UNIVERSIDADE FEDERAL DE GOIÁS

ESCOLA DE ENGENHARIA ELÉTRICA, MECÂNICA E DE COMPUTAÇÃO

CURSO DE ENGENHARIA DE COMPUTAÇÃO

RONALDO GOMES SILVA

REDES NEURAIS AUTO-ORGANIZÁVEIS NA VISUALIZAÇÃO DA FALA COMO RECURSO FONOAUDIOLÓGICO

TRABALHO DE CONCLUSÃO DE CURSO

GOIÂNIA

2019

TERMO DE CIÊNCIA E DE AUTORIZAÇÃO PARA DISPONIBILIZAR VERSÕES ELETRÔNICAS DE TRABALHO DE CONCLUSÃO DE CURSO DE GRADUAÇÃO NO REPOSITÓRIO INSTITUCIONAL DA UFG

Na qualidade de titular dos direitos de autor, autorizo a Universidade Federal de Goiás (UFG) a disponibilizar, gratuitamente, por meio do Repositório Institucional (RI/UFG), regulamentado pela Resolução CEPEC nº 1204/2014, sem ressarcimento dos direitos autorais, de acordo com a Lei nº 9610/98, o documento conforme permissões assinaladas abaixo, para fins de leitura, impressão e/ou *download*, a título de divulgação da produção científica brasileira, a partir desta data.

1. Identificação do Trabalho de Conclusão de Curso de Graduação (TCCG):

Nome completo do autor: Ronaldo Gomes Silva

Título do trabalho: Redes Neurais Auto-Organizáveis na Visualização da Fala como Recurso Fonoaudiológico

2. Informações de acesso ao documento:

Concorda com a liberação total do documento [x] SIM [ ] NÃO[1]

Havendo concordância com a disponibilização eletrônica, torna-se imprescindível o envio do(s) arquivo(s) em formato digital PDF do TCCG.

Ronaldo Gomes Silva

Ronaldo Gomes Silva[2]

Ciente e de acordo:

Weber V Martins

Prof. Weber Martins, PhD (EMC/UFG)[2] Data: 22 / 07 / 2019

[1] Neste caso o documento será embargado por até um ano a partir da data de defesa. A extensão deste prazo suscita justificativa junto à coordenação do curso. Os dados do documento não serão disponibilizados durante o período de embargo.
Casos de embargo:
- Solicitação de registro de patente;
- Submissão de artigo em revista científica;
- Publicação como capítulo de livro;
- Publicação da dissertação/tese em livro.

[2] A assinatura deve ser escaneada.

RONALDO GOMES SILVA

REDES NEURAIS AUTO-ORGANIZÁVEIS NA VISUALIZAÇÃO DA FALA COMO RECURSO FONOAUDIOLÓGICO

Trabalho de conclusão de Curso apresentado como como requisito parcial à obtenção do título de Bacharel em Engenharia de Computação, da Escola de Engenharia Elétrica, Mecânica e de computação, da Universidade Federal de Goiás.

Orientador: Prof. Weber Martins, PhD

GOIÂNIA

2019

Ficha de identificação da obra elaborada pelo autor, através do
Programa de Geração Automática do Sistema de Bibliotecas da UFG.

Gomes Silva, Ronaldo
Redes Neurais Auto-Organizáveis na Visualização da Fala como Recurso Fonoaudiológico [manuscrito] / Ronaldo Gomes Silva. - 2019.
53 f.: il.

Orientador: Prof. Weber Martins.
Trabalho de Conclusão de Curso (Graduação) - Universidade Federal de Goiás, Escola de Engenharia Elétrica, Mecânica e de Computação (EMC), Engenharia de Computação, Goiânia, 2019.
Bibliografia. Apêndice.
Inclui siglas, abreviaturas, símbolos, gráfico, tabelas, lista de figuras, lista de tabelas.

1. coeficientes cepstrais de frequência em escala mel. 2. mapa auto-organizáveis. 3. mfcc. 4. reconhecimento automático de fala. 5. visualização da fala. I. Martins, Weber, orient. II. Título.

CDU 62:004.3/.4

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE GOIÁS
ESCOLA DE ENGENHARIA ELÉTRICA, MECÂNICA E DE COMPUTAÇÃO
COORDENAÇÃO DE ESTÁGIO E PROJETO FINAL

ATA DE AVALIAÇÃO DE PROJETO FINAL

CURSO

( ) Eng. Elétrica ( ) Eng. Mecânica (X) Eng. de Computação
( ) Projeto Final 1 (X) Projeto Final II

AVALIAÇÃO DE PROJETO FINAL

Título do projeto: Redes Neurais Auto Organizáveis na Visualização da Fala como Recurso Fonoaudiológico

BANCA AVALIADORA

Membro 1: Weber Martins
Membro 2: Rodrigo Pinto Lemos
Membro 3: Wellington Santos Martins

ESTUDANTES

Matrícula	Nome
201107638	Ronaldo Gomes Silva

NOTAS

Matrícula	Membro 1				Membro 2				Membro 3				Média
	NPT	NTE	NAA	NF	NPT	NTE	NAA	NF	NPT	NTE	NAA	NF	
201107638	8,0	8,0	8,0	8,0	8,0	8,0	8,0	8,0	8,0	8,0	8,0	8,0	8,0

NPT – Nota plano de trabalho; NTE – Nota do trabalho escrito; NAA – Nota de apresentação e arguição
Para Eng. Elétrica, Mecânica e PFC2 da Eng. Da Computação: NF = 0,1 x NPT + 0,45xNTE + 0,45 x NAA
Para PFC1 da Eng. Da Computação: NF = 0,3 x NPT + 0,7 x NAA

Goiânia, 10 de julho de 2019.

Weber Martins
Membro 1

Membro 2

Membro 3

FREQUÊNCIA – a ser preenchido pelo orientador(a)	
Nome do(a) estudante	Frequência (%)
Ronaldo Gomes Silva	100%

Weber Martins
Professor(a) Orientador(a)

ATA DE AVALIAÇÃO DE PROJETO FINAL - Observações

Preencher com modificações solicitadas, caso existam. Em caso de reprovação, informar a justificativa.

A apresentação seguiu o trâmite normal e as alterações foram encaminhadas por e-mail.

Goiânia, 10 de julho de 2019.

Weber Martins

AGRADECIMENTOS

Agradeço ao meu orientador Prof. PhD Weber Martins, por me guiar com dedicação nesta importante etapa da minha vida acadêmica.

Ao suporte oferecido pela EMC através da coordenação do curso de Engenharia de Computação.

Aos meus amigos Renan S. Miranda e Tainá A. S. Marchewicz pela gentil colaboração neste trabalho.

Agradeço aos amigos e colegas que participaram e contribuíram para a minha formação.

Por fim registro profunda gratidão a minha família e em especial ao meu pai Geraldo Gomes Dutra (*In Memoriam*) por compreenderem que a educação é o único caminho para emancipação do ser humano.

RESUMO

A fala é um importante elemento de socialização e desenvolvimento do indivíduo. Entretanto, milhões de pessoas não a desenvolve parcial ou completamente. Os distúrbios de comunicação verbal normalmente surgem como anomalias secundárias decorrentes de outras patologias, como a deficiência auditiva. Embora exista número expressivo de deficientes auditivos, os estudos e emprego de recursos computacionais no treinamento e tratamento fonoaudiológico para aquisição e aprimoramento da fala ainda são tímidos. Neste contexto, o presente trabalho propõe o uso de redes neurais auto-organizáveis como auxílio no processo de treinamento fonoaudiológico de indivíduos com distúrbios da fala, por meio de *feedback* visual. São descritos os processos necessários para o desenvolvimento do sistema proposto, além da realização de experimentos com o Mapa Auto-Organizável de Kohonen, usando como descritores de características da fala os Coeficientes Cepstrais de Frequência em Escala Mel. Foram definidos alguns cenários de configuração tanto dos parâmetros do mapa auto-organizável quanto dos coeficientes cepstrais, onde se obteve o mapa fonético topológico de escopo reduzido que produziu menor percentual de erro e equívocos, sendo 35,83% e 35,42%, respectivamente.

Palavras-chave: coeficientes cepstrais de frequência em escala mel, mapa auto-organizáveis, mfcc, reconhecimento automático de fala, som, visualização da fala.

ABSTRACT

Speech is an important element of socialization and development of the individual. However, millions of people do not develop it partially or completely. Verbal communication disorders usually appear as secondary anomalies due to other pathologies, such as hearing impairment. Although there is an expressive number of hearing impaired, the studies and use of computational resources in training and speech therapy for speech acquisition and improvement are still timid. In this context, the present work proposes the use of self-organizing neural networks as an aid in the speech-language training process of individuals with speech disorders, through visual feedback. The processes required for the development of the proposed system are described, as well as experiments with Kohonen's Self-Organizing Map, using Mel Frequency Cepstral Coefficients as descriptors of speech characteristics. Some configuration scenarios were defined for both the self-organizing map parameters and the cepstral coefficients, where the topological phonetic map of reduced scope was obtained, which produced a lower percentage of error and misconceptions, being 35.83% and 35.42%, respectively.

Keywords: automatic speech recognition, mel frequency cepstral coefficients, mfcc, self-organizing maps, som

LISTA DE ILUSTRAÇÕES

LISTAS DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

ASR	*Automatic Speech Recognition*
DCT	*Discrete cosine transform*
DFT	*Discrete Fourier Transform*
FFT	*Fast Fourier transform*
IBGE	Instituto Brasileiro de Geografia e Estatística
IPA	*International Phonetic Alphabet*
LPC	*Linear Predictive Coding*
MFCC	*Mel Frequency Cepstral Coefficient*
OMS	Organização Mundial da Saúde
PNCC	*Power-Normalized Cepstral Coefficients*
Praat	Programa de análise acústica da fala, (do holandês, conversar)
RNA	Rede Neural Artificial
UTF-16	*Unicode Transformation Format (16-bit)*
WAV	*Waveform Audio File Format*

SUMÁRIO

1 INTRODUÇÃO

O **Reconhecimento Automático da Fala**[1] é a área de estudo que objetiva principalmente a transcrição da voz humana em fonemas, palavras ou sentenças. Tal objetivo resulta em tarefas complexas e multidisciplinares por lidarem com sinais não estacionários, que possuem várias propriedades trabalhadas em diversas áreas de estudos, como Processamento Digital de Sinais, Redes Neurais Artificiais e Linguística (Fonética e Fonologia).

Com aprimoramento das técnicas de Inteligência Artificial, especificamente das Redes Neurais Artificiais (RNA), o Reconhecimento Automático da Fala tem se tornado cada vez mais presente em aplicações cotidianas, como interface de comunicação em assistentes virtuais, comando de voz etc.

Kohonen (1982) apresentou o modelo de Rede Neural Artificial conhecido como **Mapas Auto-Organizáveis**, SOM (do inglês, *Self-Organizing Maps*). Os Mapas auto-organizáveis estão entre as mais importantes arquiteturas de redes neurais com aprendizado não supervisionado. São aplicados em diversas áreas da ciência e da engenharia como: **processamento de imagens**, **mecânica**, **geomorfologia**, **biologia**, **meteorologia, oceanografia**, **medicina**, **modelagem**, **processamento de fala** etc. (MWASIAGI, 2011).

Kohonen (1988) demonstrou a aplicação da rede SOM no reconhecimento e visualização da fala, com a construção de mapas fonéticos topológicos do idioma finlandês. Essa abordagem se mostra potencialmente interessante na busca por soluções em Reconhecimento Automático de Fala devido à boa relação custo-benefício.

Fonética e **Fonologia** fornecem conhecimentos da formação, transmissão e recepção do som, bem como da organização dos sons no contexto do idioma. Tais conhecimentos são importantes para o reconhecimento automático da fala por computadores e podem ser aplicados como ferramenta na aquisição e desenvolvimento da fala.

A linguagem falada é o principal meio pelo qual as pessoas se comunicam, sendo elemento relevante para socialização e desenvolvimento do indivíduo na

[1] Termo comumente apresentado pela sigla ASR (do inglês, *Automatic Speech Recognition*).

comunidade. Entretanto, milhões de pessoas não desenvolvem, parcial ou completamente, a comunicação verbal, devido a patologias como a deficiência auditiva, gagueira, afasia entre outras. Essas pessoas são afetadas pelos distúrbios da fala e carecem de acompanhamento e tratamento fonoaudiológico entre outras especialidades médicas.

