Lucas Meireles Tomaz de Alvarenga

Transcrição, alinhamento e criação de legendas em uma base de História Oral

Rio de Janeiro, Brasil

2019

Lucas Meireles Tomaz de Alvarenga

Transcrição, alinhamento e criação de legendas em uma base de História Oral

Dissertação de mestrado apresentada à Escola de Matemática Aplicada como requerimento parcial para obter o título de Mestre em Modelagem Matemática

Fundação Getulio Vargas – FGV

Escola de Matemática Aplicada

Programa de Pós-Graduação

Orientador: Dr. Moacyr Alvim Horta Barbosa da Silva

Rio de Janeiro, Brasil

2019

Dados Internacionais de Catalogação na Publicação (CIP)
Ficha catalográfica elaborada pelo Sistema de Bibliotecas/FGV

Alvarenga, Lucas Meireles Tomaz de
Transcrição, alinhamento e criação de legendas em uma base de história oral / Lucas Meireles Tomaz de Alvarenga. – 2019.
57 f.

Dissertação (mestrado) -Fundação Getulio Vargas, Escola de Matemática Aplicada.
Orientador: Moacyr Alvim Horta Barbosa da Silva.
Inclui bibliografia.

1. Reconhecimento automático da voz. 2. Modelagem de dados. 3. História oral. I. Silva, Moacyr Alvim Horta Barbosa da. II. Fundação Getulio Vargas. Escola de Matemática Aplicada. III. Título.

CDD – 006.12

Elaborada por Márcia Nunes Bacha – CRB-7/4403

FGV

LUCAS MEIRELES TOMAZ DE ALVARENGA

"TRANSCRIÇÃO, ALINHAMENTO E CRIAÇÃO DE LEGENDAS EM UMA BASE DE HISTÓRIA ORAL".

Dissertação apresentado(a) ao Curso de Mestrado em Modelagem Matemática do(a) Escola de Matemática Aplicada para obtenção do grau de Mestre(a) em Modelagem Matemática.

Data da defesa: 20/05/2019

ASSINATURA DOS MEMBROS DA BANCA EXAMINADORA

Moacyr Alvim Horta Barbosa da Silva
Orientador(a)

Paulo Cezar Pinto Carvalho
Membro Interno

Vivian Luiz Fonseca
Membro Externo

Alexandre Evsukoff
Membro Externo

Resumo

Dado a evolução da tecnologia relacionada a reconhecimento automático de fala nos é permitido criar ferramentas que automatizem processos que somente eram possíveis serem feitos por humanos, se o objetivo fosse ter uma boa qualidade no resultado. Sendo assim, o objetivo desse trabalho é pesquisar a literatura em reconhecimento automático de fala e utilizar esse conhecimento para criar ferramentas que auxiliem o Centro de Pesquisa e Documentação de História Contemporânea do Brasil (CPDOC) com o programa de História Oral. Além disso, esperamos que esse trabalho possa auxiliar outros centros que trabalhem com história oral, como por exemplo: Associação Internacional de História Oral (IOHA), Rede Latino-Americana de História Oral (RELAHO) e Associação brasileira de História Oral (ABHO). Após um estudo sobre as principais técnicas na área de reconhecimento automático de áudio, foi decidido utilizar um software proprietário para a transcrição automática da fala das entrevistas que populam o banco de dados de História Oral. Com a transcrição automática, utilizamos uma variação do algoritmo de Needleman-Wunsch para alinharmos essa transcrição com a transcrição manual das entrevistas, essas transcrições foram feitas por pessoas e não possuem o tempo das frases. Esse alinhamento possibilita a criação de legendas com um texto mais correto do que seria possível somente com a transcrição automática. Concluímos que transcrições automáticas podem ser utilizadas no contexto de História Oral, que é possível alinhar textos utilizando o algoritmo de Needleman-Wunsch e dado esse alinhamento é possível criar uma legenda, que a perda de sincronia entre o áudio e o texto não impossibilita o uso da mesma.

Palavras-chaves: Needleman-Wunsch. História Oral. *Deep Learning*. Alinhamento. *Recurrent Neural Network*. Áudio para texto.

Abstract

Given the evolution of technology related to automatic speech recognition we are able to create tools that automate processes that could only be done by humans, if we wanted to have a good quality in the result. Therefore, the objective of this work is to research the literature about automatic speech recognition and use this knowledge to create tools that support the Center for Research and Documentation of Contemporary History of Brazil (CPDOC) with the Oral History project. In addition, we hope that this work can help other centers that work with oral history, such as: International Oral History Association (IOHA), Latin American Oral History Network (RELAHO) and Brazilian Association of Oral History (ABHO). After the study on the main techniques in the area of Automatic Speech Recognition, It was decided to use proprietary software to automatic transcribe the interviews that populate the Oral History database. With the automatic transcription, we used a variation of the Needleman-Wunsch algorithm to align this transcription with a manual transcription, these transcriptions were made by human and do not have the time of the sentences. This enable the creation of subtitles with a more correct text than would be possible with automatic transcription alone. We conclude that automatic transcriptions can be used in the context of Oral History, that it is possible to align texts using the Needleman-Wunsch algorithm and given this alignment It is possible to create a subtitle, that the loss of synchrony between the audio and the text does not preclude the use of It.

Key-words: Needleman-Wunsch. Deep Learning. Alignment. Recurrent Neural Network. Speech to text.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

CPDOC	Centro de Pesquisa e Documentação de História Contemporânea do Brasil
RNN	*Recurrent Neural Networks*
ASR	Reconhecimento automática de áudio
DNN	Deep Neural Networks
LAS	Listen, Attend and Spell
RNN-T	Recurrent Neural Networks Transducer
LSTM	Long Short-Term Memory
HMM	Hidden Markov Models
CNN	Convolutional Neural Networks
GMM	Gaussian Mixture Model
IOHA	Associação Internacional de História Oral
RELAHO	Rede Latino-Americana de História Oral
ABHO	Associação brasileira de História Oral

Sumário

Parte I

Introdução

1 Introdução

1.1 A importância da história oral

A história oral pode ser vista como uma metodologia de pesquisa que traz uma rica fonte de informação para a humanidade de maneira mais viva que um texto. São histórias que são contadas por indivíduos sobre seu passado, ou sobre o passado de outras pessoas. Em Nunn e Reid (2016), podemos ver um exemplo de história que passou de geração por geração através da fala, que está sendo utilizado como fato empírico para uma catástrofe que ocorreu à mais de 7mil anos atrás.

No Brasil, temos diversos centros que tem como objetivo guardar e divulgar o testemunho vivos de entrevistados seguindo essa metodologia, sendo o CPDOC o mais antigo. No inicio do programa de história oral do CPDOC o objetivo era estudar a trajetória das elites brasileiras desde os anos 1930. Essa vertente se mantém até hoje, mas o acervo foi enriquecido com entrevistas que pretendia compreender acontecimentos e conjuntura específicas da história do Brasil. Outro exemplo é o Laboratório de História Oral e Imagem Campus de Gragoatá que tem em seu acervo entrevistas com camponeses negros descendentes da última geração de africanos capturados na África, esses registros fazem parte dos projetos Memórias do Cativeiro e Jongos, Calangos e Folias: Música negra, memória e poesia. O Museu da Imagem e do Som, a Associação Internacional de História Oral, a Rede Latino-Americana de História Oral e a Associação brasileira de História Oral são outros lugares que podemos encontrar acervos ricos de história. Essa metodologia cresceu tanto que podemos encontrar projetos até dentro de empresas ou fundações, como por exemplo: Memória Globo, Memória Petrobras e acervo de história oral da Casa de Oswaldo Cruz.

Mesmo que o principal foco do nosso projeto seja com o acervo do CPDOC, esperamo que mais programas possam vir a se beneficiar do que está estamos desenvolvendo nesse trabalho.

1.2 Machine Learning em entrevistas da história oral

O tipo de acervo criado pela história oral traz mais complexidades que um texto, por ser gravado em áudio ou vídeo. Por exemplo, é mais fácil pesquisar uma palavra em um texto do que em um áudio, principalmente se estiver em algum meio digital. Outra vantagem do texto é que ele é mais universal do que um áudio, dentro de uma mesmo idioma. Alguns problemas que podemos encontrar, específicos de áudio para um mesmo

idioma são: Diferença em sotaques, diferença na voz de locutores, qualidade do áudio e sobreposição de falas.

Felizmente, com os avanços tecnológicos vem crescendo o número de pesquisa na área de áudio dentro de *Machine Learning*, trazendo automatizações e novas ferramentas para essa área. A melhora na qualidade de transcrição automática de fala para texto nos últimos anos é facilmente notada, basta só ver um vídeo no YouTube com legendas geradas automaticamente, ou a qualidade da tradução automática de textos em diferents idiomas. O que é uma demonstração que podemos usar essa mesma tecnologia para criar transcrições automáticas de um acervo de história oral.

Entretanto, a qualidade dos modelos de *Machine Learning* variam em relação ao idioma utilizado, mesmo assim, até no melhor idioma tem erro. Por esse motivo, estamos propondo um *framework* para a criação de legendas fazendo o uso da transcrição automática e da transcrição manual, essa transcrição é feita sem o tempo que cada palavra é dita. Sendo assim, é esperado que o processo de criação de legenda fiquei mais rápido e mais barato, por não ser necessário que a legenda seja criada de forma manual.

