Uma proposta de ferramenta para a experimentação do aprendizado profundo no reconhecimento da fala para o idioma Português

Julio Cesar Duarte*, Matheus Rodrigues Affonso, Rebeca Pinheiro dos Reis
Instituto Militar de Engenharia (IME),
Praça General Tibúrcio, 80, 22290-270,
Praia Vermelha, Rio de Janeiro, RJ, Brasil.
*** duarte@ime.eb.br**

RESUMO: O Processamento de Linguagem Natural já faz parte hoje da rotina de grande parte da sociedade, em gestos simples como comandos de voz pelo celular ou instruções faladas a um assistente virtual. Compreender o que é dito verbalmente pode ser uma tarefa simples para um falante nativo de um idioma, e considerada uma das mais essenciais para o convívio interpessoal. No contexto atual, marcado pela grande busca por integração entre sistemas, Internet das Coisas e ambientes dotados de dispositivos inteligentes e robôs, tal interação entre homens e máquinas fica em evidência, e uma abordagem utilizando aprendizado de máquina para atuar nesse relacionamento torna-se uma solução possível. O presente trabalho tem o objetivo de apresentar um ambiente que possibilite a experimentação de modelos na tarefa de Reconhecimento de Fala, bem como implementar modelos de classificação utilizando redes neurais de aprendizado profundo. Ao final, é feita uma análise comparativa entre a acurácia de possíveis modelos sendo experimentados. Como contribuições deste trabalho, é apresentada uma metodologia que permite a experimentação de reconhecimento de fala, bem como uma proposta de classificador treinado em uma base de dados de áudios em Português do Brasil.

PALAVRAS-CHAVE: *Aprendizado Profundo. Redes Neurais recorrentes. Memória de longo e curto prazo. Reconhecimento de Fala.*

ABSTRACT: Natural Language Processing is now part of the routine of a large part of society, in simple gestures such as voice commands mobile phones or spoken instructions to a virtual assistant. Understanding what is said verbally can be a simple task for a native speaker of a certain language, and it is considered one of the most essential for interpersonal interaction. In the current context, marked by the great search for integration between systems, Internet of Things and environments equipped with intelligent devices and robots, such interaction between men and machines is in evidence. An approach using machine learning to act in this relationship becomes a possible solution. The present work aims at presenting an environment that allows the experimentation of models for the Speech Recognition task as well as implementing classification models using deep learning neural networks. At the end, a comparative analysis between the accuracy of possible models being tested is made. As contributions of this work, a methodology that allows the experimentation of speech recognition and the proposal of a trained classifier using a Brazilian Portuguese Audio dataset are presented.

KEYWORDS: *Deep Learning. Recurrent Neural Networks. Long short-term memory. Speech Recognition.*

1. Introdução

Compreender o que é dito verbalmente por intermédio da fala é uma tarefa corriqueira para um falante nativo de um certo idioma. Atualmente, em um mundo marcado pela grande busca por integração entre sistemas, uma abordagem utilizando aprendizado de máquina que possa compreender facilmente uma linguagem, torna-se uma solução possível e desejável nessa interação entre pessoas e máquinas.

Um problema muito importante nessa área é o denominado Reconhecimento de Fala, ou, do inglês, *Speech Recognition*, e existem na literatura diversos trabalhos que procuram resolvê-lo utilizando aprendizado profundo. Por exemplo, Hannun et al. [1] propõe uma solução completa utilizando uma rede neural recorrente bidirecional para a classificação de arquivos de áudio apresentando, além da arquitetura da rede, um modelo de linguagem para reduzir

erros fonéticos e soluções para paralelismo da fase de treinamento.

Já em um outro estudo, Amodei *et al*. [2] trabalha com uma arquitetura similar, entretanto mais complexa que a do modelo proposto por Hannun *et al*. [1], e que se mostrou muito eficiente tanto para a língua inglesa quanto para o Mandarin, dois idiomas extremamente diferentes no que diz respeito à construção linguística e escrita.

Existem também estudos sobre reconhecimento de fala aplicado à Língua Portuguesa do Brasil. Por exemplo, Quintanilha [3] desenvolve um modelo de aprendizado profundo com erro por rótulo de 25,13%, mostrando ser viável desenvolver um sistema de ponta a ponta para o português. Em termos de ferramentas disponíveis para tal tarefa, a equipe da Mozilla possui um projeto de código aberto com uma implementação baseada na arquitetura do grupo de pesquisa Baidu [4] que utiliza a biblioteca Python do Google para aprendizado profundo denominada *Tensorflow*.

Entretanto, poucos são os trabalhos encontrados que tentam resolver o problema do reconhecimento de fala para o Português e que possuam uma ferramenta de código aberto que permita realizar a experimentação de diferentes arquiteturas e hiperparâmetros.

O presente trabalho tem como finalidade apresentar uma ferramenta de experimentação que possibilite a realização de tais experimentos, utilizando modelos de aprendizado profundo baseados em redes recorrentes para a tarefa de reconhecimento da fala em língua portuguesa. Tal implementação utilizou como base a ferramenta gerenciada pela equipe do Mozilla [4].

Além disso, também são implementados modelos de classificação para a tarefa, a partir de dados de áudio brutos, de forma a definir uma metodologia que permita não só a escolha do melhor modelo para o Português do Brasil, mas que também possibilite a realização de experimentos com outros idiomas. A grande contribuição do trabalho, em relação aos resultados práticos, é apresentar a flexibilidade que a ferramenta possui na inspeção do melhor conjunto de hiperparâmetros a serem utilizados em um problema particular. Nesse sentido, foram apresentados diversos experimentos, com diferentes hiperparâmetros, que permitem mostrar a evolução dos resultados em um possível ambiente de desenvolvimento de um reconhecedor de fala para o Português. Dessa forma, são apresentados uma metodologia que permite a experimentação de reconhecimento de fala e um classificador treinado em uma base de dados de áudios em Português do Brasil. Além disso, todos os códigos e bases de dados utilizados estão disponibilizados publicamente [5], permitindo a reprodução dos experimentos realizados e a continuação do trabalho para outros possíveis idiomas.

2. A ferramenta Deepspeech

Devido à repercussão positiva das técnicas e resultados obtidos com o sistema de reconhecimento de fala proposto por Hannun *et al*. [1], a equipe da Mozilla decidiu desenvolver um sistema de código fonte aberto [6] para reconhecimento de fala completo, baseado não só nas técnicas presentes no artigo do Baidu em questão, mas em outros estudos, tais como os realizados por Park *et al*. [7] para aumentar a capacidade de generalização dos modelos através de técnicas como *Data Augmentation* dos dados brutos de áudio (sinais e espectrogramas), que consiste na complementação da base com dados artificiais gerados a partir dos dados disponíveis.