Oliveira, Goulart e Chiari (2013) destacam a falta de estudos para verificar a associação entre distúrbios de fala e deficiência auditiva, registrando que, quanto mais severa e precoce, maior interferência da deficiência na fluência do indivíduo. Assim, a população de deficientes auditivos é provavelmente afetada pelos distúrbios de linguagem, especialmente nos casos de manifestação precoce de surdez.

Segundo a Organização Mundial da Saúde (OMS), cerca de 466 milhões de pessoas no mundo sofrem de perdas auditivas, sendo que 34 milhões são crianças e adolescentes até 15 anos de idade (Deafness, 2018). Os dados oficiais brasileiros mostram mais de 7,5 milhões de pessoas com dificuldades auditivas (IBGE, 2010). Nesse contexto, aproximadamente 1,8 milhões apresentam a deficiência de forma severa. Os casos extremos, onde o deficiente é completamente surdo, alcançam cerca de 344 mil pessoas. Todos esses dados demonstram a grande necessidade de investir em soluções para a melhoria da habilidade de comunicação para essas pessoas.

Os distúrbios de fala são condições clínicas que afetam a capacidade de comunicação e, por consequência, a qualidade e os anseios de vida do indivíduo. Entretanto, são tímidos os estudos e aplicações de recursos computacionais para minimizar tais condições. O uso de redes neurais auto-organizáveis na visualização da fala como recurso fonoaudiológico surge, portanto, como oportuna contribuição da Inteligência Artificial. Mais especificamente, pode-se perguntar: soluções baseadas em redes neurais auto-organizáveis adequam-se satisfatoriamente ao modelo de treinamento fonoaudiológico com *feedback* visual?

O objetivo geral do presente trabalho é, portanto, propor o uso de redes neurais auto-organizáveis como auxílio no processo de treinamento fonoaudiológico de pessoas com distúrbios da fala.

De forma mais específica, o trabalho pretende:

1. Descrever os passos necessários para o desenvolvimento da aplicação de reconhecimento e visualização da fala;

2. Criar um mapa fonético topológico do Português Brasileiro, usando redes neurais auto-organizáveis propostas por Kohonen;
3. Verificar a efetividade da extração de coeficientes cepstrais de frequência, MFCC (do inglês, *Mel Frequency Cepstral Coefficients*) na distinção dos diferentes fonemas do português brasileiro.

Parte-se da hipótese que o Mapa Auto-Organizável aplicado ao problema de reconhecimento e visualização dos fonemas do Português Brasileiro, pode trazer precisão e tempo de resposta satisfatórios em treinamento para aquisição e aprimoramento da fala por pessoas com distúrbios de linguagem.

Assim, para verificar a hipótese, realiza-se uma pesquisa bibliográfica, de caráter descritivo e exploratório, utiliza-se de fontes primárias e secundárias, com realização de análises quantitativas e qualitativas tanto nas fontes bibliográficas quanto nas amostras de áudios que compõe o trabalho.

Este trabalho é composto por cinco capítulos. Após esta introdução, o segundo capítulo apresenta a fundamentação teórica. No Capítulo 3, é descrito o sistema proposto. Os resultados e discussões ocupam o quarto capítulo, ficando a conclusão para o capítulo seguinte.

Mais especificamente, o segundo capítulo apresenta as áreas básicas envolvidas no trabalho: Fonética, Fonologia, Redes Neurais Auto-Organizáveis, Processamento Digital de Sinais de Áudio, Extração de Atributos de Sinais da Fala por meio de coeficientes cepstrais de frequência.

O Capítulo 3 detalha a proposta do ponto de vista funcional, com ênfase no mapa fonético topológico obtido como resultado do treinamento do Mapa Auto-organizável (SOM), além de especificar diretrizes para implementação em software.

O quarto capítulo mostra os experimentos realizados com a rede SOM, bem como as análises das amostras de áudios obtidas de banco online e coleta de amostras do próprio autor.

O Capítulo 5 conclui o trabalho, mostrando as principais contribuições da proposta e possíveis refinamentos na busca futura por melhores resultados da rede SOM.

Ao final, conclui-se que os objetivos foram atingidos e a questão-problema sendo respondida com o fortalecimento da hipótese. No entanto, observa-se a necessidade de etapas de refinamento para melhor precisão do sistema e aplicação em tempo real.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo, são apresentados os conceitos relacionados ao processamento e reconhecimento automático de fala, a partir de abordagem não-determinística, baseada em Redes Neurais Artificiais. Considerando os objetivos de ASR e a complexidade da fala humana, são abordados conhecimentos sobre **Fonética e Fonologia**, **Processamento Digital de Sinais de Áudio,** extração de características MFCC e Redes Neurais Artificiais, especificamente, **Mapas Auto-Organizáveis (SOM).**

2.1 Fonética e Fonologia

Fonética e Fonologia são ramos da linguística responsáveis por estudar os sons da fala humana. Cada uma delas tem objetivos distintos quanto ao estudo dos sons, entretanto, são interdependentes (MUSSALIM e BENTES, 2001).

Fonética é basicamente uma ciência descritiva, Fonologia por sua vez é uma ciência explicativa e interpretativa. Análises fonéticas baseiam-se na produção, percepção e transmissão dos sons da fala, análises fonológicas buscam o valor dos sons em uma língua, ou seja, sua função linguística.

2.1.1 Fonética

É o campo da linguística que estuda a natureza física da produção, transmissão e percepção dos sons da fala. O estudo é subdividido em fonética acústica, fonética articulatória e fonética auditiva.

Fonética Articulatória estuda os sons do ponto de vista fisiológico. Descreve e classifica os sons de acordo com as partes do corpo envolvidas em sua produção, a saber: pulmões; traqueia; laringe; epiglote; cordas vocais; glote; faringe; véu palatino; palato duro (ou céu da boca); palato mole; língua; dentes; mandíbula; lábios e cavidades nasais (OLIVEIRA, 2009).

Assim, na articulação da consoante “p”, por exemplo, pode-se afirmar a ausência de vibrações das cordas vocais, por isso ele é **não vozeado**, o fluxo de ar segue o caminho do trato vocal caracterizando-o como **oral**, observa-se obstrução do fluxo pelos dois lábios, indicando som **oclusivo** e **bilabial** (OLIVEIRA, 2009).

Fonética Acústica compreende o estudo das propriedades físicas do som, como a fala se propaga e chega ao aparelho auditivo. A análise do som e de sua propagação são realizadas com auxílio de programas computacionais específicos, com objetivo de avaliar propriedades como intensidade, frequência fundamental, harmônicos, etc.

Mussalim e Bentes (2001) observam existência de três tipos de preocupações gerais da Fonética Acústica: pesquisa da estrutura física dos sons da fala, pesquisa de fala sintética e pesquisa do reconhecimento automático da fala. O último tipo define o escopo de interesse deste trabalho.

Fonética Auditiva concentra os estudos na percepção dos sons da fala pelo aparelho auditivo. Eventualmente os mesmos sons são percebidos de formas distintas, apenas análises mais acuradas permitem identificá-los, esse papel é desempenhado pela fonética auditiva, embora seja pouco explorado, sobretudo no Brasil (OLIVEIRA, 2009).

A unidade básica da Fonética é o fone, isto é, a menor unidade de representação da fala dentro de um idioma (MUSSALIM e BENTES, 2001). Os **fones** são segmentos fonéticos diferenciados pelos movimentos articulatórios, observando o percurso do fluxo de ar e os órgãos participantes na formação do som (SEARA, NUNES e LAZZAROTTO-VOLCÃO, 2011). Portanto, todo trecho de fala pode ser transcrito em fones.

A transcrição fonética é uma etapa fundamental, pois precede inclusive a análise fonológica. Na transcrição pode-se usar os símbolos que compõem o Alfabeto Fonético Internacional conhecido por IPA (do inglês, *International Phonetic Alphabet*) (OLIVEIRA, 2009). Por convenção, a transcrição é feita com os símbolos entre colchetes, como mostra o Quadro 1.

Quadro 1 – Exemplos de transcrição fonética

Palavra	**Transcrição Fonética**
Faca	[fakɐ]
Vaca	[vakɐ]
Barba	[baɦbɐ]
Tia	[tʃiɐ]

Fonte: autor

IPA é um sistema alfabético de notação fonética baseado no alfabeto latino, criado no século XIX com o intuito de padronizar a representação dos sons de todas as línguas (INTERNATIONAL PHONETIC ASSOCIATION, 1999). O alfabeto é apresentado no quadro fonético (ver figuras 1, 2, 3 e 4), agrupados em: consoantes produzidas com mecanismos de corrente de ar pulmonar, consoantes produzidas com corrente de ar não-pulmonar e vogais (SILVA e YEHIA, 2008). O quadro apresentado nas figuras 1, 2, 3 e 4 é adaptado do IPA – 2005, destacando em **verde** os fones que ocorrem no Português Brasileiro.

Figura 1 - Consoantes (pulmonar)

	Bilabial		Labio-dental		dental		Alveolar		Pós-alveolar		Retroflexa		Palatal		Velar		Uvular		Faringal		Glotal	
Oclusiva	p	b					t	d			ʈ	ɖ	c	ɟ	k	g	q	ɢ			ʔ	
Nasal		m		ɱ				n				ɳ		ɲ		ŋ		ɴ				
Vibrante		ʙ						r										ʀ				
Tepe(ou Flepe)								ɾ				ɽ										
Fricativa	ɸ	β	f	v	θ	ð	s	z	ʃ	ʒ	ʂ	ʐ	ç	ʝ	x	ɣ	χ	ʁ	ħ	ʕ	h	ɦ
Fricativa lateral								ɬ	ɮ													
Aproximante				ʋ				ɹ				ɻ		j		ɰ						
Aprox. lateral								l				ɭ		ʎ		ʟ						

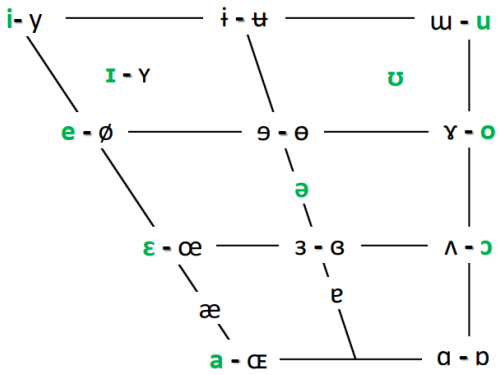
Fonte: Adaptado de Silva e Yehia (2008)

Figura 2 - Consoantes (não-pulmonar)

Cliques	Implosivas vozeadas	Ejectivas
ʘ - Bilabial	ɓ - Bilabial	pʼ - Bilabial
ǀ - Dental	ɗ - Dental/Alveolar	tʼ - Dental/alveolar
ǃ - Pós-alveolar	ʄ - Palatal	kʼ - Velar
ǂ - Palato-alveolar	ɠ - Velar	sʼ - Fricativa alveolar
ǁ - Lateral-alveolar	ʛ - Uvular	

Fonte: Adaptado de Silva e Yehia (2008)

Figura 3 - Vogais



Fonte: Adaptado de Silva e Yehia (2008)

Figura 4 - Consoantes africadas

Africadas
t͡s - Alveolar desvozeada
t͡ʃ - Palato-alveolar desvozeada
t͡ɕ - Alvéolo-palatal desvozeada
t͡ʂ - Retroflexa desvozeada
d͡z - Alveolar vozeada
d͡ʒ - Post-alveolar vozeada
d͡ʑ - Alvéolo-palatal vozeada
d͡ʐ - Retroflexa desvozeada

Fonte: Adaptado de Silva e Yehia (2008)

Estudos de Fonética fornecem fundamentos a áreas como Medicina, Ciências da Computação, Telecomunicações, Fonoaudiologia, etc. (MUSSALIM e BENTES, 2001). Sua compreensão exige maior aprofundamento, sobretudo em Fonética Articulatória, na formação da fala a partir do aparelho fonador humano. Entretanto, para o escopo deste trabalho, faz-se necessário o conhecimento sobre transcrição fonética, fones e os símbolos que os representam.

2.1.2 Fonologia

Estuda os sons de um idioma considerando sua função no sistema de comunicação linguístico. Preocupa-se com a forma como os sons se organizam dentro da língua, classificando-os em unidades capazes de distinguir significados, chamados **fonemas** (SEARA, NUNES e LAZZAROTTO-VOLCÃO, 2011). Os fonemas são representados entre barras inclinadas, /p/, /t/, /k/ (MUSSALIM e BENTES, 2001).

No Português, /f/ e /v/ são fonemas; pois possuem significados distintos no idioma. Por exemplo, as palavras [fakɐ] e [vakɐ] se distinguem apenas por [f] e [v], entretanto, possuem significados diferentes, “faca” e “vaca”, respectivamente. Por outro lado, [depojʃ] e [depojs] têm o mesmo significado “depois”, embora sejam diferentes por [ʃ] e [s].

