Universidade de São Paulo
Faculdade de Filosofia, Letras e Ciências Humanas
Departamento de Linguística

Um modelo de classificação para o Reconhecimento de Entidades Nomeadas

Versão Corrigida

Andressa Vieira e Silva

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. Marcos Lopes

São Paulo
2020

Um modelo de classificação para o Reconhecimento de Entidades Nomeadas

Dissertação de Mestrado

Programa de Mestrado em Semiótica e Linguística Geral

Universidade de São Paulo

Versão Corrigida

Andressa Vieira e Silva
Orientador: Prof. Dr. Marcos Lopes
Departamento de Linguística FFLCH-USP

São Paulo
2020

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

S586m

Silva, Andressa Vieira e
Um modelo de classificação para o Reconhecimento de Entidades Nomeadas / Andressa Vieira e Silva; orientador Marcos Fernando Lopes - São Paulo, 2020.
132 f.

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

1. Linguística . 2. Linguística Computacional. 3. Processamento de Linguagem Natural. 4. Redes Neurais. I. Lopes, Marcos Fernando, orient. II. Título.

UNIVERSIDADE DE SÃO PAULO
FACULDADE DE FILOSOFIA, LETRAS E CIÊNCIAS HUMANAS

ENTREGA DO EXEMPLAR CORRIGIDO DA DISSERTAÇÃO/TESE

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

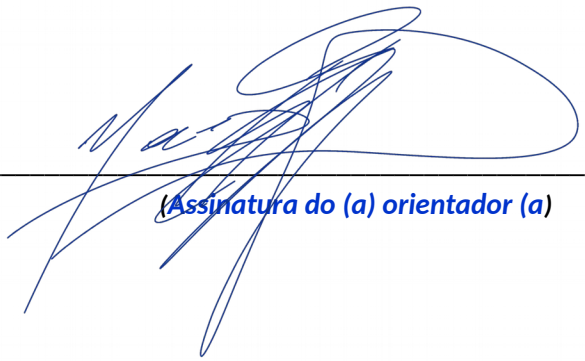
Nome do (a) aluno (a): Andressa Vieira e Silva

Data da defesa: 16/12/2020

Nome do Prof. (a) orientador (a): Marcos Fernando Lopes

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, 27/1/21.



(*Assinatura do (a) orientador (a)*)

O presente trabalho foi realizado com o apoio do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), processo nº 163439/2018-4.

This study was financed in part by The Brazilian National Council for Scientific and Technological Development (CNPq), grant #163439/2018-4.

Agradecimentos

Ao longo do meu trajeto acadêmico e pessoal, eu tive pessoas ao meu lado das quais não poderia deixar de agradecer pelo apoio, incentivo e carinho. Em primeiro lugar, agradeço à minha irmã Aline, minha primeira leitora, que nunca deixou de acreditar em mim e esteve presente quando eu precisei, e à minha mãe, pelo cuidado e preocupação com meu bem-estar e pelo apoio em todas as minhas decisões, mesmo quando ela não concordava.

À minha tia Mazé, por ser aquela que, em primeiro lugar, me incentivou em seguir minha carreira acadêmica, me oferecendo conselhos valiosos que vou levar comigo sempre.

Também não vou esquecer do suporte dos meus amigos e colegas da Linguística, em especial a minha amiga e confidente Natália, que esteve à disposição para ouvir meus desabafos, estudar comigo e me fazer rir quando eu precisava.

Ao meu orientador Marcos Lopes, que me acompanhou desde o início em meus estudos em Linguística Computacional na graduação e, depois, nesta pesquisa. Agradeço por tudo o que você me ensinou durante esses anos, sempre muito paciente e atencioso. Os seus conselhos e incentivos me fizeram crescer como pesquisadora e me trouxeram onde estou hoje.

