

O modelo Encoder-Decoder aplicado em irregularidades verbais do Português Brasileiro

Versão Corrigida

DISSERTAÇÃO DE MESTRADO

PROGRAMA DE MESTRADO EM SEMIÓTICA E LINGUÍSTICA GERAL

UNIVERSIDADE DE SÃO PAULO

Beatriz Albiero

Orientador: Prof. Dr. Marcelo Barra Ferreira

Departamento de Linguística FFLCH-USP

São Paulo

2019

Universidade de São Paulo
Faculdade de Filosofia, Letras e Ciências Humanas
Departamento de Linguística
Programa de Pós-Graduação em Linguística

O modelo Encoder-Decoder aplicado em irregularidades verbais do Português Brasileiro

Versão Corrigida

Beatriz Albiero

Dissertação apresentada ao Programa de Pós-Graduação em Linguística do Departamento de Linguística da Faculdade de Filosofia, Letras e Ciências Humanas da Universidade de São Paulo, para obtenção do título de Mestre em Letras.

Orientador: Prof. Dr. Marcelo Barra Ferreira

São Paulo
2019

Autorizo a reprodução e divulgação total ou parcial deste trabalho, por qualquer meio convencional ou eletrônico, para fins de estudo e pesquisa, desde que citada a fonte.

Catálogo na Publicação
Serviço de Biblioteca e Documentação
Faculdade de Filosofia, Letras e Ciências Humanas da Universidade de São Paulo

A335m Albiero, Beatriz
O modelo Encoder-Decoder aplicado em
irregularidades verbais do Português Brasileiro /
Beatriz Albiero ; orientador Marcelo Barra Ferreira.
- São Paulo, 2019.
92 f.

Dissertação (Mestrado)- Faculdade de Filosofia,
Letras e Ciências Humanas da Universidade de São
Paulo. Departamento de Linguística. Área de
concentração: Semiótica e Linguística Geral.

1. LINGUÍSTICA COMPUTACIONAL. 2. MORFOLOGIA
(LINGUÍSTICA). 3. APRENDIZADO DE MÁQUINA. I.
Ferreira, Marcelo Barra, orient. II. Título.

ENTREGA DO EXEMPLAR CORRIGIDO DA DISSERTAÇÃO/TESE

Termo de Ciência e Concordância do (a) orientador (a)

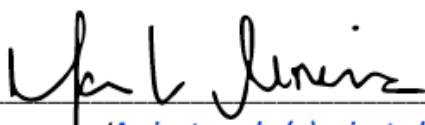
Nome do (a) aluno (a): Beatriz Albiero

Data da defesa: 29/11/2019

Nome do Prof. (a) orientador (a): Marcelo Barra Ferreira

Nos termos da legislação vigente, declaro **ESTAR CIENTE** do conteúdo deste **EXEMPLAR CORRIGIDO** elaborado em atenção às sugestões dos membros da comissão Julgadora na sessão de defesa do trabalho, manifestando-me **plenamente favorável** ao seu encaminhamento e publicação no **Portal Digital de Teses da USP**.

São Paulo, 20/01/2020



(Assinatura do (a) orientador (a))

Banca Examinadora

Prof. Dr. Marcelo Barra Ferreira (USP)
Presidente

Profa. Dra. Livy Maria Real Coelho
Membro titular

Prof. Dr. Marcos Fernando Lopes (USP)
Membro titular

Prof. Dr. Pablo Picasso Feliciano de Faria (UNICAMP)
Membro titular

*Em memória de Valéria.
Mãe, amiga, querida.*

*Ao meu pai André, que me deu a maior força
para começar, continuar e concluir este trabalho.*

Agradecimentos

Posso dizer que eu tive ao meu lado pessoas muito importantes que me ajudaram a passar pela experiência do mestrado de forma incrível. Foram muitos desafios e aprendizados, e sem as pessoas listadas a seguir, certamente esse processo não teria sido tão produtivo.

Primeiramente, agradeço à minha família, que me apoiou e incentivou nos meus estudos desde sempre. À minha mãe Valéria, que apesar de não ter tido a oportunidade de acompanhar essa fase da minha vida, observa de longe e se faz presente a cada passo que dou. Não sinto saudades, pois sinto que está comigo sempre. Ao meu pai André, meus avós Astrid e Norberto e aos meus irmãos Fábio e Juliana, pelo amor, carinho e cuidado.

Sou muito grata também ao meu orientador Marcelo Barra Ferreira. Tenho a sorte de poder dizer que a minha experiência no mestrado foi muito positiva e enriquecedora em um momento muito delicado para a pós-graduação brasileira, e certamente a sua orientação e parceria foi fundamental nesse processo.

Agradeço aos meus amigos e membros do grupo de estudos em linguística computacional da USP (o GLIC), que me ajudaram e inspiraram. Agradecimentos especiais ao meu colega Bruno Guide, que me ajudou diversas vezes com as minhas questões sobre a pós, muito antes da minha pesquisa começar. Também agradeço à Livy pelo seu olhar crítico e curioso sempre. A forma como todos do grupo amam estudar linguística é incrível. Também agradeço ao querido professor Marcos Lopes, cuja participação também foi fundamental para a minha formação acadêmica e para que o meu ingresso no mundo da linguística ocorresse de maneira leve e agradável.

Agradeço também aos meus colegas do Datalab, que me apoiaram e incentivaram durante a execução desta pesquisa. Sinto-me muito privilegiada por ter tido o apoio que eu tive no meu ambiente de trabalho e por todas as oportunidades que me foram concedidas para o meu desenvolvimento acadêmico. Agradecimento especial a Estêvão Uyrá, que contribuiu diretamente na execução desta pesquisa com sugestões valiosíssimas. Também sou muito grata ao professor Renato Vicente por ter me incluído em um time tão genial e criativo, além de todo o apoio na minha formação acadêmica e profissional. Também agradeço aos meus colegas Sami e Ricardo, pela compreensão, pelos conselhos e risadas.

Também não posso deixar de agradecer ao meu colega Felipe Salvatore, doutorando no IME-USP, que me auxiliou com sugestões essenciais para a base computacional desta pesquisa. Às professoras Esmeralda Vailati Negrão e Maria Cristina Altman que me apresentaram ao universo da linguística e me incentivaram a estudar esse assunto com mais

profundidade no mestrado.

Finalmente, a todos aqueles que me apoiaram nos momentos mais difíceis, em especial: Maria Teresa Lamberte, Carlos Joventino, Arthur Athayde, Ramon Vilarino, Thaís Dresch, Marina Nagata e Manu Bonfim.

Resumo

Albiero, B. **O modelo Encoder-Decoder aplicado em irregularidades verbais do Português Brasileiro**. 2019. Dissertação (Mestrado) - Faculdade de Filosofia e Ciências Humanas, Universidade de São Paulo, São Paulo, 2019.

Inspirada na controversa questão da aquisição de verbos irregulares na língua inglesa (Chomsky, N. & Halle (1968/1991), Pinker & Prince (1988), Albright, A. & Hayes (2003), Kirov & Cotterell (2018)), esta pesquisa tem como objetivo estudar a questão da flexão de verbos irregulares do Português Brasileiro sob a ótica do modelo computacional *Encoder-Decoder*. Para tanto, a tarefa proposta ao modelo era a de prever uma forma verbal flexionada dada uma forma primária (*Radical + Vogal Temática*). O escopo da pesquisa restringiu-se ao estudo do paradigma de 1ª Pessoa do Singular no Modo Indicativo e Tempo Presente. O modelo utilizado, por sua vez, é um modelo de caráter associativo que pertence ao grupo dos modelos de Redes Neurais Artificiais. Também, fez-se necessária a construção de um *corpus* linguístico composto pelo paradigma selecionado e em seguida transcrito em notação fonética específica para viabilizar a utilização do modelo escolhido. O *corpus* produzido é composto por 423 verbos que foram marcados como pertencendo às famílias de verbos regulares (51%) ou irregulares (49%). Ainda, dentro do escopo da família de verbos irregulares, foi possível identificar 15 subgrupos conforme a identificação de diferentes padrões de flexão. A partir da notação fonética utilizada, os verbos puderam ser associados a novas representações que englobavam informações relativas aos traços fonéticos presentes. Assim, o modelo proposto tenta prever as formas flexionadas a partir da identificação das relações fonéticas envolvidas durante o processo de flexão. O modelo apresentado foi submetido a múltiplos treinamentos e testes e apresentou uma acurácia média de 13.55%, mas chegou a acertar 17% em um dos experimentos. Considerando a segmentação entre verbos regulares e irregulares, o modelo performou melhor na classe dos regulares. Entretanto, considerando-se todas as 16 classes individualmente (15 irregulares + 1 regular), pôde-se observar que as duas primeiras classes em que o modelo performou melhor eram classes irregulares, deixando a classe regular como a terceira com os melhores resultados.

Palavras-chave: morfologia verbal, aprendizagem de máquina, conexionismo.

Abstract

Albiero, B. **The Encoder-Decoder Model Applied to Brazilian-Portuguese Verbal Irregularities**. 2019. Dissertação (Mestrado) - Faculdade de Filosofia e Ciências Humanas, Universidade de São Paulo, São Paulo, 2019.

Inspired by the controversial debate about the acquisition of irregular verbs in English-language (Chomsky, N. & Halle (1968/1991), Pinker & Prince (1988), Albright, A. & Hayes (2003), Kirov & Cotterell (2018)), this research aims to study the inflection process of irregular verbs in Portuguese through the perspective of the computational model *Encoder-Decoder*. To do this, we proposed the task of predicting an inflected verbal form given a primary form (*Stem + Thematic Vowel*). The scope of the research was restricted to the study of the singular first-person paradigm in the indicative mood and present tense. The model, in turn, is an associative model that belongs to the group of Artificial Neural Networks models. Also, it was necessary to construct a linguistic *corpus* composed by the chosen paradigm and then transcribe it into a specific phonetic notation to enable the usage of the chosen model. The resulting *corpus* consists of 423 verbs that were marked as belonging to either regular (51%) or irregular (49%) verb families. Moreover, within the scope of irregular verbs, it was possible to identify 15 subgroups through the identification of inflection patterns. Through the phonetic notation provided, verbs could be associated with new representations that included information related to the phonetic features. Thus, the proposed model attempts to predict inflected forms by identifying the involved phonetic relationships during the inflection process. The model was submitted to multiple trainings and tests and presented an average accuracy of 13.55%, but it got to 17% in one of the experiments. Considering the segmentation between regular and irregular verbs, the model performed better among the regular class. However, considering all 16 classes individually (15 irregular + 1 regular), it was observed that the first two classes in which the model performed best were irregular classes, leaving the regular class with the third place.

Key-words: verbal morphology, machine learning, connectionism.

Sumário

	i
Introdução	1
1 Aprendendo Verbos Irregulares	3
1.1 Contextualização	3
1.2 Motivação	9
1.2.1 Motivação no campo da linguística	9
1.2.2 Motivação no campo da ciência da computação	12
1.2.3 Arquitetura Estado-da-Arte	13
1.2.4 Delimitação de escopo	14
2 Pré-Processamento dos Verbos para Redes Neurais	16
2.1 Pré-processamento de Rumelhart & McClelland	17
2.2 Apresentação do corpus	19
2.2.1 Types x Tokens	20
2.3 Pré-processamento para o Encoder-Decoder	21
2.3.1 Os inputs do modelo Encoder-Decoder	25
3 Modelos de Redes Neurais	27
3.1 Aprendizado de máquina	27
3.2 Introdução teórica a Redes Neurais	31
3.3 Treinamento	34
3.4 Uma outra arquitetura	35
3.4.1 Modelo de Linguagem	35
3.4.2 Redes Neurais Recorrentes	37
3.4.3 Modelos de Linguagem com RNR's	38
3.5 Conclusão do capítulo	40
4 Encoder-Decoder	41
4.1 Introdução ao modelo Encoder-Decoder	41
4.2 Encoder-Decoder e flexão verbal	42
4.2.1 A rede Encoder desenvolvida	43

4.2.2	A rede Decoder desenvolvida	43
4.2.3	Pós-processamento	44
5	O Experimento	46
5.1	Hiperparâmetros	46
5.2	Metodologia para avaliação	47
5.2.1	Métrica de avaliação	48
5.3	Resultados	48
5.3.1	Acurácia em cada classe	49
5.3.2	Acurácia e comprimento médio dos verbos nas classes	51
5.4	Outros erros relevantes	52
5.5	Discussão	53
6	Conclusão	56
	Bibliografia	58
A	Tabela de Transcrição Fonética de Rumelhart e McClelland	62
B	Corpus	64
C	Resultados das Decodificações	75

SUMÁRIO

Introdução

Inspirada no controverso tema da aquisição de verbos irregulares na língua inglesa (Pinker (1999), Chomsky, N. & Halle (1968/1991), Pinker & Prince (1988), Rumelhart & McClelland (1986)), esta pesquisa tem como objetivo estudar o processo de flexão de verbos irregulares do Português Brasileiro fazendo uso de um modelo computacional conhecido como *Encoder-Decoder* (Bahdanau, D. (2014)).

A controvérsia em torno do processo de aprendizado de verbos irregulares teve início na década de 60, tendo sido explorada por pesquisadores de diversas áreas (linguistas, psicólogos, neurocientistas, cientistas da computação, entre outros). No Cap. 1 deste trabalho, serão apresentadas as origens e motivações da discussão, bem como os principais pesquisadores envolvidos e hipóteses levantadas. Além disso, será introduzida uma categoria de modelos computacionais associativos conhecida como Rede Neural Artificial, do qual o modelo *Encoder-Decoder* faz parte. Em seguida, na primeira parte da seção de motivação (Seç. 1.2) serão apontadas algumas especificidades gramaticais da língua portuguesa que podem dificultar o processo de aprendizado da flexão irregular. Na segunda parte da seção, serão apresentados alguns trabalhos computacionais que se seguiram nos anos subsequentes em resposta à discussão estabelecida.

Após a exposição dos diferentes trabalhos desenvolvidos, ficará evidente a razão pela qual tal tema chamou a atenção de tantas áreas distintas. Ademais, veremos que o problema da flexão de verbos irregulares tornou-se um desafio computacional para além dos problemas linguísticos ou cognitivos em questão, de modo que muitas pesquisas subsequentes acabaram se distanciando do debate linguístico e focando nos aspectos matemáticos que viabilizariam o aprendizado artificial. A presente pesquisa também dará continuidade a esse aspecto computacional do problema e não terá como objetivo assumir qualquer posição em torno dos aspectos cognitivos do debate. Desse modo, avaliaremos o uso do modelo *Encoder-Decoder* a partir de um ponto de vista prático de acordo com a sua performance na tarefa proposta e em comparação a outros modelos computacionais já apresentados em outras pesquisas.

Para a realização do objetivo proposto, primeiramente foi necessário o desenvolvimento de um corpus linguístico específico para a tarefa. O corpus resultante apresenta-se completo no Apêndice (B), mas será discutido em detalhes no Cap. 2, na Seç 2.2. Ainda nesse capítulo, veremos a importância da aplicação de tratamentos de pré-processamento adequados nas unidades do corpus para a viabilização do aprendizado pretendido. Para tanto, revisitaremos um algoritmo de pré-processamento proposto em uma pesquisa anterior, re-

alizada por Rumelhart & McClelland (1986). Em seguida, apresentaremos o algoritmo de pré-processamento desenvolvido nesta pesquisa.

O Capítulo 3 introduzirá conceitos básicos sobre o tema de *Aprendizado de Máquina* e em seguida apresentará uma introdução aos modelos de Redes Neurais Artificiais. Além disso, abordaremos os conceitos de *Modelo de Linguagem* e de arquiteturas de *Redes Neurais Recorrentes* - assuntos imprescindíveis para o entendimento do modelo em destaque desta pesquisa, o *Encoder-Decoder*. Feitas as devidas introduções, estaremos prontos para a apresentação do modelo *Encoder-Decoder* no Capítulo 4.

O Capítulo 5, por sua vez, apresentará e discutirá os resultados obtidos. Nele veremos as configurações escolhidas para a definição do modelo e as métricas utilizadas para a avaliação do mesmo. Analisaremos também os diferentes erros observados e procuraremos por possíveis explicações para os resultados obtidos.

Para concluir, o Capítulo 6 exibirá um resumo dos assuntos tratados nesta pesquisa e também apontará sugestões para pesquisas futuras sobre o assunto.

Capítulo 1

Aprendendo Verbos Irregulares

Este capítulo será dedicado primeiramente a uma apresentação a respeito da origem da discussão sobre o processo de aprendizado de flexão de verbos irregulares. Assim, construiremos uma visão geral do problema e apresentaremos as principais hipóteses e pesquisas envolvidas. Em seguida, discutiremos as motivações que levaram à produção desta pesquisa. Nesse sentido, veremos que, por um lado, há um interesse de caráter linguístico, motivado por uma maior complexidade da língua portuguesa se comparada ao inglês. Por outro, veremos também que há um interesse de cunho computacional, motivado pelo surgimento de novos recursos de modelagem. Por último, veremos uma seção destinada à delimitação do escopo da pesquisa.

1.1 Contextualização

A questão do aprendizado infantil, referente ao processo de flexão de verbos irregulares na língua inglesa, está certamente entre um dos temas de debate mais controversos do campo da linguística (Chomsky, N. & Halle (1968/1991), Pinker (1999), Pinker & Prince (1988), Albright, A. & Hayes (2003), Kirov & Cotterell (2018)). O cerne do debate está na exata caracterização dos mecanismos que possibilitam que um falante seja capaz de relacionar um verbo na forma não flexionada (*walk*, por exemplo) à sua forma flexionada no *Simple Past* (*walked*).

Os verbos no tempo passado do inglês podem ser subdivididos em uma variedade de famílias. Um primeiro grupo é a forma *regular*, cuja forma corresponde à aplicação da regra *radical + ed* (como no exemplo visto do verbo *to walk*). Dentre os verbos irregulares, estes podem ser considerados supletivos, ou seja, possuem um processo de flexão único e sem regra aparente, como por exemplo *go* → *went*, ou podem se conglomerar seguindo padrões fonéticos de flexão similares (Nelson (2010)):

1. blow – blew, grow – grew, know – knew, throw – threw
2. bear – bore, swear – swore, tear – tore, wear – wore

3. drink – drank, shrink – shrank, sink – sank, stink – stank

É possível pensar que o aprendizado de tais padrões dependeria de uma memorização caso a caso. No entanto, a pesquisa de Bybee & Moder (1983) mostra um estudo psicolinguístico em que indivíduos são apresentados a diversos verbos inventados (hipoteticamente em uma forma não flexionada). A pesquisa revelou que, ao invés de aplicarem sistematicamente a regra regular (verbo + *ed*), os indivíduos apresentaram tendências à alocação de alguns verbos em alguns subgrupos irregulares. Por exemplo, para o verbo inventado “*spling*”, a maioria dos indivíduos optou pela forma “*splang*” ou “*splung*”. Este exemplo contradiz a ideia de que os falantes poderiam estar apenas reproduzindo formas memorizadas e sugere que eles estejam ativamente identificando padrões. Além disso, possuem uma intuição natural sobre a adequabilidade da alocação de um verbo a um grupo de verbos ou a outro.

A partir do exemplo dado, é razoável deduzir que a motivação por de trás de tais tendências ocorra a partir das similaridades entre as unidades fonéticas dos verbos inventados e os verbos reais que já apresentam uma flexão de caráter irregular. Entretanto, as circunstâncias que levam à aquisição dessa *intuição* linguística são indeterminadas. Por um lado, faz sentido dizer que para que um ser humano seja capaz de introduzir-se ao mundo dos falantes, é necessário que ele seja dotado de algumas pré-disposições para tal, caso contrário seria possível observar ou ensinar essa forma de comunicação para outras espécies. Em contrapartida, estudos mostram que crianças privadas do contato com uma sociedade falante se tornam permanentemente incapazes de dominar integralmente a gramática de uma língua (Pinker (1994/2007)), o que nos leva a concluir que a experiência das crianças com a sociedade, assim como as suas próprias pré-disposições genéticas são ambas parcialmente responsáveis pelo processo de desenvolvimento da linguagem. A dificuldade, está portanto, na tentativa de se quantificar, delimitar e apontar os conhecimentos adquiridos a partir do contato cultural, bem como os conhecimentos linguísticos ditos *inatos*. É, portanto, em torno desta questão que tem início o debate a respeito do aprendizado dos verbos irregulares da língua inglesa.

De um lado do debate, encontra-se a teoria da Fonologia Gerativa de Chomsky e Halle (1968/1991). Nesta teoria, os indivíduos seriam portadores de um dispositivo de aquisição de linguagem (*LAD* - Language Acquisition Device) responsável pela *formulação* e *manipulação* de estruturas fonológicas abstratas em um sistema de regras. De modo simplificado, a teoria propõe que o falante seja capaz de identificar e formular regras intuitivamente para dar conta do aprendizado das formas irregulares da língua. Um exemplo disso é a família dos verbos terminados em “-ind”.

bind – bound, find – found, grind – ground, wind – wound

Vemos que, de modo simplificado, pode-se propor uma regra como:

$$a: \rightarrow a: / \text{ X } __\text{nd}] + \text{past}$$

A regra proposta sugere que o segmento [ai] se transforma em [aʊ] quando terminado em [nd] e flexionado para o passado. O símbolo ____ representa o local onde ocorre tal transformação e **X** representa uma unidade fonológica arbitrária.

Em outras palavras, pode-se dizer que o conhecimento dito *inato* defendido por Chomsky e Halle refere-se a uma certa capacidade cognitiva de formulação de regras a partir da identificação de alguns elementos fundamentais (como por exemplo os elementos apontados na regra proposta). Uma estrutura como essa permitiria ao falante construir generalizações e, eventualmente, abstrair as regras fonológicas de sua língua.

Do outro lado do debate, os pesquisadores Rumelhart & McClelland (1986) confrontam a teoria anterior ao argumentar que comportamentos de caráter regrado podem ser reproduzidos por mecanismos que não dependam de nenhuma manipulação simbólica. Ao invés disso, os pesquisadores sugerem que os mecanismos envolvidos no processo de flexão verbal possam ser construídos de tal forma que a sua performance possa ser descrita através de regras, mas que as regras em si não estejam representadas explicitamente em nenhuma parte do processo. Para sustentar essa ideia, Rumelhart & McClelland (1986) apresentam um modelo computacional baseado em padrões associativos que não fazem uso de construções com regras desse tipo. Posteriormente, o modelo construído foi fundamental para o surgimento de uma nova escola dentro das ciências cognitivas: o conexionismo.

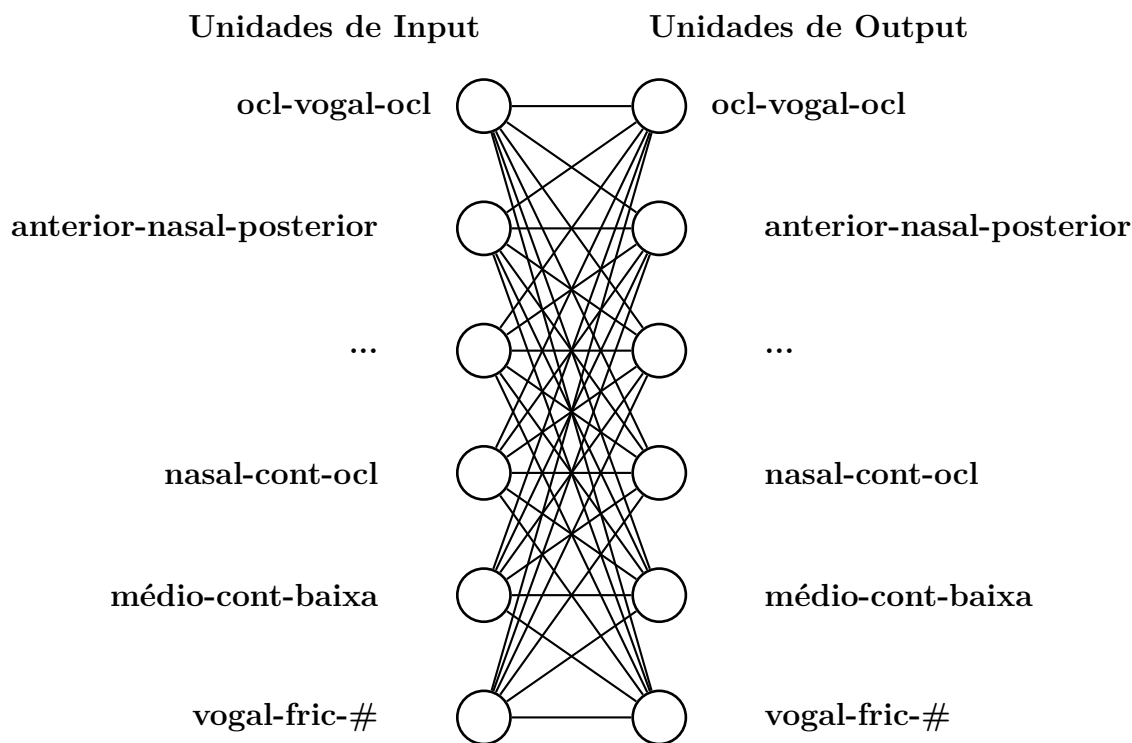


Figura 1.1: Esquema do Modelo Apresentado pelos Pesquisadores Rumelhart e McClelland

O modelo desenvolvido foi criado por analogia à estrutura em que se relacionam os

neurônios no cérebro. Ele é composto basicamente por uma rede artificial de nódulos interconectados paralelamente (Fig. 1.1).

