A. F. R., & D'Arbo Jr., H., "Partially recurrent neural networks for production of temporal sequences", *Proceedings of the IEEE International Joint Conference on Neural Networks*, Anchorage, Alaska, USA, May 04-09, 1998, pp 474-479
- [5] Auda, G., & Kamel, M. (1997). "Modular neural networks: a survey", *International Journal of Neural Systems*, vol.9, no.2, pp.129-152.
- [6] Barreto, G.A, & Araújo, A.F.R. (1999). "Unsupervised learning and recall of temporal sequences: An application to robotics", *International Journal of Neural Systems*, vol.9 no.3 pp.235-242.
- [7] Barreto, G. de A., & Araújo, A. F. R. (2000). "Time in self-organizing maps: An overview of models", *International Journal of Computer Research*, artigo aceito.
- [8] Battiti, R., & Colla, A. (1994). "Democracy in neural nets: Voting schemes for classification", *Neural Networks*, vol.7, no.4, pp.691-707.
- [9] Baxt, W. (1992). "Improving the accuracy of an artificial neural network using multiple differently trained networks", *Neural Computation*, vol.4, pp.772-780.
- [10] Becker, S. (1999). "Implicit learning in 3D object recognition: The importance of temporal context", *Neural Computation*, vol.11 no.2 pp.347-374.
- [11] Bengio, Y., Simard, P., & Frasconi, P. (1994). "Learning long-term dependencies with gradient descent is difficult", *IEEE Transactions on Neural Networks*, vol.5, no.2, pp.157-166.

- [12] Bengio, Y., & Frasconi, P. (1996). "Input-output HMM's for sequence processing", *IEEE Transactions on Neural Networks*, vol.7, no.5, pp.1231-1249.
- [13] Bourlard, H. & Wellekens, C. J., (1990). "Links between Markov models and multilayer perceptrons". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(12):1167-1178.
- [14] Campolucci, P., Uncini, A., Piazza, F., & Rao, B. D. (1999). "On-line learning algorithms for locally recurrent neural networks", *IEEE Transactions on Neural Networks*, vol.10, no.2, pp.253-271.
- [15] Carpenter, G. A., Grossberg, S. (1987). "A massively parallel architecture for a self-organizing neural pattern recognition machine", *Computer Vision, Graphics, and Image Processing*, vol. 37, pp.54-115.
- [16] Carpenter, G. A., Grossberg, S. (1995). "Adaptive resonance theory (ART)", in M. A. Arbib, ed., *The Handbook of Brain Theory and Neural Networks*, pp.79-82, Cambridge, MA:MIT Press.
- [17] Carpenter, G. A., Grossberg, S., & Rosen, D. B. (1991). "Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system", *Neural Networks*, vol.4, pp.759-771.
- [18] Catfolis, T., & Meert, K. (1997). "Hybridization and specialization of real-time recurrent learning-based neural networks", *Connection Science*, vol.9, no.1, pp.51-69.
- [19] Corke, P. I. & Armstrong-Hélouvry, B., (1994). "A search for consensus among model parameters reported for PUMA 560 Robot", *IEEE Conf. On Robotics and Automation*, pp. 1608-1613.
- [20] Corke P. I., (1996). "A robotics toolbox for MATLAB", *IEEE Robotics and Automation Magazine*, vol 3, n° 1, 24-32.
- [21] Craig, J. J., (1989). *Introduction to Robotics: Mechanics and Control*, 2nd ed. Addison-Wesley.
- [22] D'Arbo Junior, H. (1998) "Redes neurais recorrentes para produção de seqüências temporais", *Dissertação de Mestrado*, Escola de Engenharia de São Carlos, USP.
- [23] Dayan, P., Hinton, G.E., Neal, R.M., & Zemel, R.S. (1995). "The Helmholtz machine", *Neural Computation*, vol.7 pp.889-904.