A ferramenta *Deepspeech* utiliza o Tensorflow desenvolvido pelo Google e, além de oferecer modelos pré-treinados para o inglês e uma lista de APIs em diversas linguagens (C, C#, Java, JavaScript e Python) para facilitar o uso dos modelos em sistemas já em produção, permite que o modelo seja treinado com uma base de dados própria, bem como a configuração de diversos hiperparâmetros do sistema, tais como o número de células em cada camada, taxa de aprendizado e até mesmo o alfabeto e um modelo de linguagem para aumentar a precisão das transcrições.

O *Deepspeech*, por ser uma plataforma customizável para a construção de reconhecedores de fala, que utilizam aprendizado profundo, permite não só o treinamento de modelos totalmente novos, mas também o aproveitamento de modelos pré-treinados,

que possibilitam a construção de reconhecedores em ambientes de vocabulário limitado, por exemplo, melhorando o seu desempenho.

2.1 Redes reurais recorrentes

Redes neurais são um conjunto de modelos matemáticos desenvolvidos tomando como inspiração a estrutura e o funcionamento celular de um neurônio biológico. A origem matemática por trás das redes neurais data por volta de 1940, porém, apenas recentemente, com o grande volume de dados e o avanço do poder computacional, as redes neurais adquiriram um desempenho elevado em tarefas de regressão e classificação [8].

A estrutura básica de uma rede neural começa a partir da definição de um neurônio que possui uma tarefa de implementação simples, ao receber um vetor de entrada x, retornar um valor escalar y. Cada neurônio é definido por três parâmetros: (1) um vetor de pesos w; (2) um escalar b chamado bias; e (3) uma função de ativação f. A operação realizada em um neurônio é a seguinte:

$$y = f(w \cdot x + b)$$

onde w·x representa o produto escalar entre o vetor de entrada x e os pesos w.

Existem algumas funções de ativação [9] que são mais comuns no desenvolvimento de redes neurais, sendo elas a ReLU ($f(x) = \max\{0, x\}$), a Sigmoid ($\sigma(x) = 1 / (1 + e^{-x})$) e a Tangente Hiperbólica ($\tanh(x) = (e^{2x} - 1) / (e^{2x} + 1)$).

As redes neurais surgem da organização de dezenas, centenas, ou até milhares de neurônios simples em camadas. As redes neurais possuem três tipos de camadas básicas: (1) camada de entrada, responsável por receber os dados de entrada e repassar às camadas seguintes; (2) camada(s) intermediária(s), composta(s) por diversos neurônios, cada um processando os dados de entrada individualmente e repassando os resultados para as camadas seguintes; (3) camada de saída, responsável por redimensionar os dados para a dimensão da saída.

Redes neurais com uma camada escondida são consideradas redes neurais simples e redes com diversas camadas escondidas em sequência são chamadas de redes neurais profundas, ou mais comumente, *deep neural network*s. Todos os neurônios de uma camada devem possuir a mesma função de ativação, mas camadas diferentes podem ter funções de ativação diferentes.

Durante as rodadas de treinamento de uma rede neural, que normalmente é realizado utilizando o algoritmo *backpropagation*, por intermédio de uma certa taxa de aprendizado, é possível que, por conta de o ajuste dos parâmetros do modelo, ao se buscar minimizar os resultados da função de erro sobre o conjunto de treinamento, o modelo se ajuste tanto a esse conjunto de dados, que não seja capaz de generalizar para dados fora do conjunto de treinamento. Os passos do *backpropagation* são realizados em frações consecutivas do conjunto de treinamento, denominados lotes.

Esse fenômeno de ajuste excessivo é chamado de sobreajuste, ou *overfitting*, sendo o objetivo do modelo, na verdade, ter um bom desempenho sobre os dados do conjunto de teste, e não só sobre o conjunto de treino. Uma das técnicas para prevenir o sobre-ajuste é o *dropout*, que consiste em, durante a etapa de treinamento, escolher uma fração aleatória dos neurônios de cada camada a serem ignorados [10].

Redes neurais recorrentes são um tipo específico de redes neurais, no qual as unidades intermediárias alimentam não apenas as camadas seguintes, mas também a mesma camada ou as anteriores. Ao se alimentar uma célula de uma rede neural recorrente simples, é aplicada uma função de ativação, implicando que, quanto mais a informação flui pela camada intermediária, maior a quantidade de composições de funções de ativação [11].

Tal arquitetura, entretanto, apresenta um problema ao calcular o gradiente descendente devido à regra da cadeia, pois quanto maior a sequência de composição de funções de ativação, menor se torna o gradiente. Ou seja, entradas de iterações mais antigas acabam perdendo o impacto em iterações mais recentes. Esse fenômeno é chamado de esvanecimento do gradiente, do inglês, *vanishing gradient*. Para solucionar esse problema, foi desenvolvida a arquitetura *long short-term memory* (LSTM), que tem o objetivo de manter as informações importantes aprendidas pela rede neural com o passar do tempo, o que é realizado por meio da célula de memória que pode ser observada na **figura 1** [13].

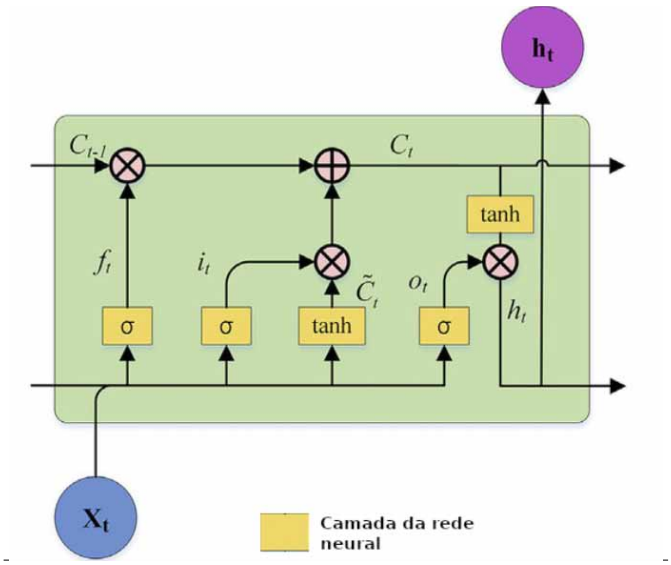

Fig. 1 – Célula LSTM. Fonte: [13].