A primeira camada de nódulos do modelo é responsável por receber os dados de entrada (os *inputs*), que são os dados referentes aos traços fonéticos distintivos que caracterizam os sons de um verbo não flexionado. Traços fonéticos podem ser caracterizados como propriedades distintivas das unidades fônicas (Seara, I. C. (2015)). Tais propriedades podem ser baseadas em critérios acústicos, articulatórios ou perceptuais. Na figura, cada nódulo é apresentado ao lado de uma sequência de três traços. O primeiro nódulo, por exemplo, refere-se à sequência **ocl-vogal-ocl**. Nesse caso, **ocl** (uma abreviação para *oclusiva*) indica uma propriedade comum entre algumas consoantes, referente à interrupção do fluxo de ar (como no fone [k], por exemplo). Na figura temos ainda **fric** para fricativas, **vogal** para vogais, **nasal** para nasalidade, locais da execução (**anterior** e **posterior**), traços de corpo da língua (**média**, **baixa**), entre outros. Ainda sobre a camada de *input*, é possível observar que cada nódulo é representado por uma tríade de traços fonéticos. Esta foi uma solução encontrada pelos autores para realizar o mapeamento entre os traços dos verbos da forma não flexionada para o *Past Simple*. Os *inputs* são estruturados dessa forma para contornar a dificuldade de inserção de dados de natureza sequencial e de tamanho variável (como é o caso de um verbo - composto por uma sequência de sons). Cada tríade é uma associação de três traços, cada um referente a um fone. Por exemplo, para o verbo *came* (transcrito em forma fonética pelos autores como /kAm/), temos que cada um dos fones possui múltiplos traços. O fone [k], por exemplo, é uma consoante oclusiva, surda e anterior. Os fones subsequentes também são constituídos a partir de seus respectivos traços fonéticos, de modo que podemos representar o verbo como uma sequência de listas de traços fonéticos (Tabela 1.1). O tema dos traços fonéticos utilizados, bem como todo o esquema de pré-processamento utilizado pelos autores Rumelhart & McClelland (1986) será abordado em maior profundidade no Cap. 2.

k	A	m
surda	longa	nasal
interrompida	vogal	interrompida
anterior	baixa	posterior
consoante	posterior	consoante

Tabela 1.1: *Tríades de Traços Fonéticos Utilizados nos Inputs do modelo de Rumelhart e McClelland*

Voltando à Fig. 1.1, vemos após a camada de *input* uma rede de conexões. Cada conexão, por sua vez, possui um *peso*. Esses pesos irão funcionar como uma espécie de filtro dos *inputs*, fazendo com que pesos com valores maiores passem as informações adiante com mais efeito (ou força), e os pesos menores, com menos. A segunda camada de nódulos da Fig.1.1 é uma camada de resposta (conhecida como camada de *output*) que tem como objetivo retornar

dados referentes aos traços que caracterizam os sons do mesmo verbo fornecido no *input*, porém no tempo passado.

Durante o processo de aprendizado do modelo, os dados de saída da camada de *output* deverão ser então comparados à forma correta do verbo no tempo passado, através de uma espécie de gabarito, usualmente conhecido como *alvo* ou também *target* (Fig.1.2). Feita essa comparação, é possível alterar a rede de conexões entre as camadas de *input* e *output* de modo a reforçar (ou enfraquecer) os pesos das mesmas para atingir o aprendizado proposto. Antes da primeira comparação, a rede é inicializada com pesos aleatórios. Conforme o número de comparações aumenta, a tendência é que os pesos sejam pouco a pouco calibrados para o modelo atingir o seu objetivo, que nesse caso, é aprender os padrões de flexão dos verbos.

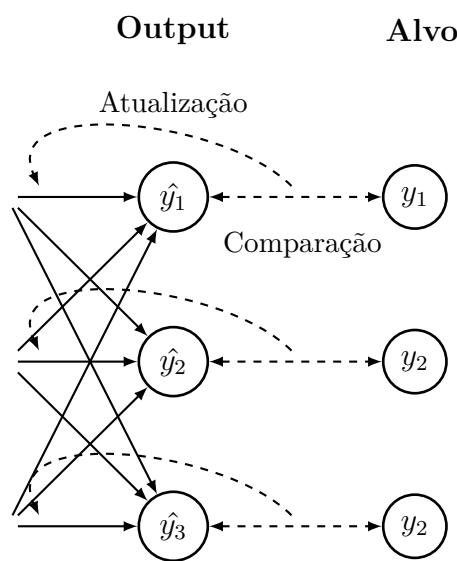


Figura 1.2: Comparações entre o Output e o Target

Para realizar o treinamento, Rumelhart & McClelland (1986) introduzem 420 verbos no modelo repetidamente (200 vezes cada um, 84.000 inserções no total). Após o treinamento, o modelo foi capaz de prever corretamente todos os 420 verbos que já haviam sido inseridos. Além disso, em um novo grupo com 86 verbos desconhecidos, acertou cerca de 2/3 do conjunto. Dentre os novos verbos irregulares apresentados, cometeu alguns erros interessantes de *super-regularização* (como *caught* (ao invés de *caught*) e *digged* (ao invés de *dug*)). Esses erros foram observados em 11 dos 14 verbos irregulares testados (Prasada, S. & Pinker (1993)).

Além desses resultados, Rumelhart & McClelland (1986) relatam que o processo de aprendizado do modelo apresentou um fenômeno interessante, reproduzindo um desempenho similar a comportamentos observáveis em crianças durante a fase de aquisição: a Curva de Desenvolvimento em U (U-shaped Development, Marcus, G. (1992)). A Curva de Desenvolvimento em U basicamente se refere a um processo de aprendizado que ocorre em três estágios:

- (i) inicialmente, crianças acertam a flexão de verbos irregulares (*come*→*came*);
- (ii) em seguida passam por um processo de *super-regularização* (em que produzem formas como *comed*), e passam a errar flexões que antes acertavam;
- (iii) por fim, elas passam a reproduzir corretamente tanto os verbos regulares quanto irregulares.

Rumelhart & McClelland (1986) descrevem como foi possível observar tal comportamento também no modelo computacional desenvolvido. Na fase inicial do processo de treinamento, o modelo foi alimentado com uma quantidade pequena de verbos, como: *come*, *get*, *give*, *look*, *take*, *go*, *have*, *live* e *feel*. A performance do modelo foi compatível com o primeiro estágio da curva, ou seja, para esses verbos foi capaz de identificar corretamente a forma correspondente no *Simple Past*. Em um segundo momento, o modelo foi alimentado com uma quantidade muito maior de verbos. Nesse estágio é possível verificar que o modelo está passando por um processo de regularização sistemática dos verbos. Ele produziu resultados como: *breaked*, *comed*, *gived*; e também combinações entre padrões regulares e irregulares (ex. *gaved*). Após uma série de muitas inserções repetidas, o modelo finalmente foi capaz de responder corretamente a uma quantidade maior de verbos, assim como no último estágio do processo de aprendizado natural.

Os resultados de Rumelhart & McClelland (1986) causaram um considerável impacto na comunidade científica da época. Muitos pesquisadores viam o novo modelo como uma completa mudança de paradigma, não apenas na linguística, mas também como uma nova forma de se estudar aprendizado em geral (Schneider (1987)).

Apesar disso, Pinker & Prince (1988) dão continuidade ao debate ao apontar uma série de questões pertinentes que o modelo falhou em explicar. Primeiramente, como o modelo recebe apenas uma representação fonética do verbo como *input*, ele é incapaz de gerar duas respostas diferentes para verbos com sonoridade idêntica (por exemplo *break*→*broke* e *brake*→*braked*). Para realizar essas previsões corretamente, o modelo precisaria de um módulo adicional para distinguir entre as duas palavras, o que o descaracterizaria como modelo puramente associativo. Em segundo lugar, o modelo é extremamente dependente dos padrões observados durante o treinamento, tendo uma capacidade baixa para generalizações. Pinker (1999) comenta que o modelo ficou mudo quando alimentado com os verbos *jump*, *pump*, *warm*, *trail* e *glare* (que dispõem de uma sonoridade razoavelmente incomum). Além disso, o modelo apresentou alguns resultados completamente distorcidos, como: *squat* – *squakt*, *tour* – *toueder* e *mail* – *membled*; associações inaceitáveis para qualquer falante nativo.

Com relação ao padrão de aprendizado observado (a Curva de Desenvolvimento em U), Pinker (1999) explica que esse comportamento foi causado segundo a forma em que os verbos foram inseridos no modelo durante o treinamento: Rumelhart & McClelland realizaram o treinamento em partes e controlando a quantidade de repetições de cada lote de verbos. Na primeira parte do treinamento, selecionaram alguns verbos de alta frequência na língua inglesa (muitos deles irregulares), reproduzindo o estágio (i) da curva. Em seguida, treinaram

o modelo com esses verbos, reintroduzindo-os múltiplas vezes até que o modelo conseguisse atingir um desempenho razoável nesses verbos. Na sequência introduziram uma quantidade maior de verbos, sendo estes menos frequentes que os anteriores, mas em sua maioria regulares. Dessa forma, o modelo começou a se ajustar para aplicar a regra regular e assim foi possível observar os estágios (ii) e (iii) da curva. Ainda, segundo Plunkett, K. & Marchman (1991), os estágios de desenvolvimento (i), (ii) e (iii) podem ser considerados parte de um comportamento *macro U-shape*, mas ainda é possível observar a ocorrência de um comportamento *micro U-shape*. Plunkett, K. & Marchman (1991) observam que a reprodução dos verbos irregulares na fala espontânea de crianças aprendizes oscila bastante entre flexões corretas e *super-regularizadas*. Eles também notam que estas oscilações ocorrem em proporções diferentes para cada verbo e que crianças raramente “*irregularizam*” verbos regulares (como *ping*→*pang*), e raramente misturam uma forma irregular com uma regular (fato que ocorreu durante o aprendizado do modelo com *gaved*).

Para concluir, Pinker & Prince (1988) apresentam a formulação de uma nova hipótese para tal questão: uma teoria híbrida na qual a fonologia gerativa se aplicaria ao processo de flexão regular e um mecanismo associativo se aplicaria ao processo de flexão irregular. Os pesquisadores propõem que as formas regulares sejam computadas a partir de um mecanismo que deve abstrair o radical do verbo e combiná-lo ao sufixo –ed. Tal mecanismo pode ser aplicado a qualquer palavra, em um processo independente da memória. As formas irregulares, por sua vez, passam por um processo diferente: verbos irregulares precisam passar por um processo de memorização, havendo não somente a associação entre uma forma verbal e outra mas também entre as propriedades (traços fonéticos, rima, radical, núcleo, etc.) das mesmas, parecido com o que foi proposto por Rumelhart e McClelland.

1.2 Motivação

Uma vez que uma contextualização acerca do problema já foi apresentada, focaremos agora nas questões que motivaram o desenvolvimento desta pesquisa. Assim, esta seção será dividida em duas partes: (i) motivação no campo da linguística, e (ii) motivação no campo da ciência da computação.

1.2.1 Motivação no campo da linguística

A morfologia verbal da língua inglesa é bastante simples, se comparada à portuguesa. Em primeiro lugar, os verbos do português se distribuem em três classes mólicas (*conjugações*), sendo cada uma destas definida a partir de uma *vogal temática* (/a/, /e/ e /i/). Dado um verbo em sua forma infinitiva, por exemplo “*amar*”, a vogal temática é a aquela que se encontra entre o morfema lexical do verbo (o radical) e a desinência de infinitivo *r*.

$$\begin{aligned} &\text{Am} + \text{a} + \text{r} \\ &\text{Radical} + \text{VT} + \text{r} \end{aligned}$$

Com isto, os três possíveis tipos de conjugação são: 1^a - ar (amar, brigar), 2^a - er (beber, comer) e 3^a (rir, descobrir). Na língua inglesa, essa distinção não existe.

Uma criança em processo de aquisição de linguagem no sistema do português brasileiro passa por muitos desafios. Uma parte do processo é justamente perceber a relação entre a vogal temática e as possíveis conjugações verbais regulares. Nesse processo, não é incomum observarmos o surgimento de trocas de conjugação. Wuerges (2014) apresenta dados linguísticos produzidos por crianças com diversas destas trocas. Alguns exemplos de seu trabalho podem ser observados na Tab. 1.2.

Verbo	Observado	Correto	Troca
botar	“eu boti”	botei	1 ^a com 2 ^a ou 3 ^a
comer	“eu comei”	comi	2 ^a com 1 ^a
jantar	“eu janti”	jantei	1 ^a com 2 ^a ou 3 ^a

Tabela 1.2: *Exemplos de Trocas de Conjugação Durante o Processo de Aquisição Verbal*

As formas verbais irregulares apresentam-se como uma dificuldade adicional nesse processo para as crianças falantes da língua portuguesa. Wuerges (2014) também aponta exemplos observados de *regularização* de verbos irregulares: “eu *consego*” (regularização do verbo conseguir) e “eu *podo*” (regularização do verbo poder).

Um verbo é dito irregular se apresentar alterações no radical (em relação ao radical da forma infinitiva) e/ou no sufixo flexional (em relação ao padrão regular imposto por cada conjugação) (Wuerges (2014)). Os sufixos flexionais (SF) são os segmentos acrescentados após o radical do verbo. Eles podem ser divididos em dois tipos: (i) sufixo modo-temporal (SMT) e (ii) sufixo número-pessoal (SNP). Para o verbo “gostávamos”, por exemplo, considera-se o segmento */gost/* como o radical do verbo, */av/* como o sufixo modo-temporal, que neste caso marca simultaneamente o modo indicativo e tempo pretérito imperfeito; e *amos* como o sufixo número-pessoal indicando primeira pessoa do plural (nós).

Seguindo a definição proposta, é necessário reforçar que o interesse deste estudo está em capturar irregularidades no nível fonético, portanto verbos como: “gosto”, “boto” e “coloco”, cuja ortografia apresenta o padrão regular, serão classificados como irregulares. Isso acontecerá em razão das transformações fonéticas que ocorrem na fala e que não são capturadas na escrita. No caso de “gosto”, por exemplo, vemos que a primeira vogal “/o/”, na verdade tem som de /ɔ/ (ó). Assim, a Tabela 1.3 exhibe alguns exemplos das classificações realizadas.

Verbo Infinitivo	Verbo Flexionado	Classificação
Falar	Falo	Regular
Gostar	Gosto	Irregular
Testar	Testo	Irregular
Ansiar	Anseio	Irregular
Pedir	Peço	Irregular
Mentir	Minto	Irregular
Por	Ponho	Irregular

Tabela 1.3: *Exemplos de Classificações de Verbos Quanto a Presença de Irregularidades*

Uma observação sobre a disposição das irregularidades presentes no português brasileiro (levando em consideração apenas a 1ª Pessoa do Singular (tempo Presente e modo Indicativo) nos permite observar algumas regularidades (padrões) dentre os verbos irregulares:

Bobear – Bobeio, Bloquear – Bloqueio, Chatear – Chateio, Clarear – Clareio, Golpear – Golpeio

Agredir – Agrido, Conseguir – Consigo, Inserir – Insiro, Perseguir – Persigo, Preferir – Prefiro, Proferir – Profiro, Repetir – Repito, Servir – Sirvo, Vestir – Visto

Cobrir – Cubro, Dormir – Durmo, Engolir – Engulo

Al[e]gar – Al[ε]go, C[e]gar – C[ε]go, Compl[e]tar – Compl[ε]to, Col[e]tar – Col[ε]to, Entr[e]gar – Entr[ε]go, Pr[e]gar – Pr[ε]g,

Ad[o]rar – Ad[ɔ]ro, Ad[o]tar – Ad[ɔ]to, B[o]tar – B[ɔ]to, C[o]lar – C[ɔ]lo, F[o]car – F[ɔ]co, M[o]rar – M[ɔ]ro, S[o]ltar – S[ɔ]lto, S[o]lar – S[ɔ]lo, T[o]car – T[ɔ]co, M[o]strar – M[ɔ]stro

Mentir – Minto, Sentir – Sinto

Os padrões observados a partir da exposição de algumas classes irregulares, permitem, assim como no inglês, a proposição de fórmulas, ou regras fonéticas, que explicam as flexões realizadas em cada classe. É possível notar, por exemplo, que um verbo da mesma família de *conseguir* segue a regra:

$$e \rightarrow i / _C]ir$$

A regra proposta indica que /e/ se transforma em /i/ quando em um contexto de terceira conjugação (ir). No caso, C indica uma consoante qualquer.

Os padrões encontrados sugerem não somente a possibilidade de elaboração de regras, como também a possibilidade do desenvolvimento de redes capazes de capturar tais dependências.

1.2.2 Motivação no campo da ciência da computação

Desde a apresentação da pesquisa de Rumelhart & McClelland (1986), o modelo associativo utilizado pelos autores já passou por diversos avanços. Na realidade, esse tipo de modelagem hoje é chamado de Rede Neural Artificial e passou a ser utilizado em uma variedade de tarefas computacionais, como classificação de imagens, classificação de texto, tradução automática, agentes conversacionais, entre outros.

Nos últimos anos, o poder computacional aumentou muito. O desenvolvimento dos *hardwares* possibilita que hoje sejam realizadas muito mais computações e com muito mais velocidade do que na década de 80. Com isso, arquiteturas mais complexas puderam ser exploradas e desenvolvidas. A título de exemplo, é possível acrescentar camadas intermediárias entre as camadas de *input* e *output*. Redes construídas dessa forma possibilitam que as informações de entrada sejam distribuídas ao longo de mais conexões e com isso, potencializem o aprendizado. Isso acontece pois o tipo de arquitetura sem camadas intermediárias consegue aproximar apenas um tipo de função, as funções lineares. Ao aumentar o número de camadas intermediárias, é possível ampliar o universo de soluções para a resolução de problemas mais complexos (ver Goodfellow, I. & Bengio (2016) para explicações mais detalhadas sobre as camadas intermediárias). Esse tipo de modelagem que segue um fluxo com um sentido único (do *input* ao *output*) é chamado de *Feedforward* (FFD). Entretanto, existem muitos tipos de arquiteturas cujos fluxos não seguem essa configuração. Nesse âmbito, um tipo de arquitetura que ficou muito conhecido é o *Convolutacional* (*Redes Neurais Convolucionais* (RNC's)) (bastante utilizado na área de visão computacional (Krizhevsky *et al.* (2012), por exemplo)). No campo de tarefas linguísticas, as redes do tipo *Recorrente* (*Redes Neurais Recorrentes* (RNR's)) são bastante utilizadas, uma vez que a arquitetura viabiliza a inserção de dados sequenciais (Liu, P. & Qiu (2016), por exemplo). O tema dos modelos de Redes Neurais, em especial a arquitetura de RNR's, será abordado com mais detalhes no Capítulo 3.

No que diz respeito à questão do aprendizado dos verbos irregulares do inglês, uma série de novos experimentos se sucederam após as críticas de (Pinker & Prince (1988)). (Plunkett, K. & Marchman (1991), (1993)) simplificam a questão ao considerar apenas verbos de tamanho fixo (3 sílabas) e abordam o problema fazendo uso de uma arquitetura com adição de camadas intermediárias (*Multi-Layered Perceptron* - MLP). Outros trabalhos transformaram a questão em um problema de *classificação*, desse modo o modelo não teria mais como objetivo encontrar uma forma flexionada. Ao invés disso, teriam um conjunto finito e pré determinado de formas possíveis. Nakisa & Hahn (1996), por exemplo, classificam os plurais dos substantivos da língua alemã. Plunkett, K. & Nakisa (1997) atacam o mesmo problema, porém para a língua Árabe.

Westermann, G. & Goebel (1997) apresentam um modelo construído para mapear verbos não flexionados da língua alemã para a forma no particípio. O modelo apresentado é capaz de lidar com dados com sequências de tamanhos variáveis e utiliza uma arquitetura baseada em RNR's. Entretanto, o modelo foi construído a partir de um mecanismo de rota dupla, de modo que verbos irregulares passavam por uma rota específica de memorização e verbos regulares por outra rota, com a aplicação da regra. Apesar disso, o desempenho do modelo deixou a desejar. Alguns experimentos diferentes foram realizados, sendo que um deles consistia no treinamento de grupos de verbos isoladamente. Para tal, os verbos regulares ficaram em um grupo e os irregulares foram divididos em outros dois. Nesse caso, a porcentagem de acerto do modelo chegou a quase 100% para os grupos treinados isoladamente. Entretanto, ao misturar os verbos em um único treinamento, a porcentagem de acerto se estabilizou em torno de 60 a 70%.

No âmbito da construção de modelos não associativos, ou seja, baseados em regras, Albright, A. & Hayes (2003) apresentam o modelo *Minimal Generalization Learner (MGL)*, cuja implementação se aproxima muito da teoria proposta por Pinker & Prince (1988). O modelo MGL se baseia na descoberta e atribuição de pesos a regras pré-estabelecidas para as transformações irregulares. Falaremos mais sobre os resultados desse modelo na seção de discussão dos resultados (Seq. 5.5).

1.2.3 Arquitetura Estado-da-Arte

Bahdanau, D. (2014) e Sutskever (2014) apresentam um novo tipo de arquitetura de Redes Neurais construído para mapear duas sequências de tamanhos variáveis. A nova arquitetura, conhecida como *Encoder-Decoder*, ou também *Seq2Seq*, é reconhecida especialmente pelo seu bom desempenho em tarefas linguísticas, em especial no ramo da tradução automática (Wu (2016)). Essa arquitetura consiste na concatenação de duas RNR's. A primeira rede, *Encoder*, lê cada símbolo de uma sequência de entrada (por exemplo, uma palavra em inglês) e gera como resposta uma representação abstrata da palavra lida. A segunda rede, *Decoder*, recebe como entrada a representação devolvida pelo *Encoder* e tem como objetivo produzir uma outra sequência alvo correspondente (sendo nesse caso de uma tarefa tradução, a palavra traduzida para outra língua). A arquitetura do modelo *Encoder-Decoder* será abordada em maiores detalhes no Capítulo 4.

Na tarefa do aprendizado dos verbos irregulares, Faruqui (2015) elabora a questão a nível de *caracteres* (letras), ou seja, não utiliza traços fonéticos como *inputs* no modelo *Encoder-Decoder*. Kann, K. & Schütze (2016) utilizam rótulos morfológicos (marcações de particípio, gerúndio, etc) como *inputs* no mesmo modelo. Cotterell *et al.* (2016) atingem performance *estado-da-arte* em um problema de flexão morfológica, postulado em uma tarefa compartilhada (*SIGMORPHON Shared Task* (<http://www.sigmorphon.org/>)). Neste problema foram introduzidos conjuntos de dados morfológicos para 10 idiomas (Espanhol, Alemão, Finlandês, Russo, Turco, Georgiano, Navajo, Árabe e Húngaro) com diversas características tipológi-

cas. Para a tarefa de geração de flexões de lemas, o sistema de Cotterell *et al.* (2016) obteve uma média de acurácia de 95,56% em todos os idiomas, variando de Maltês (88,99%) para húngaro (99,30%).

Assim, temos como motivação para esta pesquisa o uso do modelo *Encoder-Decoder* para o aprendizado de flexão verbal. O modelo *Encoder-Decoder*, apesar de ter apresentado resultados promissores, ainda não foi aplicado a nível exclusivamente fonético (como em (Rumelhart & McClelland (1986))), uma vez que as pesquisas anteriores exibem resultados a nível de caracteres e com rótulos morfológicos. Além disso, vemos também que o modelo nunca foi testado na língua portuguesa.

1.2.4 Delimitação de escopo

Em comparação ao inglês, é possível argumentar que a língua portuguesa é mais complexa. No inglês, considerando o tempo presente, vemos a distinção de apenas duas formas: (i) a forma para *I, We, They* (como, por exemplo, o verbo *walk*) e (ii) para *he/she/it (walks)*. Uma exceção a essa regra é o verbo *to be*, que apresenta três formas possíveis: (i) *I am*, (ii) *he/she/it is* e (iii) *they are*. No *Simple Past*, apresenta maior número de formas também apenas para o grupo *to be*: (i) *I/he/she/it was*, e (ii) *They/We were*; os demais não apresentam marcação de pessoa (Nelson (2010)). No português, a norma tradicional distingue seis pessoas: 1ª: Eu, 2ª: Tu, 3ª: Ele/Ela, 4ª: Nós, 5ª: Vós, 6ª: Eles. Entretanto, atualmente vemos que as formas associadas às 2ª e 5ª pessoas estão caindo em desuso. A 2ª pessoa, o “Tu”, é utilizada ainda em algumas regiões do Brasil, mas normalmente tem sido reproduzida como a forma da 3ª pessoa (ou seja, *Você/Ele gosta*). Do mesmo modo, a forma de 5ª tem sido substituída pela forma de 6ª pessoa (*Vocês/Eles gostam*) (Câmara Jr. (1999)).¹

Com relação às irregularidades, no inglês os verbos irregulares encontram-se apenas no *Simple Past* e *Past Participle*, enquanto que o sistema verbal do português é repleto de irregularidades em todos os tempos, modos e pessoas. Vejamos o verbo “*dizer*”, por exemplo. No tempo presente e modo indicativo, notamos a presença de irregularidade na primeira e terceira pessoa do singular (“*digo*” e “*diga*”). No pretérito perfeito, o paradigma é totalmente irregular, com: “*disse*”, “*disseste*”, “*disse*”, “*dissemos*”, “*dissestes*” e “*disseram*”. No futuro do presente, temos: “*direi*”, “*dirás*”, “*dirá*”, “*diremos*”, “*direis*” e “*dirão*”.²

Apesar da complexidade do português mostrar-se relevante para o desafio do aprendizado de flexão de verbos irregulares, optamos pela realização de um estudo mais restrito. A razão para tal limitação decorre principalmente da ausência de um corpus preparado para a realização da tarefa. Desse modo, ao limitarmos as irregularidades a um único paradigma, colocamos o nível de dificuldade em um patamar similar ao da pesquisa de Rumelhart & McClelland (1986), tendo como principal diferença (e possivelmente dificuldade extra) as três conjugações possíveis.

¹Também é interessante notar que ultimamente a 4ª (Nós) tem sido alternada com o uso de “A gente”, cuja forma verbal corresponde à 3ª pessoa também (*A gente gosta*).

²Também é possível observar irregularidades nos modos subjuntivo e imperativo em todos os paradigmas.

O escopo foi, portanto, restrito à 1ª Pessoa do Singular no tempo Presente e modo Indicativo (com exemplos já explorados na Seção 1.2.1). Desse modo, verbos que apresentarem irregularidade em outro tempo, modo ou pessoa que não na 1ª pessoa do singular no tempo presente e modo indicativo, serão tratados como pertencentes à classe dos regulares. Como exemplo, considere o verbo *correr*. Esse verbo apresenta flexão regular para a 1ª Pessoa (*corro*), mas é irregular para a 3ª Pessoa (*corre*). Assim, apesar de *correr* ser um verbo irregular, como ele apresenta flexão regular no paradigma escolhido, será tratado como regular para os fins dessa pesquisa.

Capítulo 2

Pré-Processamento dos Verbos para Redes Neurais

Este capítulo tratará de uma das partes mais importantes da modelagem em redes neurais: o pré-processamento dos dados. Muito se discute sobre o desenvolvimento de novas técnicas e arquiteturas, mas nem sempre a mesma importância é dada para esse estágio da modelagem - que é, na maioria das vezes, onde se dedica mais tempo.