1.3 Contribuições

As contribuições desse trabalho são:

- A possibilidade de criar transcrições automáticas para o acervo de História Oral do Centro de Pesquisa e Documentação de História Contemporânea do Brasil (CPDOC).
- Utilizar o algoritmo Needleman-Wunsch para alinhar dois textos distintos. Na literatura esse algoritmo é mais comumente utilizado para o alinhamento de proteínas e nucleótidos.
- Um *framework* para a criação de legendas para vídeos, dado que já exista uma transcrição feita de forma manual e sem a marcação do tempo.

1.4 Organização dos capítulos

1.4.1 Revisão de Literatura

Dividimos o trabalho em tópicos referentes transcrição automática de áudio para texto, alinhamento e outros trabalhos com bases de dados de história oral. Sendo assim, no capítulo de revisão de literatura trazemos as pesquisas que achamos mais importantes para este trabalho. Falaremos um pouco sobre reconhecimento de áudio antes de entramos nos trabalhos sobre transcrição de áudio para texto e discutimos que modelo o Google deve

utilizar em seu sistema. Em relação ao alinhamento, estudamos em que tipos de trabalhos o algoritmo de Needleman-Wunsch foi utilizado e falamos de trabalhos que utilizaram de forma similar a nossa. Por fim, falaremos de outros trabalhos que foram feitos no âmbito de história oral.

1.4.2 Metodologia

Essa seção traz informações práticas sobre o uso do *Google Speech-to-text*, explica os conceitos de uma *Recurrent Neural Networks (RNN)* que é a base para o estado da arte em *Speech-to-text*, como funciona o algoritmo de Needleman-Wunsh, as modificações que fizemos nesse algoritmo e como fazemos a criação da legendas após o alinhamento entre a transcrição automática e a transcrição manual.

1.4.3 Resultados

Apresentamos como foi utilizado o serviço do *Google Speech-to-text*, os resultados da transcrição automática desse sistema e quais são as informações que ele retorna. Com essa informação fazemos o alinhamento entre a transcrição automática e a transcrição manual e mostramos o resultados desse alinhamento utilizando uma métrica que contabiliza o número de palavras iguais na mesma posição menos o número de palavras que não coincidem. Por fim, com o alinhamento é possível criar a legenda e trazemos exemplos dessas legendas após a sincronização com os seus respectivos áudios.

1.4.4 Discussão

Nesse capítulo levantamos ideias sobre o tema e o porque essas ideias foram utilizadas ou não. Alguns dos pontos levantados são: a escolha do Google, fonemas não terem sido utilizados para a etapa de alinhamento, testes com a distância de Levensthein para definir se as palavras são aproximadamente idêntica e formas de validar a sincronização entre o texto e o áudio.

1.4.5 Conclusão

Falamos sobre os objetivos que foram alcançados, nosso ponto de vista sobre os nossos resultados e afirmamos que conseguimos uma resposta promissora sobre as questões propostas por este trabalho.

Parte II

Revisão de Literatura

2 Revisão da Literatura

2.1 Reconhecimento Automático de Fala

Reconhecimento Automático de Fala (*ASR*) é uma área de pesquisa que consiste no uso de sistemas computacionais para processar a voz humana, podendo ser utilizado como uma ferramenta de interação entre humano e máquina (LÓPEZ; QUESADA; GUERRERO, 2017), transcrição de áudio para texto (GIBIANSKY et al., 2017), reconhecimento de locutor (CAMPBELL, 1997), alteração de voz (GAO; SINGH; RAJ, 2018), entre outras aplicações. Essa área está ativa a mais de seis décadas, o primeiro sistema de *ASR* que se tem documentação é o Audrey, feito em 1952 (PIERACCINI; DIRECTOR, 2012). Mas por muito tempo foi uma área com pouca expressão, a tecnologia da época não era boa o suficiente para conseguir resolver problemas em condições de uso real.

No entanto, hoje em dia, uma mistura de fatores estão fazendo com que cada vez mais tenhamos contato com a área de *ASR*. Em Yu e Deng (2015) são citados 3 principais fatores para essa mudanças estarem acontecendo recentemente, elas são:

- O poder computacional que temos disponível, isso nos permite utilizar modelos que tem uma demanda maior de computação;
- Por causa do avanço da internet temos acesso a muito mais dados do que antes;
- Estamos usando cada vez mais aparelhos portáteis como *smartphones*, *smartwatchs*, assistentes pessoais e etc...

Podemos ver o impacto desses fatores em Hinton et al. (2012), nesse trabalho são apresentados resultados que mostram que *Deep Neural Networks Hidden Markov Models* (*DNN-HMM*) tiveram uma performance melhor que *Gaussian Mixture Models Hidden Markov Models* (*GMM-HMM*) em diversas tarefas. Em Baker et al. (2009), podemos ver que *HMM* já era a principal técnica utilizada em *ASR*.

Um ponto importante a ser levanto é que pela particularidade de cada idioma e pela disponibilidade de banco de dados, a qualidade da transcrição automática varia. Como apontado em Neto et al. (2011) os recursos para o português brasileiro são limitados, o que faz com que seja esperado resultados abaixo de idiomas mais comuns, como o inglês. Mesmo que esse trabalho trate de levantar novos recursos é afirmado que para o português brasileiro não se tem tantos recursos quanto para alguns outros idiomas.

Como dito no inicio dessa seção, uma das formas que *ASR* pode ser aplicada é na transcrição de áudio para texto. Sendo esse um dos objetivos desses trabalho, nós fizemos uma pesquisa mais concentrada nesse âmbito.

2.1.1 Transcrição automática de áudio para texto

Os principais trabalhos na área de transcrição automática de áudio para texto estão fazendo o uso de *Deep Learning*. Um exemplo é em Amodei et al. (2016), esse trabalho mostra um modelo que transcreve curtos áudios em Mandarim melhor do um típico falante de Mandarim, em um experimento com 250 expressões, e melhor do que um comitê de 5 chineses, em um experimento com 100 expressões, ambos os experimentos foram feitos com expressões da base de teste escolhidas de forma aleatória. Foi utilizado uma arquitetura de *Recurrent Neural Networks* (*RNN*) para alcançar esse resultado.

Outro exemplo da importância dessa arquitetura é em Sak, Senior e Beaufays (2014), em que o modelo utilizado foi o *Long Short-Term Memory Recurrent Neural Networks* (LSTM RNN), onde chegaram em um erro por palavra de 10.7%. Esse trabalho foi o primeiro a utilizar o *LSTM RNN* em uma tarefa de reconhecimento de voz com um grande vocabulário.

Mesmo que arquiteturas de *RNN* sejam as mais comuns em *ASR*, elas são computacionalmente complexas e difíceis de serem treinadas. Em contrapartida a essa arquitetura temos o uso de *Convolutional Neural Networks (CNN)*, que é uma arquitetura computacionalmente mais eficiente que *RNN* (ZHANG et al., 2017). Outro fator importante sobre a *CNN*, é que ela também consegue modelar a correlação temporal tendo profundidade suficiente. Nesse artigo chegam a um erro de 18.2%, menor do que a arquitetura de *RNN* que foi utilizada para comparação. Os testes foram feitos no popular banco de dados TIMIT, mas como os outros modelos, discutidos nessa seção, não possuem a informação da taxa de acerto nesse mesmo banco, não temos como fazer uma comparação direta do resultado.

Recentemente, conseguimos ver avanços maiores em Chiu et al. (2018), onde foi alcançado um erro por palavra de 4.1% na tarefa de transcrição de áudio para texto. Foi utilizado o modelo *Listen, Attend and Spell (LAS)* para esse feito. Falamos um pouco mais deste modelo na próxima seção (2.1.2).

2.1.2 O modelo por trás do Google Speech-to-Text

Não foi possível encontrar explicitamente qual é o modelo utilizado atualmente pelo Google[1] para a tarefa de *Speech-to-text*. Embora, em 2015 eles tenham anunciado

[1] A escolha do Google como plataforma para fazer a transcrição automática é discutida na seção de Discussão

que pararam de utilizar *Gaussian Mixture Model (GMM)* e passaram a utilizar *LSTM RNN* em seu lugar[2].

Em Chiu et al. (2018), artigo produzido pela equipe do Google, apresenta um novo modelo estado-da-arte na área de reconhecimento de fala. Esse modelo é uma melhora do modelo *LAS* (Chan et al., 2016). O modelo foi patenteado pelo Google na área de reconhecimento de fala (CHAN et al., 2018), o que é um bom indicativo que pode ser o modelo utilizado atualmente. Mas uma das desvantagens desse modelo é que ele não funciona para transcrever fala em tempo real, um serviço que o Google também oferece. Vale ressaltar, que o *LAS* utiliza *RNNs* em sua arquitetura.

Por fim, temos um anuncio mais recente[3] utilizando *RNN Transducer (RNN-T)* (GRAVES, 2012). Nesse caso, trata-se de uma aplicação de reconhecimento de fala para rodar diretamente dentro de ambientes embarcados, sem precisar de acesso a recursos mais avançados, como podemos ver em He et al. (2019). Esse trabalho consegue concluir o que se propõe, construir um modelo que rode embarcado com uma acurácia tão boa quanto modelos que precisem de mais recursos computacionais, ou seja, o modelo que vem sendo utilizado podia ser uma versão desse modelo.

Todos esses modelos são arquiteturas de *RNN*, que é uma classe de redes neurais capaz de aprender relação temporal entre as variáveis. Esse trabalho não tem como objetivo explicar essa arquitetura, mas recomendamos a leitura do *Deep Learning Book* (GOODFELLOW; BENGIO; COURVILLE, 2016).