Simplificadamente, a arquitetura LSTM apresenta três tipos de portas: entrada (i), esquecimento (f) e saída (o). A porta de entrada é responsável pela influência do estado anterior no estado atual, ao passo que a porta de esquecimento controla a parte da informação que deve ser atualizada ou esquecida. Finalmente, a porta de saída controla quais partes da informação devem ser utilizadas na saída da célula [11].

3. Uma proposta para o reconhecimento de fala em português

A proposta deste trabalho é produzir um sistema que seja capaz de permitir experimentos que utilizam modelos de aprendizado profundo baseados em redes recorrentes na área de reconhecimento de fala em língua portuguesa. As etapas seguidas ao longo do trabalho estão retratadas no esquema da **figura 1**. Inicialmente, são definidas as três arquiteturas de rede a serem utilizadas, onde são realizados diversos treinamentos variando seus hiperparâmetros, como forma de compreender como cada configuração pode influenciar nos resultados e para quais valores são observados um melhor desempenho na tarefa. Os hiperparâmetros incluem, por exemplo, número de neurônios, taxa de aprendizado, tamanho do lote, dentre outros.

A experimentação dos hiperparâmetros para cada topologia indica qual a melhor configuração do modelo de classificação correspondente deveria ser utilizada. Em seguida, os três modelos são treinados, variando-se apenas a arquitetura de rede, para que seus resultados possam ser comparados dentro de cada configuração definida. Para cada modelo, as mesmas etapas, de (1) a (5), são executadas. Tais modelos foram treinados e validados a partir da base de dados gratuita Common Voice [14], disponibilizada pela Mozilla. A versão utilizada da base de dados foi a 5.1, que contém aproximadamente 48 horas de áudio validados pela comunidade em arquivos de no máximo dez segundos cada, com 744 diferentes locutores.

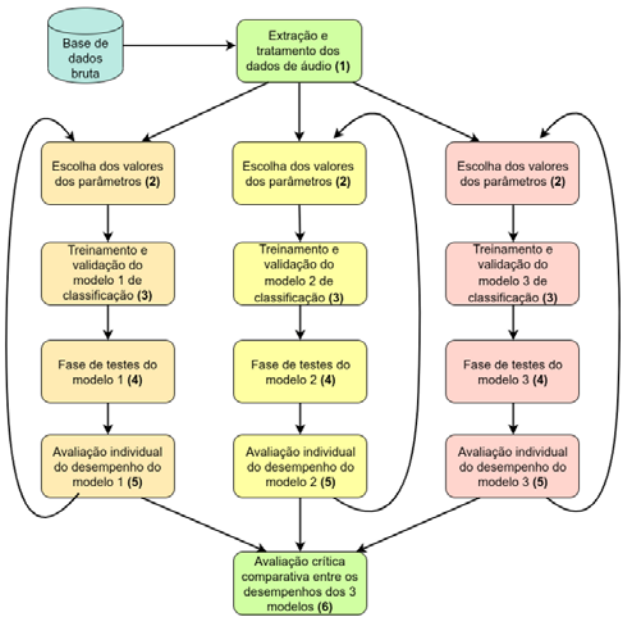

Fig. 2 – Processo de desenvolvimento iterativo do projeto.

As fases do projeto podem ser melhor descritas nas seis seguintes etapas: (1) extração e tratamento dos arquivos de áudio que são utilizados no treinamento e avaliação de cada modelo; (2) escolha dos possíveis valores para os hiperparâmetros dos algoritmos; (3) execução do treinamento e validação do modelo desenvolvido utilizando a ferramenta; (4) testes do modelo utilizando um conjunto de dados diferente; (5) avaliação individual do desempenho de cada modelo;

e (6) realização de uma avaliação crítica e comparativa do desempenho de cada modelo desenvolvido.

As etapas de (2) a (5) se repetem a cada novo modelo proposto, ou seja, para os modelos com topologias diferentes. Já as etapas (1) e (6) são realizadas apenas uma vez, respectivamente, no início do projeto, como um pré-processamento, e ao final, como uma análise dos resultados de forma global. Em cada ramo representado na **figura 2**, o ciclo de execução das etapas (2) a (5) ocorre até que sejam definidos os hiperparâmetros mais adequados para a configuração do modelo.

4. Experimentação na escolha dos hiperparâmetros para a tarefa de reconhecimento de fala

Esta seção relata a realização de diversos experimentos com configurações distintas de hiperparâmetros, para efeitos de comparação dos resultados obtidos e teste do ambiente desenvolvido.

Diversas abordagens podem ser utilizadas para a otimização dos melhores hiperparâmetros para um algoritmo. Buscas manuais e por grids são as técnicas comumente utilizadas na busca de otimização de hiperparâmetros [15]. Neste trabalho, em função do objetivo ser apresentar o ambiente de experimentação, optou-se por realizar os experimentos por intermédio de uma busca gulosa dos hiperparâmetros onde, em cada momento, um deles era otimizado, enquanto os demais se mantinham fixados.

Dessa maneira, pretende-se explorar ao máximo o grau de customização que pode ser aplicado na implementação e arquitetura das redes, de forma a analisar o desempenho sobre a base de dados da língua portuguesa, já que a ferramenta foi desenvolvida com recursos nativos que a princípio facilitam a tarefa para o idioma Inglês. Com isso, espera-se compreender de que forma as alterações nos hiperparâmetros influenciam nos erros, e quais são as melhores configurações para obtenção de melhorias nos resultados.

São realizados experimentos com três modelos distintos, variando-se as topologias da rede. O primeiro modelo possui 3 camadas totalmente conectadas, 1 camada recorrente utilizando células LSTM e 1 camada totalmente conectada ao fim (3-1-1). Os outros dois modelos apresentam arquiteturas semelhantes, incrementando apenas a quantidade de camadas recorrentes de um para o outro: o segundo modelo com 3 camadas recorrentes (3-3-1) e o terceiro modelo com 5 camadas recorrentes (3-5-1).

Com o objetivo de comparar as estratégias experimentadas, são adotadas três medidas comumente utilizadas nesse contexto: a função de custo, *Word Error Rate* (WER) e o *Character Error Rate* (CER).