- [24] De Vries, B., & Principe, J. C. (1992). "The gamma model – A new neural mode for temporal processing", *Neural Networks*, vol.5, pp.565-576.
- [25] Dominey, P. F. (1995). "Complex sensory-motor sequence learning based on recurrent state representation and reinforcement learning", *Biological Cybernetics*, vol.73, pp.265-274.
- [26] Doya, K. (1999). "What are the computations of the cerebellum, the basal ganglia and the cerebral cortex?", *Neural Networks*, vol.12, pp.961-974.
- [27] Elman, J. L. (1990). "Finding structure in time", *Cognitive Science*, vol.14, pp.179-211.
- [28] Frasconi, P., Gori, M., & Soda, G. (1992). "Local feedback multilayered networks", *Neural Computation*, vol.4, pp.120-130.
- [29] Funahashi, K. I., & Nakamura, Y. (1993). "Approximation of dynamical systems by continuous time recurrent neural networks", *Neural Networks*, vol.6, pp.801-806.
- [30] Galicki, M., Leistriz, L., & Witte, H. (1999). "Learning continuous trajectories in recurrent neural networks with time-dependent weights". *IEEE Transactions on Neural Networks* 10(4):741-756.
- [31] Grossberg, S. (2000). "The complementary brain: A unifying view of brain specialization and modularity", *Trends in Cognitive Sciences*, artigo aceito.
- [32] Happel, B., & Murre, J. (1994). "Design and evolution of modular neural network architectures", *Neural Networks*, vol.7, no.6/7, pp.895-1004.
- [33] Haykin, S. (1999). *Neural Networks – A Comprehensive Foundation*. Prentice Hall, New Jersey
- [34] Hertz, J., Krogh, A., & Palmer, R. G. (1991). *Introduction to the Theory of Neural Computation*. Addison-Wesley, Redwood City, CA.
- [35] Heyman, D.P., & Sobel, M.J. (1984). *Stochastic Models in Operations Research Volume II: Stochastic Optimization*, McGraw-Hill, New York, NY.
- [36] Hughey, R. & Krogh, A., (1996). "Hidden Markov models for sequence analysis: Extension and analysis of the basic method". *CABIOS* 12(2):95-107.

- [37] Hush, D., & Horn, B. (1993). "Progress in supervised neural networks: What's new since Lippmann?", *IEEE Signal Processing Magazine*, vol.10, no.2, pp.8-39.
- [38] Jacobs, R. A., Jordan, M. I., Nowlan, S. J., & Hinton, G. E. (1991). "Adaptive mixtures of local experts", *Neural Computation*, vol.3, pp.79-97.
- [39] Jordan, M. I., & Jacobs, R. A. (1994). "Hierarchical mixtures of experts and the EM algorithm", *Neural Computation*, vol.6, pp.181-214.
- [40] Kawato, M. (1991). "Computational schemes and neural network models for formation and control of multijoint arm trajectory", in W. T. Miller III, R. S. Sutton & P. J. Werbos (eds). *Neural Networks for Control*, pp.197-228. Cambridge, MA: MIT Press.
- [41] Kohonen, T. (1997). *Self-Organizing Maps*. 2nd extended edition, Springer-Verlag, Berlin.
- [42] Koogan, A. & Houaiss, A. (1994). *Enciclopédia e Dicionário*. Edições Delta, Rio de Janeiro.
- [43] Kosmatopoulos, E. A., Polycarpou, M. M., Christodoulou, M. A., & Ioannou, P. A. (1995). "High-order neural network structures for identification of dynamical systems", *IEEE Transactions on Neural Networks*, vol.6, no.2, pp.422-431.
- [44] Kremer, S. C. (1995). "On the computational power of Elman-style recurrent networks", *IEEE Transactions on Neural Networks*, vol.6, no.4, pp.1000-1004.
- [45] Krogh, A. & Riis, S. K., (1999). "Hidden neural networks". *Neural Computation*, vol. 11 pp.541-563.
- [46] Kundu, A. & Chen, G. C., (1997). "An integrated hybrid neural network and hidden Markov model classifier for sonar signals". *IEEE Transactions on Signal Processing*, 45(10):2566-2570.
- [47] LeCun, Y., (1993). Efficient learning and second-order methods, A tutorial at NIPS 93, Denver.
- [48] Lee, K. Y., McLaughlin, S., & Shirai, K., (1997). "Speech enhancement based on neural predictive hidden Markov model". *Signal Processing*, 65:373-381.