2.2 Needleman-Wusnch

Temos como objetivo utilizar uma transcrição automática e alinhar com um transcrição manual e escolhemos utilizar o algoritmo de Needleman-Wusnch (NEEDLEMAN; WUNSCH, 1970) para essa tarefa. escolhemos esse algoritmo por ser uma técnica de programação dinâmica, ou seja, é um problema de otimização que é resolvido salvando a pontuação ótima da solução de todos os subproblemas em vez de recalcular essa informação, o que nos permite controlar a o uso de memória do sistema. Além disso, é garantido que será encontrado um alinhamento ótimo dado uma função de pontuação em particular.

A demanda de utilizar algum algoritmo para alinhar a transcrição automática com a transcrição manual já é antiga, como podemos ver em Ellozy et al. (1997), mas não encontramos nenhum trabalho utilizando o algoritmo de Needleman-Wunsch. Em Picone et al. (1986), é apresentado o alinhamento entre dois textos utilizando programação dinâmica no domínio de fonemas, no nosso caso não iremos fazer essa transformação o

[2] "*The neural networks behind Google Voice transcription*", Google AI Blog, https://ai.googleblog.com/2015/08/the-neural-networks-behind-google-voice.html, (Abril, 2019)

[3] "An All-Neural On-Device Speech Recognizer", Google AI Blog, https://ai.googleblog.com/2019/03/an-all-neural-on-device-speech.html, (Abril, 2019)

motivo explicado está na seção de Discussão. Outra diferença para esse trabalho é que ele faz o alinhamento por letra e o nosso objetivo é fazer por palavra.

Dado a natureza do algoritmo de Needleman-Wunsch ter sido originalmente desenhado para alinhamento de aminoácidos e proteínas, esse algoritmo é muito comum na área de bioinformática e ainda existem poucos trabalhas que utilizam ele fora desse nicho. Em Dieny, Thevenon et al. (2011) o algoritmo de Needleman-Wunsch é utilizado como ponto de partida para a criação de um algoritmo de correspondência estéreo em imagens. Ou seja, é utilizado para encontrar correspondência densa entre imagens, para a reconstrução da imagem em 3D.

Eem Chua e Foo (2017), o algoritmo de Needleman-Wunsch é utilizado para alinhar árvores de decisão no domínio de seleção de sensores em *Smart Homes*. Eles afirmam que esse método teve melhor desempenho que dois métodos que utilizaram como caso base, tendo ainda uma melhor eficiência computacional.

2.3 Trabalhos com dados similares

A literatura sobre a história oral é extensa, mas a nossa revisão de literatura tem como base trabalhos que utilizaram um banco de História Oral e produziram na área de *ASR*, sendo assim, um nicho mais restrito dessa área.

Em Huijbregts, Ordelman e Jong (2005) é testado a tarefa de recuperação de informação em um acervo de História Oral. Ele utiliza dois modelos de *Broadcast new system*, são algoritmos já treinados, um funcionando com uma mistura de *RNN* e *HMM* e o outro utiliza *decision-tree state-clustered HMM*. Depois utilizaram 22 horas transmissões de notícias em holandês, para ajustar o sistema independente de gênero para o Holandês. Conseguiram 30% de erro por palavra em teste com dados de transmissão de notícias, o benchmark era menor do que 20%. Mas quando foram testar o sistema em dados de história oral o erro foi para 80%. Ajustando o modelo para o domínio de história oral conseguiram melhorar o modelo para 66.9%, um erro inviável para conseguirem fazer a recuperação de informação de um acervo de História Oral. Acreditamos que esse aumento no erro aconteceu pela mudança no vocabulário e da fala ser menos controlado do que em um ambiente de transmissões de noticias.

Podemos ver resultados em diferentes áreas de *ASR* em Byrne et al. (2004), utilizando bases de História Oral em inglês e tcheco. Os tópicos estudados de *ASR* foram: *Named entity recognition*[4], *Document Segmentation*[5], *Segment Categorization*[6] e como es-

[4] Named entity recognition é a área que estuda a classificação de entidades em categorias pré-definidas, como por exemplo: cidade, pessoa, data, etc...

[5] Document Segmentation tem como objetivo dividir as entrevistas em pequenos pedaços, segmentados em tópicos coerentes

[6] Segment Categorization tem como objetivo segmentar as entrevistas em múltiplas categorias semân-

sas três técnicas podem ser usadas juntas para busca interativa. Eles reportam que os limites dos tópicos podem ser reconhecidos de maneira confiável em conversas espontâneas, não ficaram satisfeitos com a acurácia da categorização e que a detecção de entidades foi afetada pela quantidade de exemplos na base de treino.

Foi feito um estudo sobre transcrição automática da História Oral em Tcheco (IRCING et al., 2002). Esse estudo fez parte de um grande projeto com um acervo de 116 mil horas de entrevistas, o que totaliza mais de 180 terabytes de vídeos. O projeto começou coletando testemunhos de sobreviventes do Holocausto, o que mostra a sua importância para a história da humanidade. O foco desse foi a transcrição manual de entrevistas para que fosse possível começar a trabalhar com *ASR* e a criação de uma modelo baseline para transcrição automática. A melhor taxa de acerto para uma base que possuiu até 60mil vocabulários foi de 34%, enquanto o teste em uma base de transmissões de notícias com um vocabulário de mesmo tamanho chega a um percentual de acerto de 70%.

Em Gref, Köhler e Leh (2018), temos um artigo mais recente que trata de melhorar a transcrição automática e a indexação em uma base de História Oral. O *Framework* que eles criaram, para ser o o modelo base, tinha as seguintes etapas:

- Segmentar os áudios utilizando um algoritmo próprio para isso;
- Detectar se o segmento possui alguma fala;
- Identificar o gênero do locutor;
- Agrupar os áudios do mesmo locutor;
- Utilizar um híbrido de Hidden Markov Model-Deep Neural Network (HMM-DNN)[7];
- Extrair palavras chaves utilizando tf-idf.

Eles conseguiram melhorar o resultado fazendo o uso de *LSTM RNN*, no lugar do *HMM-DNN*. Conseguindo chegar em um erro por palavra de 39.4% o que foi uma melhora de 28.3% do modelo base. Vale ressaltar que esse trabalho foi feito para o idioma Alemão.

ticas

[7] Eles utilizam o modelo pronto da Kaldi, que é um projeto que reúne diversas ferramentas para a área de reconhecimento de fala.

Parte III

Metodologia

3 Metodologia

3.1 Banco de dados

O banco de dados utilizado é composto por entrevistas do acervo de história oral do CPDOC, grande parte dessas entrevistas possuem transcrições manuais, somente uma parte dessas entrevistas tem legendas. Os entrevistados são personagens importantes da história, que podem testemunhar sobre acontecimentos, conjunturas, instituições, modos de vida ou outros aspectos da história contemporânea [1]. O programa de história oral foi criado em 1975, possui mais de 1000 entrevistas e totaliza mais de 7 mil horas de áudio. As entrevistas são feitas após o levantamento de dados e criação de um roteiro.

Em relação a qualidade do áudio das entrevistas, temos pouca influência de ruídos por serem feitas em ambiente controlado. Mesmo assim, como existem entrevistas desde 1975, a qualidade da gravação varia em relação a época. Com a evolução da tecnologia, houve uma melhora da qualidade de gravação dos áudios. Temos também o fator do interlocutor, existe uma gama grande de diferentes pessoas entrevistadas, a dicção dos entrevistados vária consideravelmente, o que também pode causar um impacto na construção da transcrição. A quantidade de pessoas nas entrevistas também varia, embora sejam grupos pequenos.

Algumas entrevistas são feitas em outros idiomas, como o inglês e o espanhol. A metodologia utilizada neste trabalho é invariante em relação ao idioma utilizado na entrevista, contanto que a mesma não varie dentro da mesma gravação. Quando for realizada a transcrição automática é possível especificar o idioma da entrevista para que seja feita a transcrição de maneira correta.

Outro fator importante sobre o banco de dados são as transcrições manuais. Por questões de regras predefinidas para as transcrições, elas não são uma cópia perfeita do que é dito na entrevistas. Alguns exemplos de regras que causam esse impacto são: O cabeçalho da entrevista é desconsiderado para a transcrição; Quando há repetição de palavras seguidamente, só uma palavra é transcrita no lugar; Pode ser que algo falado seja inaudível.

As entrevistas foram usadas para a etapa de transcrição automática e as transcrições manuais foram usadas para a etapa de alinhamento.

[1] Definição retirada do site https://cpdoc.fgv.br/acervo/historiaoral, acessado em 30 de março de 2019.

3.2 Google Speech-to-text

Pela falta de recursos e na esperança de poder fazer a melhor transcrição automática possível, optamos por utilizar serviços especializados. Logo, utilizaremos o serviço *Speech-to-text* do Google para a transcrição das entrevistas.

Nesse serviço áudios maiores do que um minuto precisam estar armazenados na nuvem para serem transcritos, isso se dá por limitações do serviço oferecido. Por esse motivo, o primeiro passo é fazer o upload do áudio de interesse para a nuvem, após o upload é possível fazer a análise do áudio utilizando o serviço de *Speech-to-text* do Google e com a análise pronta nós podemos salvar a transcrição automática. Para não consumir espaço no ambiente da nuvem nós deletamos o áudio[2] após salvarmos a transcrição automática. A transcrição do Google é feita em blocos, ou seja, ele agrupa as frases e retorna as seguintes informação para cada grupo:

- Um percentual que transmite o quanto ele está confiante da transcrição naquele bloco, quanto mais próximo de 1 melhor;
- A frase que compõe o bloco;
- Cada palavra que compõe esse bloco tem a marcação de tempo estimada que aparece na faixa de áudio.