No exemplo acima, [ʃ] e [s], são fones diferentes, entretanto, representam o mesmo fonema /S/. Em outras palavras, [ʃ] pode ser substituído por [s] sem prejuízos no reconhecimento da fala, esse fenômeno recebe o nome de **alofonia**. Portanto, define-se **fonema** como grupo composto por um **fone** e seus **alofones**, como mostra o Quadro 2 (SEARA, NUNES e LAZZAROTTO-VOLCÃO, 2011).

Quadro 2 - Quadro Fonêmico parcial do Português

Fonemas	Alguns alofones
/p/	[p], [p^w]
/b/	[b], [b^w]
/t/	[t], [t^w], [t^j], [tʃ]
/d/	[d], [d^w], [d^j], [dʒ]
/k/	[k], [w^w], [k^j]
/g/	[g], [g^w], [gk^j]
/f/	[f], [f^w]
/v/	[v], [V^w], [h]

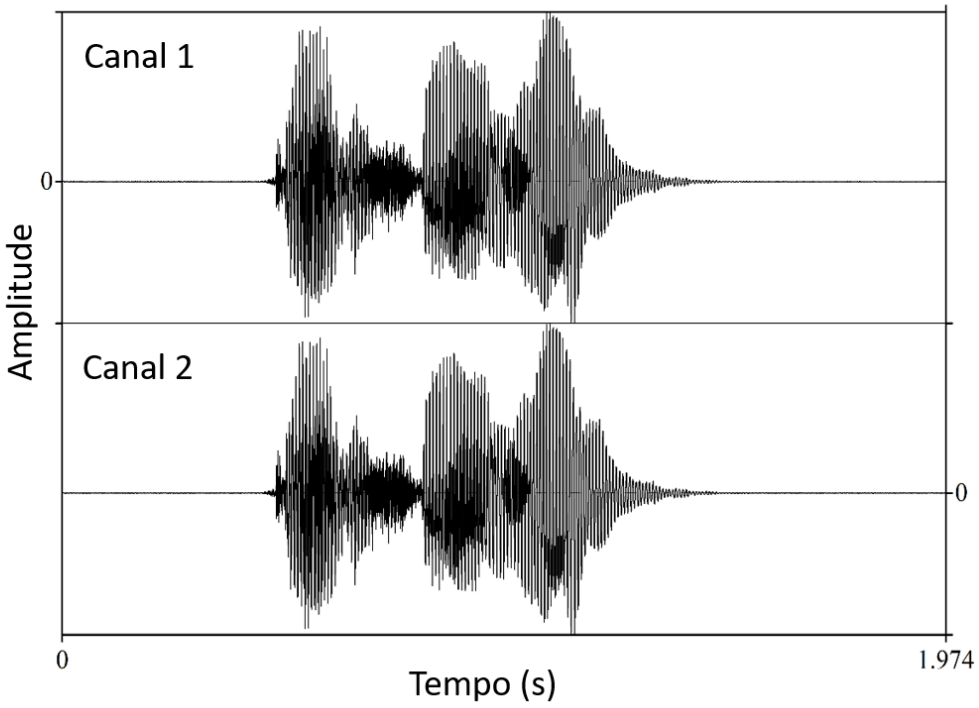
Fonte: adaptado de Silva e Yehia (2008)

2.2 Processamento Digital de Sinais de Áudio

A voz humana é composta por formas de ondas combinadas em diferentes frequências (LADEFOGET, 1996). Várias partes do corpo participam simultaneamente na produção da fala, vibrando, desviando ou obstruindo o fluxo de ar, produzindo sobreposição dos sinais sonoros. Desta forma, a fala é caracterizada como sinal não estacionário.

O sinal da fala é capturado por microfone e convertido em sinal discreto e digital para ser processado computacionalmente. A voz possui forma de onda senoidal, representado na Figura 5 em gráfico de amplitude em função do tempo. Entretanto, o sinal sonoro no domínio do tempo não é suficiente para encontrar as propriedades que distinguem os diferentes sons da fala.

Figura 5 – Forma de onda gerada no programa *Praat*



Fonte: autor

Bresolin (2008), destaca que dois sinais da mesma palavra no domínio do tempo são diferentes, independentemente de serem produzidos pela mesma pessoa. Entretanto, no domínio da frequência, são evidenciadas características importantes

que diferenciam os sons da fala. Para tanto, utiliza-se a transformada de Fourier, a fim de analisar e processar o sinal, aplicando técnicas de filtragem digital (MATUCK, 2005).

A **Transformada de Fourier** para funções contínuas representa um sinal integrável x(t) como soma de exponenciais complexas com frequência angular ω e amplitude complexa X(ω), apresentado na Equação 2.1 (BRESOLIN, 2008). A transformação permite a decomposição espectral nas frequências que constituem o sinal, mas não determina a relação entre tempo e frequência.

$$X(\omega)= \int_{-\infty}^{\infty} x(t).\, e^{-i\omega t} dt \qquad (2.1)$$

Pode-se reverter a transformada de Fourier obtendo o sinal original x(t), a partir da transformada inversa de Fourier, dada pela Equação 2.2.

$$x(t)=F^{-1}[X(\omega)] = \frac{1}{2\pi}\int_{-\infty}^{\infty} X(\omega).\, e^{-i\omega t} dt \qquad (2.2)$$

A **Transformada de Fourier de Tempo Curto**, STFT (do inglês, *Short-Time Fourier Transform*) analisa o sinal em pequenas seções. Cada seção (ou janela) é tratada com o sinal a ser analisado. STFT relaciona tempo e frequência em uma função bidimensional, mostrando quando e em quais frequências uma característica está presente no sinal (BRESOLIN, 2008).

Sinais analisados por ferramentas computacionais são amostrados, ou seja, discretos. Análises em sinais discretos são feitos por meio da **Transformada Discreta de Fourier**, DFT (do inglês, *Discrete Fourier Transform*). Bresolin (2008), observa que a DFT é uma transformação para análise de Fourier de funções discretas no tempo e de domínio finito. A DFT é definida pela Equação 2.3.

$$X_j = \sum_{k=0}^{n-1} x_k e^{\frac{2\pi i}{n} jk} \qquad j = 0, 1, 2, 3, \ldots, n-1 \qquad (2.3)$$

A **Transformada Discreta Inversa de Fourier** reconstitui o sinal original x_k, como mostra a Equação 2.4.

$$x_k = \frac{1}{n} \sum_{j=0}^{n-1} X_j \, e^{\frac{-2\pi i}{n} jk}, \qquad j = 0, 1, 2, 3, \ldots, n-1 \qquad (2.4)$$

A análise usando DFT é computacionalmente cara, pois possui complexidade $\mathcal{O}(n^2)$. A solução para este problema é a T**ransformada Rápida de Fourier**, FFT (do inglês, *Fast Fourier transform*), um algoritmo eficiente para calcular a transformada discreta Fourier e sua inversa. A FFT reduz a complexidade da DFT para $\mathcal{O}(n.\log_2 n)$ (BRESOLIN, 2008).

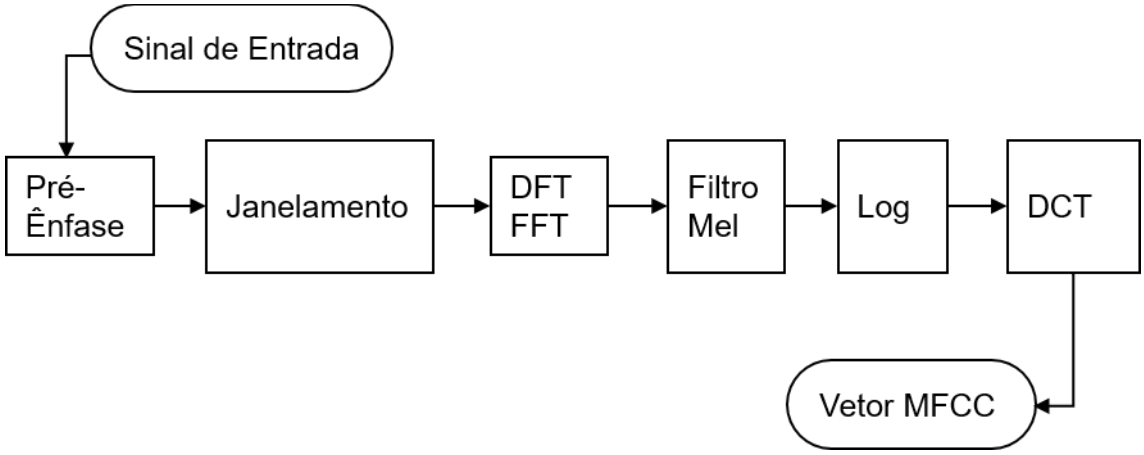
2.3 Extração de Coeficientes Cepstrais de Frequência de Escala Mel

A extração de características é importante para o reconhecimento da fala. Essa etapa permite a redução do espaço de entrada em vetores contendo atributos acústicos da voz. Portanto, aplicações em reconhecimento de fala utilizam técnicas para reduzir o tamanho da entrada de dados.

Os métodos mais utilizados na extração de atributos de áudio são LPC (do inglês, *Linear Predictive Coefficients*) e MFCC (do inglês, *Mel Frequency Cepstral Coefficients*). Coeficientes Cepstrais de Frequência de Escala Mel apresentam maior acurácia por compreender as características não lineares do sinal de voz. A técnica se baseia no sistema auditivo humano utilizando banco de filtros com espaçamento logaritmo (DAVE, 2013).

A extração de características MFCC é processada nas seguintes etapas: pré-ênfase; enquadramento e janelamento; Transformada Discreta de Fourier, banco de filtros Mel e Encapsulamento de frequência; cálculo do Log; e Transformada Discreta do Cosseno. As etapas são apresentadas na Figura 6.

Figura 6 – Etapas do método MFCC



Fonte: autor

- **Pré-ênfase**: O sinal de voz apresenta maior energia para as baixas frequências. A fim de compensar a atenuação em componentes de altas frequências utiliza-se o filtro pré-ênfase descrito pela Equação 2.5 (ROSA e VALLE, 2013). É um filtro de primeira ordem passa-altas.

$$y(n) = s(n) - \alpha s(n - 1), \quad n = 0, 1, 2, \ldots \qquad (2.5)$$

  Onde $s(n)$ é o sinal de entrada discreto no tempo, $y(n)$ o sinal compensado e α é uma constante com valores entre 0,9 e 1. Quanto mais próximo de 1 α estiver, mais ênfase o sinal receberá nas altas frequências.

- **Janelamento:** A transformada de Fourier não pode ser aplicada diretamente sobre a fala por se tratar de um sinal não estacionário. Porém, o sinal de voz pode ser segmentado em uma sequência de sinais estacionários. O áudio normalmente é dividido em janelas de 20 a 30ms, o que é condizente com a duração dos fonemas (CERÓN e BADILLO, 2011).

  Para atenuar a descontinuidade no início e fim de cada segmento é aplicado a janela de Hamming com deslocamentos típicos entre 5 e 10ms. O deslocamento produz uma sobreposição do sinal entre os segmentos justapostos (GORDILLO, 2013). A janela de Hamming é representada pela Equação 2.6.

$$w(n) = \begin{cases} 0{,}54 - 0{,}46\cos\left(\frac{2\pi n}{N-1}\right), & 0 \le n \le N\text{-}1, \\ 0 & \text{, caso contrário} \end{cases} \quad (2.6)$$

Para janelas com tamanho N, iniciando com n = 0.

- **DFT:** A Transformada Discreta de Fourier extrai informações espectrais do sinal. Assim, a DFT revela a energia contida nas diferentes faixas de frequências (GORDILLO, 2013). A DFT tem como saída um número complexo determinado pela Equação 2.3.

- **Banco de Filtros Mel:** A escala de frequência Mel foi desenvolvida com objetivo de mapear a forma que o sistema auditivo humano interpreta as diferentes frequências sonoras. Observou-se uma relação linear no intervalo [0, 1000] Hz, e logarítmica em frequências superiores (SÃO THIAGO, 2017). A relação entre as escalas Mel e frequência é aproximado pela Equação 2.7.

$$\text{Mel}(f) = 1125\ln\left(1 + \frac{f}{700}\right) \quad (2.7)$$

O banco de filtros é aplicado a todos trechos do sinal segmentado em etapa anterior. Cada filtro calcula a média espectral em torno da frequência central de acordo com a Equação 2.8 (GORDILLO, 2013). Isso permite simular a audição humana.