- [49] Lin, T., Horn, B. G., & Giles, C. L. (1998). "How embedded memory in recurrent neural network architectures helps learning long-term temporal dependencies". *Neural Networks*, vol.11, pp.861-868.
- [50] Magoulas, G. D., Vrahatis, M. N., & Androulakis, G. S., (1999). "Improving the convergence of the backpropagation algorithm using learning rate adaptation methods". *Neural Computation* 11:1769-1796.
- [51] Miller, D., Rao, A. V., Rose, K., & Gersho, A., (1996). "A global optimization technique for statistical classifier design". *IEEE Transactions on Signal Processing* vol.44 no.12 pp.3108-3122.
- [52] Mozer, M. C. (1989). "A focused back-propagation algorithm for temporal pattern recognition", *Complex Systems*, vol.3, pp.349-381.
- [53] Murre, J. (1992). *Learning and categorization in modular neural networks*. Harvester-Wheatcheaf.
- [54] Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Mateo, CA.
- [55] Pineda, F. J. (1989). "Recurrent Back-Propagation and the Dynamical Approach to Adaptive Neural Computation." *Neural Computation*, vol.1, pp 161-172.
- [56] Rabiner, L. R. (1989). "A tutorial on Hidden Markov Models and selected applications in speech recognition", *Proceedings of the IEEE*, vol.77, no.2, pp.257-286.
- [57] Resnick, S. I. (1992). *Adventures in Stochastic Processes*. Birkhäuser, Boston.
- [58] Rose, K. (1998). "Deterministic annealing for clustering, compression, classification, regression, and related optimization problems". *Proceedings of the IEEE* vol. 86 no.11 pp.2210-2239.
- [59] Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). "Learning internal representations by error propagation", in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, vol 1, ed. David E. Rumelhart, J. L. McClelland, and the PDP Research Group, MIT Press, Cambridge, Mass.
- [60] Rumelhart, D. E., Durbin, R., Golden, R., & Chauvin, Y., (1995). "Backpropagation: The basic theory", in *Backpropagation: Theory, Architectures, and Applications*, ed. Yves Chauvin e David E. Rumelhart, Lawrence Erlbaum Associates.

- [61] Sharkey, A. J. C. (1997). "Modularity, combining and artificial neural nets", *Connection Science*, vol.9, no.1, pp.3-10.
- [62] Srinivasa, N., & Ahuja, N. (1999). "A topological and temporal correlator network for spatiotemporal pattern learning recognition, and recall", *IEEE Transactions on Neural Networks*, vol.10, no.2, pp.356-371.
- [63] Stornetta, W. S., Hogg, T., & Huberman, B. A. (1988). "A dynamic approach to temporal pattern processing", in *Neural Information Processing Systems*, D. Z. Anderson (Ed.), pp.750-759.
- [64] Tsoi, A. C., & Back, A. D. (1994). "Locally recurrent globally feedforward networks: a critical review of architectures", *IEEE Transactions on Neural Networks*, vol.5, no.2, pp.229-239.
- [65] Waibel, A. (1989). "Modular construction of time-delay neural networks for speech recognition", *Neural Computation*, vol.1, pp.39-46.
- [66] Wang, D. L., & Arbib, M. A. (1990). "Complex temporal sequence learning based on short-term memory", *IEEE Proceedings*, vol.78, pp.1536-1543.
- [67] Wilson, A. D. & Bobick, A. F., (1999). "Parametric hidden Markov models for gesture recognition". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(9):884-900.
- [68] Yu, X.-H. Chen, G.-A., & Cheng, S.-X. (1995). "Dynamic learning rate optimization of the backpropagation algorithm." *IEEE Transactions on Neural Networks*, vol. 6, no. 3, pp. 669-677.
- [69] Yu, X.-H., & Chen, G.-A. (1997). "Efficient backpropagation learning using optimal learning rate and momentum". *Neural Networks*, vol. 10, no. 3, pp.517-527.

APÊNDICE A

Detalhes do modelo TOTEM

A.1. O Módulo SOM

Os pesos da unidade vencedora no módulo SOM e dos seus vizinhos são adaptados de acordo com a seguinte equação:

$$\Delta w_{ij} = \alpha h_{i^* j^*}(t) [x(t) - w_{ij}(t)] \quad (\text{A.1.1.})$$

sendo: (i^*, j^*) se referem à posição espacial do neurônio vencedor

w_{ij} é o vetor de pesos (da unidade vencedora ou de um vizinho) a ser corrigido;

α é a taxa de aprendizagem;

$x(t)$ é o padrão de entrada no instante t ;

A função $h_{i^* j^*}(t)$ é uma função do tipo Gaussiana utilizada para ponderar a modificação dos pesos de uma unidade (i, j) dependendo da distância Euclidiana para o vencedor (i^*, j^*) . Esta função é definida da seguinte forma:

$$h_{i^* j^*}(t) = e^{\frac{-[(i-i^*)^2 + (j-j^*)^2]}{2\sigma^2}} \quad (\text{A.1.2.})$$

sendo σ uma variável que define o tamanho da vizinhança que sofrerá alterações nos vetores de pesos.