Neste trabalho, os dados de interesse são *verbos*, e para um computador, um verbo é apenas uma sequência de caracteres. Ademais, um modelo de Redes Neurais é um modelo computacional, e para que o modelo funcione, é necessário preparar os dados para que eles estejam em um formato adequado para os cálculos que serão realizados. Desse modo, mesmo sem adentrar na parte teórica do funcionamento do modelo, já cabe discutir de que forma os verbos serão inseridos. Simplificadamente, os modelos de redes neurais são alimentados com *vetores numéricos* (ou *arrays*). Esses vetores são essencialmente uma lista de números (por ex. $[0, 1, 2, 3]$). Assim, pré-processar os verbos para a alimentação do modelo significa encontrar uma *representação vetorial* para cada um deles. Ainda, teremos que todos os verbos serão representados por vetores de comprimentos iguais, e que será este mesmo comprimento que definirá a dimensão (número de nódulos) da primeira camada da rede (a camada de *input*). Além disso, também é neste momento que definiremos o recorte dos dados - o nível de abstração do estudo, por assim dizer. Em outras palavras, isso significa que podemos recortar um verbo de diversas maneiras: podemos considerar que verbos são uma sequência de letras, ou uma sequência de fones, uma sequência de morfemas, etc. Porém, ao fazermos este recorte, estaremos automaticamente enviesando o resultado final do estudo. Por exemplo, se optarmos por representar um verbo utilizando a escrita ortográfica do mesmo, o modelo falhará em encontrar as relações mais sutis entre os elementos fonéticos desse verbo.

Desse modo, fica evidente a importância desta etapa. Falaremos primeiramente sobre o pré-processamento utilizado por Rumelhart & McClelland (1986). Em seguida, veremos uma apresentação do corpus utilizado nesta pesquisa para o treinamento do modelo *Encoder-Decoder*. Por fim, será exibido o algoritmo de pré-processamento utilizado neste trabalho para

a alimentação do modelo desenvolvido. Ainda, em questão de terminologia, por vezes o termo *codificação* será utilizado como opção a pré-processamento. Analogamente, *decodificação* e *pós-processamento* serão utilizados para descrever o processo inverso.

2.1 Pré-processamento de Rumelhart & McClelland

A arquitetura do modelo utilizado por Rumelhart & McClelland (1986) era um tanto limitada para o problema do aprendizado das flexões dos verbos. A limitação resultava do fato de que tanto os *inputs* quanto os *alvos* do modelo possuíam tamanhos variáveis. O *input*, por exemplo, poderia ser “like” ou “overtake”, e os *alvos*, “liked” e “overtook”. Entretanto, a arquitetura utilizada (já apresentada na Fig. 1.1) é composta por um número fixo de nódulos em cada camada. Poderíamos supor simplesmente que cada nódulo representasse cada fone do alfabeto fonético. Dessa forma, o *input* hipotético seria o conjunto de fones do verbo. Entretanto, ao fazermos isto, a rede perderia completamente a noção de sequência dos fones. Dada a limitação, Rumelhart & McClelland (1986) acabaram desenvolvendo um sistema de codificação composto por várias etapas. Utilizaremos o verbo “came” (a forma no pretérito de “to come”) como exemplo para detalhar cada um dos estágios de codificação utilizados pelos pesquisadores.

A primeira etapa consistiu na transcrição dos verbos utilizando um alfabeto compatível com o código ASCII (Mackenzie (1980)). O código ASCII é um código usado para representar textos em computadores. Ele codifica letras do alfabeto latino, sinais de pontuação e sinais matemáticos. A opção pela utilização do código ASCII é necessária, pois o código fonético não é interpretável pelas linguagens de programação. Segundo a chave de transcrição dos pesquisadores (Apêndice A), o verbo “came” foi transcrito para “#kAm#”. O símbolo “#” é utilizado para demarcar o início e o final do verbo.

Entretanto, tal codificação não seria o suficiente. Rumelhart & McClelland (1986) precisavam encontrar uma forma de apresentar os fones sequencialmente para que o modelo pudesse captar a ordem linear das informações fonéticas. O problema é que o verbo teria que ser inserido todo de uma vez, então não bastaria encontrar representações para cada fone individualmente, seria necessário encontrar uma representação completa para todos os fones do verbo e que de alguma forma ainda preservasse alguma noção de sequência. Assim, Rumelhart & McClelland (1986) tiveram a ideia de reaproveitar um conceito criado por Wickelgren (1969), o conceito de *Wickelphone*. Com isso, as transcrições foram reestruturadas em *trigramas*. Fazer isso significa analisar a sequência do verbo de três em três segmentos, ou seja, nessa etapa, a sequência “#kAm#” passou a ser tratada como “#kA” + “kAm” + “Am#”, sendo que cada uma dessas unidades é considerada um *Wickelphone*.

Apesar disso, tratar os verbos com seus respectivos *Wickelphones* ainda não seria o suficiente para o modelo de Rumelhart & McClelland (1986). A questão é que os autores gostariam que o modelo fosse capaz de identificar relações mais sutis entre as formas ver-

bais, ou seja, identificar relações mais próximas às execuções sonoras das palavras do que no nível dos fones. Desse modo, os autores apresentam uma nova estrutura inspirada pela representação de *Wickelphones* de Wickelgren (1969), intitulada de *Wickelfeatures*. Os *Wickelfeatures* são justamente as tríades de traços fonéticos introduzidas no Cap. 1. Como visto na Tabela 1.1, cada um dos fones pode ser descrito por algumas características relacionadas à execução dos seus respectivos sons. Entretanto, as características dos trigramas formados podem ser combinadas de muitas formas, mais precisamente, 4^3 possibilidades, visto que cada fone pode ser descrito por 4 traços (ver Apêndice (A)).

Assim, para a construção dos nódulos do modelo, Rumelhart & McClelland (1986) computaram todos os *Wickelfeatures* possíveis entre os verbos e excluíram alguns redundantes para simplificar a computação. Ao final, os *Wickelfeatures* dos verbos foram mapeados como chaves em um dicionário de tamanho fixo (um vetor) em que os valores poderiam assumir somente 0 ou 1; 1 caso aquele *Wickelfeature* estivesse presente no verbo e 0 caso contrário. A Figura 2.1 ilustra o esquema de codificação para o verbo “came”.

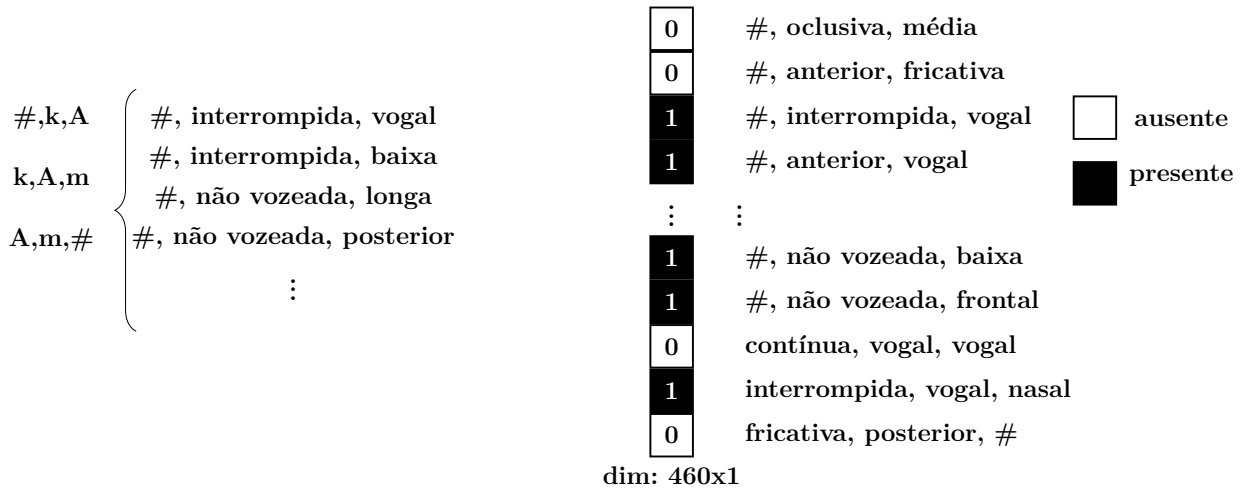


Figura 2.1: Esquema de Codificação de Rumelhart e McClelland

Como comentado no Cap. 1, o modelo de Rumelhart & McClelland (1986) atingiu um desempenho razoável, acertando 2/3 do conjunto de teste. Apesar disso, veremos que a arquitetura do tipo *Feedforward* não é a mais adequada para esse tipo de problema. Em primeiro lugar, o esquema de codificação proposto é bastante limitado, visto que os vetores de *input* da rede conseguem marcar apenas a presença ou ausência dos *Wickelfeatures*. Pinker (1999) comenta sobre esse problema utilizando como exemplo a palavra “*algalgal*” (uma palavra da língua *Oykangand*), em que ocorre repetição de trigramas. Podemos observar, nesse caso, que utilizando a representação escolhida seria impossível marcar tal repetição. Ademais, podemos notar que o problema não ocorre somente em caso de repetição de trigramas, mas também em casos de repetição de *Wickelfeatures*. Como um exemplo do problema, podemos analisar o verbo “*understand*”. A Tabela 2.3 exhibe uma comparação entre os *Wickelfeatures* dos trigramas *der* e *tan*, ambos presentes no verbo de exemplo. Na tabela, podemos perceber que muitos *Wickelfeatures* se repetem apesar dos dois trigramas serem completamente

diferentes.

d	e	r	t	a	n
consoante	vogal	consoante	consoante	vogal	consoante
vozeada	frontal	nasal	não vozeada	frontal	líquida
oclusiva	curta	média	oclusiva	curta	média
média	anterior	vozeada	média	baixa	vozeada

Tabela 2.3: *Wickelfeatures de der e tan*

Além disso, assim como os verbos tem que ser processados para entrar nos modelos como vetores, a saída do modelo também precisa ser decodificada para a reconstrução do verbo - o processo de *decodificação* dos *outputs* do modelo. A decodificação envolvia um esquema também composto por várias etapas. De maneira simplificada, Rumelhart & McClelland (1986) elencavam os trigramas (*Wickelphones*) candidatos para cada verbo e estes “competiam” pelos vetores de *Wickelfeatures* mais relevantes da saída do modelo. Como a saída da rede FFD é também um vetor que guarda apenas a presença ou ausência dos traços, e não quantas vezes eles aparecem, o processo de decodificação apresenta problemas para acertar em casos de verbos com repetições de *Wickelfeatures*. As complicações observadas tanto na construção da codificação quanto na decodificação dos dados refletem a necessidade de uma arquitetura mais adequada para o problema em questão.

2.2 Apresentação do corpus

O corpus utilizado nesta pesquisa foi construído a partir da listagem de verbos contida no endereço www.conjugacao.com.br.

Primeiramente, foi realizada uma etapa de extração dos verbos e suas respectivas formas flexionadas para um arquivo *.csv* via técnica de *web scraping*, uma técnica que extrai informações contidas nas páginas da web (Mitchell (2015)). Em seguida, os verbos irregulares foram selecionados manualmente para diferentes famílias de verbos, ou seja, grupos que continham o mesmo padrão de flexão. Alguns dos verbos irregulares listados na fonte de referência não eram irregulares no processo de flexão de interesse, e portanto foram realocados para o grupo de verbos regulares (vide exemplo do verbo *correr* na Seção 1.2.4). Na sequência, os verbos coletados na forma infinitiva tiveram o fone /r/ extraído para que no *input* do modelo entrasse apenas o radical + vogal temática dos verbos. A razão para tal foi apenas uma questão de conveniência já que essa marcação era redundante por estar presente em todos os verbos.

Um experimento foi realizado na tentativa de utilizar o transcritor fonético automático disponibilizado por Guide (2016) para tornar o processo de transcrição mais rápido. Entretanto, o transcritor falhou na transcrição de verbos cujas formas escritas coincidem com substantivos, como por exemplo: “apoio”, “peso”, “toco”, “posto”, “jogo”; ignorando as características fonéticas dessas palavras enquanto verbos.

Desse modo, os verbos coletados foram transcritos manualmente utilizando a chave de transcrição apresentada na Tabela 2.8. No total, foram obtidos 423 verbos, 83 a menos que no experimento realizado por (Rumelhart & McClelland (1986)). Dos 423 verbos extraídos, 20 foram considerados verbos sem possível agrupamento (como “ir”, “trazer” e “saber”), totalizando uma base de 214 verbos regulares e 209 irregulares (50.6% e 49.4% respectivamente). O estudo de Rumelhart & McClelland (1986), por sua vez, era composto por apenas 20% de verbos irregulares.

A Tabela 2.4 associa exemplos de verbos das classes obtidas às suas respectivas contagens e proporções no corpus.

	Exemplos	Contagem	Proporção
1	ansiar, anseio	9	2.13%
2	botar, boto	30	7.09%
3	cobrir, cubro	7	1.65%
4	dizer, digo	7	1.65%
5	fazer, faço	15	3.55%
6	crer, creio	5	1.18%
7	sentir, sinto	8	1.89%
8	pedir, peço	7	1.65%
9	pôr, ponho	27	6.38%
10	seguir, sigo	27	6.38%
11	ter, tenho	10	2.36%
12	testar, testo	20	4.73%
13	ver, vejo	6	1.42%
14	vir, venho	10	2.60%
15	saber, sei	20	4.73%
16	falar, falo	214	50.59%

Tabela 2.4: *Organização do corpus*

Apesar do volume de verbos irregulares ser consideravelmente maior do que o volume utilizado no estudo de Rumelhart & McClelland (1986), pode-se dizer que algumas das famílias coletadas apresentam certa improdutividade, no sentido de que muitos verbos são reaproveitamentos de outros através do uso de prefixos. Por exemplo, a família do verbo “fazer” é composta apenas de verbos derivados do mesmo: “desfazer”, “refazer”, etc. Isto pode ser observado em outras classes, como a do verbo “cobrir”, “pedir”, “dizer”, “ver”, entre outras.

2.2.1 Types x Tokens

Outra questão importante a respeito do corpus utilizado, refere-se à frequência em que os verbos estão presentes no mesmo. Para isso, introduzimos os conceitos de *word type* e *word token* (Peirce, S. & Santiago (1906)). O termo *type*, refere-se ao número de palavras **únicas**

presentes em um Corpus. A frase “Essa frase é uma frase de exemplo.”, portanto, possui 6 *types*. O termo *token*, em contrapartida, refere-se ao número total de termos presentes, incluindo as repetições. Nesse caso, a mesma frase de exemplo possui 7 *tokens*. Assim, tratar os verbos como *word tokens* significaria ter um corpus que respeitasse a frequência de uso dos verbos. Tratá-los como *word types*, significaria apenas tratá-los como uma lista de verbos, sem repetições.

Discutir sobre a utilização de *word types* ou *word tokens* na constituição do corpus é relevante, pois o treinamento das Redes Neurais se dá por meio de *ciclos* e já vimos no Cap. 1 que o aprendizado da rede consiste no ajuste dos pesos das conexões. Desse modo, se o modelo for alimentado com verbos em diferentes frequências, ele tenderá a priorizar o ajuste dessas conexões (e portanto do aprendizado como um todo) para os verbos mais frequentes.

Para esta pesquisa, optou-se pela utilização de *word types*, isto é, todos os verbos foram tratados igualmente e introduzidos o mesmo número de vezes no modelo.¹

2.3 Pré-processamento para o Encoder-Decoder

No caso da arquitetura *Encoder-Decoder*, diferentemente da arquitetura utilizada por Rumelhart & McClelland (1986), não há motivo para preocupação com relação à sequência dos dados, nem durante o processo de inserção, nem no processo de predição do modelo. A razão para tal ficará mais clara após os capítulos 3 e 4. Entretanto, o que já pode ser adiantado é que não há a necessidade do tratamento dos verbos em trigramas, tão pouco em *Wickelfeatures*. Neste tipo de arquitetura, os verbos podem ser normalmente tratados como uma sequência de fones. Entretanto, mesmo com essa simplificação, existem múltiplas maneiras de se abordar essa questão. Este trabalho tem como objetivo estudar as relações entre os verbos em um nível fonético, portanto é importante que as representações vetoriais construídas levem isto em consideração.

Começaremos pela apresentação do Alfabeto Fonético Internacional (AFI), apresentado na Tabela 2.5 e Fig. 2.2. O AFI é um sistema de notação fonética, criado pela Associação Fonética Internacional para promover uma padronização na transcrição de dados de diferentes idiomas. Ele organiza símbolos que representam unidades sonoras presentes nas línguas humanas a partir de características de execução dessas unidades.

A Tabela 2.5 reúne o conjunto dos sons consoantes e exibe na dimensão das colunas o ponto de articulação dos sons, isto é, o ponto onde ocorre uma perturbação da passagem do ar. A coluna “Bilabial”, por exemplo, corresponde à obstrução que ocorre nos lábios durante a produção dos seus respectivos fones. As linhas compõem os diferentes *modos* de articulação possíveis, ou seja, as formas de obstrução da passagem de ar. A obstrução pode ser total como em [p] ou parcial como em [s]. Por fim, dentro de uma mesma célula pode ocorrer um

¹Evidências na área de psicolinguística (Bybee (1995); Cotterell (2001)) indicam que humanos aprendem a generalizar padrões fonológicos baseado na contagem de *word types*, ignorando a frequência de uso das palavras.

som com ou sem a vibração das cordas vocais, é o caso do par [p b]. O símbolo da esquerda representa o som surdo e o símbolo da direita, sonoro.

As vogais estão organizadas na Fig. 2.2. As colunas dessa tabela se referem ao local de reprodução dos sons (com avanço ou recuo da língua), e as linhas, à altura da língua em relação ao céu da boca durante a execução do som. Quando os símbolos aparecem em pares, aquele da direita representa uma vogal arredondada (Seara, I. C. (2015)). Como um exemplo do funcionamento desta tabela, é interessante observar a execução da vogal [u] e compará-la com a vogal [o]. Ao pronunciá-las para uma comparação, nota-se o arredondamento dos lábios em ambas, porém a altura da língua durante a execução da segunda é levemente mais baixa que a altura da primeira. Para entender a questão de anterioridade/posterioridade, é interessante observar a movimentação (avanço e recuo) da língua durante a execução de [e] e [o]. Nesse caso, nota-se também que [o] apresenta traço de arredondamento, enquanto que [e] não.

	Bilabial	Labiodental	Dental	Alveolar	P-alveo.	Retroflexa	Palatal	Velar	Uvular	Faringal	Glotal
Oclusiva	(p) (b)			(t) (d)		t d	c ʃ	(k) (g)	q ɣ		ʔ
Nasal		(m)	ɱ	(n)		n	ɲ	(ŋ)	N		
Vibrante		β		(r)					R		
Tepe				r		r					
Fricativa	ɸ β	(f) (v)	θ ð	(s) (z)	(ʃ) (ʒ)	s z	ç ʝ	(x) ɣ	χ ʁ	ħ ʕ	h ɦ
Fric. Lateral				ɬ ɮ							
Aprox.			ʋ	ɹ		r	j	ɰ			
Aprox. Lateral				(l)		l	ʎ	(L)			

Tabela 2.5: *Consoantes AFI*

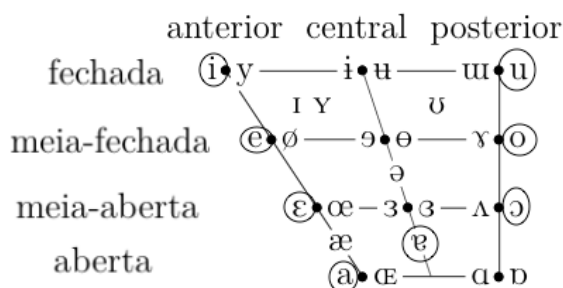


Figura 2.2: *Vogais AFI*

A ideia do AFI é mapear e caracterizar todos os sons das línguas humanas. Desse modo, cada língua aproveita apenas um subconjunto do AFI. Os fones presentes no Português-Brasileiro estão circulados. Levando em conta apenas o subconjunto dos fones do PB, e também tendo em mente que os símbolos do alfabeto fonético precisam ser transformados em código ASCII para serem interpretados, faz sentido construir uma nova tabela adaptada para o problema em questão. O resultado desta adaptação pode ser visto nas Tabelas 2.6 e 2.7. Nessas novas tabelas, além da exclusão de alguns pontos e modos de articulação

(motivada pela própria natureza do PB), apresenta-se também uma chave de transcrição alternativa que engloba apenas caracteres pertencentes ao código ASCII. Com relação à tabela das vogais (2.7), a dimensão de arredondamento foi dispensada pois não havia mais a necessidade dessa marcação já que essa informação seria redundante devido ao fato de que toda vogal posterior é arredondada no português brasileiro e nenhuma anterior é.

	Bilabial		Lab. dent.	Alveolar		P-alveo.	Velar
Oclusiva	p	b		t	d		k g
Nasal	m			n			N
Tepe				r			
Fricativa			f v	s z	x j		h
Lat. apr.				l			L

Tabela 2.6: *Consoantes na nova representação*

	Anterior	Posterior
Fechada	i	u
Meia-fechada	e	o
Meia-aberta	E	O
Aberta	a	
Nasais	A (ã)	
	3 (ẽ)	

Tabela 2.7: *Vogais na nova representação*

A Tabela 2.8 exibe a chave de transcrição proposta utilizando as tabelas fonéticas criadas e a Tabela 2.9 exibe alguns exemplos de transcrições possíveis.²

²Com intuito de facilitar o pré-processamento, algumas transcrições fonéticas foram representadas por um mesmo símbolo, é o caso de: tʃ e t → t; u, w e ʊ → u.

AFI	Transcrição Proposta
[p] - parar	p
[b] - botar	b
[t] - tocar	t
[d] - dançar	d
[k] - casar	k
[g] - gostar	g
[f] - fugir	f
[v] - voltar	v
[s] - soltar	s
[z] - presenciar	z
[ʃ] - chamar	x
[ʒ] - jantar	j
[tʃ] - sentir	t
[dʒ] - dizer	d
[h] - errar	h
[r] - encarar	r
[l] - pular	l
[ʎ] - espalhar	L
[m] - morar	m
[n] - nadar	n
[ɲ] - sonhar	N
[a] - parar	a
[e] - ler	e
[ɛ] - espero	E
[i] - rir	i
[o] - propor	o
[ɔ] - coloco	O
[u] - curtir	u
[ẽ] - entreter	3
[ã] - plantar	A
[õ] - componho	o
[u] - caso	u
[j] - saio	i
[w] - volto	u

Tabela 2.8: Chave de Transcrição Proposta

Verbo	Transcrição
ressentir	hes3ntir
paro	paru
posuo	posuu
olha	oLa
sacudir	sakudir
voltar	voutar

Tabela 2.9: Exemplos de Transcrições

2.3.1 Os inputs do modelo Encoder-Decoder

Após a construção do corpus, os verbos transcritos tiveram seus respectivos fones associados aos traços fonéticos que os compõem, os mesmos traços que originaram as Tabelas 2.6 e 2.7. A Tabela 2.10 apresenta um exemplo desse processo para o verbo “posso” - “*pOsu*”.

Fone	Traços Fonéticos
p	bilabial, oclusiva, não vozeada
O	meio-aberta, posterior
s	fricativa, alveolar, não vozeada
u	fechada, posterior

Tabela 2.10: *Traços Fonéticos para o Verbo “Posso”*

Nota-se na tabela 2.10 que é possível caracterizar cada fone com três ou dois traços fonéticos. Entretanto, para construirmos a representação vetorial para o modelo, precisamos considerar de antemão todas as dimensões possíveis do mesmo, pois cada dimensão representará um único traço fonético.

Para a construção dos vetores, utilizaremos *dicionários*. Na computação, um dicionário é uma estrutura de armazenamento de dados que associa uma chave a um valor. Essa estrutura possui um conjunto mutuamente exclusivo de chaves, cada uma associada a um valor. Desse modo, ao consultar um dicionário com um valor chave, a estrutura retorna como resposta o valor associado. No total são necessárias 20 dimensões para representar um fone. São 18 para representar os traços das tabelas fonéticas construídas (2.6 e 2.7): Oclusiva, Nasal, Tepe, Fricativa, Aproximante Lateral, Bilabial, Lábio-Dental, Alveolar, Pós-Alveolar, Velar, Glotal, Fechada, Meia-fechada, Meia-aberta, Aberta, Anterior, Posterior; e mais outras 2 para representar início e final do verbo. A necessidade dessas duas últimas dimensões ficará mais clara após o Cap. 3.

Os fones são então caracterizados pela presença (1) e ausência (0) dos traços mencionados. Desse modo, o dicionário construído possui fones nos valores das chaves e uma lista de 0's e 1's para representar as presenças e ausências dos traços fonéticos. Como visto, de acordo com as tabelas fonéticas desenvolvidas, cada fonema pode ser descrito por três ou dois traços fonéticos, de modo que cada vetor terá apenas três ou dois valores marcados como 1's. A representação de início e final do verbo serão exceções, com apenas uma marcação de presença no vetor na respectiva dimensão.

A Tabela 2.11 mostra como exemplo uma comparação entre dois fones similares (**p** e **b**) que distinguem-se apenas pelo traço fonético sonoro. O resultado é uma representação vetorial que também carrega essa noção de proximidade entre os fones. O começo do verbo (ou seja, antes do primeiro fone) é representado por um vetor que marca 0 em todos os traços fonéticos e 1 para representar o começo (o traço de fim pode ser compreendido de maneira análoga). Ademais, vale notar que as tabelas fonéticas propostas (2.6 e 2.7) somam 28 fones

possíveis. Como temos também as marcações de começo e final, teremos à disposição um total de 30 representações vetoriais distintas.

Traço	p	b
oclusiva	1	1
nasal	0	0
tepe	0	0
fricativa	0	0
l-aprox	0	0
bilabial	1	1
labiodental	0	0
alveolar	0	0
p-alveolar	0	0
velar	0	0
glotal	0	0
vozeada	0	1
fechada	0	0
m-fechada	0	0
m-aberta	0	0
aberta	0	0
anterior	0	0
posterior	0	0
<beg>	0	0
<eos>	0	0

Tabela 2.11: *Exemplo de Codificação de fones*

Juntando todos os passos expostos, o processo completo de transformação dos inputs pode ser resumido a:

1. Adição de símbolos para marcação de início e final dos verbos.
2. Divisão dos verbos em fones, seguindo chave de transcrição proposta.
3. Transformação dos fones em *arrays* de 0's e 1's, seguindo o dicionário de fones desenvolvido.