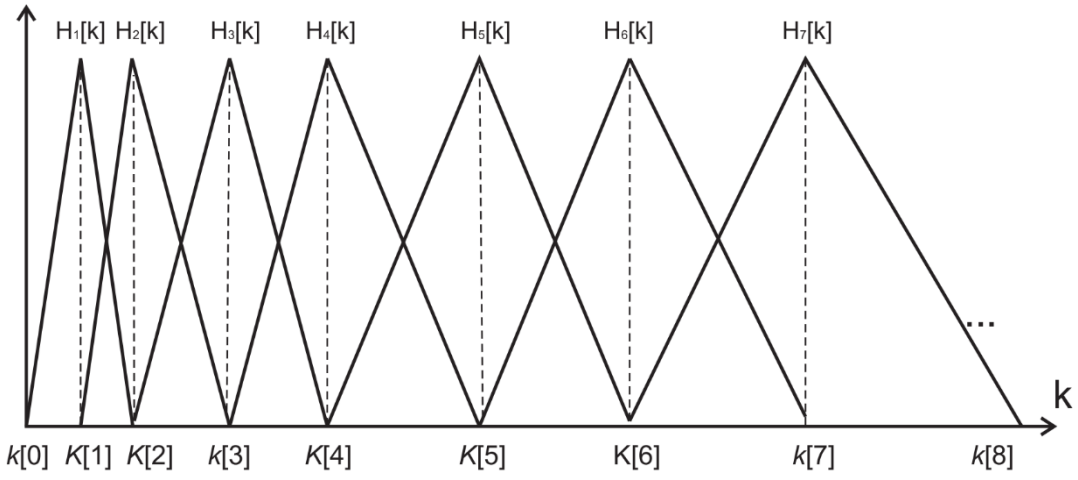
$$H_m[k] = \begin{cases} 0 & , \ k \le k[m-1] \\ \frac{2(k-k[m-1])}{(k[m+1]-k[m-1])(k[m]-k[m-1])} & , \ k[m-1] \le k \le k[m] \\ \frac{2(k[m+1]-k)}{(k[m+1]-k[m-1])(k[m+1]-k[m])} & , \ k[m] \le k \le k[m+1] \\ 0 & , k > k[m+1] \end{cases} \quad (2.8)$$

Onde:

- H_m é o m-ésimo filtro;
- K[m] é a frequência central do m-ésimo filtro;

O sinal é convertido usando bancos de filtros passa-banda, de resposta triangular cujo espaçamento e largura são obtidos por intervalos constantes de frequência Mel (SÃO THIAGO, 2017). A largura de banda aumenta com a frequência, como mostra a Figura 7.

Figura 7 – Banco de filtros usado no método MFCC



Fonte: Gordillo (2013)

- **Log:** Calcula-se o logaritmo da magnitude na saída dos filtros para obter os coeficientes cepstrais. O logaritmo retira parte da sensibilidade a variações de potência causadas pelas diferentes distâncias entre o falante e o microfone (ROSA e VALLE, 2013). Obtêm-se o logaritmo da energia por meio da Equação 2.9 na saída de cada filtro.

$$\hat{S}(m) = \ln\left(\sum_{m=0}^{\frac{N}{2}-1} S[k]H_m[k]\right),\ 1 < m < M \qquad (2.9)$$

- **DCT:** Finalmente, obtêm-se os coeficientes MFCC após aplicar a transformada inversa do cosseno (DCT) ao logaritmo obtido na etapa anterior (GORDILLO, 2013). Os coeficientes MFCC são calculados a partir da Equação 2.10.

$$c(n) = \sum_{m=0}^{M-1} \hat{S}[m] \cos\left(\frac{\pi n(m-0,5)}{M}\right), \quad 0 < n < M-1 \qquad (2.10)$$

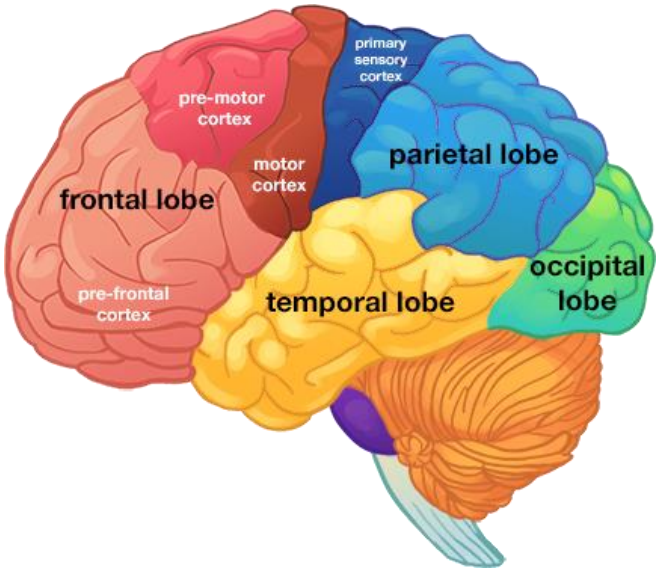
Onde n é o índice dos coeficientes MFCC e M é o número total de filtros.

2.4 Mapas Auto-Organizáveis

A rede neural artificial *Self-Organzing Maps* (SOM), também conhecida como Mapas Auto-Organizáveis foi proposta por Kohonen em 1982. Essa rede neural baseia-se no mapeamento em uma, duas ou três dimensões, onde características semelhantes apresentadas à entrada da rede são agrupadas na camada de saída. Trata-se de uma transformação não linear aplicada na visualização de dados multidimensionais para exibição em até três dimensões.

O Mapa Auto-Organizável de Kohonen é um modelo de rede neural com treinamento não supervisionado, inspirado no modelo de divisão de funções do córtex cerebral (Figura 8). O qual prevê ativação de áreas distintas do cérebro para diferentes estímulos (KOHONEN, 1997). Os mapas auto-organizáveis têm sido aplicados em problemas de diversas áreas. Tais como soluções de séries temporais, compressão e extração de características em imagens, sistemas de controle, bioinformática, reconhecimento de fala, meteorologia e oceanografia entre várias outras aplicações em ciências e engenharia (MWASIAGI, 2011).

Figura 8 – Modelo de Divisão do Córtex Cerebral



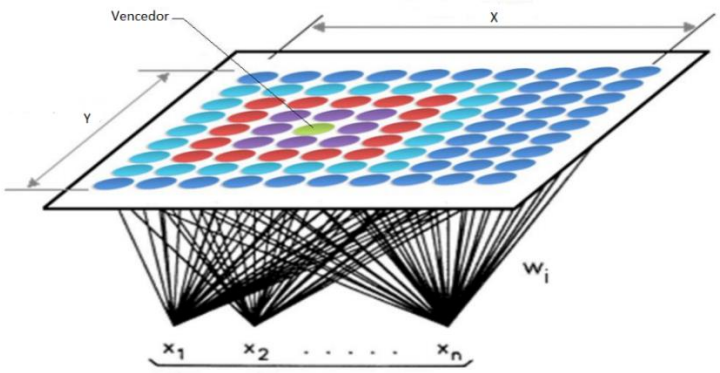
Fonte: Bugayong (2017)

Matematicamente o mapa auto-organizável é definido como mapeamento do espaço de entrada $\mathcal{R}^n$ para $\mathcal{R}^2$. Onde a cada neurônio i na camada de saída é associado um vetor de pesos sinápticos $m_i \in \mathcal{R}^n$ (Equação 2.11). E o vetor de entrada $x \in \mathcal{R}^n$ (Equação 2.12) é conectado a todos os neurônios da rede no instante t, como mostrado na Figura 9.

$$m_i=[\mu_{i1}, \mu_{i2}, \ldots, \mu_{in}]^T \in R^n \quad (2.11)$$

$$x=[\xi_1, \xi_2, \ldots, \xi_n]^T \in R^n \quad (2.12)$$

Figura 9 - Vetor de entrada x conectado ao Mapa Auto-Organizável por meio do vetor de pesos w_i



Fonte: adaptado de Sepúlveda (2012)

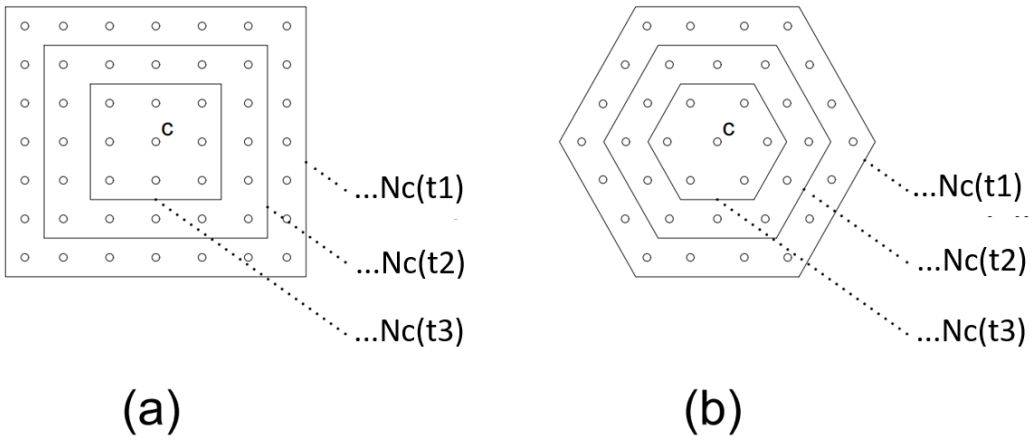
A transformação do espaço de entrada para a saída ocorre em três processos durante o treinamento: competição, cooperação e adaptação. No processo de **competição** o algoritmo busca o neurônio que melhor corresponde a uma entrada x (neurônio vencedor). Tipicamente a melhor correspondência é definida pela menor distância euclidiana dada pela Equação 2.13. No processo de **cooperação**, o neurônio vencedor estimula seus vizinhos de acordo com a função de **vizinhança** definida a priori. Por fim, os neurônios ativados têm seus pesos atualizados no processo de **adaptação** sináptica (ROSA e VALLE, 2013).

$$c = \text{argmin}\{||x - m_i||\} \quad (2.13)$$

Os neurônios vizinhos são geralmente ajustados em torno do vencedor seguindo padrão retangular ou hexagonal, como é mostrado na Figura 10. Dentro do raio de vizinhança os neurônios são excitados segundo a Equação 2.14. Usualmente, a taxa de aprendizagem η (t) e o raio da vizinhança $N_c(t)$ decrescem monotonamente com o tempo (KOHONEN, 1997).

$$h_{ci}(t) = \begin{cases} \eta(t), & i \in N_c \\ 0, & i \notin N_c \end{cases}, \quad t = 0, 1, 2, 3, \ldots, t_f \qquad (2.14)$$

Figura 10 - Exemplos de topologia de vizinhança



Fonte: Kohonen (1997)

Outra forma mais suave de ajustar os pesos dos neurônios vizinhos é a partir da gaussiana descrita pela Equação 2.15.

$$h_{ci}(t) = \exp\left(-\frac{\left|\left| r_c - r_i \right|\right|^2}{2\sigma^2(t)} \right), \quad t = 0, 1, 2, 3, \ldots, t_f \qquad (2.15)$$

Onde r_c e $r_i \in \mathcal{R}^2$ são respectivamente vetores de localização do neurônio c e i, na saída da rede. O parâmetro σ(t) define a largura da vizinhança, correspondendo ao raio $N_c(t)$.