A.2. O Módulo STM

Na representação adotada, um terceiro índice ($k = f_{ij}$) é utilizado para cada unidade do módulo STM. A ativação x da unidade (i^*, j^*, k) do 2º módulo, excitada pelo vencedor (i^*, j^*) do módulo SOM, é dada pela seguinte equação :

$$x_{i^* j^* k}(t) = \frac{A_1 J_{i^* j^* k} - CX}{B_1 + X + J_{i^* j^* k}} \quad (\text{A.2.1.})$$

sendo que $J_{i^* j^* k}$ é o peso da conexão entre o neurônio do módulo SOM e o do 2º módulo;

$X = \sum_{(i,j) \neq (i^*, j^*)} x_{ijk}(t)$ representa a inibição lateral;

A_1, B_1, C_1 são parâmetros de ajuste da função.

A ativação x da unidade (i, j, k) não referente a um vencedor da camada SOM, é dada por:

$$x_{ijk}(t) = \frac{A_2 x_{ijk}(t-1) - CX}{B_2 + X + x_{ijk}(t-1)} \quad (\text{A.2.2.})$$

Se os parâmetros forem ajustados de forma que $A_1 \ll 1$, $A_2 \ll 1$, $B_1 = B_2 \cong 1$ e $C = 0$, então, a ativação x será uma função monotonicamente decrescente no tempo. E assim sendo, a ordem dos padrões entrados será dada pela intensidade das ativações no módulo STM.

A.3. O 3º Módulo do Modelo TOTEM

Para cada unidade lida, utiliza-se seus índices i e j para encontrar pares do tipo $y_i = [y_{i1}, y_{i2}]$ através das seguintes equações:

$$y_{i1} = \frac{i}{n} \quad \text{e} \quad y_{i2} = \frac{j}{n} \quad (\text{A.3.1.})$$

E assim, uma sequência fornecida será codificada em um vetor $\mathbf{y} = [y_1, y_2, \dots, y_M]$.

A partir do vetor $\mathbf{y}$, calcula-se o seu complemento $\mathbf{y}^c = 1 - \mathbf{y}$, para formar o padrão $\mathbf{Y} = [\mathbf{y}, \mathbf{y}^c]$. Este último será apresentado para o 4º módulo do sistema, a rede fuzzy-ART. A codificação utilizando um vetor $\mathbf{y}$ e seu complemento $\mathbf{y}^c$ (chamada codificação complementar) ajuda a prevenir um problema comum em redes do tipo ART conhecido como *problema da proliferação de categorias* [17].

A.4. O Módulo Fuzzy-ART

A rede fuzzy-ART é formada por uma camada de entrada totalmente conectada à camada de saída, via pesos adaptáveis $\mathbf{Z}$. Cada unidade j na saída do Módulo fuzzy-ART é ativada utilizando operações de lógica nebulosa conforme mostra a seguinte equação:

$$T_j = \frac{|\mathbf{Y} \wedge \mathbf{Z}_j|}{\delta + |\mathbf{Z}_j|} \quad (\text{A.4.1.})$$

sendo δ um parâmetro que minimiza a recodificação;

$\wedge$ representa o operador “E” na lógica nebulosa;

$|Y \wedge Z_j|$ é definido na lógica nebulosa da seguinte forma $|Y \wedge Z_j| = \sum_i \min(Y_i, Z_{ji})$

$|Z_j|$ é definido na lógica nebulosa da seguinte forma $|Z_j| = \sum_i Z_{ji}$

A unidade de saída T_j que possuir a maior ativação indicará a categoria na qual foi agrupado o padrão Y desde que o critério de vigilância a seguir seja atendido:

$$\frac{|Y \wedge Z_j|}{|Y_j|} \geq \rho \quad (\text{A.4.2.})$$

sendo ρ conhecido como parâmetro de vigilância.

Caso o critério de vigilância seja atendido, então, os pesos Z_j são atualizados de acordo com a seguinte regra de aprendizagem:

$$Z_j = \beta |Z_j \wedge Y| + (1 - \beta) Z_j \quad (\text{A.4.3.})$$

sendo: β ($< 1,0$) é conhecido como fator de esquecimento

No caso de nenhum dos nós de saída atenderem ao critério de vigilância, então, uma nova unidade de saída é recrutada para aprender o padrão entrado.