Também agradeço ao professor Marcelo Barra Ferreira, por seus comentários sempre pontuais e relevantes, não só com relação a esta pesquisa. Pude aprender muito com você nas discussões do Grupo de Linguística Computacional que tive a oportunidade de participar. Ao professor Marcelo Finger, pelo ótimo curso de Linguística Computacional que foi enriquecedor para meu aprendizado na área e por suas sugestões pertinentes sobre esta pesquisa.

Resumo

O Reconhecimento de Entidades Nomeadas (REN) é uma tarefa de Processamento de Linguagem Natural (PLN) que busca identificar as Entidades Nomeadas de um texto, tais como nomes de pessoas, cidades e organizações, classificando-as em um conjunto pré-definido de categorias. Essa é considerada uma tarefa difícil, pois as Entidades Nomeadas constituem uma classe gramatical com muita variação lexical e de baixa frequência quando comparadas à massa total de dados textuais. Recentemente, as pesquisas com redes neurais profundas têm mostrado excelentes resultados em diversas aplicações de PLN, incluindo o REN.

Nesta pesquisa, foram investigadas duas arquiteturas de redes neurais para o REN no Harem, um corpus de língua portuguesa: BERT (DEVLIN et al., 2018) e uma rede neural bidirecional LSTM (BiLSTM). O objetivo principal foi explorar traços baseados na distribuição contextual das entidades, através de representações vetoriais *word embeddings* associadas a traços linguísticos. Foram usados traços de etiquetagem morfossintática, forma ortográfica da palavra e recursos lexicais. Esses traços foram concatenados às representações *word embeddings* para alimentar a BiLSTM. Os resultados mostraram uma melhora estatisticamente significativa no desempenho desse modelo em comparação à BiLSTM apenas com os *word embeddings*. O modelo BERT, por sua vez, obteve medidas próximas ao estado da arte no Harem.

Palavras-chave: Reconhecimento de Entidades Nomeadas, redes neurais, representações *word embeddings*, traços de representação linguística.

Abstract

Named Entity Recognition (NER) is a Natural Language Processing (NLP) task that aims at identifying Named Entities in a text, such as person, city, and organization names, classifying them into a pre-defined set of categories. NER is considered a hard task as Named Entities are a grammatical class with lots of lexical variation and relatevely low frequency if compared to the total mass of textual data. Nevertheless, deep neural network researches have recently shown excelent results in several NLP applications, including NER.

In this work, two neural network archictetures were investigated for Harem, a corpus of Portuguese: BERT (DEVLIN et al., 2018) and a bidirectional neural network LSTM (BiLSTM). The main goal was to explore features based on the entities contextual distribution by means of word embeddings vectors associated with linguistic features. We used as features part-of-speech tagging, spelling formats, and lexical resources. Those features were concatened with word embeddings vectors and fed into the BiLSTM. Our results showed a significant performance improvement with this model if compared to a BiLSTM using only word embeddings. On the other hand, BERT model obtained scores close to the Harem state-of-the-art.

Keywords: Named Entity Recognition, neural networks, word embeddings, linguistic feature representation.

Sumário

Lista de Figuras

Lista de Tabelas

1 Introdução

O Reconhecimento de Entidades Nomeadas (REN), do inglês *Named Entity Recognition*, é uma tarefa computacional que tem como objetivo identificar as Entidades Nomeadas de um corpus e classificá-las em um conjunto pré-estabelecido de tipos, tais como pessoa, local e tempo. A tarefa surgiu em 1996, em razão da 6ª *Message Understanding Conference* (MUC-6) (GRISHMAN; SUNDHEIM, 1996), um evento voltado para extração automática de informações em mensagens militares. Na época, foi notada a importância do reconhecimento de determinadas entidades para a extração de informações relevantes sobre o conteúdo textual de corpora não-estruturados, por exemplo, artigos jornalísticos e páginas da Web.

Nos primeiros trabalhos em REN, os pesquisadores estavam interessados em três tipos de entidade: Pessoa, Local e Organização, que podem ser considerados exemplos bem estabelecidos de Entidades Nomeadas. Além disso, essas categorias são extremamente frequentes em textos genéricos, o que, do ponto de vista prático, é importante quando se considera a disponibilidade de dados para o desenvolvimento de modelos computacionais.