Em suma, teremos que os vetores de *input* são portanto vetores binários de 20 dimensões, contendo de uma a três dimensões ocupadas por 1's e o restante preenchido com zeros. A representação completa do verbo inteiro, por sua vez, consistirá na concatenação consecutiva desses vetores.

Capítulo 3

Modelos de Redes Neurais

Este capítulo servirá como uma introdução à teoria das Redes Neurais. Os conceitos aqui introduzidos servirão como embasamento teórico imprescindível para a apresentação do modelo final utilizado nesta pesquisa, o *Encoder-Decoder*. Primeiramente começaremos com uma breve introdução ao conceito de *Aprendizado de Máquina*. Para tal, veremos exemplos de modelagem em aprendizados ditos *supervisionados* e *não-supervisionados*. Em seguida, veremos os principais conceitos teóricos para o modelo de *Redes Neurais*. Na sequência, seremos apresentados ao conceito de *Modelo de Linguagem*, o qual servirá como apoio para o entendimento do funcionamento das *Redes Neurais Recorrentes*.

3.1 Aprendizado de máquina

Um modelo de rede neural é, essencialmente, um modelo de aprendizado de máquina supervisionado que está à procura de identificar padrões. Um modelo de aprendizado de máquina é uma tarefa computacional que explora algoritmos que podem aprender a partir de seus erros e fazer previsões sobre dados. Os possíveis tipos de aprendizado de máquina são divididos entre *supervisionados*, *não-supervisionados* e por *reforço* (Gron (2017)).

Um modelo de aprendizado de máquina é dito *supervisionado* caso seja feito o uso de uma variável *resposta* para o treinamento do mesmo. Para exemplificar o que isso significa, podemos utilizar o *dataset “Iris”* (Dua & Graff (2017)). Esse *dataset* é composto por colunas de diferentes características analisadas em três espécies diferentes de plantas (*Iris setosa*, *Iris virginica* e *Iris versicolor*). Observe o gráfico de dispersão na Fig. 3.1. Cada ponto desse gráfico representa uma planta observada de uma única espécie (a *Iris setosa*). A posição dos pontos representa, simultaneamente, o comprimento da sépala da observação (eixo x) e o comprimento da respectiva pétala no eixo y . Observando o gráfico, vemos que saber o comprimento de uma *sépala* nos traz informação também sobre o comprimento da *pétala*, isto é, conforme o tamanho da sépala aumenta, o tamanho da pétala parece aumentar respeitando uma certa proporção, ou seja, um padrão linear. Assim, podemos tentar encontrar uma expressão para uma reta que formalize o padrão observado. Fazer isso é interessante pois,

supondo que tivéssemos um grupo de plantas dessa espécie em que dispuséssemos somente de informações sobre suas sépalas, teríamos uma forma de estimar os respectivos valores das pétalas.

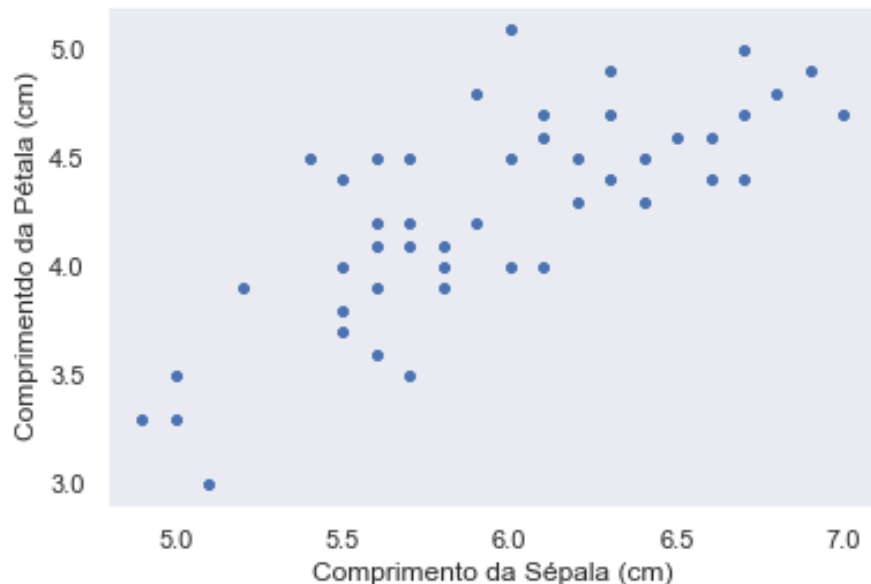


Figura 3.1: Gráfico de Dispersão das variáveis Comprimento de Sépala e Comprimento de Pétala para uma mesma espécie de planta no dataset Iris

Um dos modelos que podemos utilizar para encontrar a expressão buscada é o modelo de aprendizado de máquina de *Regressão Linear*. O objetivo desse modelo é encontrar a expressão para uma reta que represente o padrão observado da melhor forma possível. Para tanto, o treinamento desse modelo requer a utilização das observações existentes, ou seja, dos valores reais das pétalas para encontrar a expressão que buscamos. Relembrando conceitos de álgebra, uma expressão genérica para uma reta pode ser dada por:

$$y = ax + b \quad (3.1)$$

A equação 3.1 propõe que o valor de y pode ser obtido a partir do valor de x . No caso de um problema de *Regressão Linear*, não poderemos obter um valor para y , apenas estimá-lo. Dessa forma, substituímos y por \hat{y} para indicar a modelagem, resultando na equação 3.2.

$$\hat{y} = ax + b \quad (3.2)$$

Vemos, portanto, que para estimarmos y estamos essencialmente à procura de dois parâmetros: a e b . O primeiro, refere-se ao ângulo da inclinação da reta, ou seja, a variação em y a cada unidade de x (no exemplo das plantas, uma unidade de x equivale a um centímetro). O segundo, refere-se ao intercepto da reta, ou seja, o valor de y quando x é zero.

De modo simplificado, o algoritmo do modelo de *Regressão Linear* irá utilizar os valores

observados de x e y (comprimentos de sépalas e pétalas) contidos no *dataset* “*Iris*” para encontrar os parâmetros mencionados. O processo envolve conceitos de otimização e cálculo, e portanto não será abordado em detalhes neste trabalho (ver Morettin, P. A. & Bussab (2004) para o detalhamento completo). Entretanto, o importante para o entendimento deste tipo de aprendizado de máquina (o *supervisionado*) é que temos e utilizamos os valores de y , a variável resposta, para a busca dos parâmetros do modelo proposto. Na Fig. 3.2 podemos ver a reta encontrada pelo modelo de *Regressão Linear*.

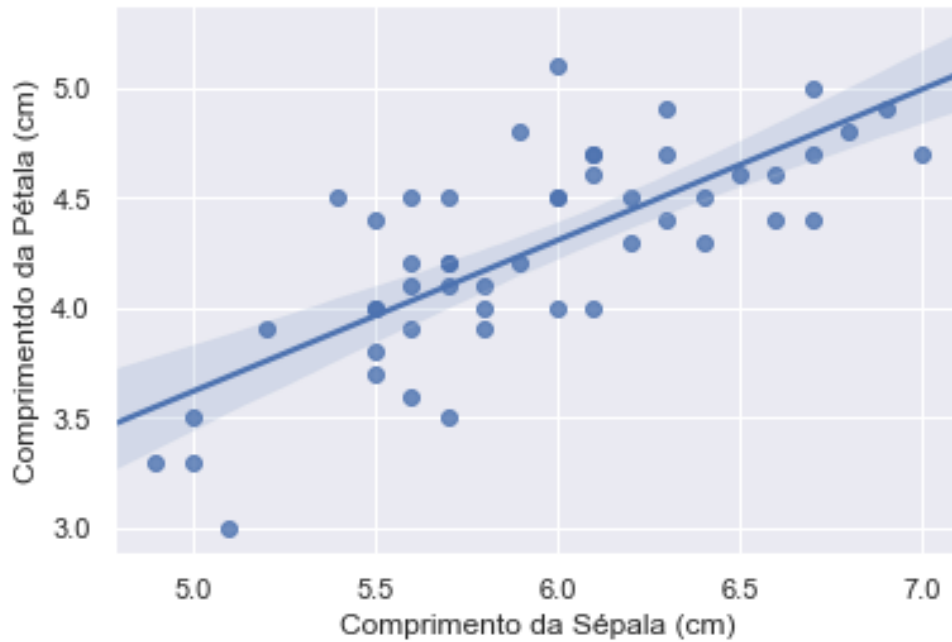


Figura 3.2: *Modelo de Regressão Linear proposto*

Os aprendizados não-supervisionados, por sua vez, podem ser caracterizados pela ausência de uma variável resposta durante o treinamento. Eles são tipicamente utilizados para tarefas de agrupamento (ou *clustering*). Na Fig. 3.3 vemos outro gráfico de dispersão relacionando comprimento de sépalas e pétalas, dessa vez com os dados observados de todas as três espécies. No caso, continuamos tendo informações sobre os comprimentos, mas não sabemos a quais espécies as observações pertencem. Podemos então recorrer a um algoritmo não-supervisionado para fazer suposições sobre essa informação.

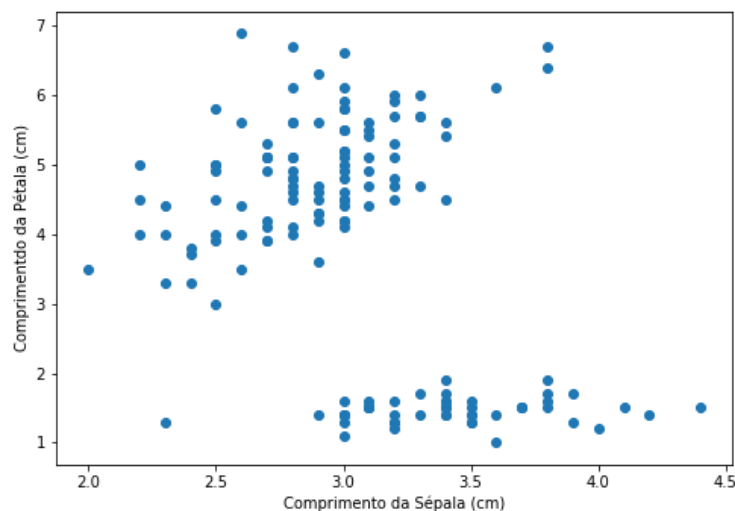


Figura 3.3: *Gráfico de Dispersão para o Comprimento de Sépalas e Pétalas para as Três Espécies do dataset Iris*

Assim como utilizamos o algoritmo de *Regressão Linear* para exemplificar um aprendizado supervisionado, utilizaremos o algoritmo de *K-means* (MacQueen (1967)) para exemplificar o aprendizado não-supervisionado que buscamos. A proposta do algoritmo de *K-means* é deduzir possíveis agrupamentos a partir de observações que compartilham de características similares.

A aplicação do algoritmo resultou nos agrupamentos que podem ser visualizados na Fig. 3.4. Nela, é possível observar que há um grupo de observações cujo tamanho das pétalas e das sépalas é em geral menor que os demais (grupo *amarelo*), isto pode indicar que estas observações sejam de uma mesma espécie. Além disso, o algoritmo também identificou os grupos *azul* e *marrom*, que não apresentam uma separação clara entre si. Entretanto, como não sabemos as classes verdadeiras das plantas, podemos apenas supor que cada agrupamento obtido se refere a uma espécie diferente.

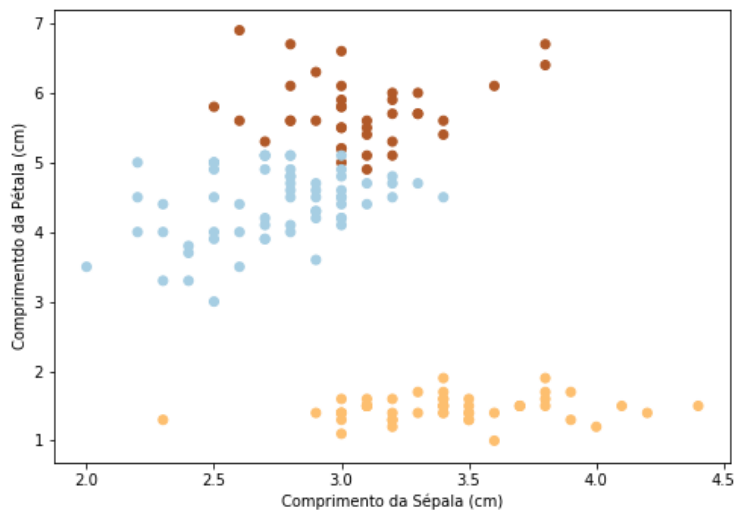


Figura 3.4: *Algoritmo não-supervisionado (K-means) realizado no dataset Iris*

Temos ainda mais um tipo de aprendizado de máquina, o de *reforço*. Os aprendizados por reforço consistem em algoritmos que recebem sinais de recompensa/punição quando acertam/erram uma tarefa de interesse. Já que a apresentação deste tipo de aprendizado não é fundamental para os fins desta pesquisa, recomenda-se a leitura de Kaelbling (1996) para maiores detalhes sobre o mesmo.

Para concluir, a partir dos conceitos expostos e retomando a Fig. 1.2, temos que um modelo de *Rede Neural* é um modelo de aprendizado de máquina supervisionado. Assim, temos que, para o aprendizado deste modelo ocorra, precisaremos informá-lo com as respostas que queremos que ele produza.

3.2 Introdução teórica a Redes Neurais

A inspiração para o desenvolvimento da modelagem em Redes Neurais Artificiais surgiu a partir de estudos em neurociência que concluíram que, diante de múltiplas apresentações de um mesmo estímulo, um mesmo conjunto de neurônios recebe estímulos e dispara. Um estímulo diferente resulta no disparo de um conjunto de neurônios diferente (Hubel & Wiesel (1962)). Analogamente, o modelo artificial é composto por uma camada denominada de *input* responsável por receber diferentes estímulos (matematicamente retratados por vetores numéricos que representam o objeto do aprendizado). A informação recebida é distribuída ao longo de múltiplas conexões com a próxima camada através de uma multiplicação com uma matriz de pesos \mathbf{W} (Fig. 3.5). A matriz de pesos funciona como uma analogia às conexões existentes entre neurônios de modo que um peso maior (mais próximo de 1) representa uma conexão reforçada e um peso menor (mais próximo de 0) representa uma conexão inibida.

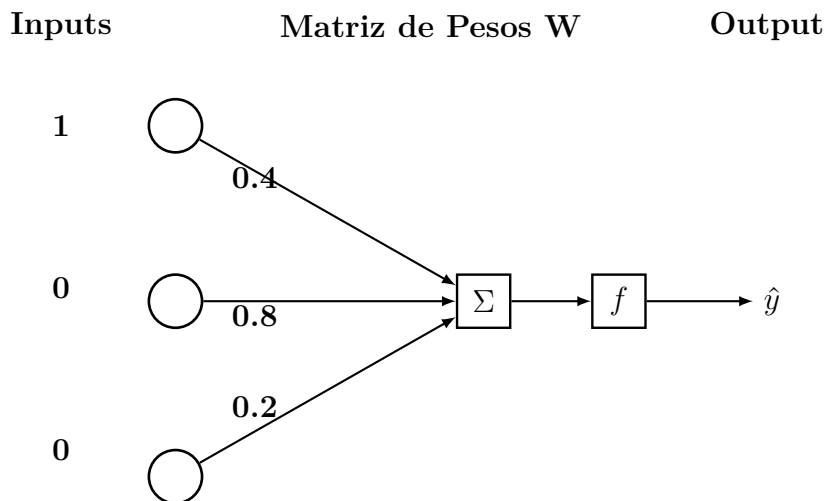


Figura 3.5: *Digrama de uma Rede Neural do Tipo Perceptron*

O resultado dessa multiplicação gera um vetor, cujos valores são em seguida somados (Σ). O valor resultante entra como argumento em uma função (que nesse contexto é chamada de *função de ativação* (f)). Uma das funções de ativação mais utilizadas nas primeiras pesquisas sobre *Redes Neurais* (e inclusive utilizada pelos pesquisadores Rumelhart e McClelland) é a função *Sigmoide*, uma função com propriedades matemáticas desejáveis e que comprime os resultados obtidos em um intervalo de $[0, 1]$. Pensando em uma tarefa de classificação, ou seja, uma tarefa com o objetivo de associar um determinado *input* a alguma classe, vemos que tal compressão é interessante, pois possibilita a associação dos *outputs* do modelo a probabilidades. A Fig. 3.6 mostra o gráfico da função *Sigmoid*. Na literatura existem ainda outras funções comumente utilizadas, como a *ReLu*, a *Tanh* e a *Softmax*. A escolha de função de ativação vai depender de alguns fatores, como por exemplo, o objetivo do aprendizado. Recomenda-se a consulta de Goodfellow, I. & Bengio (2016) para maiores detalhes sobre o assunto.

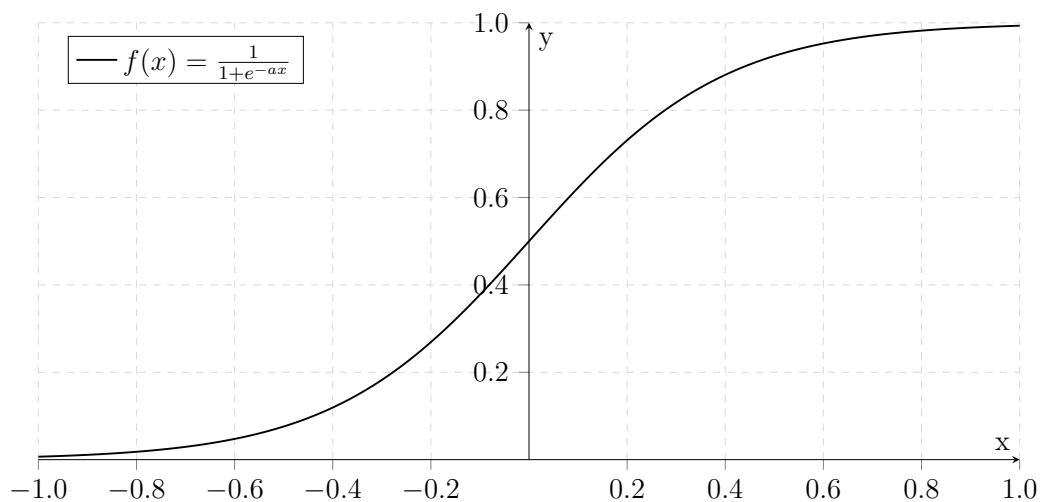


Figura 3.6: *A função Sigmoid.*

O diagrama apresentado na Fig. 3.5 representa uma arquitetura conhecida como *Perceptron*. Essa arquitetura, uma das primeiras construídas, é bastante simples. O *Perceptron* consiste de um vetor de *inputs*, uma matriz de pesos (que nesse caso tem apenas uma coluna), uma função de ativação e um *output* (Rosenblatt (1958)).

A partir dessa estrutura, as arquiteturas das Redes Neurais foram se tornando mais complexas, com por exemplo a adição de uma ou mais camadas intermediárias, antes da camada de *output*. Tais camadas intermediárias são chamadas de *camadas escondidas* (Ver Fig. 3.7.). Nesse caso, vale lembrar da álgebra linear que as dimensões de uma matriz são dadas por dois valores, o número de linhas (n) e o número de colunas (m) da mesma. No caso do *Perceptron*, temos uma matriz com $n \times m$, em que n corresponde à dimensão do *input* e m é 1. Ao acrescentarmos uma camada intermediária, temos que a matriz de pesos entre a camada de *input* e a camada adicionada terá dimensão $n \times m$, em que n continua tendo a dimensão do vetor de *input* e m corresponde ao número de nódulos da camada adicionada.

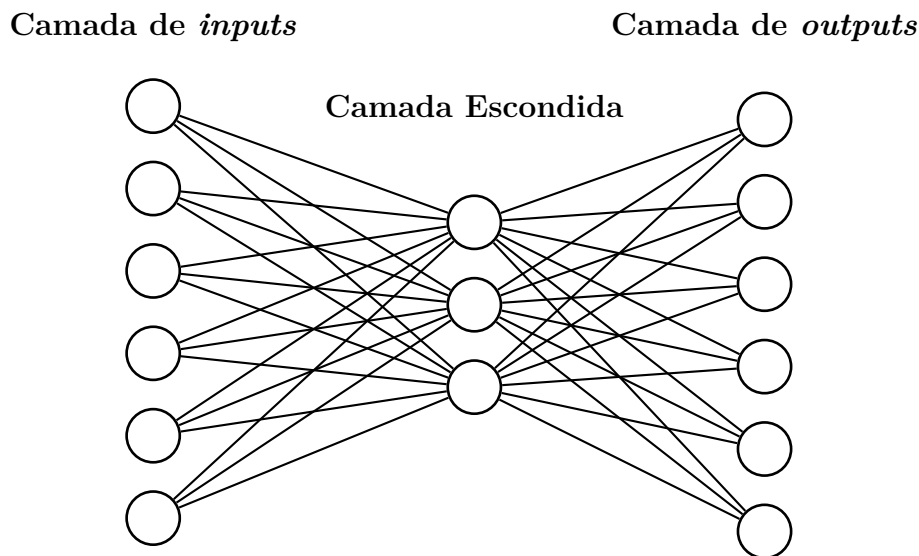


Figura 3.7: Esquema de uma Rede Neural com Camada Escondida

Como comentado na Seção 1.2.2, o acréscimo das camadas escondidas aumenta a capacidade de aprendizado dos modelos. Isso acontece porque a cada camada intermediária, o modelo aprende a identificar características relevantes para a produção de uma resposta final (o *output*). Entretanto, apontar quais características estão sendo aprendidas é uma tarefa complicada. A camada escondida gera uma representação de difícil interpretação, já que o resultado de cada nó é influenciado por todos os nós do *input*, criando assim uma representação essencialmente distribuída. Desse modo, fica difícil associar um determinado nó da camada a uma característica isolada do *input*.

3.3 Treinamento

Na Seção. 3.1 falamos sobre o fato de uma Rede Neural ser um aprendizado de máquina supervisionado. Para que a rede seja capaz de identificar os padrões desejados, é necessário provê-la com os *alvos*, pois o treinamento da mesma consiste, essencialmente, na atualização das matrizes de pesos que deve ocorrer a partir da comparação entre os valores previstos pela rede (os *outputs*) e os *alvos*. A comparação entre esses valores se dá por meio de uma função de custo (*Loss Function*), que representa uma forma de se quantificar o quão perto se está de uma rede ideal em que os resultados previstos correspondam exatamente aos *alvos*. O objetivo do aprendizado da rede é minimizar essa diferença, ou seja, encontrar os valores dos pesos na matriz \mathbf{W} que minimizam a função de custo (Patterson & Gibson (2017)). Para isso, faz-se uso de um algoritmo chamado de *backpropagation* (Goodfellow, I. & Bengio (2016)). Após a inserção de cada *input* e a aplicação do algoritmo de *backpropagation*, todos os pesos são atualizados simultaneamente com os valores que em conjunto minimizam a função de custo, e portanto, aproximam as previsões da rede aos *alvos*. Entretanto, o ajuste de um modelo de redes neurais é bastante delicado e experimental. Ele depende não somente da busca pelos melhores valores para os pesos da matriz \mathbf{W} como também de uma busca pelos chamados *hiperparâmetros*. Retomando o que vimos sobre aprendizado supervisionado na Seç. 3.1, vimos que o aprendizado de um modelo de *Regressão Linear*, consistia em encontrar valores para os parâmetros a e b para obter a expressão de uma reta. Analogamente, em um modelo de *Redes Neurais*, teremos que os *parâmetros* buscados para o aprendizado são os valores dos pesos de \mathbf{W} . Os *hiperparâmetros* são parâmetros do modelo que são escolhidos *a priori* para a sua configuração. Como exemplo, já foi falado um pouco sobre a quantidade de camadas intermediárias. Como essa questão determina a arquitetura da rede, deve ser definida *a priori*, sendo considerada um *hiperparâmetro*. Temos também a dimensionalidade (número de nódulos) das camadas, que também deve ser pré-determinada.

Além desses *hiperparâmetros*, temos também alguns outros valores que devem ser pré-determinados para dar início ao processo de treinamento do modelo. Um deles refere-se ao número de vezes em que um mesmo *input* é inserido. O treinamento de uma Rede Neural, se dá por meio de *ciclos*. Cada vez que um mesmo *input* é apresentado à rede, ocorre uma nova predição, e conseqüentemente, uma nova comparação com o *alvo*. Desse modo, a rede tem a oportunidade de ir refinando e ajustando os pesos para que os *outputs* se aproximem dos *alvos*. Esses *ciclos* são chamados na literatura de *épocas*.

Existem também outros *hiperparâmetros* importantes, como a taxa de aprendizado η , que tem a ver com a otimização do modelo. Ele essencialmente dita ao modelo o quanto alterar os pesos de \mathbf{W} a cada iteração. Existem também os *hiperparâmetros* λ e a taxa de *Dropout*, utilizados para evitar que o modelo se torne muito especialista nos dados de treinamento (enfraquecendo o poder de generalização para exemplos não vistos). Para mais detalhes sobre o uso desses e outros hiperparâmetros, ver Patterson & Gibson (2017).

Sobre a determinação dos hiperparâmetros, esta não é uma tarefa trivial, visto que há

uma variedade de hiperparâmetros que podem ser combinados de várias maneiras. Múltiplos treinamentos com diferentes configurações de hiperparâmetros são realizados para que se possa fazer essa escolha. Na prática, porém, é mais comum que esses hiperparâmetros sejam configurados a partir de resultados de experimentos semelhantes publicados na literatura. Existem também algoritmos de exploração de hiperparâmetros, como por exemplo o *Grid Search*. Dado um conjunto de hiperparâmetros para teste, este algoritmo testa todas as combinações possíveis e devolve como solução a combinação que gerou o melhor resultado (Goodfellow, I. & Bengio (2016)).