APÊNDICE B

Alguns Conceitos de Processos Estocásticos

B.1. Cadeia de Markov

Considere um sistema que pode ser descrito em qualquer instante através de um estado (S_i) pertencente a um conjunto de n estados distintos, $S_1, S_2, \dots, S_n$ como ilustrado na Figura 41 (que por simplicidade mostra um exemplo de um sistema específico com apenas 4 estados possíveis, ou seja, $n = 4$).

Figura 41: Exemplo de cadeia de Markov com 4 estados (S_1, S_2, S_3 e S_4) e probabilidades de transição do estado i para o estado j ζ_{ij} .

Considere o tempo como sendo uma variável discreta com instantes regularmente espaçados. A cada instante de tempo, o sistema sofre uma mudança de estado (ou permanece no estado presente). Os instantes de tempo associados a cada mudança de estado serão denotados por $t = 1, 2, \dots$, e o estado do sistema no instante atual t será denotado por x_t . Uma descrição probabilística completa desse sistema, em geral, necessitaria da especificação do estado atual (no instante t), assim como de todos os estados passados (histórico do sistema), ou seja:

$$P[x_t = S_j \mid x_{t-1} = S_i, x_{t-2} = S_k, \dots] \quad (\text{B.1.1.})$$

Um caso especial do sistema discreto acima, é a cadeia de Markov de primeira ordem, na qual a descrição probabilística é truncada para apenas o estado atual e o seu predecessor, ou seja:

$$P[x_t = S_j \mid x_{t-1} = S_i, x_{t-2} = S_k, \dots] = P[x_t = S_j \mid x_{t-1} = S_i] \quad (\text{B.1.2.})$$

Além disto, considere que esse processo seja independente do tempo, e assim podemos definir a probabilidade de transição de estados da seguinte forma:

$$\zeta_{ij} = P[x_t = S_j \mid x_{t-1} = S_i], \quad 1 \leq i, j \leq n \quad (\text{B.1.3})$$

Ou seja, a probabilidade de transição do estado i para o estado j , ζ_{ij} , expressa a probabilidade de que, dado que o sistema, no instante $t-1$ possuía o estado $x_{t-1} = S_i$, no instante t , o sistema mude para o estado $x_t = S_j$. E a matriz de probabilidades de transição de estado é definida por:

$$\zeta = \begin{bmatrix} \zeta_{11} & \zeta_{12} & \cdots & \zeta_{1n} \\ \zeta_{21} & \zeta_{22} & \cdots & \zeta_{2n} \\ \vdots & & \ddots & \\ \zeta_{n1} & \zeta_{n2} & \cdots & \zeta_{nn} \end{bmatrix} \quad (\text{B.1.4.})$$

Para que essa matriz seja uma matriz estocástica, é preciso que as probabilidades de transição de estado atendam às seguintes restrições.

$$\zeta_{ij} \geq 0 \quad \text{e} \quad \sum_{j=1}^n \zeta_{ij} = 1 \quad (\text{B.1.5.})$$

Além disto, define-se $\pi_i = P [x_1 = S_i]$, $1 \leq i \leq n$, como sendo a probabilidade de um estado i ser o estado inicial do processo.

O processo estocástico definido é dito ser modelo Markoviano observável, se a saída desse processo a cada instante de tempo é um estado pertencente ao conjunto de estados possíveis, e principalmente, se qualquer estado desse conjunto corresponde a um evento físico observável⁷.

Uma vez determinado um modelo (como a cadeia de Markov descrita até o momento), é possível saber qual a probabilidade da ocorrência de uma seqüência de observações $O = \{x_1, x_2, \dots, x_k\}$ sendo $x_t \in S$, $t = 1, 2, \dots, k$. Esta probabilidade pode ser encontrada como mostra o exemplo a seguir:

Considere a seqüência de observações $O = \{S_2, S_3, S_1, S_4, S_1, S_3\}$. Seja M um modelo estocástico como a cadeia de Markov definida anteriormente, a probabilidade de ocorrência da seqüência O é dada por:

$$\begin{aligned} P(O|M) &= P[x_1 = S_2, x_2 = S_3, x_3 = S_1, x_4 = S_4, x_5 = S_1, x_6 = S_3 | M] \\ &= P[x_1 = S_2] \cdot P[x_2 = S_3 | x_1 = S_2] \cdot P[x_3 = S_1 | x_2 = S_3] \cdot P[x_4 = S_4 | x_3 = S_1] \cdot \\ &\quad \cdot P[x_5 = S_1 | x_4 = S_4] \cdot P[x_6 = S_3 | x_5 = S_1] \\ &= \pi_2 \cdot \zeta_{23} \cdot \zeta_{31} \cdot \zeta_{14} \cdot \zeta_{41} \cdot \zeta_{13} \end{aligned} \quad (\text{B.1.6.})$$

Estudos mais aprofundados a respeito de processos Markovianos podem ser encontrados em [35][57].