A função de custo representa o valor cuja arquitetura de rede neural implementada dentro da ferramenta *DeepSpeech* busca minimizar. Nesse caso, a função de custo consiste, em escala logarítmica, calcular eficientemente a probabilidade inversa de um certo áudio X = [X1, X2, ..., XT], ser transcrito em uma sequência Y = [Y1, Y2, ..., YU]. Tal medida está ligada diretamente ao treinamento da rede e seu processo de validação.

A medida WER é uma medida de erro a nível de palavras que considera as suas alterações, inserções e supressões ao se comparar com a saída correta. O WER normalmente penaliza muito os erros, uma vez que apenas um caractere errado em uma palavra a invalida completamente. Nesse caso, também foi considerada a medida CER, que considera os erros a nível de caractere isoladamente.

4.1 Modelo de linguagem

A ferramenta *DeepSpeech* apresenta a implementação de um modelo de linguagem em inglês que pode ser utilizado para refinar os resultados das inferências realizadas, mas que não interfere no treinamento. Para garantir a não interferência do modelo de linguagem nativo do inglês, as *flags*, *lm_alpha* e *lm_beta*, podem ser configuradas para 0, de forma a se manterem nulos os valores de α e β, que regulam o peso da aplicação do modelo de linguagem.

Para os experimentos realizados com o modelo 3-1-1, as flags em questão são zeradas, de modo que nenhum

modelo de linguagem é aplicado à inferência. Para os experimentos com o segundo (3-3-1) e terceiro (3-5-1) modelos de classificação, entretanto, um modelo de linguagem em português é aplicado para que possam ser observadas as diferenças nos resultados obtidos com tal modelo. Espera-se que o modelo de linguagem melhore as predições dos classificadores. O modelo de linguagem, entretanto, é aplicado apenas na etapa final do reconhecimento e, portanto, não interfere no treinamento dos modelos.

O modelo de linguagem em português foi construído a partir das próprias transcrições contidas na base de dados do Common Voice, utilizando a ferramenta chamada *KenLM* [16], sugerida pela equipe da *DeepSpeech*.

Em relação às *flags lm_alpha, lm_alpha_max, lm_beta e lm_beta_max*, também é feito um processo de otimização, conhecido como ajuste dos hiperparâmetros, com o objetivo de encontrar os melhores valores de α e β que se adequem a cada modelo treinado. Para tal otimização, utiliza-se um script fornecido pela própria ferramenta *DeepSpeech*. Tal código busca os valores das flags, a partir de uma biblioteca chamada Optuna [17].

4.2 Experimentos realizados com o modelo 3-1-1

Aqui são descritos os 9 experimentos realizados para a primeira arquitetura, composta de 3 camadas totalmente conectadas, 1 camada recorrente com células do tipo LSTM e 1 camada totalmente conectada. Ao fim de cada treinamento, é executada uma inferência com um mesmo subconjunto de transcrições de áudio já vistas pelos modelos, como forma de comparar o desempenho da rede para cada configuração de hiperparâmetros utilizada e observar qualitativamente algum grau de sobre-ajuste.

Os experimentos estão separados em 4 conjuntos, cada um responsável pela análise dos impactos causados pela variação de um dos hiperparâmetros: número de neurônios (a mesma quantidade foi utilizada em todas as camadas), tamanho do lote, taxa de aprendizado ou tamanho da base de dados. Dentro de um mesmo grupo de experimentos, apenas um dos hiperparâmetros varia, mantendo as outras configurações fixas, como pode ser observado no esquema da **tabela 1**. Por fim, foi possível tirar conclusões acerca dos hiperparâmetros que mais favoreceram o desempenho na tarefa de reconhecimento de fala para este modelo de classificação 3-1-1. O último conjunto possui apenas uma configuração I, pois o seu objetivo é realizar a avaliação dos modelos com um tamanho de base maior, utilizando-se os melhores hiperparâmetros encontrados nos conjuntos anteriores.

Após a realização de todos os experimentos e comparação das métricas obtidas, o eleito é aquele que obtiver o melhor desempenho dentre os outros, sendo este o experimento denominado I. Tal experimento é configurado com 512 neurônios, tamanho do lote 10, taxa de aprendizado 0.0005 e base de dados em 50% do seu tamanho.

Tab. 1 – Esquema representativo dos 4 grupos de experimentos realizados, especificando os valores dos hiperparâmetros para cada treinamento do modelo 3-1-1.

		Número de neurônios					Tamanho do lote				Taxa de Aprendizado		Tamanho da Base de Dados			
Análises	Experimento	64	128	256	512	1024	1	5	10	15	0.001	0.0005	1%	10%	25%	50%
Número de neurônios	A	■					■				■			■		
	B		■				■				■			■		
	C			■			■				■			■		
	D				■		■				■			■		
Taxa de Aprendizado	D*				■		■				■			■		
	E											■				
Tamanho do lote	F					■		■			■		■			
	G					■			■		■		■			
	H					■				■	■		■			
Tamanho da Base	I				■				■			■	■			■

A **tabela 2** contém o valor médio do CER, WER e da função de custo obtidos para o experimento I.

Tab. 2 –Resultados obtidos a respeito do CER, WER e da função de custo sobre o conjunto de teste do experimento I no modelo 3-1-1.

Custo	CER	WER
98,55	0,64	1,13

O gráfico da **figura 3** retrata o valor da função de custo ao longo do experimento I, tanto para o conjunto de treino quanto para o conjunto de validação. Observa-se, entretanto, um comportamento não desejável para o erro de validação.

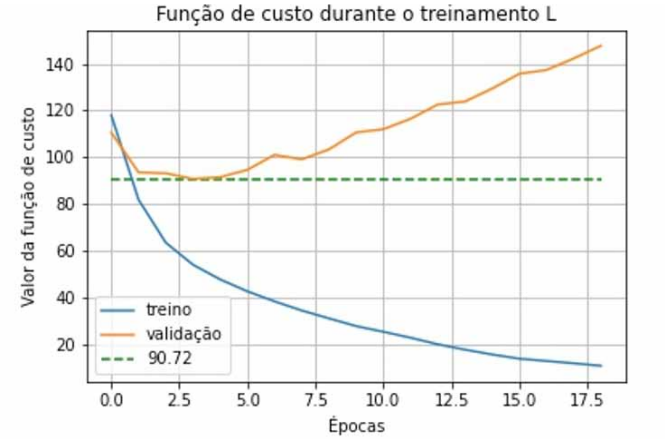

Fig. 3 – Evolução da função de custo para o conjunto de treino e validação ao longo do experimento I no modelo 3-1-1.