Mesmo com uma extensa busca pelos hiperparâmetros do modelo, nem sempre é possível encontrar valores em \mathbf{W} que possibilitem o aprendizado de forma satisfatória. Na prática, é bastante comum a ocorrência de problemas conhecidos como *overfit* (quando o modelo se torna especialista nos dados de treino mas tem dificuldade de generalizar para dados nunca vistos antes) e *underfit* (quando o modelo falha em capturar o padrão subjacente dos dados) (Patterson & Gibson (2017)). Na literatura existem algumas técnicas disponíveis para tratar dessas questões, como *Dropout*, *Data Augmentation* e *Resampling* (Goodfellow, I. & Bengio (2016), Patterson & Gibson (2017)).

3.4 Uma outra arquitetura

No Capítulo 2 exploramos dois tipos de pré-processamentos de verbos: (i) O processo de codificação de Rumelhart & McClelland (1986), que fazia uso de tríades de traços fonéticos para manter um registro de ordem sequencial dos *inputs*, e (ii) o pré-processamento escolhido para alimentação do modelo *Encoder-Decoder* desta pesquisa.

Como comentado, a arquitetura do *Encoder-Decoder* é mais adequada do que a *Feed-forward* para a inserção de dados de comprimentos variáveis. Isso é possível graças a uma arquitetura composta por *Redes Neurais Recorrentes*.

Antes, porém, da apresentação desta arquitetura, faz-se necessária uma introdução ao conceito de *Modelo de Linguagem*.

3.4.1 Modelo de Linguagem

Um modelo de linguagem é um modelo que propõe uma distribuição probabilística para uma sequência de termos em uma linguagem natural (Manning & Schütze (1999)). Visto de outro ângulo, por exemplo, o modelo ajuda a responder perguntas do tipo: “Qual a probabilidade de uma sentença s ? E de uma sentença s' ? Qual delas é mais provável?”. Como exemplo, suponha uma sentença incompleta, como: “*Pedro vai à _____*”. Com um modelo de linguagem, poderemos completar a frase com novos termos, como: “*missa*”, “*praia*”, “*festa*”, etc. A resposta para tal questão dependerá basicamente do modelo de linguagem proposto e principalmente do corpus no qual o modelo foi treinado.

Modelos de linguagem são fundamentais para inúmeras aplicações computacionais rotineiras. Os casos de uso mais intuitivos são os serviços de auto-preenchimento, bastante utilizados em aplicativos de celulares para trocas de mensagens de texto. Entretanto, também são utilizados para correção automática, transcrição de áudio para texto, tradução automática, entres outros.

Para entender como os modelos de linguagem são obtidos, começaremos formalizando a questão que tentam responder, ou seja, atribuir uma probabilidade a um termo \mathbf{t} , dado um histórico \mathbf{h} , isto é, propor uma $P(t | h)$. Pode-se representar uma sequência de N termos como $t_1, t_2, t_3, \dots, t_n$. Assim, a probabilidade que queremos computar agora pode ser escrita como a probabilidade conjunta $P(t_1, t_2, \dots, t_n)$. Considera-se também o **histórico** como sendo a sequência t_1, t_2, \dots, t_{n-1} ou também, de forma simplificada, t_1^{k-1} (que são simplesmente os respectivos termos anteriores de cada item da sequência).

Seguindo a regra da cadeia para probabilidades (Jurafsky, D. & Martin (2009)), podemos decompor a probabilidade conjunta que queremos utilizando a seguinte fórmula:

$$P(t_1, t_2, \dots, t_n) = P(t_1) \cdot P(t_2|t_1) \cdot P(t_3|t_1, t_2) \cdots P(t_n|t_{n-1}, t_{n-2}, \dots, t_1) = \prod_{k=1}^n P(t_k|t_1^{k-1}) \quad (3.3)$$

Para estimar as probabilidades de 3.3, podemos utilizar um corpus de referência (um livro, por exemplo), e observar todas as vezes em que os termos apareceram um após o outro, obtendo uma frequência relativa da sequência dada. Entretanto, a probabilidade de se encontrar exatamente essa sequência no corpus é muito baixa. Isso acontece pois a linguagem é um processo criativo e o número de combinações novas diferentes é imensurável. Entretanto, ainda queremos ser capazes de atribuir probabilidades para qualquer sequência. Desse modo, o que se pode fazer é uma aproximação.

O modelo de linguagem de *N-gramas* oferece uma aproximação de maneira intuitiva (Jurafsky, D. & Martin (2009)). A proposta do modelo é, basicamente, limitar o histórico a apenas alguns dos últimos termos, e não à sequência toda.

Observemos o modelo de *bigramas*. Ele consiste em aproximar a probabilidade de ocorrência de uma palavra dada uma sequência fazendo uso apenas da última palavra do histórico, ou seja:

$$P(t|t_1^{n-1}) \approx P(t|t_{n-1}) \quad (3.4)$$

Para estimar as probabilidades do modelo de *bigramas*, podemos utilizar as *contagens* ($C()$) em que os bigramas ocorrem no corpus. Em seguida, os termos devem ser adaptados para que os valores se enquadrem entre 0 e 1. Para tanto, basta dividir a contagem obtida pelo número total de vezes em que t_{n-1} apareceu no corpus:

$$P(t_n|w_{t-1}) = \frac{C(t_{n-1}, t_n)}{C(t_{n-1})} \quad (3.5)$$

É possível demonstrar que o estimador em 3.5 é também o estimador de *máxima verossimilhança* para 3.4, ou seja, aquele cuja fixação de parâmetros maximiza a probabilidade do corpus dado.

Considerando o exemplo de “*Pedro vai à _____*”, poderíamos estimar as probabilidades do próximo termo ser “*missa*”, por exemplo, contando todas as vezes que “*missa*” ocorreu após “*à*” no corpus. Para normalizar a contagem (restringir essa informação entre 0 e 1 para uma aproximação probabilística), dividimos o valor obtido pelo número total de ocorrências do termo “*à*”.

Também é possível aumentar o histórico e construir modelos de trigramas, quadrigramas, etc, basta aumentar o tamanho do histórico a ser considerado. Entretanto, deve-se considerar as implicações computacionais decorrentes da escolha de **N**. Isto se deve ao fato de que a memória necessária para computar o modelo de n-gramas cresce exponencialmente. Suponha um corpus cujo vocabulário seja um conjunto de 10.000 palavras. Para o modelo de bigramas, é necessário computar 10.000^2 probabilidades; um trigrama 10.000^3 , e assim por diante.

3.4.2 Redes Neurais Recorrentes

Os modelos de Redes Neurais Recorrentes (em português, **RNR's** e em inglês **RNN's**) (Goodfellow, I. & Bengio (2016)) possuem uma arquitetura que favorece o armazenamento do histórico de termos precedentes, sendo assim muito útil para a criação de modelos de linguagem pois não necessita de uma limitação de janela, como no caso dos modelos de *N-Gramas*. A ideia central dessa arquitetura consiste na retroalimentação dos elementos sequenciais, de modo que o *input* de cada um deles serve, não somente para a previsão do próximo item da sequência, mas também para a formação de um componente intermediário, um *estado*. Esses estados, representados na Figura 3.8 como **h**'s são na prática matrizes, e funcionam como uma espécie de memória condensada dos elementos precedentes. Eles também entram como *input* para os estados posteriores. Essa é uma maneira de retransmitir a cada momento os efeitos dos *inputs* anteriores para o restante da sequência (Goodfellow, I. & Bengio (2016)).

No grafo da Fig. 3.8, considere x e \hat{y} como vetores de dimensão qualquer (porém, fixa). O estado **h**, como comentado, é uma matriz. As setas do grafo representam as matrizes de pesos, as mesmas introduzidas na Seç. 3.2. Os estados são calculados a partir da equação recorrente:

$$\mathbf{h}^{(t)} = g(\mathbf{h}^{(t-1)}, \mathbf{x}^{(t)}) \quad (3.6)$$

O estado **h(0)** é normalmente inicializado de maneira aleatória e entra, em conjunto com o primeiro *input* (o primeiro termo da sequência), no estado **h(1)**. O alvo desse primeiro

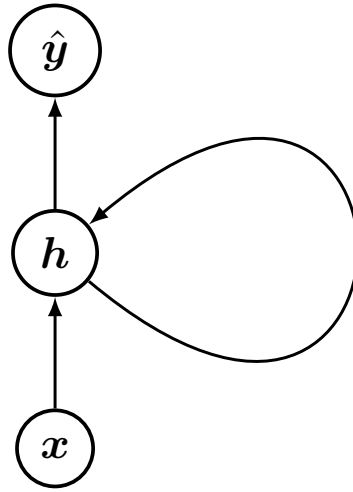


Figura 3.8: *Grafo Computacional de uma Rede Neural Recorrente*

passo é o segundo termo da sequência ($\hat{\mathbf{y}}^{(1)}$). Em seguida, o segundo termo da sequência tem como seu respectivo alvo o próximo termo e assim por diante. Após o treinamento da rede e a aplicação do algoritmo de *backpropagation*, espera-se que o último estado tenha incorporado uma certa memória de todos os estados anteriores e capturado as relações de dependência entre os termos.

3.4.3 Modelos de Linguagem com RNR's

Assim como no Cap. 2 foi necessário encontrar representações vetoriais para os verbos para que os mesmos servissem como *inputs* para os modelos, o mesmo deve ser feito com as palavras de um corpus para a construção dos modelos de linguagem. A representação vetorial mais intuitiva para a representação de palavras é conhecida como *One-hot Encoding*. Nesta representação, um vetor com o tamanho do vocabulário é inicializado com valores nulos. Cada termo é associado a uma dimensão específica do vetor, e portanto, para representá-lo, o valor 1 é atribuído na dimensão correspondente (Harris & Harris (2013)).

Desse modo, retomemos o exemplo “*Pedro vai à*”. Cada palavra terá a sua própria representação vetorial. A Fig. 3.9 exemplifica uma possível representação para a palavra “*missa*”. Assim, os vetores de *one-hot encoding* são inseridos no modelo seguindo a sequência em que as palavras aparecem nas sentenças do corpus de treinamento.

A Fig. 3.10 sintetiza a arquitetura da RNR¹ aplicada para a construção de um modelo de linguagem. Para o treinamento do modelo, é construída uma lista com todas as sentenças

¹As RNR's na verdade constituem uma classe de arquiteturas. Nessa classe, enquadram-se a *Long-Short Term Memory* (LSTM) (Hochreiter, S. & Schmidhuber (1997)) e também a *Gated Recurrent Unit* (GRU) (Ver Patterson & Gibson (2017) para maior detalhamento sobre essas arquiteturas).

0	Pedro
0	João
0	livro
1	missa
⋮	⋮
0	festa
0	amigo
0	do

Figura 3.9: *Representação One-hot Encoding*

do corpus. Em seguida, acrescentam-se a cada sentença termos para marcar o início e o final das mesmas.² No primeiro passo do treinamento, alimenta-se a rede com o vetor associado ao termo *<beg>*, e associado a ele, o seu alvo, isto é, a primeira palavra da sentença. Em seguida, o termo apresentado como alvo entrará como *input* e o seu respectivo alvo será o próximo termo da sentença, e assim por diante até o termo *<eos>*. Após o treinamento e calibragem dos pesos da rede, espera-se que o modelo resultante possa agora servir para estimarmos probabilidades de sequências ou também gerarmos sequências completamente novas.

Para o caso de uma tarefa de geração de sentenças, o modelo escolherá como próxima palavra aquela que possui maior probabilidade associada. A predição tem início com a apresentação do vetor de início, o *<beg>*. Após a inserção desse termo, o modelo passará a escolher os termos mais prováveis de acordo com o que foi aprendido durante o treinamento. Para que a sequência tenha uma conclusão e o modelo não fique predizendo valores indefinidamente, em determinado momento o modelo terá como termo mais provável o termo de final de sequência (o *<eos>*), e assim, conclui-se a tarefa de predição.

²Esses termos já foram apresentados no Cap. 2 durante a composição do pré-processamento do modelo (*<beg>* e *<eos>*).

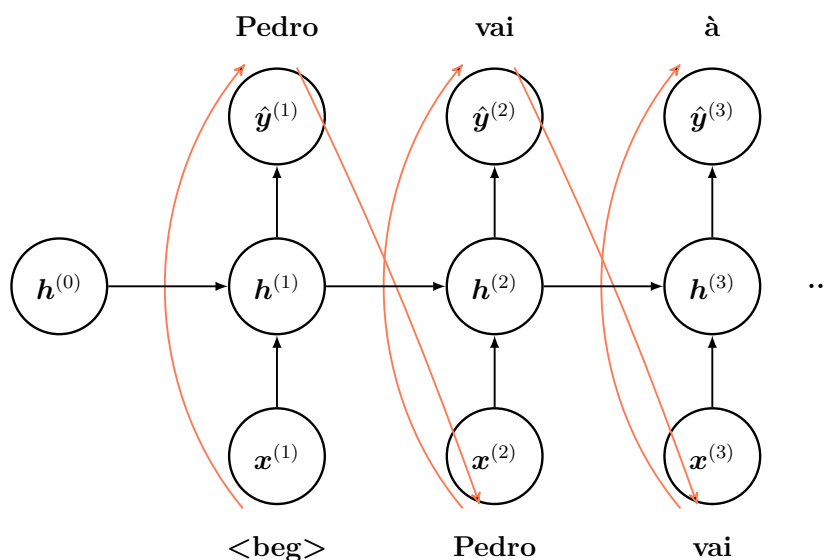


Figura 3.10: *Modelo de Linguagem com Rede Neural Recorrente*

3.5 Conclusão do capítulo

O objetivo deste capítulo foi introduzir o leitor aos modelos de Redes Neurais e apresentar conceitos teóricos fundamentais para o entendimento do modelo *Encoder-Decoder*, que é o ponto central desta pesquisa. Para isso, vimos que os modelos de Redes Neurais são ditos modelos de *aprendizado de máquina supervisionados*, pois utilizam-se *alvos* durante o treinamento dos mesmos. Também foram introduzidos os chamados *hiperparâmetros*, que são valores que devem ser definidos *a priori* para a configuração dos modelos. Além disso, introduzimos a noção de modelo de linguagem, necessária para a apresentação do modelo de **RNR**'s que ocorreu na sequência.

Com esses conceitos em mente, estamos prontos para entender a arquitetura do *Encoder-Decoder*.

Capítulo 4

Encoder-Decoder

Neste capítulo, será introduzida a arquitetura do modelo *Encoder-Decoder*. Além disso, falaremos sobre a aplicação da arquitetura aplicada ao problema em questão, ou seja, a proposição de um modelo para a predição de formas verbais flexionadas (no paradigma já estabelecido). Por último, será apresentado o algoritmo de *Pós-Processamento* adotado nesta pesquisa.

4.1 Introdução ao modelo Encoder-Decoder

Um modelo de mapeamento *Encoder-Decoder* (Bahdanau, D. (2014)) é um sistema composto por duas RNR's cuja principal função é mapear uma sequência à outra. Modelos do tipo *Encoder-Decoder* (também conhecidos como *Seq2Seq* (Sutskever (2014))) têm sido bastante utilizados em tarefas linguísticas, especialmente no desenvolvimento de sistemas de diálogo e em tradução automática.

Em um contexto de tradução, por exemplo, o modelo recebe como *input* uma sequência de uma língua de origem e produz como *output* uma sequência em uma língua alvo. A sequência gerada precisa, além de preservar o conteúdo semântico da sequência de origem, apresentar uma sintaxe aceita pelos falantes da língua alvo.

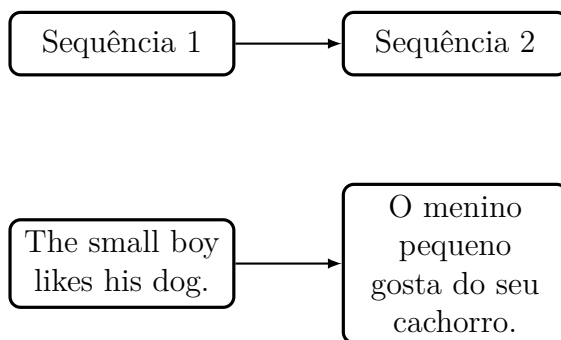


Figura 4.1: *Objetivo do Modelo Encoder-Decoder: Mapeamento de uma Sequência à Outra*

Observa-se no caso da tradução, que não há uma correspondência exata entre os termos

de cada uma das línguas. A sentença na língua inglesa possui um termo a menos, sendo que o pronome “his” corresponde aos termos “do seu” no português. Além disso, observa-se que cada uma das línguas possui uma sintaxe diferente. No primeiro caso, o adjetivo *small* se posiciona antes do substantivo *boy*. No segundo, essa ordem é invertida (*menino pequeno*).

As duas redes recorrentes funcionam da seguinte maneira (ver Fig. 4.4): uma primeira rede, a denominada *Encoder*, é alimentada com uma sentença da língua de origem (em inglês, por exemplo). Os termos dessa sentença entram um após o outro na rede e alimentam os estados ($\mathbf{h}^{(t)}$), porém nesse caso não é necessário relacionar cada termo a um correspondente (y_t) como no caso do modelo de linguagem. A rede *Encoder*, portanto, não possui *alvos*, ela serve apenas para acumular os dados da língua de origem. Assim, o resultado do último estado, gerado após a inserção de todos os termos da sequência de origem, servirá como estado inicial ($\mathbf{h}^{(0)}$) para uma segunda rede recorrente, denominada de *Decoder*. A rede *Decoder*, por sua vez, recebe como seu primeiro *input*, um marcador que representa o início da sequência alvo ($\langle \text{beg} \rangle$), inicializando o modelo de linguagem dessa língua e com ordem de parada ao predizer o marcador de final de palavra ($\langle \text{eos} \rangle$), como no modelo de rede neural recorrente apresentado na Seção 3.4.2.

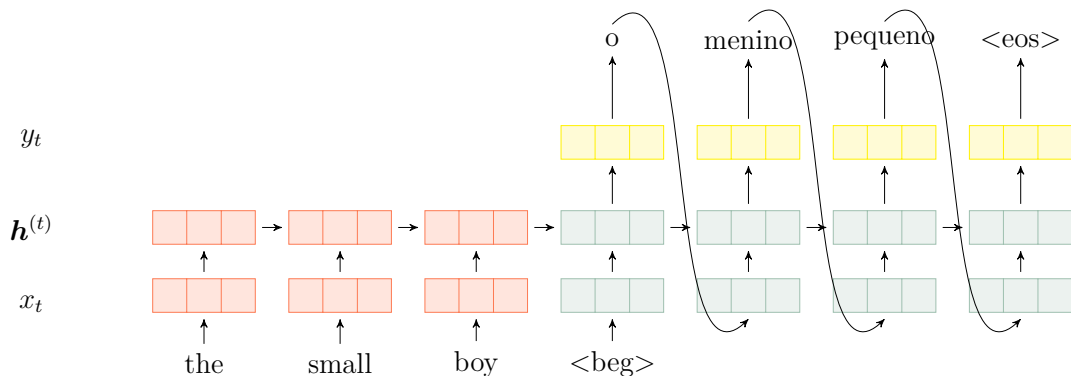


Figura 4.2: Esquema de Encoder-Decoder para Tradução Automática

Por fim, o aprendizado do modelo ocorre como já mencionado no Capítulo 3: os *outputs* do modelo são comparados com os *alvos*, o erro é propagado de volta através do algoritmo de *backpropagation*, os pesos da rede são atualizados de modo a diminuir esse erro e o processo se repete até a conclusão de todas as *épocas*. É importante ressaltar também que, embora a figura ilustre como exemplo duas sequências de três palavras, o modelo comporta sequências de tamanhos quaisquer e que não precisam coincidir entre si.

4.2 Encoder-Decoder e flexão verbal

Assim como no problema da tradução automática, pode-se retratar a questão do aprendizado de flexão dos verbos através de uma relação entre duas sequências. No caso desta

pesquisa, pensaremos nessa relação como partindo de uma forma base para a forma flexionada escolhida (como mencionado na Seq. 1.2.4).

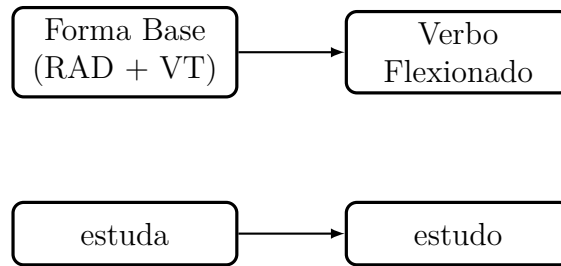


Figura 4.3: *Relação entre um Verbo na Forma Base e o Mesmo Flexionado*

Diferentemente do caso de tradução, em que se busca o aprendizado das relações entre uma sentença e outra das línguas de origem e de alvo, no caso do aprendizado de flexão de verbos busca-se aprender as relações existentes entre uma forma verbal básica e uma flexionada a nível fonético. Desse modo, resta explicar como se deu a transformação dos verbos em vetores adequados para que o modelo fosse capaz de aprender tais relações.

4.2.1 A rede Encoder desenvolvida

Relembrando a Seq 2.3.1, temos que cada fone é representado através de uma forma vetorial com 20 dimensões (correspondentes ao número de traços fonéticos do estudo). Desse modo, a camada de *inputs* da rede *Encoder* tem dimensão 20, e é responsável por receber os verbos na forma base (RAD + VT). Em seguida, o *input* é direcionado para uma rede de tipo LSTM *unidirecional*. Unidirecional refere-se a um mecanismo que regula o foco de atenção da rede. Nessa configuração, os *inputs* são inseridos sequencialmente seguindo o fluxo padrão (do primeiro ao último termo da sequência). Existe também o tipo *bidirecional*, em que os *inputs* são inseridos de forma dupla, seguindo o fluxo padrão e também o inverso (Schuster & Paliwal (1997)).

4.2.2 A rede Decoder desenvolvida

A rede *Decoder*, por sua vez, funciona como um modelo de linguagem. O estado inicial desse modelo ($\mathbf{h}^{(0)}$) é o estado final gerado pela rede *Encoder*. Além disso, também entram como *input* na rede os respectivos verbos flexionados. A rede recorrente utilizada também é de tipo LSTM unidirecional.

A Figura 4.4 apresenta um esquema do modelo desenvolvido utilizando como exemplo o verbo irregular “*ler*” (já com a exclusão da desinência de infinitivo).

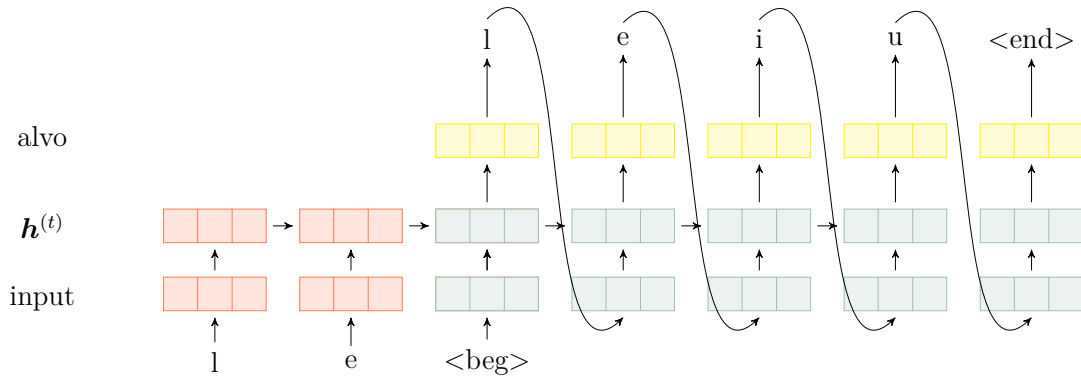


Figura 4.4: *Esquema do Encoder-Decoder Desenvolvido*

4.2.3 Pós-processamento

Assim como é necessária uma etapa de pré-processamento para transformar os verbos em vetores numéricos para a alimentação do modelo, também é necessário traduzir a saída do modelo (que neste momento corresponde a uma sequência de vetores numéricos) para uma sequência de caracteres e para a reconstrução do verbo flexionado predito.

Primeiramente, temos que a predição do verbo flexionado é o resultado da concatenação das predições do modelo de linguagem aprendido durante o treinamento. Dessa forma, para realizar uma predição, o algoritmo libera um vetor de cada vez. O algoritmo finaliza a predição de fones de um verbo assim que prevê um marcador de final. Tais predições não constituem mais vetores binários, ao invés disso, o modelo solta vetores com valores variando de 0 a 1 (conforme a função de ativação utilizada). Por essa razão, para o pós-processamento dos vetores de saída são feitas duas aproximações: (i) valores abaixo de 0.5 foram substituídos por 0 e valores acima são substituídos por 1; (ii) o resultado dessa operação pode não resultar em um fone válido, com por exemplo um vetor com presença de dois traços contraditórios (ex: fricativo e oclusivo) e portanto é substituído (se necessário) pela representação vetorial cuja distância seja mínima quando comparada a todos os fones. Aqui, portanto, vemos que a restrição quanto à validade dos fones envolve um conhecimento externo ao modelo proposto, por parte da linguística. De todo modo, esse critério teve de ser adotado para viabilizar o algoritmo de pós-processamento.

Na álgebra linear existem várias maneiras para se medir a distância entre dois vetores. Nesta pesquisa, optou-se arbitrariamente pela *distância euclidiana* (ver Boulos & Oliveira (2009) para maiores detalhes). Entretanto, é importante atentar para o fato de que há um custo computacional embutido nessa operação: calculam-se as distâncias entre cada vetor predito pelo modelo e as 30 representações vetoriais possíveis.

A distância euclidiana entre os pontos \mathbf{p} e \mathbf{q} é equivalente ao comprimento do segmento de reta que os conecta ($\overline{\mathbf{pq}}$). Utilizando coordenadas cartesianas, e sejam $\mathbf{q} = (q_1, q_2, \dots, q_n)$ e $\mathbf{p} = (p_1, p_2, \dots, p_n)$ dois pontos em um espaço n -dimensional, temos:

$$\begin{aligned}
 d(\mathbf{q}, \mathbf{p}) &= \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \dots + (q_n - p_n)^2} \\
 &= \sqrt{\sum_{i=1}^n (q_i - p_i)^2}.
 \end{aligned} \tag{4.1}$$

Desse modo, o esquema apresentado na Fig. 4.5 apresenta um resumo do processo de pós-processamento utilizado neste trabalho.