⁷ Um evento físico observável é simplesmente um evento cuja existência é conhecida, e sua ocorrência pode ser observada.

B.2. Os Modelos Markovianos Escondidos - HMMs

Um modelo Markoviano escondido é similar a uma cadeia de Markov pois em ambos os casos, a transição de um estado para outro é probabilística [33]. Porém, existe uma diferença fundamental. Enquanto em uma cadeia de Markov, a produção de um símbolo de saída é determinística, no modelo Markoviano escondido, os símbolos de saída também são probabilísticos, consequentemente, vários símbolos são possíveis para cada estado. Sendo assim, para cada estado de um modelo Markoviano escondido, tem-se uma distribuição de probabilidade dos símbolos de saída [33].

Os modelos Markovianos observáveis são muito restritivos⁸, e sendo assim, existem inúmeros problemas para os quais tais modelos não se aplicam [56]. Para tais problemas, é possível estender os conceitos de modelos Markovianos de forma a incluir casos onde a observação é uma função probabilística do estado, ou seja, o modelo resultante, conhecido como modelo Markoviano escondido, tem como base um processo estocástico que não é observável, e só pode ser acessado através de um outro conjunto de processos estocásticos que produzem seqüências de observações. Um exemplo prático para visualizar a idéia dos modelos Markovianos escondidos é o caso do jogo de moedas escondidas:

Imagine uma sala com dois ambientes separados por uma cortina, de forma que, para uma pessoa de um lado da sala seja impossível observar o que se passa do outro lado da sala. Esta pessoa apenas ouve um jogador (que está do outro lado) informando uma seqüência de caras e coroas. A forma mais simples de se modelar o experimento pode ser vista na Figura 42.

Figura 42: Cadeia de Markov simples para o experimento da moeda. Esse modelo considera apenas uma moeda sendo lançada.

Neste caso, acredita-se que o jogador utiliza uma única moeda (que pode ser viciada ou não) para gerar a seqüência de caras e coroas. Este é um exemplo onde foi utilizado um modelo Markoviano observável simples. Para que este modelo esteja completamente determinado, basta encontrar a probabilidade $P(\text{cara})$ (que será 0,5 para o caso em que a moeda não é viciada). No entanto, é possível que o jogador esteja utilizando aleatoriamente duas moedas

⁸ Os modelos Markovianos observáveis podem não ser adequados para inúmeros problemas que contenham estados escondidos (ou não observáveis). Gutterp (1995) mostra alguns exemplos nas áreas de meteorologia (para modelar a precipitação de chuvas em certas regiões), e da neurofisiologia (para modelar as trocas de íons pela membrana celular) onde o desempenho dos modelos Markovianos observáveis não é satisfatório.

diferentes, e além disso, tais moedas sejam viciadas de forma diferente. Nesse caso, o modelo Markoviano observável da Figura 42 não consegue modelar o processo de forma apropriada. Uma melhor forma de modelar esse processo passa a ser então considerar o modelo Markoviano escondido da Figura 43.

Figura 43: Modelo Markoviano Escondido que considera duas moedas diferentes sendo lançadas aleatoriamente no experimento da moeda.

Para esse modelo da Figura 43, cada estado implica na utilização de uma moeda correspondente, e além disto, as probabilidades de cara e coroa são definidas independentemente para as moedas 1 e 2. E dessa forma, se as moedas são viciadas de forma diferente, P_1 será diferente de P_2 . Se esse modelo também não servir, é possível então que o jogador esteja utilizando mais de duas moedas, todas diferentemente viciadas. Percebe-se então uma primeira dificuldade na obtenção de modelos Markovianos não observáveis, ou seja, não se sabe quantos estados escondidos estão envolvidos no processo. Além disso, é importante notar que no caso do modelo como uma moeda, havia apenas um parâmetro desconhecido (que é a probabilidade $P(\text{cara})$), já o segundo caso (com duas moedas), possui quatro parâmetros desconhecidos (ζ_{11} , ζ_{22} , P_1 , e P_2). Da mesma forma, se fosse utilizado um modelo supondo a existência de três moedas, haveriam nove parâmetros desconhecidos. Ou seja, considerar processos com mais estados escondidos significa considerar modelos com mais graus de liberdade, tais modelos conseguem abranger uma maior quantidade de experimentos, porém, em troca de um aumento do tamanho e da complexidade do processo, que terá mais parâmetros para serem estimados.