Ao longo dos anos, o REN foi se expandindo para novas áreas de interesse, como Medicina, Direito e Investimentos, e, consequentemente, para outros domínios de aplicação, fazendo surgir uma variedade de tipos de Entidades Nomeadas. Podemos distingui-las quanto ao domínio de aplicação (específico ou genérico), grau de refinamento de classificação, modelo de classificação (horizontal ou hierárquico), entre outros. Para citar um exemplo, Sekine e Nobata (2004) propuseram um sistema de REN que identifica 200 tipos de entidades distintos, subdivididos hierarquicamente.

A aplicação do REN na área de Processamento de Língua Natural (PLN) é bastante diversificada, sendo essa uma tarefa importante em sistemas de Sumarização de Texto, Recuperação e Extração de Informação e Tradução Automática. Em sistemas de tradução automática, por exemplo, é necessário fazer o reconhecimento de nomes próprios, uma vez que esses termos, em geral, não são traduzidos entre línguas. O REN também é muito usado em etapas de pré-processamento em tarefas de classificação como inferência textual e pergunta-e-resposta.

Os bons resultados obtidos em alguns corpora, como o ConLL-2003 (TJONG KIM SANG; DE MEULDER, 2003), com o estado-da-arte em 94,3% de medida-F (YAMADA et al., 2020), podem fazer crer que o REN é uma tarefa resolvida, uma visão otimista sobre sua condição. Na verdade, os melhores resultados obtidos na tarefa surgem de corpora específicos, de domínio fechado, como textos jornalísticos e artigos da Wikipédia. Em corpora informais, como as redes sociais e blogs, em que a linguagem usada contêm diversas abreviações e as entidades citadas, em geral, são mais locais e específicas, as medidas de avaliação caem drasticamente. Em uma revisão apresentada por Li et al. (2020), os autores apontam que o melhor resultado obtido no W-NUT-2017 (DERCZYNSKI et al., 2017), um corpus informal, ficou um pouco acima de 40,0% de medida-F.

1.1 Abordagens para o REN

Podemos pontuar duas grandes abordagens para a modelagem de sistemas de REN: uma baseada em regras manuais e, outra, a de aprendizado de máquina. Os primeiros sistemas de REN eram modelos baseados em regras, desenvolvidos a partir de um conjunto de postulados definidos pelo pesquisador. Em geral, essas regras tentam capturar padrões linguísticos relevantes para a classificação de entidades, como a forma ortográfica (a identificação de letra inicial maiúscula, por exemplo), morfológica (como a terminação da palavra) e sintática (como a análise de dependência). De acordo com Jiang, Banchs e Li (2016), os modelos baseados em regras tendem a obter alta precisão

na tarefa, mas normalmente têm baixa portabilidade, isto é, apresentam resultados ruins quando aplicados em novos domínios.

Hoje em dia, os modelos baseados em aprendizado de máquina são mais utilizados. Murphy (2012) define aprendizado de máquina como um conjunto de métodos capaz de detectar automaticamente padrões em dados e usá-los para prever dados futuros. Em tarefas de PLN, os algoritmos de aprendizado de máquina são treinados em um corpus. Pela definição do dicionário online Priberam[1], um corpus é um conjunto de documentos que servem de base para a descrição ou o estudo de um fenômeno. Cada entrada recebida pelo algoritmo é chamada de *exemplo*, constituído de traços (do inglês, *features*), que são valores simbólicos ou numéricos extraídos do corpus para representar características relevantes, tais como a categoria gramatical da palavra, a frequência de um termo, etc.

Nadeau e Sekine (2007) dividem os algoritmos de aprendizado de máquina em três categorias de acordo com o tipo de treinamento: supervisionado, não supervisionado e semi-supervisionado. No aprendizado supervisionado, o algoritmo recebe como entrada um exemplo associado com uma etiqueta, a resposta esperada para classificação, e o seu objetivo é aprender uma função que mapeia exemplos em etiquetas. Já no aprendizado não supervisionado, o modelo é treinado somente com os exemplos e tem que aprender propriedades úteis dos dados sem supervisão. O semi-supervisionado, por sua vez, é treinado combinando os dois métodos anteriores.