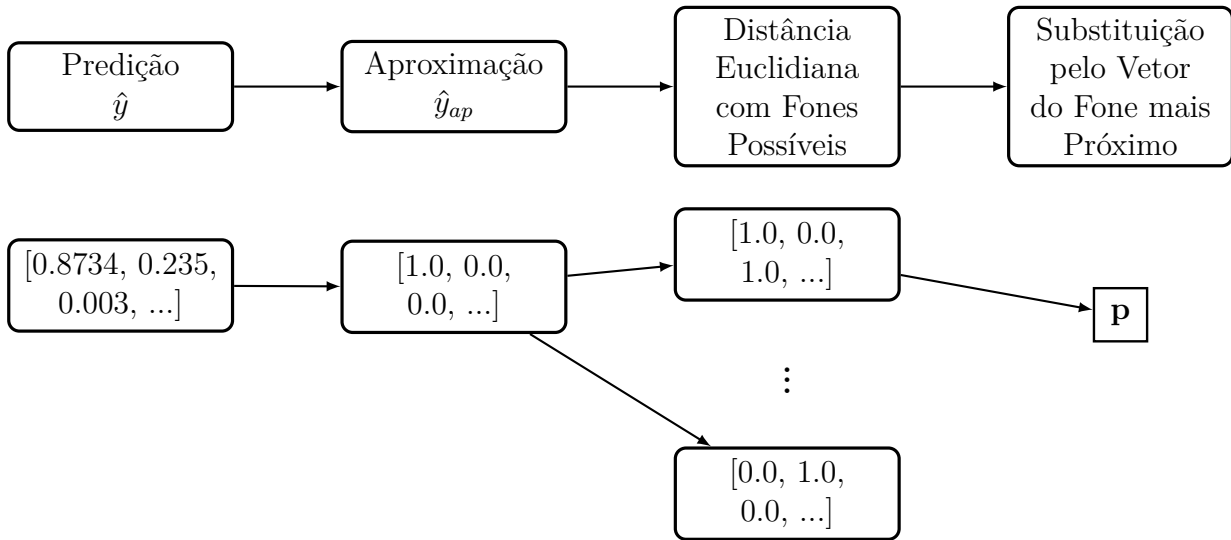


Figura 4.5: Pós-Processamento do Modelo

Capítulo 5

O Experimento

Neste capítulo falaremos sobre o modelo *Encoder-Decoder* aplicado, bem como os hiperparâmetros utilizados. Além disso, será exibida a metodologia adotada para avaliação dos resultados e discussões sobre os resultados obtidos.

5.1 Hiperparâmetros

Os hiperparâmetros selecionados para a realização do experimento foram principalmente baseados em uma tarefa similar de Chollet (2017b), em que textos são traduzidos de uma língua para outra utilizando o nível de caracter como referência.

Após alguns experimentos, o número de *épocas* do modelo foi definido como 300. A escolha se deu após a verificação de que após esse valor o modelo tinha seu aprendizado estacionado.

Sem entrar em detalhes teóricos, também vale comentar os demais hiperparâmetros utilizados. As redes *Encoder* e *Decoder* são ambas arquiteturas do tipo LSTM com dimensão latente de 256 unidades cada (ou seja, apresentam uma camada intermediária com 256 nós). Para a otimização do modelo, foi utilizado o algoritmo *Adam* (Kingma, D. P. & Ba (2014)), disponível na API do *Keras* (Chollet *et al.* (2015)) com hiperparâmetros pré-definidos (taxa de aprendizado = 0.001, $\beta_1=0.9$, $\beta_2=0.999$, ϵ = Vazio, Decaimento = 0.0, amsgrad = Falso). A função de custo utilizada foi a de Entropia Cruzada Binária (Chollet (2017a)), já que temos que classificar uma série de valores como sendo **0**'s ou **1**'s. Além disso, o treinamento foi realizado em lotes (*batches*) de 128 verbos com uma taxa de 20% para validação cruzada (*cross-validation*). Por último, ainda baseado na tarefa de Chollet (2017b), uma camada intermediária foi acrescentada entre o *Decoder* e a camada de *output*.

A Fig. 5.1 exibe a arquitetura completa do modelo.

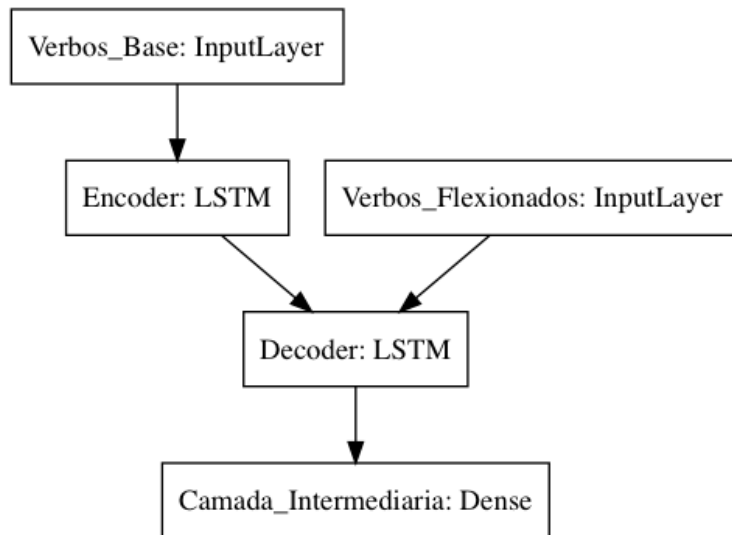


Figura 5.1: *Arquitetura Final Utilizada.*

5.2 Metodologia para avaliação

Para a avaliação dos resultados do modelo, foi utilizada uma técnica de validação cruzada chamada *K-Fold* (Raschka (2018)). A análise K-Fold permite que todos os verbos do corpus sejam testados. A técnica consiste, primeiramente, na formação de K subconjuntos de verbos mutuamente exclusivos de tamanhos próximos (idênticos caso a divisão por K seja exata). Em seguida, um desses subconjuntos é escolhido como o conjunto de teste enquanto que os K-1 restantes são utilizados como treino. O tamanho de K é definido a partir da escolha de proporção em que se deseja realizar a segmentação entre treino e teste, ou seja, para sempre manter a proporção de testes em 20% de modo que estes verbos sejam sempre diferentes, o corpus precisa ser segmentado em 5 subconjuntos distintos. O desenho 5.2 exibe o particionamento dos *datasets* segundo à proposta do algoritmo.

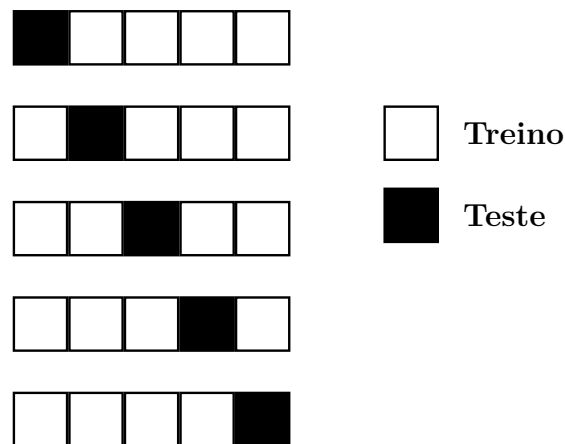


Figura 5.2: *Técnica de segmentação do Corpus via K-Fold*

Além disso, há ainda a vantagem do algoritmo de estratificação, ou seja, a certeza de que, em cada um dos treinamentos, as proporções das classes de verbos se mantenham no

treinamento. Isso significa que, por exemplo, a classe de irregularidades do verbo “*testar*” (testar → testo) que possui 20 verbos, terá sempre 16 verbos no treinamento e 4 no teste. Assim, após os 5 treinamentos diferentes, todos os verbos da classe foram testados e fica garantido que havia verbos dessa classe no treinamento.

5.2.1 Métrica de avaliação

Para a avaliação dos resultados, considerou-se a métrica de *Acurácia*. A acurácia pode ser obtida através da fórmula:

$$Acurácia = \frac{contagem(acertos)}{contagem(total)} \quad (5.1)$$

Ainda, define-se como um acerto um verbo predito exatamente igual ao esperado.

5.3 Resultados

Como há um fator de aleatoriedade nas inicializações dos pesos, e também como os verbos de treino e teste são sorteados pelo algoritmo no momento da segmentação do K-Fold, os resultados da rede podem variar. Para a verificação dos resultados levando em consideração as variações, foram realizadas 30 análises K-Fold. Isso significa que cada verbo foi testado 30 vezes. Em média, temos que o modelo acerta 13.55% dos verbos. Entretanto, a Fig. 5.3 mostra um gráfico do tipo *boxplot* (Morettin, P. A. & Bussab (2004)) onde podemos verificar a variação da acurácia. Na figura, podemos notar que a taxa média de acerto do modelo desenvolvido se concentra entre 12.5-14.7%, mas chega até 17%.

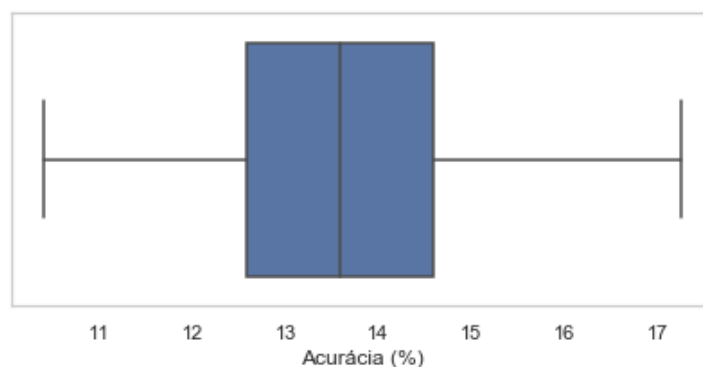
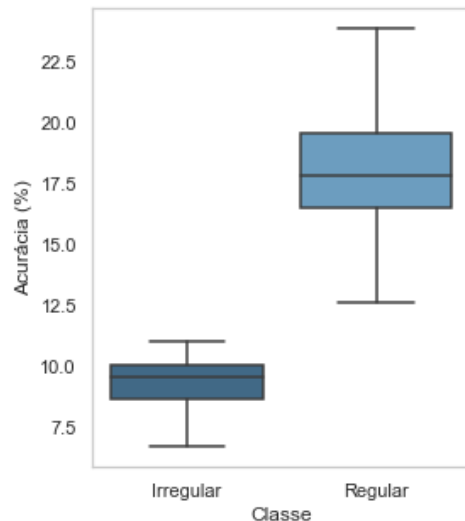


Figura 5.3: *Acurácia Para Todos os Verbos*

Ainda, agrupando todas as classes irregulares em apenas uma, é possível avaliar o desempenho do modelo comparativamente entre o grupo dos verbos regulares e o grupo de irregulares. Nesse caso, a Fig. 5.4 e a Tab. 5.1 mostram que o modelo obteve resultados melhores para a classe dos verbos regulares.



	Regulares	Irregulares
Acurácia Média	17.88 %	9.23 %
Acurácia DP	2.30 %	1.15 %
Acurácia Min	12.65 %	6.70 %
Acurácia Max	23.83 %	11.00 %

Figura 5.4: *Boxplots Para Acurácias por Classe* **Tabela 5.1:** *Acurácias Médias por Classe com Desvio Padrão (DP)*

5.3.1 Acurácia em cada classe

Também é interessante avaliar o desempenho do modelo em cada uma das 16 classes formadas. Na Fig. 5.6 observamos as acurácias de cada classe ordenadas pela proporção das mesmas no corpus. Nota-se que, apesar de a classe regular apresentar a maior proporção no corpus, as classes com maior acurácia são as classes dos verbos “botar” e “seguir”. Para avaliar a correlação entre a proporção no corpus e a acurácia foi utilizado o coeficiente de correlação de Pearson (Morettin, P. A. & Bussab (2004)). Esse coeficiente assume valores entre -1 e 1, sendo que valores próximos às extremidades indicam maior correlação e valores próximos a zero indicam que as variáveis independem linearmente uma da outra. Para a interpretação do valor obtido utilizaremos a seguinte escala (Clarke GM (1989)):

- 0.9 positivo ou negativo indica uma correlação muito forte.
- 0.7 a 0.9 positivo ou negativo indica uma correlação forte.
- 0.5 a 0.7 positivo ou negativo indica uma correlação moderada.
- 0.3 a 0.5 positivo ou negativo indica uma correlação fraca.
- 0 a 0.3 positivo ou negativo indica uma correlação desprezível.

O coeficiente de correlação observado entre essas variáveis foi de **0.42**, o que indica uma correlação *fraca*.

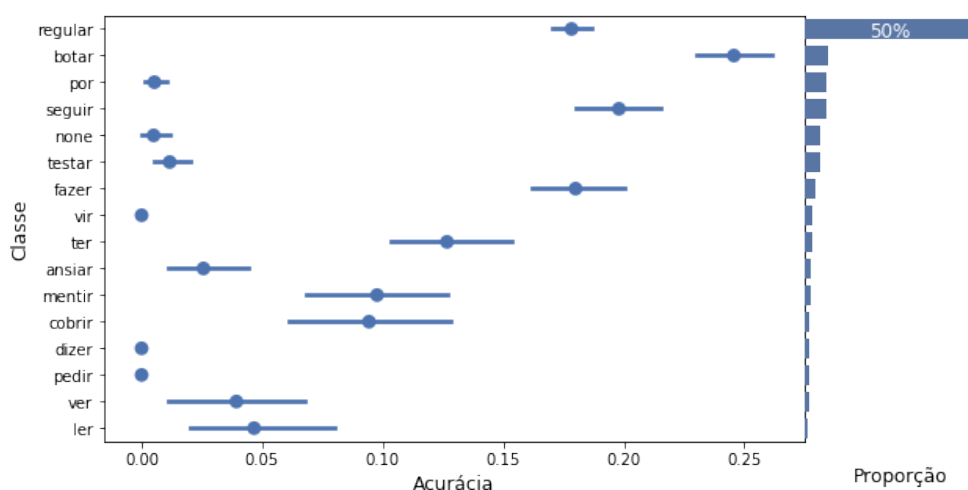


Figura 5.5: *Acurácia Para Cada Classe*

Para a análise dos resultados obtidos, utilizaremos o experimento K-Fold de acurácia total mais alta como referência (17%, vide apêndice C). Como se pode observar, diferentes tipos de erros podem acontecer, sendo que alguns são piores do que outros. Vejamos, por exemplo, o verbo “parecer” (com transcrição fonética e RAD + VT = “parese”). Nesse caso vemos que o modelo produziu a forma “peresu”. Nota-se que essa transformação em que o modelo errou alguns traços de um único fone, é diferente do erro que ocorre com o verbo “retrair” (com transcrição fonética e RAD + VT = “hetrai”), para o qual o modelo produziu a forma “setEruu”. Vemos que no segundo caso, a forma produzida é praticamente incompreensível.

Ao analisar os erros ocorridos na classe do verbo “Pôr” (Tab. 5.2), verifica-se que 7/26 dos erros cometidos foram erros de *regularização*, ou seja, o modelo se confundiu com a classe mais presente (a classe dos verbos regulares) e manteve o padrão de flexão regular.

Input	Output	Alvo
komp	komp	kompNu
despo	despu	despoNu
espo	espu	espoNu
antepo	antepu	antepoNu
supo	supu	supoNu
prosupo	prosupu	prosupoNu
propo	propu	propoNu

Tabela 5.2: *Erros de regularização na classe do verbo “Pôr”*

Em seguida, analisamos os verbos sem agrupamento (na Fig. 5.6, “none”). A acurácia baixa nesta classe era esperada pois não havia outros verbos no corpus para que o modelo conseguisse detectar possíveis padrões para fazer predições corretas. A Tab. 5.3 exhibe alguns exemplos de erros nesta classe.

Input	Output	Alvo
kabe	kabu	kaibu
idea	adeiu	ideiu
esta	estu	estou
traze	trasu	tragu
prove	provu	proveNu
estrea	estrei	estrEiu

Tabela 5.3: *Exemplos de erros na classe sem agrupamento*

Exemplos de erros na classe do verbo “testar” podem ser vistos na Tab. 5.4. Chama a atenção que, para o verbo “pegar”, o modelo acertou a flexão irregular transformando a vogal anterior meio-fechada em meio-aberta ($e \rightarrow \varepsilon$), porém não acertou o traço de vozeamento para caracterizar o fone “g” ao invés de “k”, o que resultou na flexão equivocada de pegar \rightarrow pEku.

Input	Output	Alvo
seka	seku	sEku
leva	levu	lEvu
sega	segu	sEgu
fexa	fexu	fExu
pega	pEku	pEgu

Tabela 5.4: *Exemplos de erros na classe do verbo “testar”*

A classe do verbo “vir”, por sua vez, apresentou 5/11 de erros por regularização. Além disso, é interessante que para o verbo “revir” (*hevi*) tenha flexionado como “*heviju*”, flexão próxima ao grupo do verbo “ver”. A Tab. 5.5 exhibe alguns exemplos de erros nesta classe.

Input	Output	Alvo
sobrevi	sobrevu	sobreveNu
hevi	heviju	heveNu
adivi	adivu	adiveNu
avi	aviu	aveNu
provi	provu	proveNu

Tabela 5.5: *Exemplos de erros da classe “vir”*

5.3.2 Acurácia e comprimento médio dos verbos nas classes

Como o coeficiente de *Pearson* observado entre a proporção das classes no corpus e a acurácia indicou uma correlação fraca, podemos buscar por outras variáveis que também possam influenciar no desempenho do modelo. Uma delas é o comprimento dos verbos. O verbo “3ntret3Nu” (entretenho), por exemplo, contém 9 fones para serem preditos. Como cada fone contém 20 traços, no total o modelo tem que conseguir prever 180 números. O verbo “falu” (falo), em contrapartida, possui 4 fones, ou seja, 80 números a serem preditos.

Dessa forma, é natural pensar que quanto maior o número de predições necessárias, maior a probabilidade do modelo cometer algum erro. Com isso, a Figura 5.6 exibe as acurácias das classes ordenadas de acordo com os comprimentos médios das mesmas. Na figura vemos que os grupos apresentam comprimentos médios que variam de 5 a 8 fones, mas com barras de confiança com limites próximos, indicando uma certa homogeneidade nesse aspecto.

No caso desse estudo, o coeficiente de *Pearson* obtido foi de **0.39**, o que indica uma correlação também fraca entre estas variáveis. Entretanto, vemos nas barras de confiança que há pouca diferença significativa entre os comprimentos médios das classes. Além disso, é interessante notar que a classe em que o modelo apresentou melhor desempenho (a classe de “botar”) é simultaneamente uma classe com alta proporção no corpus e também apresenta o comprimento médio mais baixo do grupo.

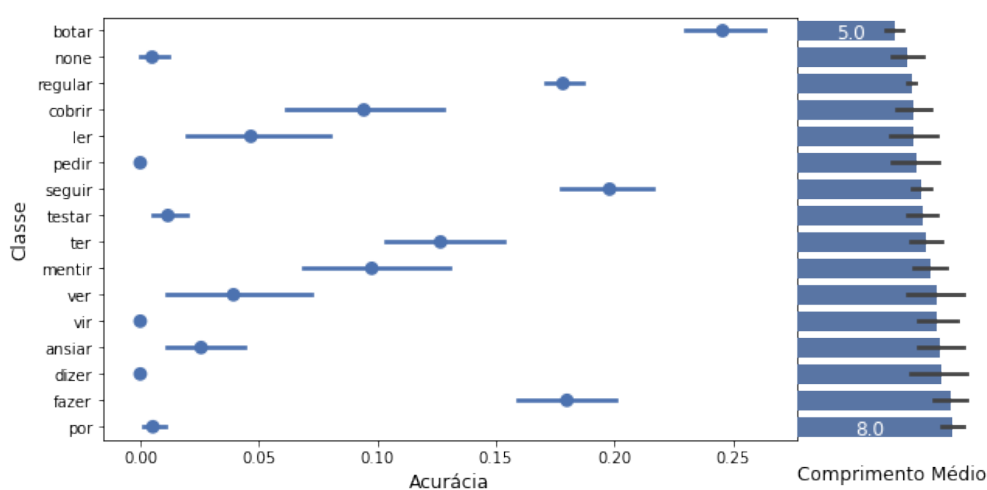


Figura 5.6: Acurácia por Comprimento Médio

5.4 Outros erros relevantes

Alguns erros interessantes como troca de classes e regularizações já foram apresentados na Seção 5.3.1, mas ainda há outros erros que merecem destaque. (Ver Apêndice C para tabela completa com Classe, Input, Output e Alvo)

Na classe de verbos sem agrupamento, um erro notável foi a flexão realizada para o verbo trazer (“trasu”), o que mostra que o modelo identificou o padrão de flexão da classe do verbo “fazer”.

Na classe do verbo “pedir”, o modelo apresentou a flexão “espиду” para o verbo “expedir” (com alvo “espEsu” - “expeço”), o que denota uma possível confusão com a classe do verbo “conseguir”.

Nos verbos regulares nota-se a presença de vários erros devido à troca de apenas um traço fonético. Como por exemplo para o verbo “convidar”, o *output* resultante foi “konviru” (trocou [d] - oclusiva, alveolar, sonora; por [r] - tepe, alveolar, sonora) . Para o verbo “convencer”, “konfensu” (trocou [v] - fricativa, lab. dental, sonora; por [f] - fricativa, lab.

dental, surda).

Para os verbos da classe do verbo “seguir”, três verbos foram flexionados de acordo com a família do verbo “testar”. São eles: “ferir” (fEru), “vestir” (vEstu) e “repetir” (hepEtu).

A classe do verbo “ver” também apresentou erros interessantes: a confusão com a classe do verbo “vir” nos verbos “prever” (“preveNu”) e “entrever” (“entreveNu”). Também não acertou o alvo do verbo “rever” (hefexu), ficou faltando o traço de sonoridade nas duas últimas consoantes.

5.5 Discussão

Durante a execução do presente trabalho, foi divulgado um estudo semelhante realizado pelos pesquisadores Kirov & Cotterell (2018). Nesse estudo, Kirov e Cotterell revisitam a questão dos verbos irregulares do inglês utilizando a arquitetura *Encoder-Decoder*. Em razão da similaridade entre os estudos, é interessante pautar a discussão dos resultados obtidos através de uma comparação entre os mesmos.

No grupo dos verbos regulares, Kirov & Cotterell (2018) obtiveram um desempenho próximo a 100%. Entretanto, ao observarmos exclusivamente o desempenho no grupo dos verbos irregulares, a acurácia cai para 28.6%. Kirov & Cotterell (2018) explicam que os erros obtidos nessa circunstância foram erros de *regularização*, e que em nenhum momento o modelo misturou regulares e irregulares como em “*gaved*” (erro observado e criticado no modelo de Rumelhart & McClelland (1986)). Em números absolutos, os autores apresentaram um corpus composto por 4039 verbos (aproximadamente 10 vezes maior que o utilizado nesta pesquisa), sendo 168 destes considerados irregulares. Além disso, Kirov & Cotterell (2018) realizaram uma partição aleatória tripla no corpus com proporções 80-10-10 para treino, desenvolvimento e teste. Desse modo, a base de teste dos pesquisadores continha apenas em torno de 17 verbos irregulares. Com uma acurácia de 28%, isso significa que o modelo *Encoder-Decoder* apresentado pelos autores acertou apenas cinco verbos irregulares. Sobre estes verbos, três deles eram verbos derivados de outros através de prefixação: *retell* (derivado de *tell*), *partake* (derivado de *take*) e *withdraw* (derivado de *draw*). Outro era o verbo *sling*, similar a outros verbos presentes no treino (*fling*, por exemplo). O último foi o verbo *forsake*, cuja terminação se assemelha bastante ao verbo *take*.

Além do modelo *Encoder-Decoder* apresentado, Kirov & Cotterell (2018) também reproduziram o modelo de regras de Albright, A. & Hayes (2003) (comentado na Seq. 1.2.2) como referência (*benchmark*) utilizando o mesmo corpus. Segundo os autores, para este modelo a acurácia dentre os verbos regulares ficou em torno de 95%, porém não foi capaz de acertar nenhum verbo irregular em nenhuma das três partições realizadas.

Em termos de pré-processamento, Kirov & Cotterell (2018) não explicitam a codificação utilizada, mas explicam que não utilizaram traços fonéticos como dados de entrada para a rede. A base de dados utilizada pelos autores foi a *CELEX*, retirada de Baayen *et al.* (1993),

e era composta pelos verbos transcritos com a notação do AFI.

Voltando ao modelo desenvolvido nesta pesquisa, observamos que a acurácia obtida no grupo dos verbos regulares foi a mais baixa em comparação aos demais trabalhos aqui referidos (com uma acurácia média igual a 13.55% e máxima = 17%). Entretanto, isto pode ser explicado em razão do tamanho do corpus necessário para o aprendizado em arquiteturas mais complexas, como é o caso do *Encoder-Decoder*. Como visto, o modelo de Kirov & Cotterell (2018) utilizou 4039 verbos para a tarefa (um corpus 10 vezes maior que o utilizado nesta pesquisa). Entretanto, como a ordem de grandeza obtida (423 verbos) estava próxima à ordem do conjunto de Rumelhart & McClelland (1986) (506 verbos), acreditava-se que o corpus obtido seria suficiente para a obtenção de acurácias mais altas.

Com relação aos verbos irregulares, apesar da acurácia média obtida ter sido 9.23%, em números absolutos isso representa em torno de 20 verbos. Dessa forma, é difícil comparar os dois modelos nessa questão. Se, por um lado, o montante de verbos regulares obtidos por Kirov & Cotterell (2018) os levou a quase 100% de acurácia nesse grupo, também foi um fator desfavorável para o desempenho no grupo dos verbos irregulares. Também pode-se questionar se os cinco verbos corretos obtidos por Kirov & Cotterell (2018) refletem de fato o potencial de 23% de acurácia do modelo. Certamente a discussão seria mais interessante caso tivéssemos uma acurácia média com desvio-padrões, como foi apresentado neste trabalho. Em comparação com os resultados do *benchmark* de Kirov & Cotterell (2018) (baseado no modelo de Albright, A. & Hayes (2003)), o modelo apresentado nesta pesquisa apresentou desempenho melhor nesse quesito (9.23% contra 0%).

No que diz respeito aos tipos de erros encontrados, pode-se dizer que alguns erros observados nesta pesquisa mostram-se bastante incompreensíveis de um ponto de vista linguístico em comparação aos apresentados por Kirov & Cotterell (2018) (erros como “*aAsaxu*”, como *output* para o verbo “*analisar*”). Entretanto, como os processos de codificação e decodificação, bem como os tamanhos dos *corpus* e os objetos dos estudos são diferentes (traços fonéticos x fonemas), também seria injusta uma comparação entre os modelos nesse sentido.