4.3 Experimentos realizados com o modelo 3-3-1

Esta subseção descreve os experimentos realizados com o modelo 3-3-1, composto de 3 camadas totalmente conectadas, 3 camadas recorrentes e 1 totalmente conectada, apresentando a discussão teórica a respeito dos resultados obtidos. Os experimentos com esse modelo utilizaram a técnica de *Data Augmentation*, com inserção de ruídos aditivos e multiplicativos no sinal de áudio para aumento do conjunto de treino. Já as inferências, realizadas com os hiperparâmetros mais favoráveis escolhidos ao final, foram executadas com e sem o modelo de linguagem construído em português, para efeitos de visualização dos impactos desse refinamento.

Diferente da metodologia seguida na experimentação com o modelo 3-1-1, os testes a seguir seguem a mesma abordagem gulosa. Os experimentos estão separados em 3 conjuntos, cada um responsável pela análise dos impactos causados pela variação de um dos hiperparâmetros: número de neurônios, tamanho do lote e taxa de aprendizado. A base de dados completa foi utilizada em todos os experimentos. Dentro de um mesmo grupo de experimentos, apenas um dos hiperparâmetros varia, mantendo-se as outras configurações. O tamanho da base foi fixo em 50%.

Os grupos de experimentos seguem a seguinte ordem: número de neurônios, variando entre 200, 400 e 600; taxa de aprendizado, variando entre 0.001, 0.0005 e 0.0001; tamanho do lote, variando entre 20, 60 e 100. Para o primeiro conjunto de experimentos, quando ainda não se conhecem os melhores hiperparâmetros, foram fixados os valores da taxa de aprendizado em 0.0005 e tamanho do lote 100, fazendo variar apenas a quantidade de neurônios, como pode ser observado na **tabela 3**.

Tab. 3 – Esquema representativo do grupo de experimentos a serem realizados para determinação da quantidade ótima de neurônios, especificando os valores dos hiperparâmetros para cada treinamento no modelo 3-3-1.

		Número de neurônios			Taxa de Aprendizado			Tamanho do lote		
Análise	Experimento	200	400	600	0.001	0.0005	0.0001	20	60	100
Número de Neurônios A										
	B									
	C									

Após a análise das métricas obtidas, conclui-se que o modelo apresenta seu melhor desempenho com 600 neurônios. Descoberto tal valor para esse hiperparâmetro, segue-se para o segundo grupo de experimentos, aqueles em que a taxa de aprendizado varia nos 3 valores possíveis, fixando o número de neurônios em 600 e o tamanho do lote novamente em 100, conforme a **tabela 4**.

Tab. 4 – Esquema representativo do grupo de experimentos a serem realizados para determinação da melhor taxa de aprendizado, especificando os valores dos hiperparâmetros para cada treinamento no modelo 3-3-1.

		Número de neurônios			Taxa de Aprendizado			Tamanho do lote		
Análise	Experimento	200	400	600	0.001	0.0005	0.0001	20	60	100
Taxa de Aprendizado	D									
	E									

Após a análise das métricas obtidas, conclui-se que o modelo tem seu melhor desempenho com a taxa de

aprendizado 0.001. Tendo em vista o resultado obtido, os experimentos que se seguem foram fixados com 600 neurônios e taxa de aprendizado 0.001, variando o lote entre 20, 60 e 100, conforme o esquema da **tabela 5**.

Tab. 5 – Esquema representativo do grupo de experimentos a serem realizados para determinação do tamanho do lote, especificando os valores dos hiperparâmetros para cada treinamento no modelo 3-3-1.

		Número de neurônios			Taxa de Aprendizado			Tamanho do lote		
Análise	Experimento	200	400	600	0.001	0.0005	0.0001	20	60	100
Tamanho do lote	F									
	G									
	H ou D									

Após a análise das métricas obtidas, conclui-se que o modelo teve seu melhor desempenho com tamanho do lote 100, equivalendo à configuração de hiperparâmetros do experimento D. A **tabela 6** contém o valor médio do CER, WER e da função de custo obtidos para o experimento D.

Tab. 6 – Resultados obtidos a respeito do CER, WER e da função de custo sobre o conjunto de teste do experimento D no modelo 3-3-1.

Custo	CER	WER
93,50	0,56	1,05

O gráfico da **figura 4** retrata o valor da função de custo ao longo do experimento D, tanto para o conjunto de treino quanto para o conjunto de validação. Observa-se, nesse gráfico, o comportamento desejável onde tanto o erro de treinamento quanto o de validação diminuem à medida que o algoritmo realiza o seu treinamento.

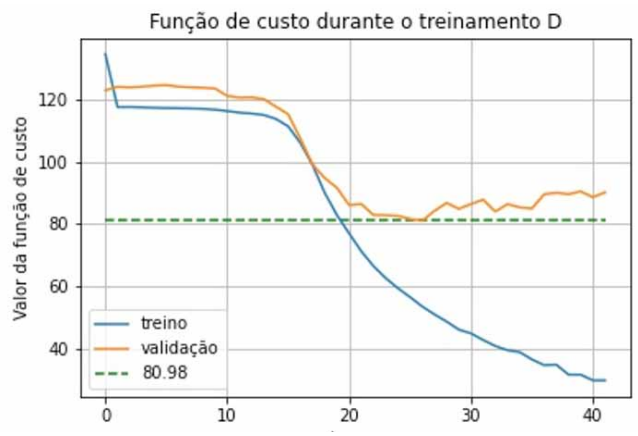

Fig. 4 – Evolução da função de custo para o conjunto de treino e validação ao longo do experimento D no modelo 3-3-1.

Observando os gráficos, nota-se que o experimento D atinge o menor valor de função de custo dentre os comparados, com a curva de validação chegando a 80,98. Isso pode ser confirmado pelas métricas da **tabela 6**, que revelaram o menor valor médio para a função de custo no experimento D, com tamanho de lote 100. De maneira geral, também se percebe um menor valor para o CER no caso do experimento D, reforçando que essa configuração de hiperparâmetros foi a mais favorável para o desempenho do modelo 3-3-1.

Para demonstrar os resultados obtidos nessas condições, foram realizadas inferências com e sem o modelo de linguagem construído em português, como forma de observar os impactos deste para a predição. Vale destacar que tais inferências foram realizadas com um subconjunto de treino. Após a otimização do modelo de linguagem para a arquitetura atual, foram encontrados os valores 0.00483 para a flag α e 0.038042 para a *flag* β. A **tabela 7** mostra os valores das métricas obtidas nessas inferências, com e sem o modelo de linguagem, para efeitos de comparação.