Nós salvamos essas informações em um arquivo *txt*, a duração desse processo varia em relação ao tamanho da entrevista, taxa de uploud da rede e o processamento do Google. Na seção 4.1 serão apresentadas durações reais dessa etapa.

O serviço do *Speech-to-text* recomenda algumas boas práticas para a melhora de qualidade na transcrição. Uma dessas recomendações é utilizar o formato *flac* nos áudios. O nosso banco de dados está em formato *wav*, portanto, antes do upload nós fazemos a transformação de *wav* para *flac*, o que acaba ajudando na redução do tamanho do arquivo e agilizando o processo.

Um dos problemas do serviço é que retorna menos palavras do que a transcrição manual, até o presente momento, já foi visto uma discrepância de até 32.5% na diferença do número de palavras. Como o número de palavras é menor existe o acúmulo de tempo em algumas palavras, dado que isso acontece porque ele mistura mais de uma palavra em uma palavra só, ou até em um número menor de palavras do que o correto. Na seção 4.1 será possível ver um exemplo real, esse problema é tratado durante a criação da legenda.

[2] O serviço da Google cobra um preço pelo uso de armazenamento continuo, por isso fazemos essa etapa de deletar os arquivo que estão na nuvem.

Nós utilizamos a linguagem de programação Python para automatizar esse processo de conversão, upload, transcrição e deleção da informação que fica armazenada na nuvem após o upload[3].

3.3 Algoritmo Needleman-Wunsch

Nós utilizamos o algoritmo Needleman-Wunsch para alinhar a transcrição automática e a transcrição manual. Esse algoritmo é muito utilizado na área de ciências biológicas para o alinhamento de proteínas e nucleótidos, como discutido na revisão de literatura. Sendo assim, tivemos que fazer a nossa própria implementação do algoritmo, as implementações que foram testadas só faziam o alinhamento por letras e não por palavras. Podemos dividir o funcionamento da nossa implementação nos seguintes passos:

1. Criar uma matriz S de tamanho $S_{N+1,M+1}$, sendo N o número de palavras de um texto e M o número de palavras do outro texto (Como por exemplo, uma transcrição manual e uma transcrição automática). A primeira posição, tanto da linha quanto da coluna, será referente a um espaço em branco. Na tabela 1, podemos ver um exemplo de como seria essa matriz para o alinhamento da palavra 'Eu quero comer sorvete' e 'Eu gostaria de comer sorvete'.

		Eu	Gostaria	de	comer	sorvete
Eu						
quero						
comer						
sorvete						

Tabela 1: Exemplo da matriz criada para o algoritmo de Needleman-Wunsch. Nas linhas temos a frase 'Eu quero comer sorvete' e nas colunas 'Eu gostaria de comer sorvete'.

2. Escolher o sistema de pontuação para cada par de palavra. No algoritmo original, temos que escolher um valor quando houver correspondência entre duas palavras (C), quando não houver correspondência (NC) e um valor para a palavra ser alinhada com uma lacuna. Na nossa implementação, podemos escolher dois valores diferentes para a criação de lacunas. Uma referente a alinhar a palavra que está na linha com uma lacuna e a outra a alinhar a palavra que está na coluna com uma lacuna, essas

[3] O código utilizado para execução desse processo pode ser encontrado em https://github.com/lucasmeirelesta/cpdoc-text-align/blob/master/speech-to-text.py, acessado em 30 de abril de 2019.

pontuações vão se chamadas de lacuna na maior frase (LMaiorF) e lacuna na menor frase (LMenorF), respectivamente[4].

3. Preencher a primeira posição da matriz com 0, ou seja, $S_{1,1} = 0$. Depois, começamos a preencher a primeira linha, cada valor será preenchido com a soma entre informação anterior e a pontuação LMenorF, ou seja, $S_{1,2} = S_{1,1} + LMenorF$ e assim por diante. O mesmo vale para a primeira coluna, só que cada valor será a soma entre o valor que está acima e a pontuação LMaiorF, ou seja, $S_{2,1} = S_{1,1} + LMaiorF$ e assim por diante. Na tabela 2 podemos ver esse passo para o seguinte esquema de pontuação escolhido: $C = 1$, $NC = -1$, $LMenorF = -1$ e $LMaiorF = -2$.

		Eu	Gostaria	de	comer	sorvete
	0	-1	-2	-3	-4	-5
Eu	-2					
quero	-4					
comer	-6					
sorvete	-8					

Tabela 2: Exemplo do primeiro passo de pontuação do algoritmo de Needleman-wunsch, utilizando o seguinte esquema de pontuação: $C = 1$, $NC = -1$, $LMenorF = -1$ e $LMaiorF = -2$. Nas linhas temos a frase 'Eu quero comer sorvete' e nas colunas 'Eu gostaria de comer sorvete'.

4. Calcular a pontuação de cada par de palavras de forma ordenada, começando pelo valor na posição $S_{2,2}$. O valor de $S_{2,2}$ será o maior valor entre as seguintes informações: A soma entre diagonal superior e a pontuação de correspondência; A soma entre o valor à esquerda e a pontuação LMenorF; A soma entre o valor que está acima e a pontuação LMaiorF. Utilizando a matriz da tabela 2, temos o seguinte exemplo:

$$\begin{aligned} S_{1,1} + C &= 0 + 1 = 1; \\ S_{2,1} + LMenorF &= -2 - 1 = -3; \\ S_{1,2} + LMaiorF &= -1 - 2 = -3; \\ S_{2,2} &= max(1, -3, -3) = 1. \end{aligned}$$

   Que resulta na matriz da tabela 3.

5. Com a informação de $S_{2,2}$, agora podemos preencher $S_{2,3}$ e $S_{3,2}$. Para preencher essas novas informações, fazemos o mesmo que o passo anterior. Podemos generalizar com a seguinte fórmula quando o par de palavra coincidem:

$$S_{i,j} = max(S_{i-1,j-1} + C, S_{i,j-1} + LMenorF, S_{i-1,j} + LMaiorF),$$

[4] Na nossa implementação deixamos sempre a menor frase na linha e a maior frase na coluna, a escolha foi arbitraria e foi feita só para organizar o código.

		Eu	Gostaria	de	comer	sorvete
	0	-1	-2	-3	-4	-5
Eu	-2	1				
quero	-4					
comer	-6					
sorvete	-8					

Tabela 3: Exemplo do resultado do cálculo para o valor $S_{2,2}$, utilizando o seguinte esquema de pontuação: $C = 1$, $NC = -1$, $LMenorF = -1$ e $LMaiorF = -2$. Nas linhas temos a frase 'Eu quero comer sorvete' e nas colunas 'Eu gostaria de comer sorvete'.

caso o par de palavras não corresponda usamos o valor de NC no lugar de C. Podemos ver na tabela 4, todos os seus valores calculados.

		Eu	Gostaria	de	comer	sorvete
	0	-1	-2	-3	-4	-5
Eu	-2	1	0	-1	-2	-3
quero	-4	-1	0	-1	-2	-3
comer	-6	-3	-2	-1	0	-1
sorvete	-8	-5	-4	-3	-2	1

Tabela 4: Exemplo da matriz de pontuação com todos os seus valors calculados, utilizando o seguinte esquema de pontuação: $C = 1$, $NC = -1$, $LMenorF = -1$ e $LMaiorF = -2$. Nas linhas temos a frase 'Eu quero comer sorvete' e nas colunas 'Eu gostaria de comer sorvete'.

6. Escolher o caminho que maximize a pontuação partindo da última célula $S_{N+1,M+1}$ até a primeira célula $S_{1,1}$. Sendo assim, começamos em $S_{N+1,M+1}$ e procuramos a célula com o maior valor entre a diagonal anterior, a célula de cima e a célula da esquerda. O caminho percorrido irá depender de qual das células tem o maior valor, o alinhamento será feito da seguinte forma dependendo do caminho:

   - Caso a célula da diagonal superior seja o maior valor, as palavras correspondentes dessa diagonal serão alinhadas uma com a outra;
   - Caso a célula da esquerda seja o maior valor, iremos alinhar a palavra da maior frase com uma lacuna na menor frase;
   - Caso a célula acima seja o maior valor, iremos alinhar a palavra da menor frase com uma lacuna na maior frase;
   - Em caso de empate, damos preferência ao caminho da diagonal superior e depois a criação de lacuna na menor frase.

   A escolha dessa ordem de preferência em caso de empate, se dá pela natureza de nosso problema. Não temos informação do tempo quando criamos lacunas nas frases, então é melhor que duas palavras sejam alinhadas entre si do que com uma lacuna.

Em relação a preferir criar uma lacuna na menor frase, é porque, se criarmos uma lacuna na maior frase, em algum momento vamos ter que igualar o tamanho de ambas as frases, ou seja, teríamos que criar mais lacunas na menor frase. Na tabela 5, temos que a seta azul demonstra o caminho escolhido pelo nosso algoritmo e a seta vermelha seria uma segunda possibilidade, caso não tivéssemos a ordem de importância.

		Eu	Gostaria	de	comer	sorvete
	0	-1	-2	-3	-4	-5
Eu	-2	1	0	-1	-2	-3
quero	-4	-1	0	-1	-2	-3
comer	-6	-3	-2	-1	0	-1
sorvete	-8	-5	-4	-3	-2	1

Tabela 5: Exemplo dos caminhos traçados pelo nosso algoritmo. Em azul temos o caminho principal escolhido e em vermelho um segundo caminho, se não tivessemos nossa ordem de importância. Utilizando o seguinte esquema de pontuação: $C = 1$, $NC = -1$, $LMenorF = -1$ e $LMaiorF = -2$. Nas linhas temos a frase 'Eu quero comer sorvete' e nas colunas 'Eu gostaria de comer sorvete'.