Na etapa de adaptação sináptica, o vetor de pesos de cada neurônio m_i é atualizado segundo a Equação 2.16. Os valores iniciais dos pesos podem ser arbitrários, definidos de forma aleatória (KOHONEN, 1997).

$$m_i(t+1) = m_i(t) + \eta(t)h_{ci}(t)[x(t) - m_i(t)], \quad t=0, 1, 2, 3, \ldots, t_f \tag{2.16}$$

Deve-se ter alguns cuidados quanto a taxa de aprendizagem η(t) e a largura de vizinhança σ(t), definidos pelas equações 2.17 e 2.18, respectivamente. A primeira é responsável pela estabilidade, a segunda é fundamental para ordenamento e convergência da rede. Ambas decrescem monotonamente com o tempo.

$$\eta(t) = \eta_0 \left(\frac{\eta_f}{\eta_0}\right)^{t/t_f}, \quad t = 0, 1, 2, 3, \ldots, t_f \tag{2.17}$$

$$\sigma(t) = \sigma_0 \left(\frac{\sigma_f}{\sigma_0}\right)^{t/t_f}, \quad t = 0, 1, 2, 3, \ldots, t_f \tag{2.18}$$

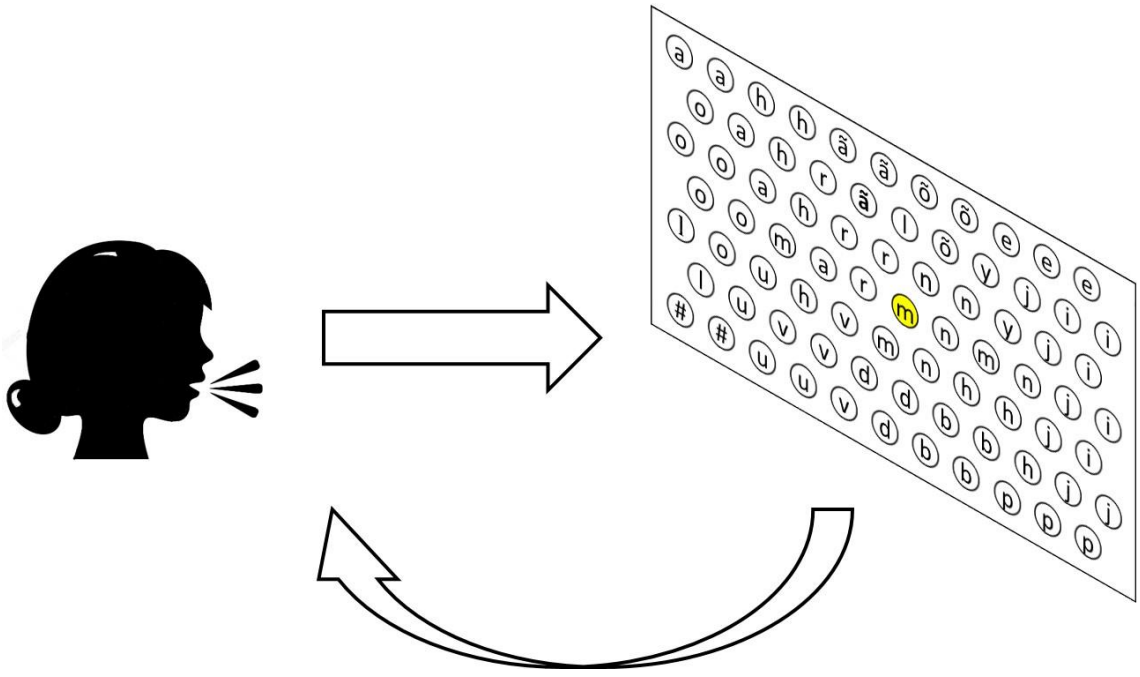
Algumas variações do algoritmo SOM foram desenvolvidas e são apresentadas no livro *Self-Organizing Maps* (KOHONEN, 1997). Várias aplicações são descritas no livro *Self Organizing Maps Applications and Novel Algorithm Design* (MWASIAGI, 2011), inclusive em ASR. O Mapa Auto-Organizável se mostrou amplamente eficiente em aplicações de agrupamento e classificação em diversas áreas desde sua formalização em 1982.

3 SISTEMA PROPOSTO

No sentido de resgatar as informações essenciais, é importante lembrar que milhões de pessoas sofrem de distúrbios da fala. Muitos dos quais podem se beneficiar de recursos computacionais no auxílio de tratamento e treinamento fonoaudiológico. O presente trabalho tem por objetivo, propor um sistema que auxilie o treinamento fonoaudiológico por meio de feedback visual, usando o Mapa Auto-Organizáveis de Kohonen.

Propõe-se um sistema de reconhecimento de fala, a partir dos fones presente no Português Brasileiro. Como resultado do treinamento da rede SOM, espera-se obter um mapa fonético topológico que guarde relações de proximidade dos fones na saída da rede. Desta forma, o usuário do sistema poderá se “localizar” e se corrigir a medida que se aproxima da pronúncia alvo (Figura 11).

Figura 11 – Treinamento Fonoaudiológico com Feedback Visual



Fonte: autor

A partir do *feedback* visual, os usuários podem ajustar a intensidade e a precisão da pronúncia. Além de acompanhar a evolução do treinamento por meio de

pontuação dos acertos e da precisão. O sistema pode ajudar deficientes auditivos a adquirir, manter ou aprimorar a fala.

3.1 Diretrizes para implementação em software

Para dar corpo a proposta é necessário a definição de uma diretriz que englobe aspectos técnicos e conceituais relativos à estrutura e ao desenvolvimento do sistema. Entretanto, algumas definições técnicas não estão neste documento, ficando a cargo do desenvolvedor a decisão de como implementá-las. As seguintes diretrizes são propostas como guia para o desenvolvimento:

1. Na apresentação do sistema (interfaces gráficas), terá uma área para treinamento do mapa auto-organizável e outra área para exercícios fonoaudiológicos;
2. Na interface gráfica (tela) destinada a configuração do mapa deve conter os campos para os seguintes parâmetros: dimensões do mapa (x, y), taxa de aprendizagem (η_0, η_f), largura da vizinhança (σ_0, σ_f) e número máximo de iterações (t_f);
3. Na interface gráfica destinada a configuração dos parâmetros MFCC deve conter os seguintes campos: tamanho das janelas (em segundos), tamanho do passo entre janelas sucessivas (em segundos), número de coeficientes cepstrais por janela, número de filtros no banco de filtros, tamanho da Transformada Rápida de Fourier e valor do coeficiente no filtro de pré-ênfase;
4. Todos os campos devem ter texto explicativo de valores típicos e intervalos aceitáveis;
5. Deve-se ter uma tela destinada à evolução do treinamento do mapa com informações de porcentagem e tempo transcorrido, assim com estimativa para conclusão do treinamento da rede;
6. Deve-se ter campos para importar arquivos (definidos a diante) necessários ao treinamento, rotulação e teste do mapa;
7. Deve-se ter uma tela destinada ao relatório de precisão do mapa (geral e por fones);

8. Na interface gráfica destinado ao exercício fonoaudiológico deve existir opção de carregar mapa fonético, praticar exercícios e exibir relatório de desempenho;
9. Na tela de exercícios deve ter o mapa fonético topológico, a pronúncia alvo e a realizada pelo usuário. Além de mostrar a intensidade da fala do usuário (baixa, média e alta);
10. O sistema necessita de “arquivo anotado” (<nome>.TextGrid) do software *Praat*, o qual permite definir intervalos com rótulos no áudio.
11. O sistema deve ter codificação de símbolos (UTF-16) que permita a representação do alfabeto fonético;
12. É usado a extensão WAV para os arquivos de áudio;
13. O sistema deve garantir que os parâmetros do arquivo de áudio, da rede e os parâmetros MFCC usados no treinamento do mapa sejam os mesmos na etapa de uso no exercício fonoaudiológico.

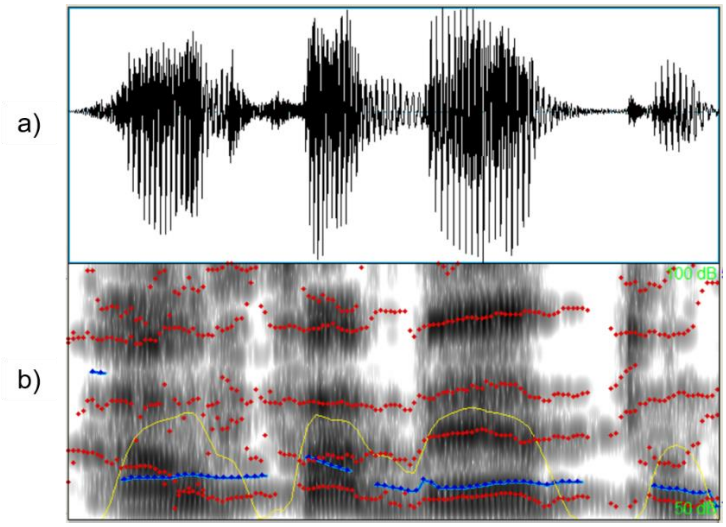
3.2 Uso do Software *Praat*

Software livre e gratuito desenvolvido por Paul Boersma e David Weenink no Instituto de Ciências Fonéticas da Universidade de Amsterdam, em 1992. O programa oferece recursos técnicos para manipulação, análise, produção e reprodução de ondas sonoras (FONSECA, 2009).

O programa sofre atualizações frequentes e possui uma comunidade ativa que contribui com sua evolução. Além disso, ele oferece recursos de análises espectrais (espectrogramas), curvas de *pitch*[2], formantes[3], intensidade[4], instabilidade, entre outras (Figura 13). Por essas e muitas outras qualidades, o *Praat* é um dos softwares de análise acústica mais usados por pesquisadores em todo o mundo.

[2] Sensação de altura do som associado a frequência fundamental.
[3] Ressonâncias intensificadas através de tubo acústico. Na fala, o tubo são as cavidades oral e nasal.
[4] Propriedade associada a amplitude da vibração sonora.

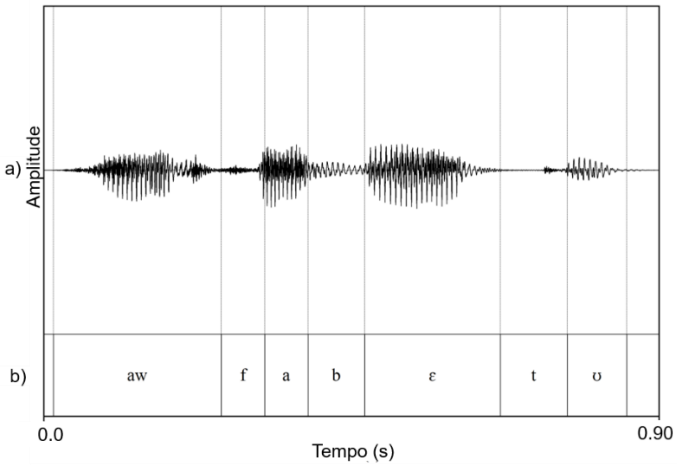
Figura 12 – Análises geradas pelo programa *Praat* a) forma de onda do sinal b) curva de intensidade em amarelo, formantes em vermelho, curvas de pitch em azul e espectrograma em escalas de cinza



Fonte: autor

Como mencionado na diretriz de número 10 na secção anterior, a implementação da proposta deste trabalho em software deve se beneficiar dos recursos do *Praat*, especificamente a parte que permite segmentação e anotações ao longo do sinal, relacionando o trecho de áudio à grafia que o representa (Figura 13). Assim, pode-se usar as marcações para extrair os intervalos relevantes para treinamento, rotulação e validação do mapa. Dessa forma, o arquivo de anotação (exemplo no apêndice A) pode ser consumido de forma automática pelo sistema aqui proposto.

Figura 13 – Áudio anotado no software *Praat para elocução* “alfabeto"



Fonte: autor

4 EXPERIMENTOS

Os passos básicos para implementar o sistema proposto passa pela formação de uma **base de dados** (áudio) para treinamento, obtenção de **descritores de características** (MFCC), **treinamento** do mapa auto-organizável e finalmente a **rotulagem e validação**. Em seguida, a rede pode ser usada para reconhecimento. Neste experimento, o mapa auto-organizável foi testado em condições de laboratório, isso significa que as entradas de teste foram gravadas e tratadas previamente.

4.1 Base de dados

Para o conjunto de treinamento foram gravados uma sequência de palavras contendo as consoantes oclusivas do Português Brasileiro[5]. Foi usado o programa *Praat* na captura de áudio, remoção de ruídos e para seleção dos trechos onde ocorrem os fones. Para cada fone foram usadas seis ocorrências a fim de cobrir a variabilidade na produção da fala.

Para rotulagem do mapa, foram usados os fones pronunciados entre duas vogais, por exemplo: ‘aba’. As amostras de áudio foram captadas e tratadas pelo *Praat* e são provenientes de único locutor (autor). Essas restrições evidenciam os objetivos do experimento: descrever e demonstrar os processos envolvidos no reconhecimento e visualização da fala, criar um mapa fonético topológico do Português Brasileiro e verificar a efetividade dos Coeficientes Cepstrais de Frequência em Escala Mel na distinção dos fones.

O sinal da fala foi captado pelo microfone interno de notebook da marca DELL, modelo *Inspiron* 14 7000. Foi usado o programa *Praat* para gravação e remoção de ruídos, instalado no Sistema Operacional Windows 10. Os arquivos foram salvos no formato WAV, canal mono, taxa de amostragem 44,1 kHz e formato de amostragem 16-bits.