Tab. 7 – Resultados obtidos a respeito do CER, WER e da função de custo do modelo 3-3-1 para o experimento D, com e sem modelo de linguagem.

Experimento	**Custo**	**CER**	**WER**
D	93,50	0,56	1,05
D+ML		0,59	1,02

Pode ser observada uma leve melhora no WER após aplicação do modelo de linguagem. Outro detalhe é que o valor da função de custo de fato não é alterado, já que o modelo de linguagem não afeta o treinamento, apenas é aplicado na decodificação após a saída do modelo. Um possível fator que pode contribuir para a influência do modelo de linguagem ter sido pouco significativo reside na forma como o modelo foi construído, utilizando transcrições da própria base de dados de entrada, ou seja, um conjunto pequeno. Dessa forma, deve-se levar em consideração que tal base de dados é pequena para o universo do modelo de linguagem, ou seja, a construção de um bom modelo requer muito mais informações para melhorar seu desempenho. Além disso, o tamanho das transcrições pode ser um problema, pois são

frases relativamente curtas, enquanto seria mais adequado utilizar sentenças longas para a construção de um modelo mais preciso, por questões inerentes à forma como as probabilidades são calculadas.

Vale destacar também que o otimizador, em função dos motivos apresentados, encontrou valores muito baixos para as flags α e β, indicando que o modelo de linguagem influencia pouco na saída final do classificador. Caso esse impacto fosse mais efetivo, a otimização retornaria valores mais altos, justamente para dar maior peso ao modelo de linguagem.

4.4 Experimentos realizados com o modelo 3-5-1

Esta subseção descreve os experimentos realizados com o modelo 3-5-1, composto de 3 camadas totalmente conectadas, 5 camadas recorrentes e 1 totalmente conectada, apresentando a discussão a respeito dos resultados obtidos. Assim como no anterior, os experimentos com esse modelo utilizaram a mesma técnica de *Data Augmentation*.

Inicialmente, pretendia-se seguir a mesma abordagem gulosa utilizada para o modelo 3-3-1, separando as análises nos 3 conjuntos, cada um responsável por verificar os impactos causados pela variação de um dos hiperparâmetros: número de neurônios, tamanho do lote e taxa de aprendizado. Os grupos de experimentos seguiriam a mesma ordem: número de neurônios, variando entre 200, 400 e 600; taxa de aprendizado, variando entre 0.001, 0.0005 e 0.0001; tamanho do lote, variando entre 20, 60 e 100. O tamanho da base, como no experimento anterior, foi fixado em 50%.

Entretanto, logo na primeira análise, do número de neurônios, nota-se que o modelo não consegue aprender de maneira satisfatória e apresentar bons resultados, de forma que os treinamentos terminam após 15 épocas sem apresentar melhora com as curvas de treino e validação por volta de 120 em termos de função de custo. De forma visual, é como se existisse uma barreira nesse patamar, que não consegue ser superada, diferente do modelo anterior, em que as curvas ultrapassam esse valor após algumas épocas estáveis naquele nível, como pode ser visto no gráfico da **figura 4**, por exemplo.

Tendo em vista as dificuldades encontradas pelo modelo, para esta terceira arquitetura seguiu-se uma abordagem distinta das anteriores: buscou-se encontrar, a princípio, alguma combinação de número de neurônios e taxa de aprendizado que fosse capaz de melhorar o desempenho do modelo 3-5-1. Para o tamanho do lote, em função do melhor resultado obtido pelo modelo anterior, e pensando também em favorecer os tempos de execução do treinamento, foi mantido seu valor constante em 100. Os experimentos seguiram uma tendência de diminuição do número de neurônios, variando entre 600, 400, 200 e 150. De forma equivalente, as taxas de aprendizado exploradas foram 0.005, 0.001, 0.0005 e 0.0001. A configurações de cada experimento podem ser observadas na **tabela 8**.

Tab. 8 – Esquema representativo do grupo de experimentos a serem realizados para determinação do número de neurônios e taxa de aprendizado, especificando os valores dos hiperparâmetros para cada treinamento no modelo 3-5-1.

		Número de neurônios				Taxa de Aprendizado				Tamanho do lote		
Análise	Experimento	600	400	200	150	0.005	0.001	0.0005	0.0001	20	60	100
Análise Global	A	■					■					■
	B		■				■					■
	C		■					■				■
	D			■		■						■
	E			■			■					■
	F			■				■				■
	G			■					■			■
	H				■		■					■

De maneira geral, os gráficos obtidos a respeito da evolução da função de custo para o conjunto de treinamento e validação ao longo dos experimentos foram muito semelhantes, todos mantendo a curva de validação em torno de 120, sem conseguir ultrapassar esta barreira. Como exemplo, a **figura 5** apresenta um desses gráficos para o experimento C.

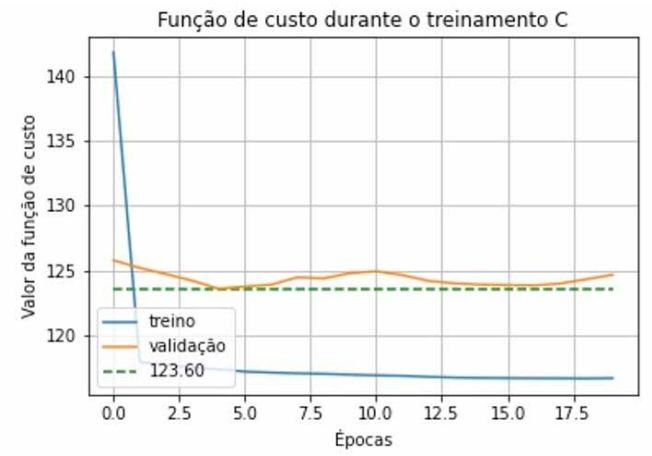

Fig. 5 – Evolução da função de custo para o conjunto de treino e validação ao longo do experimento C, no modelo 3-5-1.

Além disso, a **tabela 9** contém o valor médio do CER, WER e da função de custo para os oito experimentos realizados, como forma de visualização geral dos resultados e comparação.

Tab. 9 – Resultados obtidos a respeito do CER, WER e da função de custo para a análise do número de neurônios e taxa de aprendizado referente ao modelo 3-5-1.