Após feito o alinhamento é possível calcular uma pontuação. Somamos um para cada par de palavra correspondente e subtraímos um para cada par de palavra que não corresponde. Podemos ver na tabela 6, o resultado do alinhamento e a pontuação do caminho traçado pela seta azul. Já na tabela 7, podemos ver o resultado do caminho traçado pela seta vermelha. O símbolo # representa uma lacuna.

Frases alinhadas utilizando o algoritmo de Needleman-Wunsch

Posição	1	2	3	4	5	Total
Frase Maior	Eu	gostaria	de	comer	sorvete	-
Frase Menor	Eu	#####	quero	comer	sorvete	-
Pontuação	1	-1	-1	1	1	1

Tabela 6: Exemplo do alinhamento entre duas frases utilizando o caminho traçado pela seta azul, utilizando o seguinte esquema de pontuação: $C = 1$, $NC = -1$, $LMenorF = -1$ e $LMaiorF = -2$. Nas linhas temos a frase 'Eu quero comer sorvete' e nas colunas 'Eu gostaria de comer sorvete'.

A nossa modificação do algoritmo foi programada utilizando a linguagem Python. A complexidade do algoritmo é $O(NM)$, sendo N e M o número de colunas e de linhas, a nossa implementação ainda tem espaço para otimização. Mesmo assim, um alinhamento entre um texto com 8977 palavras e outro com 6053 palavras demorou 8 minutos e 32 segundos, o áudio de origem desses textos tinha 57 minutos e 58 segundos de duração. Já no caso de um áudio de 12 minutos e 34 segundos, com 1373 palavras na transcrição

Frases alinhadas utilizando o algoritmo de Needleman-Wunsch						
Posição	1	2	3	4	5	Total
Frase Maior	Eu	gostaria	de	comer	sorvete	-
Frase Menor	Eu	quero	comer	#####	sorvete	-
Pontuação	1	-1	-1	-1	1	-1

Tabela 7: Exemplo do alinhamento entre duas frases utilizando o caminho traçado pela seta vermelha, utilizando o seguinte esquema de pontuação: $C = 1$, $NC = -1$, $LMenorF = -1$ e $LMaiorF = -2$. Nas linhas temos a frase 'Eu quero comer sorvete' e nas colunas 'Eu gostaria de comer sorvete'.

manual e 1108 palavras na transcrição automática demorou 13 segundos para completar o alinhamento[5].

3.3.1 Legenda

Com o alinhamento completo, é necessário passar a informação do tempo das palavras da transcrição automática para a transcrição manual. Quando tivermos essa informação na transcrição manual, podemos criar a legenda para a entrevista. Entretanto, alguns problemas são encontrados, eles são:

- Não temos a informação do tempo de cada lacuna da transcrição automática;
- O acúmulo de tempo em determinadas palavras, as vezes sequencialmente.

Outro ponto importante é que boa parte dos áudios costumam ter um sinal no início. Sendo assim, nós não formamos uma frase com a primeira palavra do alinhamento, nós forçamos essa palavra a sair sozinha na legenda. Com as palavras subsequentes são formadas frases para a criação de cada linha da legenda. Cada frase receberá somente o tempo do seu inicio e do seu fim, a marcação do tempo será retirado da transcrição automática utilizando o alinhamento entre as duas transcrições. Só é necessário o tempo do inicio e do fim de cada frase, porque o objetivo é a criação de uma legenda, ou seja, ter o tempo das palavras do meio da frase não afeta o resultado e ter essa informação iria tornar o problema mais complexo. Para montar as frases, seguimos as seguintes regras:

- Cada frase tem no mínimo 10 palavras, contando as lacunas criadas na transcrição manual, embora exista uma exceção, que será falada mais à frente;
- Se a décima palavra for uma lacuna na transcrição automática é procurada a próxima palavra que não for;

[5] A programação desse algoritmo pode ser encontrado em https://github.com/lucasmeirelesta/cpdoc-text-align/blob/master/nwalgorithm.py, acessado em 30 de abril de 2019.

- Se a frase teve um número de carácteres por segundo (CPS) menor do que 5 ou maior do que 21, é procurada a próxima palavra até essa condição ser respeitada;
- Caso não seja possível encontrar esse valor, é utilizado o conjunto de palavras que tiveram o menor CPS;
- Se a frase tiver mais do que 20 palavras, ela é dividida em partes menores, de tamanho iguais e de no máximo 20 palavras. É calculado o tempo médio por palavra que é dito nessa frase, antes de particionar. A primeira partição começa no tempo original da frase e termina no tempo inicial mais o tempo médio por palavra, a próxima partição começa no tempo da partição anterior e termina seguindo a mesma regra. A última partição segue a mesma regra para o seu inicio e finaliza no tempo original da frase.

Essas frases são gravadas em um arquivo de texto, utilizamos a extensão desse arquivo como srt por ser um padrão utilizado em legenda. Com o seguinte formato:

- Primeira linha possui o número da frase;
- Na segunda linha vem o tempo de inicio e do fim da frases, separados por -->";
- Na terceira linha temos a frase;
- As próximas linhas seguem essa mesma regra, mas utilizando as próximas frases.

As regra baseadas no CPS, são justificadas porque uma frase com um CPS alto indica que temos muitos caracteres em pouco tempo, ou seja, mais chance de estarmos criando uma em um tempo mais curto do que a realidade. Já se o CPS for pequeno, temos a situação contrária, em um faixa de áudio temos poucos caracteres alinhados.

Programamos em Python a automatização desse processo de criação de legendas.[6].

[6] A função utilizada para criar a legenda pode ser encontrada em https://github.com/lucasmeirelesta/cpdoc-text-align/blob/master/utils.py, acessado em 30 de abril de 2019.

Parte IV

Resultados

4 Resultados

4.1 Google Speech-to-Text

Como discutido na metodologia, é necessário fazer o *uploud* do áudio para a nuvem. Quando falarmos do tempo de execução do código, não vamos estar levando em conta essa parte, porque é afetada unicamente pela velocidade de *upload*. Em nossos testes o áudio mais longo teve duração de $3h$ $20m$ e $56s$, o tempo para a criação da transcrição automática foi de 47 minutos. Testando em um áudio de $01h$ $02m$ e $19s$ de duração, o tempo para a transcrição foi de 15.3 minutos. Em contrapartida, a menor entrevista tinha $12m$ e $39s$ de duração, o tempo de execução foi de 5.76 minutos. Outro exemplo, é uma entrevista de $24m$ e $42s$ de duração, que demorou 6.04 minutos para a criação da transcrição automática. Tudo indica que não é somente a duração da entrevista que afeta o tempo de execução.

Na figura 1, temos um exemplo do que o Google Speech-to-Text retorna utilizando nosso código, a entrevista utilizada foi com a Irene Gala[1]. Esse serviço divide automaticamente as frases em pedaços, cada pedaço tem um nível de confiança associado a ele e cada palavra que compõe um pedaço, tem o tempo do inicio e do fim que o serviço estima que essa palavra tenha sido dita. A estrutura foi feita dessa forma:

- 'Transcription:' antecede cada conjunto de palavras;
- 'confidence:' é uma métrica que indica o quanto o algoritmo está certo da transcrição;
- 'Word:' antecede cada palavra que aparece, seguido do seu respectivo tempo.

Vale ressaltar que normalmente a primeira frase sempre vai ter um nível de confiança baixo, porque o primeiro minuto dos nossos áudios tem um ruído, correspondente ao *Color bar* que inicia o vídeo.

A transcrição automática da entrevista com a Irene Gala era 11.06% menor do que a transcrição manual, o percentil 10 dos valores retornados pela a informação de *confidence* foi de 87.07%[2]. Já a entrevista com Carlos Santos Cruz[1] a diferença é de 32.57%, o percentil 25 dos valores da variável *confidence* foi de 86.55%[3]. Na amostra

[1] Entrevista ainda não publicada no site da História Oral do CPDOC para acesso externo.

[2] O percentil 10 indica que somente em 10% dos casos o valor da variável *confidence* foi menor do que 87.07%, ou seja, em 90% dos casos a transcrição automática está mais do 87.07% confiante de que a transcrição está inteiramente correta.

[3] O percentil 25 indica que somente em 25% dos casos o valor da variável *confidence* foi menor do que 86.55%.

```
Transcript: São Paulo seja
confidence: 0.7599719762802124

Word: São, start_time: 0.0, end_time: 59.2
Word: Paulo, start_time: 59.2, end_time: 59.5
Word: seja, start_time: 59.5, end_time: 59.9

Transcript: junho de 2016 primeira sessão entrevista Embaixadora Irene vida gala
estamos presentes André palatinik o Vitor Sion o Bruno Lopes e Mathias espectador
para a gente começar acho que o ideal seria a gente cobrir o início da sua carreira
no Itamaraty porque a sua carreira tem uma peculiaridade que é muito distintiva que
é que a sua trajetória das poucas no Itamaraty que tem fofoca nitidamente Regional
né ao longo de sua carreira os principais impostos foram na África na Secretaria de
Estado a sua tarefa foi principalmente a África também então vamos lá 1983 a senhora
terminação Fran a gente tudo em branco conta pra gente
confidence: 0.9311943650245667

Word: junho, start_time: 59.9, end_time: 60.2
Word: de, start_time: 60.2, end_time: 60.3
Word: 2016, start_time: 60.3, end_time: 61.4
Word: primeira, start_time: 61.4, end_time: 62.0
Word: sessão, start_time: 62.0, end_time: 62.2
Word: entrevista, start_time: 62.2, end_time: 63.2
Word: Embaixadora, start_time: 63.2, end_time: 63.8
Word: Irene, start_time: 63.8, end_time: 64.4
Word: vida, start_time: 64.4, end_time: 64.7
Word: gala, start_time: 64.7, end_time: 64.7
Word: estamos, start_time: 64.7, end_time: 65.9
Word: presentes, start_time: 65.9, end_time: 66.6
Word: André, start_time: 66.6, end_time: 67.2
Word: palatinik, start_time: 67.2, end_time: 67.8
```

Figura 1: Exemplo da transcrição automática da entrevista com a Irene Gala.

de entrevistas utilizadas a transcrição automática é em média 19.71% menor do que a transcrição manual, o percentil 10 da variável *confidence* é 85.68% e a mediana é 90.31%.