A seguir será feita uma definição formal dos elementos de um modelo Markoviano escondido. Tais modelos são caracterizados por:

- 1) Um número total n de estados do sistema. O conjunto de estados é denotado por $S = \{S_1, \dots, S_n\}$. Apesar da utilização de estados escondidos, em muitas aplicações práticas, existe um significado físico para tais estados ou conjuntos de estados do modelo. Como no exemplo anterior, no qual os estados escondidos correspondiam a moedas viciadas distintas.
- 2) Um número m de símbolos de observação, $V = \{v_1, v_2, \dots, v_m\}$. No exemplo anterior existiam dois símbolos, $V = \{\text{cara}, \text{coroa}\}$.

3) A distribuição de probabilidade de transição $\zeta = \{\zeta_{ij}\}$ sendo

$$\zeta_{ij} = P[x_t = S_j \mid x_{t-1} = S_i], \quad 1 \leq i, j \leq n \quad (\text{B.3.1.})$$

4) A distribuição de probabilidade de observação dos símbolos no estado j , $\mathbf{B} = \{b_j(k)\}$, sendo

$$b_j(k) = P[v_k \text{ em } t \mid x_t = S_j], \quad \begin{matrix} 1 \leq j \leq n \\ 1 \leq k \leq m \end{matrix} \quad (\text{B.3.2.})$$

5) A distribuição de estados iniciais $\Pi = \{\pi_i\}$ sendo

$$\pi_i = P[x_1 = S_i], \quad 1 \leq i \leq n \quad (\text{B.3.3.})$$

Dados os valores apropriados de n , m , ζ , $\mathbf{B}$, e π , o HMM pode ser utilizado como um gerador para uma dada seqüência de observação $\mathbf{O} = O_1, O_2, \dots, O_T$, sendo cada observação O_t é um dos símbolos de $\mathbf{V}$, e T é o número de observações na seqüência). A geração pode ser feita da seguinte forma:

- 1) Escolha um estado inicial $q_1 = S_i$ de acordo com a distribuição de estados inicial π .
- 2) Faça $t = 1$.
- 3) Escolha $O_t = v_k$ de acordo com a probabilidade de distribuição do símbolo no estado S_i , ou seja, de acordo com $b_i(k)$.
- 4) Faça a transição para o novo estado $q_{t+1} = S_j$ de acordo com a distribuição de probabilidade de transição de estado para o estado S_j , ou seja, de acordo com ζ_{ij} .
- 5) Faça $t = t + 1$; se $t < T$ então volte para o passo 3; senão termine o procedimento.

Perceba então que uma especificação completa de um HMM requer a especificação dos parâmetros n e m , assim como a especificação dos símbolos de observação e das probabilidades (ζ , $\mathbf{B}$ e Π). Por uma questão de conveniência e de simplificação de notação, é comum utilizar a notação compacta $\lambda = (\zeta, \mathbf{B} \text{ e } \Pi)$ para indicar um HMM.

Para um estudo mais aprofundado a respeito de HMM's, Rabiner [56] apresenta um tutorial, explicando detalhadamente o funcionamento de tais modelos, suas características e propriedades, as dificuldades encontradas no modelamento, além de aplicações em problemas de reconhecimento da fala.

APÊNDICE C

Resultados

C.1. Resultados Obtidos com a Rede Recorrente Operando Sozinha e com o Sistema Modular Proposto na Recuperação das Trajetórias do Tipo CC

Figura 44: Resultado obtido com a rede parcialmente recorrente reproduzindo as trajetórias CC, quando os pontos inicial e final fornecidos são iguais aos treinados.

Figura 45: Resultado obtido com o sistema modular proposto reproduzindo as trajetórias CC, quando os pontos inicial e final fornecidos são iguais aos treinados.

Figura 46: Resultado obtido com a rede parcialmente recorrente reproduzindo os ângulos das juntas para as trajetórias CC, quando os pontos inicial e final fornecidos são iguais aos treinados.