Experimento	Custo	CER	WER
A	128,47	0,77	1,05
B	127,98	0,81	1,02
C	128,27	0,79	1,03
D	127,77	0,78	1,02
E	127,91	0,79	1,04
F	127,75	0,79	1,03
G	128,55	0,78	1,07
H	127,53	0,81	1,01

Comparando-se as métricas obtidas, observa-se que os resultados foram muito semelhantes, de forma que todas as configurações realmente tiveram um desempenho equivalente. Isso pode ser também confirmado pelos gráficos, uma vez que todos seguiram essa mesma tendência e ficaram estacionados com a curva de validação em torno de 120. Diferente do modelo 3-3-1, que também apresentava inicialmente tal barreira no aprendizado, o modelo 3-5-1 não conseguiu superar este patamar e ampliar sua capacidade de generalização.

Esse comportamento pode estar associado ao fato de o modelo 3-5-1 precisar de uma quantidade muito maior de dados e tempo para treinamento, em função da quantidade de parâmetros a serem aprendidos. Outro detalhe a ser levado em consideração é em relação à base de dados utilizada. Vale destacar que o modelo fica restrito às características e nuances disponíveis na amostra fornecida, de forma que alguns comportamentos são inerentes a esses dados, como por exemplo os patamares observados em ambos os modelos. De maneira geral, então, notou-se que o modelo 3-5-1 não se adaptou bem a essa base, talvez por questões de insuficiência de dados disponíveis para uma arquitetura mais robusta como esta, ou por razões de a topologia utilizada não ter sido a mais adequada para esta tarefa, em razão da presença de muitas camadas recorrentes ou até mesmo do tipo de célula aplicado (LSTM).

4.5 Comparação entre os modelos

Concluídos os experimentos de cada modelo, pode-se realizar uma comparação entre seus desempenhos utilizando os hiperparâmetros mais favoráveis de cada um. Para o modelo 3-1-1, já que foi seguida uma metodologia distinta dos outros, pode-se olhar para todos os experimentos de uma maneira geral. Observando os gráficos, verifica-se que o valor mínimo para a função de custo atingido pelas curvas de validação manteve-se na faixa de 105 para grande parte dos experimentos, alcançando o melhor resultado no experimento I, em 90,72. Quanto às outras métricas, este mesmo experimento também se destaca, com o menor valor médio para a função de custo, valendo 98,55, e para o CER, valendo 0,64. Dessa maneira, como forma de comparar o desempenho do modelo 3-1-1 com os outros, pode-se adotar o experimento I como referência.

Em relação ao modelo 3-3-1, foi eleito o experimento D como o mais eficiente, alcançando

um valor mínimo de 80,98 no gráfico da evolução da curva de validação. As outras métricas obtidas para este também foram muito positivas: 93,5 de Custo, 0,56 de CER e 1,05 de WER.

A experimentação com o modelo 3-5-1, como visto na subseção anterior, não apresenta resultados satisfatórios, de forma que todos os experimentos são equivalentes em seus resultados, impossibilitando a eleição de um melhor. De maneira geral, todos os gráficos estacionam na faixa de 120, e a média da função de custo ficou em 128,03, com o CER em torno de 0,79 e o WER em 1,03.

Sendo assim, os resultados apontam que o segundo modelo, 3-3-1, teve melhor desempenho dentre os demais, apresentando as menores métricas de custo, WER e CER, e revelando em seus gráficos que tal rede foi realmente capaz de aprender mais e superar as barreiras observadas, como aquela no nível 120. Vale destacar que, em testes realizados com a ferramenta DeepSpeech para o modelo 3-1-1 no idioma inglês por Ardila *et al*. [18], por exemplo, verifica-se um alto desempenho do classificador, obtido a partir de uma base de dados em inglês com tamanho 50 vezes maior que a utilizada [14], o que indica que o desempenho do modelo é diretamente relacionado com a quantidade de dados disponibilizados durante o seu treinamento.

Tais resultados apontam, pelo menos inicialmente que utilizar uma arquitetura com mais camadas recorrentes leva a melhores resultados no reconhecimento do Português, diferentemente do que foi reportado para o Inglês na ferramenta.

5. Conclusão

O momento atual revela um crescente uso de inteligência artificial no cotidiano dos indivíduos. Espera-se que, em breve, as pessoas estejam em grande parte inseridas em ambientes dotados de dispositivos inteligentes e robôs, de maneira que a interação entre homens e máquinas fique em evidência. Analisando esse contexto, verifica-se a aplicação do Processamento de Linguagem Natural e sua importância para desenvolver essas interações entre humanos e máquinas. Este trabalho, que foca no reconhecimento de fala a partir de áudios da língua portuguesa, é uma etapa essencial que pode ser empregada em qualquer projeto que busque melhorar tal relacionamento entre pessoas e dispositivos inteligentes, por exemplo em tarefas que envolvam comandos de voz.

O presente trabalho apresentou o desenvolvimento de um ambiente que possibilita a experimentação de modelos na tarefa de Reconhecimento de Fala em português brasileiro, bem como a implementação de modelos de classificação utilizando redes neurais de aprendizado profundo, para que no fim possa ser feita uma análise comparativa entre seus desempenhos. Em sua primeira fase, foram realizados diversos experimentos variando-se hiperparâmetros de configuração do ambiente, de forma a analisar e compreender como esses valores impactam no desempenho da rede.

Como etapa seguinte, realizaram-se alterações na topologia da rede, sendo necessário não apenas variar hiperparâmetros, mas também realizar modificações no código base dos modelos. Dadas as definições das melhores configurações de hiperparâmetros para cada arquitetura, foi possível avaliar o desempenho de cada um dos modelos e compará-los. Além disso, foi possível estudar o desempenho da ferramenta DeepSpeech sobre a base de dados da língua portuguesa, uma vez que esta foi implementada com recursos nativos que facilitam tal tarefa para a língua inglesa.

A realização dos experimentos permitiu comprovar a eficácia do ambiente que permitiu a realização de diferentes modelagens com configurações de hiperparâmetros para as topologias das redes testadas, bem como a avaliação do desempenho dos classificadores gerados. A dificuldade em se encontrar rapidamente uma configuração que tenha um desempenho ótimo, demonstra as inúmeras possibilidades de emprego do ambiente em conjunto com outros algoritmos ou sendo aplicado na tarefa para outros idiomas.