Um dos problema da transcrição automática pode ser visto na figura 2, a primeira palavra dessa frase foi transcrita como 'Ele' e tem uma duração total de 2.5 segundos. Um teste simples com um conômetro permite ver que isto não é natural. Se escutarmos essa faixa do áudio, a entrevistada fala 'É então', prolongando bastante a palavra 'É' como se estivesse se preparando para a resposta. Quando essa aglutinação acontece, o tempo total é dividido em mais de uma palavra e nem sempre começa na primeira palavra a divisão do tempo.

```
Transcript: Ele é bem pessoal né mas eu conectei no branco eu já comecei a estudar
África e foi uma coisa que gera 85 eu queria fazer uma coisa que ninguém tivesse
fazendo porque a competição era muito grande então eu descobri que ninguém mexer com
a filha tá muito eu vou começar a estudar na África em duas tia tudo branco eu
comecei a fazer todos os trabalhos sobre temas africanos então por exemplo o
embaixador ricúpero da Vitória diplomática brasileira eu fiz um trabalho sobre
relações brasil-áfrica do Sul do período desde os anos 50 60 e depois eu fiz o
trabalho de direito internacional Fiz um trabalho sobre fronteiras africanas então
desde 85 eu comecei a ideia foi essa aí eu saí do Rio Branco e fui trabalhar na
divisão da África 2 de lá eu fui para a divisão já fica um fui para Assessoria do
departamento depois fui para Lisboa
confidence: 0.9432033896446228

Word: Ele, start_time: 107.7, end_time: 110.3
Word: é, start_time: 110.3, end_time: 110.3
Word: bem, start_time: 110.3, end_time: 110.9
Word: pessoal, start_time: 110.9, end_time: 111.1
Word: né, start_time: 111.1, end_time: 111.5
Word: mas, start_time: 111.5, end_time: 111.9
```

Figura 2: Exemplo da transcrição automática da entrevista com Irene Gala, a primeira palavra fica com um acumulo de tempo maior do que o que seria necessário.

4.2 Alinhamento

Foram testados diferentes hiperparâmetros para o algoritmo de Needlman-Wunsch, podemos ver alguns exemplos do impacto na matriz de pontuação ocasionado pela escolha dos hiperparâmetros. Na figura 3, temos que o caminho em branco é o mais provável de ter sido percorrido pelo algoritmo, então é esperado que esses hiperparâmetros estejam minimizando o uso de lacunas. O valores utilizados foram $C = 1$, $NC = -1$, $LMenorF = -1$ e $LMaiorF = -1$.

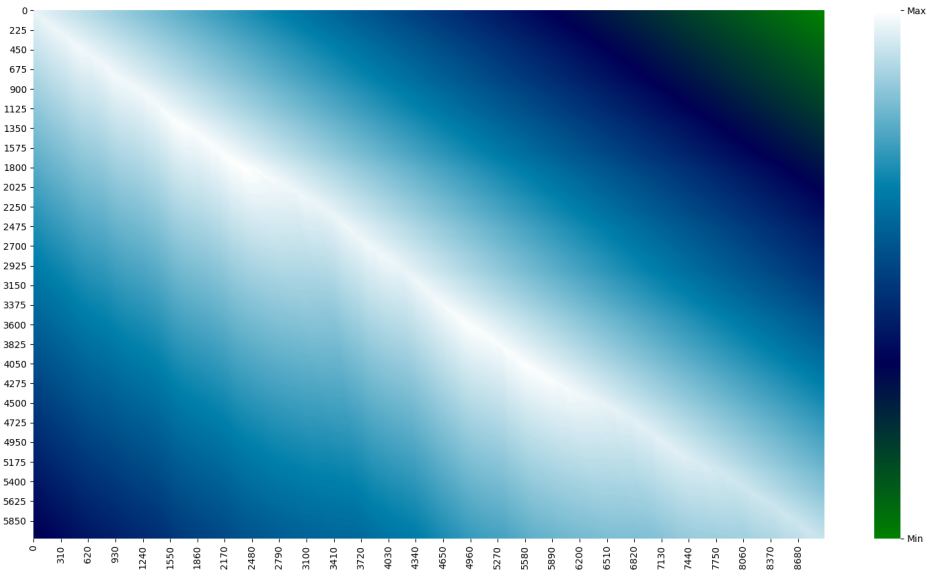

Figura 3: Exemplo da matriz de pontuação da entrevista com Carlos Santos Cruz em 21-10-2016, com os seguintes hiperparâmetros: $C = 1$, $NC = -1$, $LMenorF = -1$ e $LMaiorF = -1$. Cada linha corresponde a uma palavras na transcrição automática e cada coluna corresponde a uma palavra na transcrição manual, ambas na ordem original de cada transcrição e com uma lacuna na primeira posição. Quanto mais branca mais próxima do máximo, quanto mais azul o valor vai diminuindo até chegar em tom de verde que é o mais próximo do mínimo.

Na figura 4, utilizamos os seguintes hiperparâmetros : $C = 2$, $NC = -1$, $LMenorF = -2$ e $LMaiorF = -1$, podemos ver que o comportamento muda. Estamos penalizando a criação de lacuna na transcrição manual, comparando com a figura 3 fica menos suave a pontuação na diagonal superior da matriz. Com esses hiperparâmetros, aumentamos o valor para o caso de pares de palavras que coincidem, podemos perceber que essa combinação gerou uma área mais incerta de branco a esquerda da diagonal inferior, não deixando claro qual caminho o algoritmo irá seguir.

Podemos ver exemplos do alinhamento em ambos os casos na tabela 8 e 9, escolhemos o tempo do áudio 119.9 segundos até 125.9 segundos, da entrevista com o Carlos Santos Cruz. O primeiro ponto a observar, é que o número de palavras na mesma faixa do tempo aumentou, sendo de 11 palavras na tabela 8 e de 15 palavras na tabela 9. Na tabela

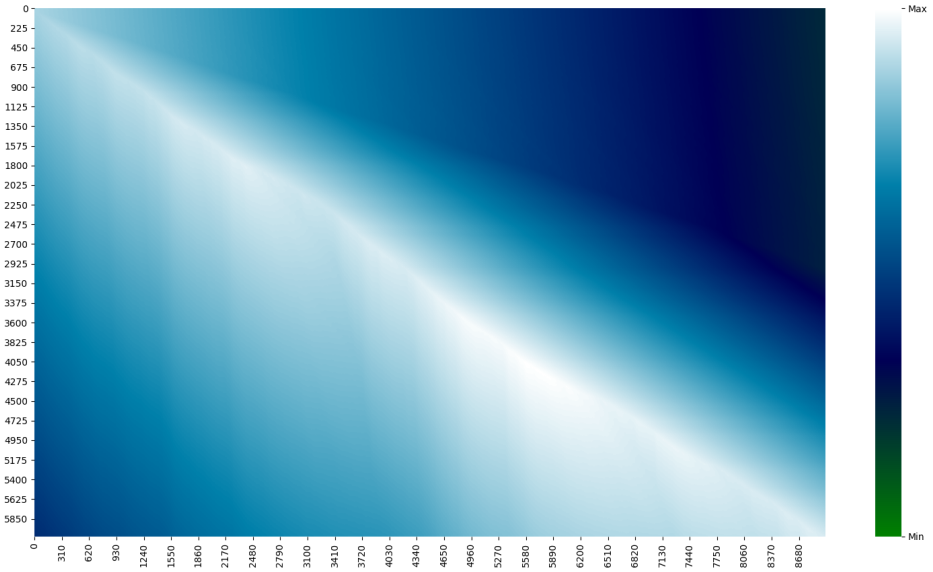

Figura 4: Exemplo da matriz de pontuação da entrevista com Carlos Santos Cruz em 21-10-2016, com os seguintes hiperparâmetros: $C = 2$, $NC = -1$, $LMenorF = -2$ e $LMaiorF = -1$. Cada linha corresponde a uma palavras na transcrição automática e cada coluna corresponde a uma palavra na transcrição manual, ambas na ordem original de cada transcrição e com uma lacuna na primeira posição. Quanto mais branca mais próxima do máximo, quanto mais azul o valor vai diminuindo até chegar em tom de verde que é o mais próximo do mínimo.

8, até as palavras que não coincidem fazem sentido estarem alinhadas na mesma posição. Entretanto, na 9 não parece ocorrer esse fator. Sendo assim, podemos calcular uma pontuação para esses trechos, teríamos uma pontuação de 3 para o trecho da tabela 8 e -7 9. Se fizermos pontuação na transcrição inteiro, temos uma pontuação de -390 para o texto utilizando os hiperparâmetros $C = 1$, $NC = -1$, $LMenorF = -1$ e $LMaiorF = -1$ e de -2033 utilizando os hiperparâmetros $C = 2$, $NC = -1$, $LMenorF = -2$ e $LMaiorF = -1$.