Como visto na seção 2.1, os fones são classificados de acordo com a articulação. As articulações podem ser: oclusiva, nasal, fricativa, africada, *tepe*, vibrante, retroflexa, aproximante e lateral. Os fones são detalhados no Quadro 3 onde

[5] Modo de articulação ao qual estão presentes os fones [p], [b], [t], [d], [k] e [g].

é mostrado sua representação gráfica, classificação e a duração de algumas amostras usadas no experimento.

Quadro 3 – Fones consonantais classificados de acordo com modo de articulação

FONEMA	MODO DE ARTICULAÇÃO	Duração (ms)
p	Oclusiva bilabial desvozeada	20,9
b	Oclusiva bilabial vozeada	21
t	Oclusiva alveolar desvozeada	22
d	Oclusiva alveolar vozeada	21
k	Oclusiva velar desvozeada	21
g	Oclusiva velar vozeada	26

Fonte: autor

4.2 Extração dos Coeficientes Cepstrais de Frequência em Escala Mel

A extração de características (MFCC) depende de alguns parâmetros que variam de acordo com o problema. Para obter o vetor de coeficientes cepstrais é usado neste experimento a biblioteca *python speech features* (LYONS, 2013), a qual requer os parâmetros listados a seguir:

- *signal* – sinal de áudio. *Array* unidimensional;
- *samplerate* – taxa de amostragem do sinal;
- *winlen* – tamanho da janela em segundos;
- *winstep* – passo entre as janelas sucessivas em segundos;
- *numcep* – número de *cepstrum* a retornar. Isto é, número de coeficientes por janela;
- *nfilt* – Número de filtros no banco de filtros (padrão 26);
- *nfft* – tamanho da FFT (padrão 512);
- *lowfreq* – borda de banda mais baixa dos filtros mel, em Hz (padrão 0);
- *highfreq* – borda de banda mais alta dos filtros mel (padrão *samplerate*/2);
- *preemph* – constante do filtro usado na pré-ênfase (padrão 0,97);
- *ceplifter* – eleva os coeficientes cepstrais finais (padrão 22);
- *appendEnergy* – se marcado como *True*, substitui o coeficiente *cepstral* de índice zero pelo log da energia total.

O coeficiente de índice zero por vezes é descartado em aplicações de reconhecimento de fala por conter muitas informações do meio de transmissão. No caso de treinamento do mapa com voz de único locutor, c(0) pode aproximar os padrões de entradas no espaço. Isto é, ao considerar o primeiro coeficiente MFCC, os vetores são representados mais próximos no espaço de entrada, necessitando mais iterações para o mapa convergir. Foram considerados os dois casos neste experimento.

O tamanho da janela (*winlen)* foi definido em 20ms devido a duração dos fones, apresentados no Quadro 3. O passo entre as janelas (*winstep)* ficou em 10ms, gerando uma sobreposição de metade da janela. Foram usados 13 coeficientes cepstrais por janela (*numcep*). O valor do coeficiente de pré-ênfase foi usado em 0,95 e 0,97 em cenários distintos. O tamanho da FFT (nfft) foi obtido multiplicando o tamanho da janela pela taxa de amostragem (*winlen* e *samplerate*) onde obtém-se 882. Os demais parâmetros foram mantidos em valor padrão da biblioteca. Os valores definidos no experimento são próximos dos praticados tipicamente em reconhecimento de voz.

Considerando a duração dos fones, o tamanho da janela e o passo entre janelas sucessivas, cada fone terá no mínimo dois vetores de coeficientes MFCC, os quais devem estar próximos no espaço $\mathcal{R}^n$. Desta forma, cada fone terá mais de um vetor de 13 coeficientes normalizados, segundo a Equação 4.1, onde cada componente pertence ao intervalo [-1, 1].

$$\mathbf{v} = \frac{\mathbf{V}}{||\mathbf{V}||}, \quad \mathbf{V} = (V_1, V_2, V_3, \ldots, V_n) \tag{4.1}$$

Os coeficientes cepstrais de frequência representam uma posição no espaço multidimensional de um fone específico. Portanto, dois fones distintos devem guardar distância relativa. Desta forma, quanto mais próximos os vetores estão, maior deve ser os cuidados na escolha dos parâmetros, pois isso impacta a convergência do mapa auto-organizável. O Quadro 4 mostra a distância relativa entre os vetores (normalizados) de coeficientes de alguns fones.

Quadro 4 – Distância relativa entre vetores de coeficientes Cepstrais de Frequência em Escala Mel

	p	b	t	d	k	g
p	0.0					
b	1.249	0.0				
t	0.811	0.997	0.0			
d	0.967	0.913	0.466	0.0		
k	0.942	1.495	0.838	0.926	0.0	
g	1.074	1.223	0.818	1.063	0.818	0.0

Fonte: autor

A distância relativa entre os vetores de coeficientes cepstrais usados na rotulagem revelam condição mínima para considerar que um fone seja reconhecido pelo mapa. Além da similaridade entre o neurônio e o padrão de entrada, deve-se garantir que a distância entre o vetor de pesos e a entrada seja inferior à menor distância relativa entre os padrões de entrada usados na rotulagem.

4.3 Algoritmo – Mapa Auto-Organizável

Existem algumas variações do algoritmo proposto por Kohonen em 1982. Principalmente na definição dos vizinhos e como estes participam do ajuste de pesos. Neste experimento, a vizinhança é ajustada segundo a Equação *2.15*. O Algoritmo 1 descreve as etapas de treinamento do mapa auto-organizável[6].

Ao implementar mapas auto-organizáveis deve ser levado em consideração os argumentos que podem ser manipulados em busca de melhores resultados. O número de neurônios no mapa deve ser superior ao número de elementos a serem representados. Entretanto, um mapa com neurônios em excesso pode exigir um procedimento de poda[7] e aumentar o tempo de treinamento significativamente. Além disso, a taxa de aprendizagem (η_0 e η_f), a largura da vizinhança (σ_0 e σ_f), são parâmetros que interferem na estabilidade e convergência do mapa.

[6] Neste experimento o algoritmo foi implementado em Python.
[7] Procedimento que remove neurônios da rede que não são ativados por entradas válidas.

Algoritmo 1 – Mapas Auto-Organizáveis de Kohonen

Inicialização:
  Parâmetros:
    Dimensão (x, y)
    η_0, η_f
    σ_0, σ_f
  Pesos – valores aleatórios e pequenos
Defina t_f, número máximo de iterações para t = 0, 1, 2, 3, …, t_f
Enquanto: mapa apresentar alterações topológicas ou t≤ t_f **faça**
  Apresentar um padrão de entrada aleatório x(t) ao mapa
  Definir o neurônio vencedor por meio da Equação 2.13
  Atualizar os pesos sinápticos dos neurônios por meio da Equação 2.16
  Incrementar t

Fonte: Adaptado de Rosa e Valle (2013)

A implementação do Algoritmo 1 está disponível no apêndice A com codificação na linguagem Python. O arquivo <kohonen.py> possui duas classes *Neuron* e *Network*, onde está presente o método de treinamento. O mapa de saída está representado em uma lista de neurônios, os quais guardam o vetor de pesos e a posição (x, y) no *grid*.

4.4 Treinamento e Rotulagem

Considerando a quantidade de parâmetros tanto do mapa auto-organizável quanto da extração de atributos, fez-se necessário combinar algumas variações desses valores dentro do espectro tipicamente praticado para aplicações semelhantes. Entretanto, foram fixados os parâmetros na extração de características (seção 4.2), alternando apenas coeficiente de pré-ênfase. Neste sentido, foram criados alguns cenários para averiguar quais produzem melhores resultados.

Os cenários foram definidos considerando os coeficientes cepstrais com a presença e ausência do primeiro coeficiente e para os valores de pré-ênfase de 0,95 e 0,97. O mapa teve suas dimensões fixadas em 8 e 12, variando os valores de taxa de aprendizagem e largura da vizinhança como mostrado no Quadro 5. Também foi definido como condição de parada 16 mil iterações, ou até que cada neurônio

vencedor tenha distância entre seu vetor de pesos e a entrada inferior a um décimo da menor distância relativa entre vetores de entrada (Quadro 4) por 100 iterações consecutivas, visando garantir menor taxa de erro na etapa de reconhecimento.

Quadro 5 – Cenários de treinamento da rede SOM

Argumentos	C1	C2	C3	C4	C5	C6	C7	C8
Coeficiente c_0	sim	sim	sim	sim	não	não	não	não
α	0,95	0,95	0,97	0,97	0,95	0,95	0,97	0,97
σ_0	8	3	8	3	8	3	8	3
σ_f	0,73	0,01	0,73	0,01	0,73	0,01	0,73	0,01
η_0	0,9	0,7	0,9	0,7	0,9	0,7	0,9	0,7
η_f	0,11	0,01	0,11	0,01	0,11	0,01	0,11	0,01

Fonte: autor

Após o mapa treinado, foi realizada rotulagem usando áudio anotado do programa *Praat*, onde os fones foram identificados manualmente. A partir do arquivo TextGrid, obtiveram-se os intervalos de áudio referente a cada fone. As amostras aplicadas na rotulagem foram diferentes das usadas no treinamento do mapa auto-organizável.

4.5 Resultados e Discussões

Na realização dos experimentos, em todos os cenários foram usados os mesmos conjuntos de dados no treinamento. Isto é, 10 ocorrências de cada um dos seis fones oclusivos ([p], [b], [t], [d], [k] e [g]) encontrados no idioma Português Brasileiro. E os mesmos trechos de áudios foram usados na rotulagem de cada um dos oito mapas resultantes.

Para testar os oito cenários descritos, foram usadas 40 ocorrências de cada fone no contexto de palavras presentes no idioma. A acurácia dos mapas foi medida considerando o erro e o equívoco para cada ocorrência no conjunto de testes. O **erro** foi definido como percentual de amostras identificadas como um padrão de entrada, quando pertencem a outro. **Equívoco** é o percentual de amostras rejeitadas indevidamente, indicando que o padrão apresentado não pertence ao mapa.

Nos quatro primeiros cenários os experimentos foram realizados considerando a presença do primeiro coeficiente cepstral c_0. Alternando a constante do filtro de pré-ênfase em 0,95 e 0,97. Esses cenários foram construídos com dois valores para os parâmetros de vizinhança (σ_0 e σ_f) e taxa de aprendizagem (η_0 e η_f) como mostrado nos quadros Quadro 6 e Quadro 7. Estes cenários verificam que a escolha da constante do filtro de pré-ênfase interfere pouco no percentual de erros e equívocos do mapa.

Quadro 6 –cenários de treinamento do mapa com o primeiro coeficiente cepstral

Parâmetros	Cenário 1		Cenário 2		Cenário 3		Cenário 4	
α	0,95				0,97			
σ_0	8		3		8		3	
σ_f	0,73		0,01		0,73		0,01	
η_0	0,9		0,7		0,9		0,7	
η_f	0,11		0,01		0,11		0,01	
Fones	Erro	Equívoco	Erro	Equívoco	Erro	Equívoco	Erro	Equívoco
p	45,0%	42,5%	45,0%	42,5%	45,0%	45,0%	45,0%	42,5%
b	45,0%	42,5%	45,0%	45,0%	45,0%	42,5%	42,5%	40,0%
t	42,5%	40,0%	40,0%	37,5%	42,5%	40,0%	37,5%	40,0%
d	37,5%	37,5%	37,5%	40,0%	40,0%	42,5%	37,5%	37,5%
k	47,5%	47,5%	42,5%	45,0%	47,5%	45,0%	42,5%	37,5%
g	45,0%	47,5%	42,5%	45,0%	47,5%	47,5%	45,0%	42,5%
Total	43,75%	42,92%	42,08%	42,50%	44,58%	43,75%	41,67%	40,00%

Fonte: autor

O coeficiente cepstral c_0 foi removido nos quatro últimos cenários. Enquanto os demais parâmetros sofreram as mesmas variações presentes nos primeiros cenários. Nestas condições, foi observado evolução mais significativa na mudança da constante do filtro de pré-ênfase de 0,95 (cenário 5) para 0,97 (cenário 7). Os quais mantem parâmetros de vizinhança (σ_0, σ_f) em 8 e 0,73 e taxa de aprendizagem (η_0, η_f) em 0,9 e 0,11, respectivamente.