Como contribuições deste trabalho, um sistema de experimentação de modelos para a tarefa de reconhecimento de fala foi desenvolvido, mostrando

ser possível utilizar tal ferramenta para realização de experimentos em busca de um modelo mais adequado para a língua portuguesa. Além disso, foi apresentado um classificador treinado em uma base de dados de áudios em português do Brasil, sendo o modelo escolhido aquele que obteve melhor desempenho dentre os demais desenvolvidos. O projeto contribuiu também para enriquecer o conhecimento acerca do problema em questão, aumentando a experiência com as ferramentas de aprendizado profundo, manipulação de redes neurais recorrentes, treino do modelo e configuração dos hiperparâmetros.

Esse trabalho permitiu a construção de um modelo base de comparação para o reconhecimento da fala no idioma do Português utilizando a ferramenta DeepSpeech. Futuros trabalhos poderão utilizar não só o modelo, mas a metodologia de experimentação para melhorar os resultados e, com uma base de dados significativa, construir um reconhecedor que possa ser utilizado em tarefas que lidem com o idioma Português.

No tocante a trabalhos futuros, pretende-se buscar uma base de dados maior em Português para testar as mesmas arquiteturas, e comparar os desempenhos. Além disso, é importante pensar na construção de um modelo de linguagem mais efetivo, utilizando para isso uma base de dados maior que priorize sentenças longas. Mediante a obtenção de resultados melhores, pode-se pensar também na possibilidade de experimentar novas arquiteturas, diferentes tipos de células e outros idiomas. Além disso, pretende-se aplicar o treinamento de tais classificadores em ambientes com muito ruído, criando reconhecedores particulares para certos ambientes ruidosos.

Agradecimentos

Este material é baseado no trabalho apoiado pela Pesquisa Científica do Escritório da Força Aérea sob o número de concessão FA9550-19-1-0020.

Referências bibliográficas

[1] HANNUN, A.; CASE, C.; CASPER, J.; CATANZARO, B.; DIAMOS, G.; ELSEN, E.; PRENGER, R.; SATHEESH, S.; SENGUPTA, S.; COATES, A.; NG, A. Y. Deep speech: Scaling up end-to-end speech recognition. ArXiv, v. 1412.5567, 2014. Disponível em: <https://arxiv.org/abs/1412.5567>.

[2] AMODEI, D.; ANANTHANARAYANAN, S.; ANUBHAI, R.; BAI, J.; BATTENBERG, E.; CASE, C.; CASPER, J.; CATANZARO, B.; CHEN, J.; CHRZANOWSKI, M.; COATES, A.; DIAMOS, G.; ELSEN, E.; ENGEL, J.; FAN, L.; FOUGNER, C.; HANNUN, A. Y.; JUN, B.; HAN, T.; LEGRESLEY, P.; LI, X.; LIN, L.; NARANG, S.; NG, A. Y.; OZAIR, S.; PRENGER, R.; QIAN, S.; RAIMAN, J.; SATHEESH, S.; SEETAPUN, D.; SENGUPTA, S.; SRIRAM, A.; WANG, C.; WANG, Y.; WANG, Z.; XIAO, B. S.; XIE, Y.; YOGATAMA, D.; ZHAN, J.; ZHU, Z. Deep speech 2 : End-to-end speech recognition in english and mandarin. ArXiv, abs/1512.02595, 2015. Disponível em: <http://proceedings.mlr.press/ v48/amodei16.pdf>.

[3] QUINTANILHA, I. M. End-to-end speech recognition applied to brazilian portuguese using deep learning. Ph. D. dissertation, MSc dissertation, PEE/COPPE, Federal University of Rio de Janeiro, 2017.

[4] Battenberg, E.; Chen, J.; Child, R.; Coates, A.; Gaur, Y.; Li, Y.; Liu, H.; Satheesh, S.; Seetapun, D.; Sriram, A.; Zhu, Z. Exploring neural transducers for end-to-end speech recognition. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), 2017.

[5] CORPORATION, M. PFC DeepSpeech. 2020. 19 out. de 2020. Disponível em: <https://github.com/M-Rodrigues/pfc-deepspeech>.

[6] MOZILLA. Welcome to DeepSpeech's documentation. 2021. Disponível em: <https://deepspeech.readthedocs.io/en/r0.9/>.

[7] PARK, D. S.; CHAN, W.; ZHANG, Y.; CHIU, C.-C.; ZOPH, B.; CUBUK, E. D.; LE, Q. V. Specaugment: A simple data augmentation method for automatic speech recognition. Interspeech, ISCA, v. 1st, 2019.

[8] KROSE, B.; SMAGT, P. van der. An introduction to Neural Network, 1993.

[9] PROGRAMMERSOUGHT. Activation function. 2018. 18 abr 2020. Disponível em: <https://www.programmersought.com/article/92601995466/>.

[10] SRIVASTAVA GEOFFREY HINTON, e. a. N. Dropout: A simple way to prevent neural networks from overfit-

ting. Journal of Machine Learning Research 15, v. 6, 2014.
[11] OLAH, C. Understanding LSTM Networks. 2015. 10 mai. de 2019. Disponível em: <https://colah.github.io/posts/2015-08-Understanding-LSTMs/>.
[12] EREMENKO, K. Deep Learning A-ZTM: Recurrent Neural Networks (RNN) - Module 3. 2018. Disponível em: <https://www.slideshare.net/KirillEremenko/deep-learning-az-recurrent-neural-networks-rnn-module-3>.
[13] LI, X.; PENG, L.; YAO, X.; CUI, S.; HU, Y.; YOU, C.; CHI, T. Long short-term memory neural network for air pollutant concentration predictions: Method development and evaluation. Environmental pollution (Barking, Essex: 1987), v. 231, 2017.
[14] MOZILLA. Common Voice. 2017. Disponível em: <https://commonvoice.mozilla.org/pt/datasets>.
[15] Bergstra, J.; Bengio, Y. Random Search for Hyper-Parameter Optimization. Journal of Machine Learning Research, 13, 2012.
[16] HEAFIELD, K. KenLM Language Model Toolkit. 2020. Disponível em: <https://kheafield.com/code/kenlm/>.
[17] OPTUNA. A hyperparameter optimization framework. 2019. Disponível em <https://github.com/optuna/optuna>.
[18] ARDILA, R.; BRANSON, M.; DAVIS, K.; HENRETTY, M.; KOHLER, M.; MEYER, J.; MORAIS, R.; SAUNDERS, L.; TYERS, F. M.; WEBER, G. Common Voice: A Massively-Multilingual Speech Corpus. 2020.