Figura 47: Resultado obtido com a rede parcialmente recorrente reproduzindo os torques das juntas para as trajetórias CC, quando os pontos inicial e final fornecidos são iguais aos treinados.

Figura 48: Resultado obtido com o sistema modular proposto reproduzindo os ângulos das juntas para as trajetórias CC, quando os pontos inicial e final fornecidos são iguais aos treinados.

Figura 49: Resultado obtido com o sistema modular proposto reproduzindo os torques das juntas para as trajetórias CC, quando os pontos inicial e final fornecidos são iguais aos treinados.

C.2. Resultados Obtidos com a Rede Recorrente Operando Sozinha e com o Sistema Modular Proposto na Recuperação das Trajetórias do Tipo 1-m

Figura 50: Resultado obtido com a rede parcialmente recorrente reproduzindo as trajetórias 1-m, quando os pontos inicial e final fornecidos são iguais aos treinados.

Figura 51: Resultado obtido com o sistema modular proposto reproduzindo as trajetórias 1-m, quando os pontos inicial e final fornecidos são iguais aos treinados.

Figura 52: Resultado obtido com a rede parcialmente recorrente reproduzindo os ângulos das juntas para as trajetórias 1-m, quando os pontos inicial e final fornecidos são iguais aos treinados.

Figura 53: Resultado obtido com a rede parcialmente recorrente reproduzindo os torques nas juntas para as trajetórias 1-m, quando os pontos inicial e final fornecidos são iguais aos treinados.

Figura 54: Resultado obtido com o sistema modular proposto reproduzindo os ângulos das juntas para as trajetórias 1-m, quando os pontos inicial e final fornecidos são iguais aos treinados.

Figura 55: Resultado obtido com o sistema modular proposto reproduzindo os torques nas juntas para as trajetórias 1-m, quando os pontos inicial e final fornecidos são iguais aos treinados.

C.3. Resultados Obtidos com a Rede Recorrente Operando Sozinha e com o Sistema Modular Proposto na Recuperação das Trajetórias do Tipo m-1

Figura 56: Resultado obtido com a rede recorrente reproduzindo as trajetórias m-1, quando os pontos inicial e final fornecidos são iguais aos treinados.

Figura 57: Resultado obtido com o sistema modular proposto reproduzindo as trajetórias m-1, quando os pontos inicial e final fornecidos são iguais aos treinados.

Figura 58: Resultado obtido com a rede recorrente reproduzindo os ângulos das juntas para as trajetórias m-1, quando os pontos inicial e final fornecidos são iguais aos treinados.

Figura 59: Resultado obtido com a rede recorrente reproduzindo os torques nas juntas para as trajetórias m-1, quando os pontos inicial e final fornecidos são iguais aos treinados.

Figura 60: Resultado obtido com o sistema modular proposto reproduzindo os ângulos das juntas para as trajetórias m-1, quando os pontos inicial e final fornecidos são iguais aos treinados.

Figura 61: Resultado obtido com o sistema modular proposto reproduzindo os torques nas juntas para as trajetórias 1-m, quando os pontos inicial e final fornecidos são iguais aos treinados.

C.4. Resultados Obtidos com o Sistema Modular Proposto na Recuperação das Trajetórias Fechadas Partindo de Situações Treinadas

Figura 62: Resultado obtido com o sistema modular proposto reproduzindo as trajetórias fechadas, quando os pontos inicial e final fornecidos são iguais aos treinados.

Figura 63: Resultado obtido com o sistema modular proposto reproduzindo os ângulos das juntas para as trajetórias fechadas, quando os pontos inicial e final fornecidos são iguais aos treinados.

Figura 64: Resultado obtido com o sistema modular proposto reproduzindo os torques nas juntas para as trajetórias fechadas, quando os pontos inicial e final fornecidos são iguais aos treinados.

C.5. Resultados Obtidos com a Rede Parcialmente Recorrente e com o Sistema Modular Proposto na Recuperação das Coordenadas XYZ da Trajetória Circular

Figura 65: Resultados da reprodução da trajetória circular fechada, obtidos com os dois modelos implementados, partindo de diferentes pontos intermediários: (a) e (b) partindo do 10º ponto; (c) e (d) partindo do 25º ponto; (e) e (f) partindo do 40º ponto. As figuras da esquerda se referem à rede parcialmente recorrente, enquanto que as figuras da direita se referem ao sistema modular desenvolvido nesta pesquisa.