Quadro 7 – cenários de treinamento do mapa sem o primeiro coeficiente cepstral

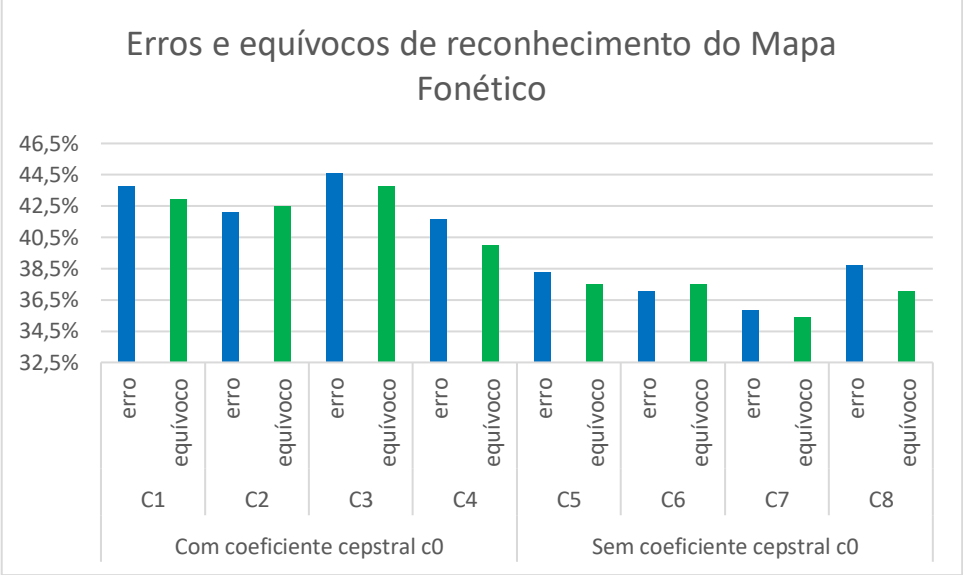
Parâmetros	Cenário 5		Cenário 6		Cenário 7		Cenário 8	
α	0,95				0,97			
σ_0	8		3		8		3	
σ_f	0,73		0,01		0,73		0,01	
η_0	0,9		0,7		0,9		0,7	
η_f	0,11		0,01		0,11		0,01	
Fones	Erro	Equívoco	Erro	Equívoco	Erro	Equívoco	Erro	Equívoco
p	40,0%	37,5%	37,5%	40,0%	40,0%	37,5%	40,0%	37,5%
b	42,0%	37,5%	40,0%	37,5%	37,5%	37,5%	37,5%	40,0%
t	37,5%	40,0%	37,5%	35,0%	35,0%	35,0%	40,0%	35,0%
d	35,0%	37,5%	35,0%	32,5%	35,0%	32,5%	37,5%	35,0%
k	37,5%	35,0%	35,0%	40,0%	32,5%	35,0%	37,5%	40,0%
g	37,5%	37,5%	37,5%	40,0%	35,0%	35,0%	40,0%	35,0%
Total	38,25%	37,50%	37,08%	37,50%	35,83%	35,42%	38,75%	37,08%

Fonte: autor

A retirada do primeiro coeficiente cepstral influenciou positivamente a capacidade de reconhecimento do mapa. Observou-se menor percentual de erros e equívocos totais nas combinações envolvendo coeficiente de pré-ênfase (α), parâmetros de vizinhança (σ_0, σ_f) e taxas de aprendizagem (η_0, η_f). Portanto, para as restrições apresentadas neste trabalho, o coeficiente cepstral c0 prejudica o reconhecimento dos fones.

O mapa que apresentou o melhor desempenho foi o descrito pelo cenário sete, como é mostrado no gráfico da Figura 14. Onde foi descartado o primeiro coeficiente cepstral e definido o valor de 0,97 para a constante de pré-ênfase, foram escolhidos para largura de vizinhança (σ_0, σ_f) 8 e 0,73, respectivamente. As taxas de aprendizagem (η_0, η_f) ficou definido em 0,9 e 0,11. Neste cenário, houve 35,83% de amostras rejeitadas indevidamente pelo mapa e 35,42% aceitas equivocadamente para um dado padrão de entrada.

Figura 14 – comparação de erros e equívocos em oito cenários, C1, C2, …, C8



Fonte: autor

O número de equívocos e erros podem ser considerados altos se comparados com a bibliográfica consultada. Um dos fatores para esse resultado foi o uso de número reduzido de amostras dos fones. Outra explicação deve-se ao fato deste trabalho tratar o reconhecimento independente de vocabulário, esbarrando na variabilidade que cada fone apresenta a depender do contexto[8] ao qual ele está inserido.

É importante ressaltar a complexidade envolvida no reconhecimento automático da fala por tratar de sinal não estacionário. Mesmo dois sinais da mesma palavra produzidos por único locutor podem ser distintos significativamente, no caso do reconhecimento de fones individuais a variabilidade do sinal é ainda maior. Principalmente as consoantes sofrem influência dos fones que as precedem e sucedem, criando um conjunto grande de possibilidades de representação. Nestes experimentos foram observadas essas complexidades tanto na identificação dos fones no programa Praat quanto na rotulagem do mapa.

[8] Principalmente os fones consonantais sofrem influência das vogais que os precedem e os sucedem

5 CONCLUSÃO

Este trabalhou apresenta uma proposta de uso do Mapa Auto-Organizável de Kohonen como recurso em treinamentos fonoaudiológicos por meio de *feedback* visual da fala. Como descritores de características da voz, foram usados Coeficientes Cepstrais de Frequência em Escala Mel (MFCC). Foram realizadas pesquisas bibliográficas para confirmar a viabilidade da proposta, além da realização de experimentos com escopo reduzido[9] contendo oito cenários de configuração de parâmetros MFCC e do mapa auto-organizável.

Diante das pesquisas bibliográficas e dos experimentos realizados neste trabalho, foi possível constatar a complexidade e os desafios encontrados na implementação de reconhecimento automático de fala. Observaram-se dificuldades em estimar com precisão razoável os limites de cada fone no sinal sonoro, principalmente devido à sobreposição dos sinais de fones adjacentes nas palavras.

Para verificar a acurácia do mapa, foram adotadas medidas de erro e equívoco. No primeiro caso, definiu-se como erro o percentual de amostras reconhecidas como um fone, enquanto representavam outro padrão. O equívoco foi definido como fones rejeitados pelo mapa, tratados como padrão de entrada inválido. Pôde-se verificar a plausibilidade da proposta, entretanto observou-se a necessidade de conjunto maior de amostras de áudio para treinamento do mapa.

Embora o método de extração de características MFCC tenha se mostrado satisfatório para diferenciar os fones pronunciados por único locutor em ambiente controlado, sugere-se para trabalhos futuros a investigação de outros descritores mais robustos a ruídos e múltiplos locutores como o PNCC (do inglês, *Power Normalized, Cepstral Coefficients*), para implementação do sistema proposto.

[9] Foram usados apenas os fones cujo modo de articulações são classificados como oclusivos: [t], [d], [p], [b], [k] e [g]

REFERÊNCIAS

BRESOLIN, A. D. A. **Reconhecimento de voz através de unidades menores do que a palavra , utilizando Wavelet Packet e SVM, em uma nova Estrutura Hierárquica de Decisão**. Natal: Universidade Federal do Rio Grande do Norte - Centro de Tecnologia, 2008.

BUGAYONG, K. Brain Lobes. **HOPES Huntington's Outreach Project For Education, At Stanford**, 2017. Disponivel em: <https://hopes.stanford.edu/the-hopes-brain-tutorial-text-version/brain-lobes/>. Acesso em: 15 maio 2019.

CERÓN, I. F. C.; BADILLO, A. G. G. A Keyword Based Interactive Speech Recognition System for Embedded Applications Master's Thesis, Västerås, junho 2011.

CUADROS, C. D. R. **Reconhecimento de Voz e de Locutor em Ambientes Ruidosos:** Comparação das Técnicas MFCC e ZCPA. Niterói: [s.n.], 2007.

DAVE, M. Feature Extraction Methods LPC, PLP and MFCC In Speech Recognition. **INTERNATIONAL JOURNAL FOR ADVANCE RESEARCH IN ENGINEERING AND TECHNOLOGY**, v. 1, n. VI, p. 1-5, july 2013.

DEAFNESS. **World Health Organization - WHO**, 15 março 2018. Disponivel em: <http://www.who.int/features/factfiles/deafness/en/>. Acesso em: 13 março 2019.

FONSECA, A. A. Análise do Tutorial do Programa de Análises Acústicas Praat. **Texto Livre: Linguagem e Tecnologia**, Belo Horizonte, 2, junho 2009. 13-16. Disponivel em: <http://www.periodicos.letras.ufmg.br/index.php/textolivre/article/view/23/7313>.

GORDILLO, C. D. A. Reconhecimento de Voz Contínua Combinando os Atributos MFCC e PNCC com Métodos de Robustez SS, WD, MAP e FRN, Rio de Janeiro, março 2013. Disponivel em: <http://www.maxwell.vrac.puc-rio.br/Busca_etds.php?strSecao=resultado&nrSeq=23090@1>.

GOULART, B. N. G. D.; OLIVEIRA, N. D.; CHIARI, B.. Distúrbios de Linguagem Associados à Surdez. **Rev. bras. crescimento desenvolv. hum.**, v. 23, p. 41-45, 2013. Disponivel em:

<http://pepsic.bvsalud.org/scielo.php?script=sci_arttext&pid=S0104-12822013000100006&lng=pt&nrm=iso>. Acesso em: 13 março 2019.

IBGE. **Censo Demográfico**. [S.l.]. 2010.

INTERNATIONAL PHONETIC ASSOCIATION. **Handbook of the International Phonetic Association:** A Guide to the Use of the International Phonetic Alphabet. Cambridge: Cambridge University Press, 1999. Disponivel em: <https://www.internationalphoneticassociation.org/>. Acesso em: 05 abril 2019.

KOHONEN, T. Self-organized formation of topologically correct feature maps. **Biological Cybernetics**, Berlim, v. 43, n. 1, p. 59-60, janeiro 1982. ISSN 0340-1200.

KOHONEN, T. The "Neural" Phonetic Typewriter. **IEEE**, março 1988. 11 - 22. Disponivel em: <https://ieeexplore-ieee-org.ez49.periodicos.capes.gov.br/document/28/authors#authors>. Acesso em: 26 fevereiro 2019.

KOHONEN, T. **Self-Organizing Maps**. 2. ed. Berlin: Springer, 1997.

LADEFOGET, P. **The acoustic analysis of speech.** 2. ed. Chicago: The University of Chicago Press, 1996.

LYONS, J. **Python Speech Features**, 2013. Disponivel em: <https://python-speech-features.readthedocs.io/en/latest/#welcome-to-python-speech-features-s-documentation>. Acesso em: 20 maio 2019.

MATUCK, G. R. **PROCESSAMENTO DE SINAIS DE VOZ PADRÕES COMPORTAMENTAIS POR REDES NEURAIS ARTIFICIAIS**. São José dos Campos: INPE, 2005.

MUSSALIM, F.; BENTES,. **Introdução à Linguística - Domínios e Fronteiras**. 2. ed. São Paulo: Cortez, v. 1, 2001.

MWASIAGI, J. I. (Ed.). **Self Organizing Maps - Applications and Novel Algorithm Design**. [S.l.]: IntechOpen, 2011.

OLIVEIRA, D. D. H. Fonética e Fonologia, João Pessoa, 2009. Disponivel em: <http://biblioteca.virtual.ufpb.br/publicacoes/view/278>. Acesso em: 22 março 2019.

ROSA, R. A. F.; VALLE, M. E. Um Estudo da Aplicação de Redes Neurais Auto-Orgazáveis para a Identificação Autônoma de Fonemas Portugueses. **RECEN - Revista Ciências Exatas e Naturais**, p. 199 - 2018, 2013.

SÃO THIAGO, E. R. D. Reconhecimento de Voz Utilizando Extração de Coeficientes Mel-Cepstrais e Redes Neurais Artificiais, São José, dezembro 2017.

SEARA, I. C.; NUNES, V. ; LAZZAROTTO-VOLCÃO, C. **Fonética e fonologia do português brasileiro**. Florianópolis: LLV/CCE/UFSC, 2011.

SEPÚLVEDA, H. H. Representación Topológica de las Características de la Señal Digitalizada de la Música para la Generación Automática de Listas de Reproducción, Valparaíso, junho 2012.

SILVA , ; YEHIA,. Quadro Fonético. **Sonoridade em Artes, Saúde e Tecnologia**, 2008. Disponivel em: <http://www.fonologia.org/quadro_fonetico.php>. Acesso em: 26 janeiro 2019.

SILVA , T. C.; YEHIA , H. C. Quadro fonêmico do Português. **Sonoridade em Artes, Saúde e Tecnologia**, 2008. Disponivel em: <http://www.fonologia.org/quadro_fonemico.php>. Acesso em: 28 fevereiro 2019.