Sobre o fato das métricas das correlações obtidas entre acurácia e tamanho médio e proporção no corpus não terem se mostrado fortes, chama a atenção a classe do verbo “botar”. Essa classe, além ter apresentado a maior acurácia, apresenta também proporção no corpus alta e é também a classe com comprimento médio mais baixo. Desse modo, supõe-se que estes fatores combinados tenham levado a esse resultado. Uma análise multivariada poderia ser feita para investigar a suposição em mais profundidade.

Para concluir a discussão, pode-se dizer que o desempenho do modelo *Encoder-Decoder* está bastante condicionado à quantidade do corpus disponível. De certo que para qualquer modelo de Rede Neural, quanto maior o número de exemplos, melhor. Entretanto, vemos também que quanto mais profunda a arquitetura do modelo de Rede Neural, maior a demanda por mais exemplos. Ainda, vimos que a questão da proporção da classe de verbos irregulares mostrou-se relevante. Desse modo, os resultados obtidos nesta pesquisa não permitem dizer que o modelo *Encoder-Decoder* seja adequado para esta tarefa. Contudo, pode-se

argumentar que, como os processos de codificação e decodificação são módulos independentes da arquitetura, essa conclusão a respeito do modelo *Encoder-Decoder* seria injusta. Além disso, para que pudéssemos chegar a conclusões mais contundentes, mais experimentos deveriam ser realizados. Dito isto, direções para possíveis trabalhos futuros serão abordadas no próximo capítulo (Cap. 6).

Capítulo 6

Conclusão

Este trabalho teve como principal objetivo estudar o aprendizado de verbo irregulares do Português Brasileiro utilizando um modelo do tipo *Encoder-Decoder* (Bahdanau, D. (2014), Sutskever (2014)).

O escopo da pesquisa foi restrito à 1ª pessoa do singular, no tempo presente e modo indicativo. Em seguida, um corpus com 423 verbos foi criado e transcrito para uma representação fonética utilizando a metodologia desenvolvida no Capítulo 2. O corpus criado foi primeiramente particionado segundo classes de irregularidade. No total, quinze classes irregulares foram formadas. A proporção de verbos regulares e irregulares no corpus montado foi de respectivamente 50.6% e 49.4%. Após a coleta e organização dos verbos, os mesmos foram pré-processados para serem introduzidos na camada de *input* do modelo.

O modelo *Encoder-Decoder* foi configurado da seguinte maneira: As redes *Encoder* e *Decoder* são ambas redes recorrentes (do tipo LSTM) com uma camada escondida de 256 nós. Para o treinamento foram utilizadas 300 épocas com lotes de 128 verbos por vez. O modelo foi otimizado com o algoritmo *Adam* disponível na API do *Keras* (Chollet *et al.* (2015)) com hiperparâmetros pré-definidos pela API e disponíveis na documentação. A função de custo utilizada foi a Entropia Cruzada Binária (Chollet (2017a)).

Para a avaliação do modelo construído, foi utilizada a técnica de validação cruzada chamada *K-fold* (Raschka (2018)), que permite que todos os verbos do corpus sejam testados pelo modelo. A métrica de avaliação escolhida foi a acurácia. Para considerar as variações possíveis durante os treinamentos, o algoritmo K-fold foi executado trinta vezes. Desse modo, foi possível estudar o comportamento médio do modelo nas diferentes classes do estudo.

A acurácia máxima atingida pelo modelo foi de 17% considerando todos os verbos do corpus (73/423). Considerando apenas verbos regulares *versus* verbos irregulares, o desempenho do modelo foi melhor no primeiro grupo, sendo as respectivas acurácias médias 17.88% e 9.23%. Considerando-se todas as classes do estudo, destaca-se a classe do verbo “botar” que, além de possuir alta proporção de exemplos no corpus, é também a classe com menor comprimento médio. Ainda, foram observados alguns erros interessantes de troca de famílias nos verbos irregulares, como por exemplo: repetir (hepeti → hepEtu), uma possível confusão

com o grupo do verbo *testar*. Também ocorreram alguns erros de super-regularização (32 erros no total).

Durante a seção de discussão (Seç. 5.5), observamos que o tamanho do corpus obtido mostrou-se incompatível com a arquitetura *Encoder-Decoder*. Por outro lado, observa-se que os módulos desenvolvidos de pré e pós processamento dos verbos são independentes da arquitetura proposta. Nesse sentido, pode-se argumentar que o resultado obtido nesta pesquisa não é conclusivo quanto ao desempenho do *Encoder-Decoder*.

Para pesquisas futuras, ficam algumas sugestões:

1. Obtenção de um corpus maior para a língua portuguesa para que o algoritmo apresentado nesta pesquisa possa ser reavaliado;
2. Desenvolvimento de novos algoritmos de pré e pós processamento dos verbos. Outras representações vetoriais podem ser desenvolvidas;
3. Construção de um modelo do tipo *Transformer* (Vaswani *et al.* (2017)). A arquitetura *Transformer* é considerada a nova arquitetura estado-da-arte para modelos de tradução automática.

Também fica como sugestão a realização de um teste psicolinguístico com verbos irregulares inventados para uma comparação entre as predições do modelo e as opiniões de falantes da língua, retomando assim o caráter cognitivo da questão.

Para concluir, esperamos que a presente pesquisa tenha apresentado contribuições para o tema do aprendizado de verbos irregulares dentro do domínio dos modelos de Redes Neurais Artificiais.

Bibliografia

- ALBRIGHT, B., A. & HAYES, “Rules vs. analogy in english past tenses: a computational/experimental study”, *Cognition*, volume 90:119–161, 2003. , 3, 13, 53, 54
- BAAYEN, R. HARALD, PIEPENBROCK, RICHARD & VAN RIJN, HEDDERIK, “The celex lexical data base on cd-rom”, 1993. 53
- BAHDANAU, CHO K. BENGIO Y., D., “Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation”, *arXiv e-prints*, 2014. 1, 13, 41, 56
- BOULOS, P. & OLIVEIRA, I., *Geometria analítica: um tratamento vetorial*, PRENTICE HALL BRASIL, 2009.
URL <https://books.google.com.br/books?id=lmnqAAAACAAJ> 44
- BYBEE, J. L., “Language and cognitive processes”, *Language*, volume 10:425–455, 1995. 21
- BYBEE, J. L. & MODER, C. L., “Morphological classes as natural categories. language”, *Language*, volume 59(2):251–270, 1983. 4
- CHOLLET, F., “Deep learning with python”, 2017a. 46, 56
- CHOLLET, F., *A ten-minute introduction to sequence-to-sequence learning in Keras*, python Package, 2017b.
URL <https://blog.keras.io/a-ten-minute-introduction-to-sequence-to-sequence-learning-in-keras.html> 46
- CHOLLET, FRANÇOIS *et al.*, “Keras”, <https://keras.io>, 2015. 46, 56
- CHOMSKY, M., N. & HALLE, *The sound pattern of English*, MIT Press, 1968/1991. , 1, 3, 4
- CLARKE GM, COOKE D., “A basic course in Statistics.”, 1989. 49
- COTTERELL, CHRISTO KIROV & RYAN, “Stochastic phonology”, *GLOT*, volume 5, 2001. 21
- COTTERELL, RYAN, KIROV, CHRISTO, SYLAK-GLASSMAN, JOHN, YAROWSKY, DAVID, EISNER, JASON & HULDEN, MANS, “The sigmorphon 2016 shared task—morphological reinfection”, *in*: “Proceedings of the 2016 Meeting of SIGMORPHON”, Berlin, Germany: Association for Computational Linguistics, 2016. 13, 14
- CÂMARA JR., J. M., *Estrutura da Língua Portuguesa*, 30 ed., Vozes, 1999. 14
- DUA, DHEERU & GRAFF, CASEY, “UCI machine learning repository”, 2017.
URL <http://archive.ics.uci.edu/ml> 27

- FARUQUI, ET AL., “Morphological Inflection Generation Using Character Sequence to Sequence Learning”, *arXiv e-prints*, 2015. 13
- GOODFELLOW, Y. & COURVILLE A., I. & BENGIO, *Deep Learning*, MIT Press, <http://www.deeplearningbook.org>, 2016. 12, 32, 34, 35, 37
- GRON, A., *Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*, 1st ed., O’Reilly Media, Inc., 2017. 27
- GUIDE, BRUNO FERRARI, *Abordagem computacional para a questão do acento no português brasileiro*, Tese de Mestrado, University of Sao Paulo, 2016. 19
- HARRIS, D. & HARRIS, S., *Digital design and computer architecture*, 2 ed., Morgan Kaufmann, 2013. 38
- HOCHREITER, J., S. & SCHMIDHUBER, “Long short-term memory”, *Neural computation*, volume 9:1735–80, 1997. 38
- HUBEL, D. H. & WIESEL, T. N., “Receptive fields, binocular interaction and functional architecture in the cat’s visual cortex”, *The Journal of Physiology*, volume 160:106–154, 1962. 31
- JURAFSKY, J. H., D. & MARTIN, *Speech and Language Processing (2Nd Edition)*, Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2009. 36
- KAEHLING, L. P. ET. AL., “Reinforcement learning: A survey”, *Journal of Artificial Intelligence Research*, volume 4:237–285, 1996. 31
- KANN, H, K. & SCHÜTZE, “MED: The LMU system for the SIGMORPHON 2016 shared task on morphological reinflection”, *in: “Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology”*, 62–70, Berlin, Germany: Association for Computational Linguistics, 2016. 13
- KINGMA, J., D. P. & BA, “Adam: A Method for Stochastic Optimization”, *arXiv e-prints*, 2014. 46
- KIROV, CHRISTO & COTTERELL, RYAN, “Recurrent neural networks in linguistic theory: Revisiting pinker and prince (1988) and the past tense debate.”, *Transactions of the Association for Computational Linguistics*, 6:651–665, 2018. , 3, 53, 54
- KRIZHEVSKY, ALEX, SUTSKEVER, ILYA & HINTON, GEOFFREY E., “Imagenet classification with deep convolutional neural networks”, *in: “Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1”*, NIPS’12, 1097–1105, USA: Curran Associates Inc., 2012. 12
- LIU, X. & HUANG X., P. & QIU, “Recurrent Neural Network for Text Classification with Multi-Task Learning”, *arXiv e-prints*, 2016. 12
- MACKENZIE, C.E., *Coded character sets: history and development*, The Systems programming series, Addison-Wesley Pub. Co., 1980. 17
- MACQUEEN, J., “Some methods for classification and analysis of multivariate observations”, *in: “Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics”*, 281–297, Berkeley, Calif.: University of California Press,

1967.

URL <https://projecteuclid.org/euclid.bsmsp/1200512992> 30

MANNING, CHRISTOPHER D. & SCHÜTZE, HINRICH, *Foundations of Statistical Natural Language Processing*, Cambridge, MA, USA: MIT Press, 1999. 35

MARCUS, ET AL., G., *Overregularization in language acquisition. Monographs of the Society for Research in Child Development*, 1992. 7

MITCHELL, R., *Web Scraping with Python*, O'Reilly Media, Inc., 2015. 19

MORETTIN, W. O., P. A. & BUSSAB, *Estatística básica*, Saraiva, 2004. 29, 48, 49

NAKISA, RAMIN CHARLES & HAHN, ULRIKE, “Where defaults don’t help: the case of the german plural system”, *ArXiv*, volume cmp-lg/9605020, 1996. 12

NELSON, G., *English: an Essential Grammar*, Routledge., 2010. 3, 14

PATTERSON, J. & GIBSON, A., *Deep Learning*, O'Reilly Media, Inc, 2017. 34, 35, 38

PEIRCE, C., S. & SANTIAGO, “Prolegomena to an Apology for Pragmaticism.”, *The Monist*, volume 16:506, 1906. 20

PINKER, S., *The Language Instinct*, New York: NY: Harper Perennial Modern Classics, 1994/2007. 4

PINKER, S., *In Single Combat. "Words and Rules: The Ingredients of Language*, Harper Perennial, 1999. 1, 3, 8, 18

PINKER, S. & PRINCE, A., “On language and connectionism: Analysis of a parallel distributed processing model of language acquisition”, *Cognition*, volume 28(2):73–193, 1988. , 1, 3, 8, 9, 12, 13

PLUNKETT, KIM & MARCHMAN, VIRGINIA, “From rote learning to system building: acquiring verb morphology in children and connectionist nets”, *Cognition*, volume 48(1):21 – 69, 1993. 12

PLUNKETT, R. C., K. & NAKISA, “A connectionist model of the arabic plural system”, *Language and Cognitive Processes*, volume 12(5-6):807–836, 1997. 12

PLUNKETT, V., K. & MARCHMAN, “U-shaped learning and frequency effects in a multi-layered perception: Implications for child language acquisition.”, *Cognition*, volume 38(1):43–102, 1991. 9, 12

PRASADA, S., S. & PINKER, “Generalisation of regular and irregular morphological patterns”, *Language and Cognitive Processes*, volume 8(1):1–56, 1993. 7

RASCHKA, S., “Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning”, 24–26, 2018. 47, 56

ROSENBLATT, F., “The perceptron: A probabilistic model for information storage and organization in the brain”, *Psychological Review*, 65–386, 1958. 33

RUMELHART, D. E. & MCCLELLAND, J. L, *On learning the past tenses of English verbs*, Bradford Books/MIT Press, 1986. 1, 2, 5, 6, 7, 8, 12, 14, 16, 17, 18, 19, 20, 21, 35, 53, 54, 63

- SCHNEIDER, W., “Connectionism: Is it a paradigm shift for psychology?”, *Behavior Research Methods, Instruments, & Computers*, volume 19(2):73–83, 1987. 8
- SCHUSTER, MIKE & PALIWAL, KULDIP K, “Bidirectional recurrent neural networks”, *IEEE Transactions on Signal Processing*, volume 45(11):2673–2681, 1997. 43
- SEARA, ET AL., I. C., *Fonética e Fonologia do Português Brasileiro*, editora contexto, 2015. 6, 22
- SUTSKEVER, I. ET. AL., “Sequence to Sequence Learning with Neural Networks”, *arXiv e-prints*, 2014. 13, 41, 56
- VASWANI, ASHISH, SHAZEER, NOAM, PARMAR, NIKI, USZKOREIT, JAKOB, JONES, LLION, GOMEZ, AIDAN N., KAISER, LUKASZ & POLOSUKHIN, ILLIA, “Attention is all you need”, *in*: “NIPS”, 2017. 57
- WESTERMANN, R., G. & GOEBEL, “Connectionist rules of language.”, 1997. 12
- WICKELGREN, W. A., “Auditory or articulatory coding in verbal short-term memory”, *Psychological review*, volume 76:232–5, 1969. 17, 18
- WU, Y. ET. AL., “Google’s Neural Machine Translation System: Bridging the Gap between Human and Machine Translation”, *arXiv e-prints*, 2016. 13
- WUERGES, E. T., *A Aquisição da Morfologia Verbal por Crianças Falantes de Português Brasileiro e o Uso de Formas Variantes*, Tese de Mestrado, University of Sao Paulo, 2014. 10

Apêndice A

Tabela de Transcrição Fonética de Rumelhart e McClelland

TABLE 5
CATEGORIZATION OF PHONEMES ON FOUR SIMPLE DIMENSIONS

		Place					
		Front		Middle		Back	
		V/L	U/S	V/L	U/S	V/L	U/S
Interrupted	<i>Stop</i>	b	p	d	t	g	k
	<i>Nasal</i>	m	-	n	-	ŋ	-
Cont. Consonant	<i>Fric.</i>	v/D	f/T	z	s	ʒ/j	ʃ/C
	<i>Liq/SV</i>	w/l	-	r	-	y	h
Vowel	<i>High</i>	E	i	O	ʌ	U	u
	<i>Low</i>	A	e	I	a/α	W	*/o

Key: N = ng in *sing*; D = th in *the*; T = th in *with*; Z = z in *azure*; S = sh in *ship*; C = ch in *chip*; E = ee in *beer*; i = i in *bir*; O = oa in *boat*; ʌ = u in *but* or schwa; U = oo in *boor*; u = oo in *book*; A = ai in *bair*; e = e in *ber*; I = i_e in *bite*; a = a in *bar*; α = a in *father*; W = ow in *cow*; * = aw in *saw*; o = o in *hot*.

Figura A.1: Tabela de Pré-Processamento retirada de Rumelhart & McClelland [1986] (pág. 235)

Apêndice B

Corpus

Infinitivo	Infinitivo pre-processado	Alvo	Alvo pre-processado
Ansiar	Ansia	Anseio	Anseiu
odiar	odia	odeio	odeiu
pleitear	pleitea	pleiteio	pleiteiu
incendiar	ins3ndia	incendeio	ins3ndeiu
sortear	sortea	sorteio	sorteiu
remediar	hemediã	remedeio	hemedeiũ
intermediar	intermedia	intermedeio	intermedeiũ
mediar	media	medeio	medeiũ
olhar	oLa	olho	OLu
colocar	koloka	coloco	kolOku
informar	informa	informo	infOrmu
tornar	torna	torno	tOrnu
boiar	boia	boio	bOiu
botar	bota	boto	bOtu
tocar	toka	toco	tOku
focar	foka	foco	fOku
mostrar	mostra	mostro	mOstru
chorar	xora	choro	xOru
notar	nota	noto	nOtu
tostar	tosta	tosto	tOstu
trocar	troka	troco	trOku
robocar	heboka	reboco	hebOku
socar	soka	soco	sOku
cocar	kosa	coco	kOsu
tomar	toma	tomo	tOmu
recortar	hekorta	recorto	hekOrtu
entortar	entorta	entorto	entOrtu
encostar	enkosta	encosto	enkOstu
retornar	hetorna	retorno	hetOrnu
descobrir	deskobri	descubro	deskubru
recobrir	hekobri	recubro	hekubru
engolir	engoli	engulo	engulu
encobrir	enkobri	encubro	enkubru
sortir	sorti	surto	surtu
dizer	dize	digo	digu
redizer	hedize	redigo	hedigu
desdizer	desdize	desdigo	desdigu
condizer	kondize	condigo	kondigu
entredizer	entredize	entregido	entredigu
contradizer	kontradize	contradigo	kontradigu
interdizer	interdize	interdigo	interdigu
perfazer	perfaze	perfaco	perfasu
trAnsfaizer	trAnsfaize	trAnsfacu	trAnsfasu
insatisfazer	insatisfaze	insatisfacu	insatisfasu

Infinitivo	Infinitivo pre-processado	Alvo	Alvo pre-processado
malfazer	maufaze	malfaco	maufasu
liquefazer	likefaze	liquefaco	likefasu
benfazer	b3nfaze	benfaco	b3nfasu
satisfazer	satisfaze	satisfaco	satisfasu
putrefazer	putrefaze	putrefaco	putrefasu
contrafazer	kontrafaze	contrafaco	kontrafasu
torrefazer	tohefaze	torrefaco	tohefasu
rarefazer	harefaze	rarefaco	harefasu
esfazer	esfaze	esfaco	esfasu
desprazer	despraze	despraco	desprasu
fazer	faze	faco	fasu
adjazer	adijaze	adjaco	adijasu
descrer	deskre	descreio	deskreiu
reler	hele	releio	heleiu
ler	le	leio	leiu
crer	kre	creio	kreiu
tresler	tresle	tresleio	treleiu
ressentir	hes3nti	ressento	hesintu
persentir	persnti	persento	persintu
desmentir	desm3nti	desminto	desmintu
pressentir	pres3nti	pressinto	presintu
assentir	as3nti	assinto	asintu
mentir	m3nti	mintu	mintu
sentir	s3nti	sinto	sintu
consentir	kons3nti	consinto	konsintu
impedir	impedi	impeco	impEsu
pedir	pedi	peco	pEsu
desimpedir	dezimpedi	desimpeco	dezimpEsu
despedir	despedi	despeco	despEsu
medir	medi	meco	mEsu
expedir	espedi	expeco	espEsu
desmedir	desmedi	desmeco	desmEsu
contrapor	kontrapo	contraponho	kontrapoNu
dispor	dispo	disponho	dispoNu
trAnspor	trAnspo	trAnsponho	trAnspoNu
redispor	hedispo	redisponho	hedispoNu
expor	espo	exponho	espoNu
despor	despo	desponho	despoNu
depor	depo	deponho	depoNu
superpor	superpo	superponho	superpoNu
pospor	pospo	posponho	pospoNu
Antepor	Antepo	Anteponho	AntepoNu
trespor	trespo	tresponho	trespoNu

Infinitivo	Infinitivo pre-processado	Alvo	Alvo pre-processado
interpor	interpo	interponho	interpoNu
decompor	dekompo	decomponho	dekompoNu
impor	impo	imponho	impoNu
supor	supo	suponho	supoNu
justapor	justapo	justaponho	justapoNu
descompor	deskompo	descomponho	deskompoNu
propor	propo	proponho	propoNu
pressupor	presupo	pressuponho	presupoNu
recompor	hekompo	recomponho	hekompoNu
entrepôr	entrepo	entreponho	entrepoNu
prossupor	prosupo	prossuponho	prosupoNu
predispor	predispo	predisponho	predispoNu
indispor	indispo	indisponho	indispoNu
desapor	dezapo	desaponho	dezapoNu
excluir	esklui	excluo	eskluu
aproveitar	aproveita	aproveito	aproveitu
imaginar	imajina	imagino	imajinu
acabar	akaba	acabo	akabu
merecer	merese	mereso	meresu
viajar	viaja	viajo	viaju
conjugar	konjuga	conjugo	konjugu
plAntar	plAnta	plAnto	plAntu
enviar	envia	envio	enviu
crescer	krese	cresso	kresu
Andar	Anda	Ando	Andu
lambuzar	lAmbuza	lambuzo	lAmbuzu
estudar	estuda	estudo	istudu
continuar	kontinua	continuo	kontinuu
aguardar	aguarda	aguardo	aguardu
ligar	liga	ligo	ligu
chover	xove	chovo	xovu
gritar	grita	grito	gritu
exercer	ezerse	exerco	ezersu
agir	aji	ajo	aju
falar	fala	falo	valu
levAntar	levAnta	levAnto	levAntu
ficar	fika	fico	fiku
deixar	deixa	deixo	deixu
julgar	julga	julgo	julgu
possuir	posui	posuo	posuu
voar	voa	voo	vou
nadar	nada	nado	nadu
parar	para	paro	paru

Infinitivo	Infinitivo pre-processado	Alvo	Alvo pre-processado
comer	kome	como	komu
permAnecer	permAnese	permAneco	permAnesu
participar	partisipa	participo	partisipu
cativar	kativa	cativo	kativu
agradecer	agradese	agradeco	agradesu
pular	pula	pulo	pulu
acompanhar	akompANa	acompanho	akompANu
dividir	dividi	divido	dividu
chegar	xega	chego	xegu
desejar	dezeja	desejo	dezeju
deslinguar	deslingua	deslinguo	deslinguu
casar	kaza	caso	kazu
zoar	zoa	zoo	zou
acontecer	akontese	aconteco	akontesu
conhecer	koniese	conheco	koniesu
correr	kohe	corro	kohu
compreende	kompre3nde	compreendo	kompre3ndu
vender	v3nde	vendo	v3ndu
ensinar	ensina	ensino	ensinu
realizar	healiza	realizo	healisu
mudar	muda	mudo	modu
permitir	permiti	permito	permitu
sofrer	sofre	sofro	sofru
atribuir	atribui	atribuo	atribuu
convidar	konvida	convido	konvidu
erigir	eriji	erijo	eriju
Analisar	Analiza	Analizo	Analisu
ocorrer	okohe	ocorro	okohu
dissolver	disouve	dissolvo	disouvu
comprazer	kompraze	comprazo	komprazu
mexer	mexe	mexo	mexu
precisar	presiza	preciso	presizu
corrigir	kohiji	corrijo	kohiju
respeitar	hespeita	respeito	hespeitu
lavar	lava	lavo	lavu
varrer	vahe	varro	vahu
beijar	beija	beijo	beiju
educar	eduka	educo	eduku
descer	dese	desco	desu
dirigir	diriji	dirijo	diriju
comprar	kompra	compro	kompru
achar	axa	acho	axu
fingir	finji	finjo	finju

Infinitivo	Infinitivo pre-processado	Alvo	Alvo pre-processado
contar	konta	conto	kontu
pensar	p3nsa	penso	p3nsu
existir	ezisti	existo	ezistu
sacudir	sakudi	sacudo	sakudu
trabalhar	trabaLa	trabalho	trabaLu
assistir	asisti	assisto	asistu
submeter	subimete	submeto	subimetu
atender	at3nde	atendo	at3ndu
cAntar	kAnta	cAnto	kAntu
surgir	surji	surjo	surju
procurar	prokura	procuro	prokuru
frigir	friji	frijo	friju
caminhar	kamiNa	caminho	kamiNu
marcar	marka	marco	marku
esquecer	eskese	esqueco	eskesu
tentar	t3nta	tento	t3ntu
decidir	desidi	decido	desidu
desistir	dezisti	desisto	dezistu
lembrar	lembra	lembro	lembu
aparecer	aparese	apareco	aparesu
brincar	brinka	brinco	brinku
usar	uza	uso	uzu
envolver	envouve	envolvo	envouvuvu
omitir	omiti	omito	omitu
dever	deve	devo	devu
romper	hompe	rompo	hompu
ceder	sede	cedo	sedu
buscar	buska	busco	busku
escrever	eskreve	escrevo	eskrevu
encaminhar	enkamiNa	encaminho	enkamiNu
cumprir	kumpri	cumpro	kumpru
amar	Ama	amo	Amu
ajudar	ajuda	ajudo	ajudu
discutir	diskuti	discuto	diskutu
entrar	entra	entro	entru
parecer	parese	pareco	paresu
partir	parti	parto	partu
apresentar	aprez3nta	apresento	aprez3ntu
beber	bebe	bebo	bebu
adquirir	adikiri	adquiro	adikiru
oferecer	oferese	ofereco	oferesu
encher	enxe	encho	enxu
passar	pasa	passo	pasu