Posição	1	2	3	4	5	6	7	8	9	10	11
Automática	distância	pequena	composta	por	um	coronel	##	######	tenente-coronel	ou	equivalente
Manual	aditância	pequena	composta	por	um	coronel	um	tenente	coronel	ou	equivalente

Tabela 8: Exemplo do alinhamento entre duas frases da entrevista de Carlos Santos Cruz, do segundo 119.9 até o 125.9, com os seguintes hiperparâmetros: $C = 1$, $NC = -1$, $LMenorF = -1$ e $LMaiorF = -1$. Cada conjunto de hashtag é uma lacuna que foi adiciona na frase.

Posição	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Automática	distância	pequena	#	###	##########	########	composta	por	um	coronel	tenent-coronel	ou	########	##	equivalente
Manual	da	defesa	e	uma	aditancia	pequena	composta	por	um	coronel	um	tenente	coronel	ou	equivalente

Tabela 9: Exemplo do alinhamento entre duas frases da entrevista de Carlos Santos Cruz, do segundo 119.9 até o 125.9, com os seguintes hiperparâmetros: $C = 2$, $NC = -1$, $LMenorF = -2$ e $LMaiorF = -1$.Cada conjunto de hashtag é uma lacuna que foi adiciona na frase.

Pelo decorrer dos nossos testes, escolhemos utilizar os hiperparâmetros $C = 1$, $NC = -1$, $LMenorF = -1$ e $LMaiorF = -1$, porque se mostraram mais consistentes em relação a nossa métrica de pontuações. Foi possível alcançar até pontuação positiva com essa escolha de hiperparâmetros em outros áudios, como por exemplo na entrevista do Juscelino Kubitschek de 01-03-1974 (OLIVEIRA, 1979), onde conseguimos uma pontuação de 300 no alinhamento das transcrições.

4.3 Legenda

Após o alinhamento, rodamos o script de criação de legends, como descrito na metodologia. Utilizamos uma ferramenta gratuita, chamada Aegisub, para poder ouvir nossos resultados. Essa ferramenta recebe a legenda, o áudio e separa quais frases pertencem a quais trechos de áudio. Nossos resultados precisam ser validados manualmente, mas pela nossa percepção os resultados foram satisfatórios. Na seção de discussão comentamos por que os resultados não foram validados automaticamente.

Mesmo assim, podemos trazer exemplos de como foram nossos resultados. Na figura 5, podemos ver que a segunda legenda está selecionada e a faixa de áudio correspondente a essa parte. Se escutarmos essa faixa em específico, temos que o início dela está alinhada corretamente, só que em vez do áudio terminar no fim da palavra 'entrevistada', o áudio termina só com o interlocutor tendo pronunciado às 3 primeiras letras. Se ouvirmos a faixa seguinte, a entrevistadora termina de pronunciar a palavra 'entrevistada' e continua corretamente o restante do áudio, só faltando a pronuncia do último 'k' em 'Kubitschek'.

A grande maioria das frases que avaliamos seguem esse mesmo comportamento. No entanto, temos casos onde ocorrem muitas lacunas consecutivas na transcrição automática, o que aumenta o erro entre o fim de frase e o início da próxima frase e voltando a se estabilizar depois de algumas frases. Na tabela 10, podemos ver um desses casos, o áudio começa de maneira correta e só na sexta frase se normaliza. Isso acontece por ser uma área com um grande quantidade de lacunas na transcrição automática. O que resulta em uma frase muito grande, essa frase é particionada em frases menores e o tempo total da frase é dividido entre elas, podendo resultar em erro de alinhamento.

Pelos nossos testes, temos que esses casos são minoria e a sincronização permanece viável para ser utilizada como legenda.

início	Fim	Frase	Tipo
00:51:06.00	00:51:11.10	brasil o general da a ordem para o coronel o coronel da a ordem para o capitao o coronel	Alinhamento
		brasil o general da a ordem para o coronel o coronel da a ordem para o capitao o coronel vai fazer da a ordem para o capitao	Verdadeiro
00:51:11.10	00:51:16.20	vai fazer da a ordem para o capitao o capitao vai fazer na onu se voce nao for na	Alinhamento
		o capitao vai fazer na onu se voce nao for na frente nao acontece porque as motivacoes sao diferentes	Verdadeiro
00:51:16.20	00:51:21.30	frente ## ####### nao acontece porque as motivacoes sao diferentes e diferente voce estar lutando pelo seu pais e	Alinhamento
		e diferente voce estar lutando pelo seu pais e voce estar lutando por uma bandeira de e a	Verdadeiro
00:51:21.30	00:51:26.40	voce estar lutando por uma bandeira de e a cadeia de comando e mais complexa tambem ## como e	Alinhamento
		cadeia de comando e mais complexa tambem como e a relacao os caras eu dei sorte porque	Verdadeiro
00:51:26.40	00:51:31.50	a relacao os caras eu dei sorte porque nesse contingente os comandantes eram muito bons o comandante do peru	Alinhamento
		nesse contingente os comandantes eram muito bons o comandante do peru era um cara fantastico	Verdadeiro
00:51:31.50	00:51:36.60	era um cara fantastico muito bom comandante uma lideranca a companhia peruana era todo mundo eram forcas especiais	Alinhamento
		muito bom comandante uma lideranca a companhia peruana era todo mundo eram forcas especiais	Verdadeiro

Tabela 10: Exemplo da sincronização entre a legenda e o que é falado no áudio na mesma faixa de tempo.

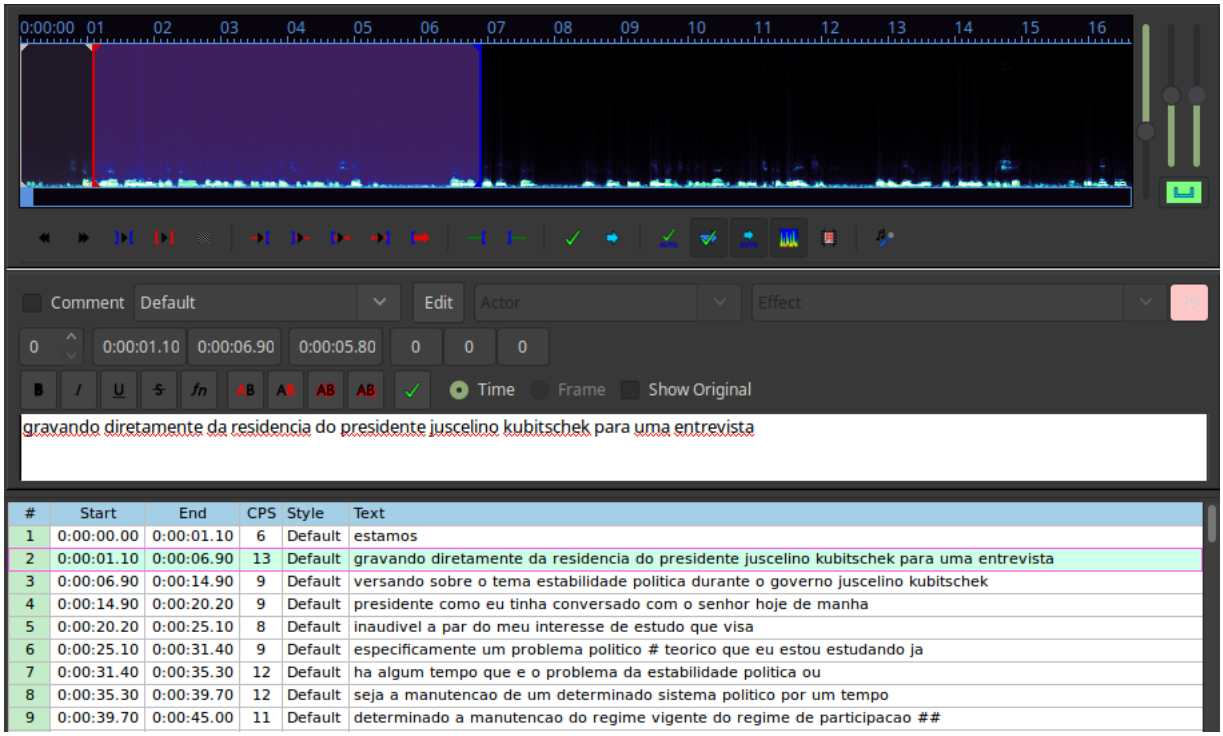

Figura 5: Exemplo da sincronização entre a legenda criada pelo nosso algoritmo e o áudio da entrevista do Juscelino Kubitschek (OLIVEIRA, 1979), utilizando o programa Aegisub

Discussão

Durante a construção desse trabalho foram levantadas ideias em relação ao tema, algumas chegamos a testar, mas não incluímos no trabalho e outras mesmo antes de testar nós seguimos em frente sem elas. Então, nessa seção nós discutimos essas ideias e as motivações por trás delas.

Chegamos a discutir o uso de fonemas no lugar das palavras, para a etapa de alinhamento. Ou seja, seria possível converter cada palavra da transcrição manual e da transcrição automática para o seu fonema correspondente e depois fazer o alinhamento. Assim, palavras cujos fonemas são iguais mas a escrita é diferente poderiam ser consideradas uma correspondência, por exemplo: as palavras seção e sessão tem o mesmo som, mas grafia diferente. Essa ideia não foi utilizada no trabalho, porque o nosso *framework* é invariante em relação ao idioma, dado que seja possível criar uma transcrição automática no idioma desejado. No nosso caso, temos entrevistas em inglês, espanhol e português.