APÊNDICE A – CÓDIGO FONTE

Neste apêndice encontram-se os códigos usados nos experimentos, exceto os executados no ambiente “jupyter-notebook”. Também consta o conteúdo de um dos arquivos <.TextGrid> usados no experimento.

Arquivo tools.py

```
#!/usr/bin/env python
# -*- coding: UTF-8 -*-
import re
import scipy.io.wavfile as wav
import numpy as np


def intervals_labels(path_textgrid, intervals_name, samplerate):
    arq = open(file=path_textgrid, mode='r', buffering=-1, encoding="utf-
16")
    line = arq.readline()
    tab = re.compile("\s{4,4}")
    token =
re.compile("(\w+\s*\[\d*\])|(name\s=\s\"\w*\")|(x\w{3,3}\s=\s\d{1,}\.{0,}\d
{0,})|(text\s=\s\".*\")")
    # item=re.compile("item\s*\[\d+\]")
    num = re.compile('\d+\.*\d*')
    x_min = re.compile('xmin')
    x_max = re.compile('xmax')
    text = re.compile('text')
    rot = re.compile('\".*\"')
    r = re.compile('name\s=\s\"\\w*\"')
    pilha = []
    nome_item = ""
    itens = {}

    ocorrencias = []
    intervals = {}
    while line:
        prox_linha = arq.readline()
        topo = len(pilha)-1

        if len(token.findall(line)) > 0 and topo < 1:
            pilha.append(len(tab.findall(line)))
            pilha.append(token.search(line).group(0))
            len(pilha)-1
        elif len(token.findall(line)) > 0 and len(tab.findall(line)) <
(pilha[topo-1]):
            if pilha[topo-1] - len(tab.findall(line)) > 1 or
len(prox_linha)==0:
                itens[nome_item] = ocorrencias
                ocorrencias = []
                intervals = {}
            rotulo = ""
            vetor = [-1, -1]

            while pilha[topo-1] > len(tab.findall(line)):
```

```
                tok = pilha.pop()
                pilha.pop()
                topo = len(pilha)-1
                if len(x_min.findall(tok)) > 0:
                    vetor[0] = float(num.search(tok).group(0))
                elif len(x_max.findall(tok)) > 0:
                    vetor[1] = float(num.search(tok).group(0))
                elif len(text.findall(tok)) > 0:
                    rotulo = rot.search(tok).group()[
                        1:len(rot.search(tok).group(0))-1]
                if (vetor[0] > -1 and vetor[1] > -1) and len(rotulo) > 0:
                    ocorrencias.append([rotulo, vetor])
                    intervals[rotulo] = vetor
                    vetor = [-1, -1]
                    rotulo = ""

            pilha.pop()
            pilha.pop()
            len(pilha)-1
            pilha.append(len(tab.findall(line)))
            pilha.append(token.search(line).group(0))
            len(pilha)-1

        elif len(token.findall(line)) > 0 and len(tab.findall(line)) >=
int(pilha[topo-1]):
            if len(prox_linha) == 0 :

                rotulo = line.split("=")[1].split('"')[1]
                ocorrencias.append([rotulo, [float(num.search(pilha[topo-
2]).group(0)), float(num.search(pilha[topo]).

group(0))]])
                itens[nome_item] = ocorrencias
                ocorrencias = []
                intervals = {}

            pilha.append(len(tab.findall(line)))
            pilha.append(token.search(line).group(0))
            len(pilha)-1
            if len(r.findall(line)) > 0:
                nome_item = r.search(line).group(0).split('"')[1]
        line = prox_linha
    list_label_intervals = []

    for l in itens[intervals_name]:
        ro = l[0]
        v1 = int(l[1][0]*samplerate)
        v2 = int(l[1][1]*samplerate)
        list_label_intervals.append([ro, [v1, v2]])

    return list_label_intervals

def intervals_sounds(path_arq_wav, intervals_labels):
    (rate, sig) = wav.read(path_arq_wav)
    sounds_intervals =[]
    for intervals in intervals_labels:
        sounds_intervals.append([intervals[0],
sig[intervals[1][0]:intervals[1][1]]])
    return (sounds_intervals, rate)
```

```
def normalize(ndarray):
    if isinstance(ndarray, list):
        ndarray = np.array(ndarray)
    if isinstance(ndarray, np.ndarray):
        if len(ndarray.shape) == 1:
            return ndarray/np.linalg.norm(ndarray)
        if len(ndarray.shape) == 2:
            aux = []
            for arr in ndarray:
                aux.append(arr/np.linalg.norm(arr))
            return np.array(aux)
    else:
        return "Error"
```

Arquivo Kohonen.py

```
import numpy as np
from scipy.spatial import distance


class Neuron:
    def __init__(self,  x_position_neuron, y_position_neuron,
index_in_gridList, length_weights, range_for_weights):
        self.x_position = x_position_neuron
        self.y_position = y_position_neuron
        self.index_in_gridList = index_in_gridList

        weights = []
        for i in range(length_weights):
            weights.append(np.random.uniform(range_for_weights[0],
range_for_weights[1]))
        self.weights = np.array(weights)

    def update_weights(self, input_vector, winner_neuron,  sigma,
learning_rate):
        raio = 3
        x_inf = winner_neuron.x_position - raio
        x_sup = winner_neuron.x_position + raio
        y_inf = winner_neuron.y_position - raio
        y_sup = winner_neuron.y_position + raio
        if x_inf < self.x_position and self.x_position < x_sup and  y_inf <
self.y_position and self.y_position < y_sup:
            dist = distance.euclidean([winner_neuron.x_position,
winner_neuron.y_position], [self.x_position, self.y_position])
            neighborhood = np.exp(-dist**2/(2*sigma**2))
            self.weights = self.weights +
learning_rate*neighborhood*(input_vector - self.weights)

class Network:
    def __init__(self, num_lines, num_columns, length_weights,
range_for_weights=[0, 1]):
        self.num_lines = num_lines
        self.num_columns = num_columns
        self.gridList = []
        self.sigma = None
```

```
        self.sigma_start = None
        self.learning_rate = None
        self.learning_rate_start = None
        self.epochs = None
        self.sigma_final = None
        self.learning_rate_final = None
        counter = 0
        for i in range(num_lines):
            for j in range(num_columns):
                self.gridList.append(Neuron(x_position_neuron=i,
y_position_neuron=j, index_in_gridList=counter,
                                            length_weights=length_weights,
range_for_weights=range_for_weights))
                counter += 1

    def update_sigma(self, t):
        self.sigma =
self.sigma_start*(self.sigma_final/self.sigma_start)**(t/self.epochs)

    def update_learning_rate(self, t):
        self.learning_rate =
self.learning_rate_start*(self.learning_rate_final/self.learning_rate_start
)**(t/self.epochs)

    def winner(self, input_vector):
        neuron_winner = self.gridList[0]
        shorter_distance = np.finfo(np.float).max
        for neuron in self.gridList:
            if distance.euclidean(input_vector, neuron.weights) <
shorter_distance:
                neuron_winner = neuron
                shorter_distance = distance.euclidean(input_vector,
neuron.weights)
        return neuron_winner

    def training(self, data_training, epochs, sigma_start,
learning_rate_start, sigma_final, learning_rate_final):
        self.epochs = epochs
        self.sigma = sigma_start
        self.sigma_start = sigma_start
        self.sigma_final = sigma_final
        self.learning_rate = learning_rate_start
        self.learning_rate_final = learning_rate_final
        self.learning_rate_start = learning_rate_start

        for t in range(epochs):
            if t % 100 == 0:
                print("\rTraining SOM...
"+str(int(t*100.0/self.epochs))+"%", sep=' ', end='')

            self.update_sigma(t)
            if t > epochs*0.1:
                self.update_learning_rate(t)

            input_vector = data_training[np.random.randint(0,
len(data_training))].reshape(np.array([data_training.shape[1]]))
            neuron_winner = self.winner(input_vector)
            for neuron in self.gridList:
                neuron.update_weights(input_vector=input_vector,
winner_neuron=neuron_winner,  sigma=self.sigma,
                                      learning_rate=self.learning_rate)
```

```
        print("\rTraining SOM... done!")
```

oclusivas1.TextGrid

```
File type = "ooTextFile"
Object class = "TextGrid"

xmin = 0
xmax = 18.972857142857144
tiers? <exists>
size = 1
item []:
    item [1]:
        class = "IntervalTier"
        name = "fones"
        xmin = 0
        xmax = 18.972857142857144
        intervals: size = 38
        intervals [1]:
            xmin = 0
            xmax = 0.4686560009110979
            text = ""
        intervals [2]:
            xmin = 0.4686560009110979
            xmax = 0.49104116650193874
            text = "p"
        intervals [3]:
            xmin = 0.49104116650193874
            xmax = 1.6857528327373454
            text = ""
        intervals [4]:
            xmin = 1.6857528327373454
            xmax = 1.7136649913587505
            text = "p"
        intervals [5]:
            xmin = 1.7136649913587505
            xmax = 2.783580474660781
            text = ""
        intervals [6]:
            xmin = 2.783580474660781
            xmax = 2.810693342355876
            text = "p"
        intervals [7]:
            xmin = 2.810693342355876
            xmax = 3.778591921410084
            text = ""
        intervals [8]:
            xmin = 3.778591921410084
            xmax = 3.8143255171402033
            text = "t"
        intervals [9]:
            xmin = 3.8143255171402033
            xmax = 4.887300450103481
            text = ""
        intervals [10]:
            xmin = 4.887300450103481
            xmax = 4.9077645249367485
            text = "t"
        intervals [11]:
            xmin = 4.9077645249367485
```

```
        xmax = 5.977733873522616
        text = ""
    intervals [12]:
        xmin = 5.977733873522616
        xmax = 6.021978676029613
        text = "t"
    intervals [13]:
        xmin = 6.021978676029613
        xmax = 7.06836436924429
        text = ""
    intervals [14]:
        xmin = 7.06836436924429
        xmax = 7.092432334251664
        text = ""
    intervals [15]:
        xmin = 7.092432334251664
        xmax = 7.121033246895737
        text = "k"
    intervals [16]:
        xmin = 7.121033246895737
        xmax = 8.061576688989922
        text = ""
    intervals [17]:
        xmin = 8.061576688989922
        xmax = 8.111659482592495
        text = "k"
    intervals [18]:
        xmin = 8.111659482592495
        xmax = 8.996166342364226
        text = ""
    intervals [19]:
        xmin = 8.996166342364226
        xmax = 9.08526105895983
        text = "k"
    intervals [20]:
        xmin = 9.08526105895983
        xmax = 9.986188643499768
        text = ""
    intervals [21]:
        xmin = 9.986188643499768
        xmax = 10.021849553499232
        text = "b"
    intervals [22]:
        xmin = 10.021849553499232
        xmax = 11.119714485536646
        text = ""
    intervals [23]:
        xmin = 11.119714485536646
        xmax = 11.150029260199265
        text = "b"
    intervals [24]:
        xmin = 11.150029260199265
        xmax = 12.026435871788511
        text = ""
    intervals [25]:
        xmin = 12.026435871788511
        xmax = 12.064117325237328
        text = "b"
    intervals [26]:
        xmin = 12.064117325237328
        xmax = 13.119998908402374
```

```
        text = ""
    intervals [27]:
        xmin = 13.119998908402374
        xmax = 13.166202482413558
        text = "d"
    intervals [28]:
        xmin = 13.166202482413558
        xmax = 14.24607098791767
        text = ""
    intervals [29]:
        xmin = 14.24607098791767
        xmax = 14.311603278411004
        text = "d"
    intervals [30]:
        xmin = 14.311603278411004
        xmax = 15.347142151993074
        text = ""
    intervals [31]:
        xmin = 15.347142151993074
        xmax = 15.388695955027096
        text = "d"
    intervals [32]:
        xmin = 15.388695955027096
        xmax = 16.1764889363161
        text = ""
    intervals [33]:
        xmin = 16.1764889363161
        xmax = 16.231833076790974
        text = "g"
    intervals [34]:
        xmin = 16.231833076790974
        xmax = 17.38586671886642
        text = ""
    intervals [35]:
        xmin = 17.38586671886642
        xmax = 17.429529940275387
        text = "g"
    intervals [36]:
        xmin = 17.429529940275387
        xmax = 18.406070937916716
        text = ""
    intervals [37]:
        xmin = 18.406070937916716
        xmax = 18.44918129601552
        text = "g"
    intervals [38]:
        xmin = 18.44918129601552
        xmax = 18.972857142857144
        text = ""
```