Infinitivo	Infinitivo pre-processado	Alvo	Alvo pre-processado
iniciar	inisia	inicio	inisiu
dAncar	dAnsa	dAnco	dAnsu
encontrar	enkontra	encontro	enkontru
sonhar	soNa	sonho	soNu
compartilhar	kompartiLa	compartilho	kompartiLu
aprender	apr3nde	aprendo	apr3ndu
avisar	aviza	aviso	avizu
entender	ent3nde	entendo	ent3ndu
mAndar	mAnda	mAndo	mAndu
preocupar	preukupa	preocupo	preukupu
garAntir	garAnti	garAnto	garAntu
sentar	s3nta	sento	s3ntu
chamar	xAma	chamo	xAmu
murchar	murxa	murcho	murxu
rir	hi	rio	hiu
incorrer	inkohe	incorro	inkohu
mobiLar	mobiLa	mobilio	mobiLu
bulir	buli	bulo	bulu
fugir	fuji	fujo	fuju
advetir	adiverti	adverto	adivertu
desprover	desprove	desprovo	desprovu
consumir	konsumi	consumo	konsumu
distrair	distrai	distraio	distraiu
acudir	akudi	acudo	akudu
subir	subi	subo	subu
eleger	eleje	elejo	eleju
construir	konstrui	contruo	konstruu
tender	t3nde	tendo	t3ndu
defender	def3nde	defendo	def3ndu
jazer	jaze	jazo	jazu
sumir	sumi	sumo	sumu
morrer	mohe	morro	mohu
contravir	kontravi	contravo	kontravu
afligir	afliji	aflijo	afliju
incluir	inklui	incluo	inkluiu
trAns fugir	trAns fuji	trAns fujo	trAns fuju
emergir	emerji	emerjo	emerju
redar	heda	redo	hedu
despolir	despoli	despolo	despolu
desaguar	dezagua	desaguo	dezaguu
remedir	hemedi	remedo	hemedu
influir	influi	influo	influiu
salvar	sauva	salvo	sauvu

Infinitivo	Infinitivo pre-processado	Alvo	Alvo pre-processado
prender	pr3nde	prendo	pr3ndu
expulsar	espulsa	expulso	espulsu
revolver	hevolve	revolvo	hevolvuu
enxugar	enxuga	enxugo	enxugu
escapular	eskapuli	escapulo	eskapulu
convencer	konv3nse	convenco	konv3nsu
retrair	hetrai	retraio	hetraiu
moer	moe	moo	mou
abrir	abri	abro	abru
concluir	konklui	concluo	konkluu
atingir	atinji	atinjo	atinju
ocultar	okulta	oculto	okultu
isentar	iz3nta	isento	iz3ntu
limpar	limpa	limpo	limpu
exprimir	esprimi	exprimo	esprimu
fartar	farta	farto	fartu
trair	trai	traio	traiu
minguar	mingua	minguo	minguu
desmobiLar	desmobiLa	desmobilio	desmobiLu
sorrir	sohi	sorrio	sohiu
perverter	perverte	perverto	pervertu
precluir	preklui	precluo	prekluu
contrair	kontrai	contraio	kontraiu
extinguir	estingi	extingo	estingu
desprecaver	desprekave	desprecavo	desprekavu
telever	televe	televo	televu
entupir	entupi	entupo	entupu
descalcar	deskalsa	descalco	deskalsu
gastar	gasta	gasto	gastu
findar	finda	findo	findu
cuspir	kuspi	cuspo	kuspu
pretender	pret3nde	pretendo	pret3ndu
dormir	dormi	durmo	durmu
destruir	destrui	destruo	destruu
esvair	esvai	esvaio	esvaiu
suspender	susp3nde	suspendo	susp3ndu
cair	kai	caio	kaiu
imprimir	imprimi	imprimo	imprimu
imergir	imerji	imerjo	imerju
submergir	subimerji	submerjo	subimerju
benzer	b3nze	benzo	b3nzu
gAnhar	gANa	gAnho	gANu
aceitar	aseita	aceito	aseitu

Infinitivo	Infinitivo pre-processado	Alvo	Alvo pre-processado
confugir	konfuji	confujo	konfuju
segurar	segura	seguro	seguru
juntar	junta	junto	juntu
acender	as3nde	acendo	as3ndu
corromper	kohompe	corrompo	kohompu
atrair	atrai	atraio	atraiu
cultivar	kultiva	cultivo	kultivu
escutar	eskuta	escuto	eskutu
perseguir	persegi	persigo	persigu
expelir	espeli	expilo	espilu
vestir	vesti	visto	vistu
inserir	inseri	insiro	insiru
agredir	agredi	agrido	agridu
ferir	feri	firo	firu
prevenir	preveni	previno	previnu
preferir	preferi	prefiro	prefiru
competir	kompeti	compito	kompitu
servir	servi	sirvo	sirvu
conferir	konferi	confiro	konfiru
sugerir	sujeri	sujiro	sujiru
regredir	hegrede	regrido	hegridu
seguir	segi	sigo	sigu
divertir	diverti	divirto	divirtu
trAnsgredir	trAnsgrede	trAnsgrido	trAnsgridu
digerir	dijeri	digiro	dijiru
progredir	progrede	progrido	progridu
aderir	aderi	adiro	adiru
aferir	aferi	afiro	afiru
preterir	preteri	pretiro	pretiru
convergir	konverji	convirjo	konvirju
deferir	deferi	defiro	defiru
compelir	kompeli	compilo	kompilu
conseguir	konsegi	consigo	konsigu
repetir	hepeti	repito	hepitu
entreter	entrete	entretenho	entret3Nu
obter	obite	obtenho	obit3Nu
sobrevir	sobreve	sobrevenho	sobrev3Nu
conter	konte	contenho	kont3Nu
obvir	obivi	obvenho	obiv3Nu
ater	ate	atenho	at3Nu
deter	dete	detenho	det3Nu
abster	abiste	abstenho	abist3Nu
reter	hete	retenho	het3Nu

Infinitivo	Infinitivo pre-processado	Alvo	Alvo pre-processado
mAnter	mAnte	matenho	mAnt3Nu
ter	te	tenho	t3Nu
suster	suste	sustenho	sust3Nu
entregar	entrega	entrego	entrEgu
começar	komesa	comeco	komEsu
readequar	headekua	readequo	headEkuu
completar	kompleta	completo	komplEtu
protestar	protesta	protesto	protEstu
esperar	espera	espero	espEru
adequar	adekua	adequo	adEkuu
mAnifestar	mAnifesta	mAnifesto	mAnifEstu
infectar	infekita	infecto	infEkitu
emprestar	empresta	empresto	emprEstu
detestar	detesta	detesto	detEstu
prever	preve	prevejo	preveju
Antever	Anteve	Antevejo	Anteveju
interver	interve	intervejo	interveju
rever	heve	revejo	heveju
entrever	entreve	entrevejo	entreveju
ver	ve	vejo	veju
reconvir	hekonvi	reconvenho	hekonv3Nu
vir	vi	venho	v3Nu
advir	adivi	advenho	adiv3Nu
desconvir	deskonvi	deconvenho	deskonv3Nu
avir	avi	avenho	av3Nu
desavir	dezavi	desavenho	dezav3Nu
provir	provi	provenho	prov3Nu
revir	hevi	revenho	hev3Nu
convir	konvi	convenho	konv3Nu
pegar	pega	pego	pEgu
cegar	sega	cego	sEgu
secar	seka	seco	sEku
levar	leva	levo	lEvu
orar	ora	oro	Oru
morar	mora	moro	mOru
postar	posta	posto	pOstu
jogar	joga	jogo	jOgu
compor	kompo	componho	kompoNu
por	po	ponho	poNu
tossir	tosi	tusso	tusu
cobrir	kobri	cubro	kubru
matar	mata	mato	matu
pagar	paga	pago	pagu

Infinitivo	Infinitivo pre-processado	Alvo	Alvo pre-processado
sair	sai	saio	saiu
bater	bate	bato	batu
requerer	hekere	requero	hekEru
entrequerer	entekere	entrequero	entrekEru
querer	kere	quero	kEru
saber	sabe	sei	sei
equivaler	ekivale	equivalho	ekivaLu
prover	prove	provenho	prov3Nu
trazer	traze	trago	tragu
poder	pode	posso	pOsu
desdar	desda	desdo	dEsdu
idear	idea	ideio	ideiu
estrear	estrea	estreio	estrEiu
dar	da	dou	dou
sobrestar	sobresta	sobrestou	sobrestou
estar	esta	estou	estou
caber	kabe	caibo	kaibu
ouvir	ouvi	ouco	ousu
entreouvir	entreouvi	ouco	entreousu
reouvir	heouvi	reouco	heousu
parir	pari	pairo	pairu
perder	perde	perco	perku
testar	testa	testo	tEstu
observar	obiserva	observo	obisErvu
escolher	eskolie	escolho	eskoLu
conversar	konversa	converso	konvErsu
fechar	fexa	fecho	fExu
gostar	gosta	gosto	gOstu
voltar	volta	volto	vOltu
expressar	espresa	expresso	esprEsu
passear	pasea	passeio	paseiu
acordar	akorda	acordo	akOrdu
cortar	korta	corto	kOrtu
almocar	aumosa	almoco	aumOsu
ingerir	injeri	ingiro	injiru

Apêndice C

Resultados das Decodificações

Classe: ansiar -> anseio

infinitivo	predição	alvo
odia	\$tuu	odeiu
media	medeiu	medeiu
hemia	hemeti	hemedeiu
sortea	sotreiu	sorteiu
intermedia	entenkenuu	intermedeiu
insendia	insenedu	insendeiu
pleitea	pe\$keiu	pleiteiu
ansia	anasu	anseiu
pasea	pareiu	paseiu

Acertou Processo Irregular	Regularizado	Errou	Acertou	Total
1	0	7	1	9
11%		78%	11%	100%

Classe: botar -> boto

infinitivo	predição	alvo
soka	sOku	sOku
heboka	hepupu	hebOku
nota	nOru	nOtu
boia	bO	bOiu
koloka	kolpu	kolOku
oLa	o	OLu
korta	kortu	kOrtu
akorda	akurtu	akOrdu
kosa	kOsu	kOsu
aumosa	emus\$u	aumOsu
gosta	gostu	gOstu
volta	juv\$u	vOltu
joga	jugu	jOgu
mostra	purteNu	mOstru
troka	tOttu	trOku
foka	fOku	fOku
xora	xOru	xOru
tosta	tustu	tOstu
entorta	entrEku	entOrtu
hetorna	senpruu	hetOrnu
bota	bOsu	bOtu
torna	trO	tOrnu
informa	infEru	infOrmu
enkosta	enkOstu	enkOstu
toka	tOku	tOku
ora	Oru	Oru
mora	mOru	mOru

Classe: botar -> boto

infinitivo	predição	alvo
posta	postu	pOstu
hekorta	hekOrtu	hekOrtu
toma	tomu	tOmu

Acertou Processo Irregular	Regularizado	Errou	Acertou	Total
5	4	12	9	30
17%	13%	40%	30%	100%

Classe: cobrir -> cubro

infinitivo	predição	alvo
kobri	kondu	kubru
tosi	tusu	tusu
engoli	etkulu	engulu
deskobri	desprkuu	deskubru
enkobri	enkopru	enkubru
hekobri	hepOttu	hekubru
sorti	sutru	surtu

Acertou Processo Irregular	Regularizado	Errou	Acertou	Total
0	0	7	0	7
0%	0%	100%	0%	100%

Classe: dizer -> digo

infinitivo	predição	alvo
dize	dEsu	digu
hedize	hedetu	hedigu
desdize	dest\$fu	desdigu
entredize	entretezu	entredigu
interdize	intrezezu	interdigu
kondize	kons\$su	kondigu
kontradize	kontravatu	kontradigu

Acertou Processo Irregular	Regularizado	Errou	Acertou	Total
0	0	7	0	7
0%	0%	100%	0%	100%

Classe: fazer -> faço

infinitivo	predição	alvo
despraze	destrevu	desprasu
faze	fasu	fasu
adijaze	adivasu	adijasu
putrefaze	pondissu	putrefasu
benfaze	besfasu	benfasu

Classe: fazer -> faço

infinitivo	predição	alvo
perfaze	perfesu	perfasu
satisfaze	sasisisu	satisfasu
insatisfaze	ensentefasu	insatisfasu
likefaze	amajezuu	likefasu
transfaze	transsazu	transfasu
kontrafaze	kontravasu	kontrafasu
tohefaze	seşpefu	tohefasu
harefaze	saassazu	harefasu
esfaze	esfanu	esfasu
maufaze	amusfaxu	maufasu

Acertou Processo Irregular	Regularizado	Errou	Acertou	Total
4	2	8	1	15
27%	13%	53%	7%	100%

Classe: ler -> leio

infinitivo	predição	alvo
kre	kreu	kreiu
tresle	treseşu	treleiu
le	şeu	leiu
hele	hele	heleiu
deskre	destriu	deskreiu

Acertou Processo Irregular	Regularizado	Errou	Acertou	Total
0	0	5	0	5
0%	0%	100%	0%	100%

Classe: mentir -> minto

infinitivo	predição	alvo
desmenti	desumnu	desmintu
konsenti	konsintu	konsintu
senti	sintu	sintu
presenti	prenşşu	presintu
menti	mentu	mintu
asenti	ansamu	asintu
hesenti	hesteiu	hesintu
persenti	persentu	persintu

Acertou Processo Irregular	Regularizado	Errou	Acertou	Total
0	2	4	2	8
0%	25%	50%	25%	100%

Classe: Sem Agrupamento

infinitivo	predição	alvo
entreouvi	entreviuu	entreousu
reouvi	er\$uju	reousu
pari	paru	pairu
perde	perteuu	perku
hekere	hetetu	hekEru
traze	trasu*	tragu
desda	destu	dEsdu
kabe	kabu	kaibu
da	ti	dou
ekivale	eteleju	ekivaLu
sabe	saku	sei
idea	adeiu	ideiu
esta	estu	estou
prove	provu	proveNu
sobresta	desprEsu	sobrestou
kere	katu	kEru
entrekere	entremet\$u	entrekEru
pode	pobu	pOsu
estrea	estreu	estrEiu
ouvi	osuvu	ousu

Acertou Processo Irregular	Regularizado	Errou	Acertou	Total
0	4	13	0	20
0%	20%	65%	0%	85%

*confundiu com a classe do fazer

Classe: pedir -> peço

infinitivo	predição	alvo
dezimpedi	despumpu	dezimpEsu
medi	metu	mEsu
despedi	desp\$u	despEsu
impedi	emid\$u	impEsu
pedi	pEtu	pEsu
espedi	espidu*	espEsu
desmedi	despinu	desmEsu

Acertou Processo Irregular	Regularizado	Errou	Acertou	Total
	0	6	0	7
14%	0%	86%	0%	100%

*confundiu com a classe de "conseguir" e regularizou (manteve o d)

Classe: por -> ponho

infinitivo	predição	alvo
predispo	pretustu	predispoNu
indispo	intittu	indispoNu
dezapo	desku	dezapoNu
kompo	kompu	kompoNu
po	poNu	poNu
kontrapo	kontrenu	kontrapoNu
pospo	popu	pospoNu
dispo	despu	dispoNu
despo	despu	despoNu
transpo	trenkinu	transpoNu
espo	espu	espoNu
interpo	ent\$\$ti	interpoNu
hedispo	hede\$piu	hedispoNu
antepo	antepu	antepoNu
deskompo	despopu	deskompoNu
justapo	su	justapoNu
trespo	tremku	trespoNu
dekompo	tompupu	dekompoNu
supo	supu	supoNu
presupo	prenudu	presupoNu
hekompo	hekuptu	hekompoNu
entrepo	entremu	entrepoNu
prosupo	prosupu	prosupoNu
propo	propu	propoNu
depo	depu	depoNu
impo		impoNu
superpo	\$up\$\$r	superpoNu

Acertou Processo Irregular	Regularizado	Errou	Acertou	Total
0	7	17	2	26
0%	27%	65%	8%	100%

Classe: Regulares

infinitivo	predição	alvo
sofre	sos\$	sofru
disouve	desuvu	disouvuvu
kohiji	konfu	kohiju
envouve	engoffu	envouvuvu
mexe	mexu	mexu
kompraze	komprazu	komprazu
presiza	presizu	presizu
analiza	aAsaxu	analisu
eriji	ensu	eriju

Classe: Regulares**infinitivo**

hompe
konvida
planta
levanta
fala
anda
kumpri
konjuga
xama
murxa
eskreve
kontinua
krese
ezerse
liga
brinka
trabaLa
subimete
hespeita
vende
sede
agradese
koNese
voa
nada
para
kome
permiti
lava
dese
beija
kohe
permanese
uza
esklui
kompreende
dezisti
aparese
envia
kai
imprimi
imerji
subimerji

predição

hempu
konviru
pendi
xenvaru
falu
atru
kompru
konjugu

mur
esterfeju
konsinu
tregu
estreuu
ligu
prapu
tarkilu
ŞumpEstu
hepentiu
fendu
setu
katrazinu
konfi
fO
tanu
paru
koNu
pentumu
lavu
desesu
beavu
kohiu
pentsentu
suu
eskluu
komprantŞu
despisu
apendeiu
esfu
kaiu
mmpiru
imŞrxu
subinviru

alvo

homp
konvidu
plantu
levantu
falu
andu
kumpru
konjugu
xamu
murxu
eskrevu
kontinuu
kresu
ezersu
ligu
brinku
trabaLu
subimetu
hespeitu
vendu
sedu
agradesu
koNesu
vou
nadu
paru
komu
permitu
lavu
desu
beiju
kohu
permanesu
uzu
eskluu
kompreendu
dezistu
aparesu
enviu
kaiu
imprimu
imerju
subimerju

Classe: Regulares**infinitivo**

benze
gaNa
aseita
konfuji
segura
suspende
junta
kohompe
atrai
kultiva
eskuta
mata
paga
sai
bate
eskoLe
asende
esvai
destrui
dormi
okohe
diriji
muda
esprimi
farta
traí
mingua
desmobilia
sohi
perverte
preklui
kontraí
estingi
desprekave
televe
entupi
deskalsa
gasta
finda
kuspi
pretende
deve
enkamiNa

predição

benzu
gaNu
aseitu
konvuju
setudu
sundendu
xuttu
kompoxu
atri
guliz\$u
eskutu
mantu
padu
seiu
bati
esko
ansedu
esfi
destruu
turpu
opoxu
dis\$u
mudu
entripu
fartu
treiu
mumdu
desemliluu
sohu
pers\$\$tu
proku
kontreiu
estidu
deskrevizu
te\$leju
entupu
des\$\$\$
gastu
findu
kuspu
prenttiu
devexu
anmanupu

alvo

benzu
gaNu
aseitu
konfuju
seguru
suspendu
juntu
kohompu
atraiu
kultivu
eskutu
matu
pagu
saiu
batu
eskoLu
asendu
esvaiu
destruu
durmu
okohu
diriju
mudu
esprimu
fartu
traiu
minguu
desmobiliu
sohiu
pervertu
prekluu
kontraiu
estingu
desprekavu
televu
entupu
deskalsu
gastu
findu
kuspu
pretendu
devu
enkamiNu

Classe: Regulares**infinitivo**

atribui
estuda
sauva
prende
espulsa
hevolve
enxuga
despoli
eskapuli
hetrai
moe
abri
konklui
atinji
okulta
izenta
limpa
konvense
heda
emerji
transfuji
mobilia
posui
fuji
adiverti
desprove
konsumi
distrai
akudi
subi
eleje
influi
hemedi
dezagua
inkohe
tenta
healiza
desidi
eskese
imajina
lembra
aproveita
omiti

predição

apridiu
estOtu
si\$xu
prentu
\$\$x\$ltu
hoju
enjuku
deslupu
et\$kliu
setEruu
mo
abiru
konpulu
atis\$u
kol\$tu
is\$atu
limpu
konfensu
heku
em\$su
trasuzsu
pupEliu
pOsuu
xuju
idistenu
desprfazu
konsumu
distantu
agutu
subu
e\$lexu
injulu
hepitu
desk\$\$u
enko
tentu
halisu
desidu
eskenu
imirau
lombru
emprevu
ompiru

alvo

atribuu
istudu
sauvu
prendu
espulsu
hevolvu
enxugu
despolu
eskapulu
hetraiu
mou
abru
konkluu
atinju
okultu
izentu
limpu
konvensu
hedu
emerju
transfuju
mobiliu
posuu
fuju
adivertu
desprovu
konsumu
distraiu
akudu
subu
eleju
influu
hemedu
dezaguu
inkohu
tentu
healisu
desidu
eskesu
imajinu
lembu
aproveitu
omitu

Classe: Regulares**infinitivo**

ensina
marka
kamiNa
friji
prokura
surji
konstrui
kanta
vahe
merese
grita
xove
kativa
viaja
dezeja
pula
akompaNa
dividi
aguarda
zoa
xega
buli
atende
pasa
ajuda
dansa
enkontra
buska
inisia
enxe
oferese
adikiri
bebe
apresentar
parti
parese
soNa
kaza
julga
hi
inklui
afliji
kontravi

predição

ensinu
mattu
kaAmu
fisxu
progutu
sustu
kontriuu
kantu
ves
meresu
dErtu
fOju
kataxu
vaxu
devizu
pulu
akOmtiNu
dividu
aku
sO
xEgu
bulu
atendu
pasu
ajudu
tra
enkOrntu
duptu
inŞanu
eneju
espersu
adiŞtiu
beipu
atrebesseu
partu
peresu
somu
kasu
juguu
heiu
atulŞu
ajiju
kontraxu

alvo

ensinu
marku
kamiNu
friju
prokuru
surju
konstruu
kantu
vahu
meresu
gritu
xovu
kativu
viaju
dezeju
pulu
akompaNu
dividu
aguardu
zou
xegu
bulu
atendu
pasu
ajudu
dansu
enkontru
busku
inisiu
enxu
oferesu
adikiru
bebu
aprezentü
partu
paresu
soNu
kazu
julgu
hiu
inkluiu
afliju
kontravu

Classe: Regulares

infinitivo	predição	alvo
mohe	mohu	mohu
sumi	sumu	sumu
jaze	vinju	jazu
denfende	des\$nteNu	defendu
tende	tendu	tendu
sakudi	atugu	sakudu
aprende	apretezu	aprendu
kompartiLa	kompraagu	kompartiLu
entende	entendu	entendu
deixa	dexiu	deixu
deslingua	despalisu	deslinguu
lambuza	lampruu	lambuzu
akontese	ankontru	akontesu
pensa	per\$u	pensu
fika	fiku	fiku
senta	sentu	sentu
ezisti	esfaru	ezistu
konta	kontu	kontu
finji	fisxu	finju
axa	axu	axu
aviza	avizu	avizu
kompra	kumpru	kompru
aji	aju	aju
asisti	asistu	asistu
eduka	eduku	eduku
ama	amu	amu
akaba	apatu	akabu
diskuti	diskutu	diskutu
entra	entriu	entru
garanti	kartinu	garantu
preukupa	porpituu	preukupu
manda	mandu	mandu
partisipa	pratirtu	partisipu

Confundiu classe	Errou	Acertou	Total
8	153	53	214
4%	67%	25%	100%

Classe: seguir -> sigo

infinitivo	predição	alvo
dijeri	difiru	dijiru
persegi	pertu	persigu
espeli	espilu	espilu

Classe: seguir -> sigo

infinitivo	predição	alvo
servi	sesfu	sirvu
preveni	preseuu	previnu
agredi	adr\$gu	agridu
progridi	kr\$trkuu	progridu
inseri	insiru	insiru
kompeti	kompitu	kompitu
diverti	devirtu	divirtu
transgredi	trantr\$gu	transgridu
sujeri	sus\$gu	sujiru
aderi	adinu	adiru
preferi	presitu	prefiru
feri	fEru*	firu
regredi	ettridu	regridu
segi	segu	sigu
deferi	deizu	defiru
konsegi	konfidu	konsigu
kompeli	kompliu	kompilu
injeri	inviru	injiru
konferi	konfiru	konfiru
vesti	vEstu*	vistu
aferi	asisu	afiru
preteri	pretiru	pretiru
konverji	konvirju	konvirju
hepeti	hepEtu*	hepitu

Acertou Processo Irregular	Regularizado	Errou	Acertou	Total
2	1	18	6	27
7%	4%	67%	22%	100%

*confundiu com a classe do testar

Classe: ter -> tenho

infinitivo	predição	alvo
entrete	entrenu	entreteNu
dete	detu	deteNu
konte	kont\$u	konteNu
obite	opOru	obiteNu
mante	manteNu	manteNu
abiste	abistu	abisteNu
hete	hekeNu	heteNu
te	te\$u	teNu
suste	sustu	susteNu
ate	ateiu*	ateNu

Acertou Processo Irregular	Regularizado	Errou	Acertou	Total
1	3	5	1	10
10%	30%	50%	10%	100%

*confundiu com a classe do ansiar

Classe: testar -> testo

infinitivo	predição	alvo
headekua	hete\$tiu	headEkuu
komesa	kom\$	komEsu
adekua	agitiu	adEkuu
seka	seku	sEku
leva	levu	lEvu
sega	seku	sEgu
pega	pEku	pEgu
kompleta	kompas\$u	komplEtu
infekita	infakkuu	infEkitu
espresa	estresu	esprEsu
detesta	tespenu	detEstu
fexa	fexu	fExu
konversa	kons\$ss\$u	konvErsu
obiserva	ko\$ vessu	obisErvu
protesta	pretetu	protEstu
espera	espri	espEru
testa	testu	tEstu
entrega	entr\$ku	entrEgu
manifesta	aminse\$iu	manifEstu
empresta	menpregu	emprEstu

Acertou Processo Irregular	Regularizado	Errou	Acertou	Total
1	4	14	1	20
5%	20%	70%	5%	100%

Classe: ver -> vejo

infinitivo	predição	alvo
ve	feu	veju
interve	ent\$ \$fasu	interveju
anteve	anesvu	anteveju
entreve	entreveNu*	entreveju
preve	preveNu*	preveju
heve	hefexu	heveju

Acertou Processo Irregular	Regularizado	Errou	Acertou	Total
0	0	6	0	6
0%	0%	100%	0%	100%

*confundi com a classe do vir

Classe: vir -> venho

infinitivo	predição	alvo
dezavi	desizu	dezaveNu
sobrevi	sobrevu	sobreveNu
obivi	oubu	obiveNu
hevi	heviju	heveNu
vi	fu	veNu
adivi	adivu	adiveNu
deskonvi	deskuzu	deskonveNu
avi	aviu	aveNu
hekonvi	hefpO	hekonveNu
konvi	konvu	konveNu
provi	provu	proveNu

Acertou Processo Irregular	Regularizado	Errou	Acertou	Total
0	5	6	0	11
0%	45%	55%	0%	100%