Foi testado a Distância de Levensthein. Essa métrica calcula a distância entre duas palavras, pela quantidade de edição que é necessária realizar para que as palavras fiquem iguais. Utilizamos essa métrica dentro do algoritmo de Needleman-Wunsch, em vez de testarmos se o par de palavras eram iguais, nós testamos se a distância entre o par de palavras era menor do que um limite. Os testes foram realizados somente para pares de palavras que a soma da quantidade das letras fosse maior do que 6. Quando fizemos os testes adicionando esse passo, o tempo de execução aumentou consideravelmente. Essa métrica tem complexidade O(nm), sendo n e m o tamanho de cada palavra em um par de palavras. A pontuação alcançada não justifica a utilização do algoritmo, se compararmos a pontuação de um alinhamento com essa métrica com a pontuação de um alinhamento sem essa métrica, temos que a pontuação é próxima. Nessa caso, a pontuação para ambos os alinhamentos é calculada usando a Distância de Levensthein. No lugar de compararmos se o par de palavras é igual, nós adicionamos uma pontuação positiva se essa distância for menor do que um limite.

Outro ponto importante para se discutir, é a escolha do Google para a transcrição automática, porque poderíamos usar qualquer serviço que tivesse uma transcrição automática com o tempo de cada palavra. Quando começamos o projeto, os serviços encontrados, que se diziam melhores do que o do Google, não tinham o português como uma das opções de transcrição. Outra vantagem do serviço do Google, é uma ferramenta que permite identificar o idioma utilizada no áudio. Essa ferramenta pode facilitar a generalização da nosso código, dado que é necessário informar a linguagem do áudio para o serviço de transcrição automático.

Foi discutida qual seria a melhor forma de validar a sincronização de maneira automática. O que nós pensamos, seria necessário uma base de dados com o tempo que cada palavra, assim seria possível criar um áudio sintético. Com esse áudio sintético, seria possível ver o quanto a legenda está desalinhada com o tempo original. Não foi encontrado um banco de dados que pudesse suprir essa necessidades, o mais próximo disso são bancos que possuem sentenças. Como a nossa legenda é feita automaticamente, as frases formadas pelo nosso processo não corresponderiam necessariamente as sentenças do banco de dados.

Conclusão

Conseguimos fazer o uso de um serviço de transcrição automática de áudio, possibilitando assim a criação de transcrição automática para entrevistas que ainda não tem transcrição manual, podendo assim auxiliar o acervo de história oral com arquivos em formato de texto dessas entrevistas.

Vimos que a utilização do algoritmo de Needleman-Wunsch é viável para o alinhamento entre dois textos, mesmo esses textos tendo uma pontuação baixa em relação a qualidade do alinhamento e tendo um número considerável de palavras. Essa pontuação baixa pode ser explicada pela diferença do tamanho entre dois texto, na melhor da hipóteses o menor número de lacunas, necessário para o alinhamento, será o mínimo para igualar os dois textos em tamanho, o que já gera uma grande penalização.

A criação da legenda para ser sincronizada com a entrevista também foi concluída. Fazendo testes com áudios sincronizados com a legenda, o resultado é bem satisfatório. Por mais que algumas partes da sincronização não sejam exatas, são próximas o suficiente para o ouvinte continuar a entender a entrevista.

Com os resultados desses trabalho, será possível encapsular o que foi construído e entregar em um ambiente de fácil utilização, um gerador de legendas e de transcrição automática para o CPDOC.

Referências

AMODEI, D. et al. Deep speech 2 : End-to-end speech recognition in english and mandarin. In: BALCAN, M. F.; WEINBERGER, K. Q. (Ed.). *Proceedings of The 33rd International Conference on Machine Learning*. New York, New York, USA: PMLR, 2016. (Proceedings of Machine Learning Research, v. 48), p. 173–182. Disponível em: <http://proceedings.mlr.press/v48/amodei16.html>. Citado na página 24.

Baker, J. M. et al. Developments and directions in speech recognition and understanding, part 1 [dsp education]. *IEEE Signal Processing Magazine*, v. 26, n. 3, p. 75–80, May 2009. ISSN 1053-5888. Citado na página 23.

Byrne, W. et al. Automatic recognition of spontaneous speech for access to multilingual oral history archives. *IEEE Transactions on Speech and Audio Processing*, v. 12, n. 4, p. 420–435, July 2004. ISSN 1063-6676. Citado na página 26.

CAMPBELL, J. P. Speaker recognition: A tutorial. *Proceedings of the IEEE*, IEEE, v. 85, n. 9, p. 1437–1462, 1997. Citado na página 23.

Chan, W. et al. Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. In: *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. [S.l.: s.n.], 2016. p. 4960–4964. ISSN 2379-190X. Citado na página 25.

CHAN, W. et al. *Speech recognition with attention-based recurrent neural networks*. [S.l.]: Google Patens, 2018. US Patent 9,990,918. Citado na página 25.

CHIU, C.-C. et al. State-of-the-art speech recognition with sequence-to-sequence models. In: . [S.l.: s.n.], 2018. p. 4774–4778. Citado 2 vezes nas páginas 24 e 25.

CHUA, S.-L.; FOO, L. Tree alignment based on needleman-wunsch algorithm for sensor selection in smart homes. *Sensors*, Multidisciplinary Digital Publishing Institute, v. 17, n. 8, p. 1902, 2017. Citado na página 26.

DIENY, R.; THEVENON, J. et al. Bioinformatics inspired algorithm for stereo correspondence. Citeseer, 2011. Citado na página 26.

ELLOZY, H. A. et al. *Automatic indexing and aligning of audio and text using speech recognition*. [S.l.]: Google Patents, 1997. US Patent 5,649,060. Citado na página 25.

GAO, Y.; SINGH, R.; RAJ, B. Voice impersonation using generative adversarial networks. In: IEEE. *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. [S.l.], 2018. p. 2506–2510. Citado na página 23.

GIBIANSKY, A. et al. Deep voice 2: Multi-speaker neural text-to-speech. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2017. p. 2962–2970. Citado na página 23.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. [S.l.]: MIT Press, 2016. http://www.deeplearningbook.org. Citado na página 25.

GRAVES, A. Sequence transduction with recurrent neural networks. 11 2012. Citado na página 25.

GREF, M.; KÖHLER, J.; LEH, A. Improved transcription and indexing of oral history interviews for digital humanities research. In: *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)*. [S.l.: s.n.], 2018. Citado na página 27.

He, Y. et al. Streaming end-to-end speech recognition for mobile devices. In: *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. [S.l.: s.n.], 2019. p. 6381–6385. ISSN 2379-190X. Citado na página 25.

HINTON, G. et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *Signal Processing Magazine, IEEE*, v. 29, p. 82–97, 11 2012. Citado na página 23.

HUIJBREGTS, M.; ORDELMAN, R.; JONG, F. M. de. A spoken document retrieval application in the oral history domain. In: UNIVERSITY OF PATRAS/WCL MOSCOW STATE LINGUISTICS UNI. *Proceedings of 10th international conference Speech and Computer, Patras, Greece (SPECOM 2005)*. [S.l.], 2005. v. 2, p. 699–702. Citado na página 26.

IRCING, P. et al. Automatic transcription of czech language oral history in the malach project: Resources and initial experiments. In: . [S.l.: s.n.], 2002. p. 219–234. Citado na página 27.

LÓPEZ, G.; QUESADA, L.; GUERRERO, L. A. Alexa vs. siri vs. cortana vs. google assistant: a comparison of speech-based natural user interfaces. In: SPRINGER. *International Conference on Applied Human Factors and Ergonomics*. [S.l.], 2017. p. 241–250. Citado na página 23.

NEEDLEMAN, S. B.; WUNSCH, C. D. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, Elsevier, v. 48, n. 3, p. 443–453, 1970. Citado na página 25.

NETO, N. et al. Free tools and resources for brazilian portuguese speech recognition. *Journal of the Brazilian Computer Society*, v. 17, n. 1, p. 53–68, Mar 2011. ISSN 1678-4804. Disponível em: <https://doi.org/10.1007/s13173-010-0023-1>. Citado na página 23.

NUNN, P. D.; REID, N. J. Aboriginal memories of inundation of the australian coast dating from more than 7000 years ago. *Australian Geographer*, Taylor & Francis, v. 47, n. 1, p. 11–47, 2016. Citado na página 17.

OLIVEIRA, J. K. de. *Juscelino Kubitschek I (depoimento, 1974)*. Rio de Janeiro: CPDOC, 1979. 15 p. dat. Citado 3 vezes nas páginas 7, 45 e 47.

Picone, J. et al. Automatic text alignment for speech system evaluation. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, v. 34, n. 4, p. 780–784, August 1986. ISSN 0096-3518. Citado na página 25.

PIERACCINI, R.; DIRECTOR, I. From audrey to siri. *Is speech recognition a solved problem*, 2012. Citado na página 23.

SAK, H.; SENIOR, A.; BEAUFAYS, F. Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In: *Fifteenth annual conference of the international speech communication association*. [S.l.: s.n.], 2014. Citado na página 24.

YU, D.; DENG, L. *Automatic Speech Recognition*. [S.l.]: Springer, 2015. Citado na página 23.

ZHANG, Y. et al. Towards end-to-end speech recognition with deep convolutional neural networks. 01 2017. Citado na página 24.