Entre as três abordagens, o aprendizado supervisionado é a mais empregada no REN (JIANG; BANCHS; LI, 2016). Há uma distinção entre os algoritmos de aprendizado supervisionado clássicos baseados em traços e os de aprendizado profundo. No primeiro, os exemplos de treinamento são representados por um conjunto de traços pré-estabelecidos, assim como os modelos de regras, em que o algoritmo será treinado para inferir os padrões de reconhecimento para classificação. Alguns exemplos são *Hidden Markov Models* (HMM), *Conditional Random Fields* (CRF) e *Support Vector Machines* (SVM). Por sua vez, no aprendizado profundo, os traços do modelo são inferidos automaticamente durante o treinamento, eliminando a necessidade

[1] https://dicionario.priberam.org/corpus

de inserção de traços manuais. Os principais modelos entre esses são as redes neurais, como *Long Short-Term Memory* (LSTM) e *Convolutional Neural Network* (CNN).

1.2 Objetivos

Um dos objetivos centrais desta pesquisa foi a elaboração de um modelo de classificação semântica para o REN em língua portuguesa. Optou-se por uma abordagem baseada na distribuição contextual de palavras, explorando representações *word embeddings* geradas a partir de redes neurais e de traços manuais. Do ponto de vista linguístico, o intuito foi analisar a influência do contexto para a classificação de entidades nomeadas em um corpus e a variação de distribuição entre categorias. Na etapa final, os resultados obtidos foram comparados ao de outros modelos de REN do português.

Pontualmente, os objetivos de pesquisa foram:

1. Explorar a distribuição contextual de entidades nomeadas como um critério de classificação.
2. Apresentar um modelo de classificação para o REN na língua portuguesa.
3. Testar o modelo em um corpus e comparar seu desempenho com o de outros trabalhos.

1.3 Motivações e contribuições

A recuperação de informações contextuais é um aspecto importante para o REN, uma vez que fornece evidência para a classificação de entidades. Nesse sentido, as representações vetoriais baseadas na distribuição contextual, como os *word embeddings*, têm se mostrado uma maneira efetiva de capturar características relevantes de palavras. Inúmeros trabalhos apontam que a utilização de *word embeddings* melhora o desempenho na tarefa de REN (SEOK et al., 2016; AUGENSTEIN; DERCZYNSKI; BONTCHEVA, 2017).

Augenstein, Derczynski e Bontcheva (2017) sugerem que um dos motivos para essa melhora de desempenho está na capacidade de generalização de palavras. Isso porque essas representações são baseadas em traços de distribuição contextual da palavra, então, palavras ocorrendo em distribuições semelhantes têm uma representação similar. Todavia, os autores apontam que os modelos de REN ainda têm baixa capacidade de generalização, tendo dificuldade em classificar entidades raras ou não-vistas.

Uma técnica mais recente que tem sido investigada é a combinação de *word embeddings* e *embeddings* gerados a partir de traços manuais, chamados de *feature embeddings*, em modelos neurais. Os *feature embeddings* podem ser concatenados aos *word embeddings* e treinados conjuntamente para uma tarefa específica. Trabalhos recentes em REN adotaram essa abordagem (CHIU; NICHOLS, 2016; GHADDAR; LANGLAIS, 2018; SONG et al., 2020), mostrando melhora de desempenho da tarefa. Alguns traços comumente utilizados são baseados na forma ortográfica da palavra e no mapeamento por correspondência de palavras.

Dito isso, uma das etapas desta pesquisa foi explorar a combinação entre um modelo de representação *word embedding* e traços linguísticos em uma rede neural LSTM bidirecional. Foram testados *word embeddings* BERT (DEVLIN et al., 2018) combinados com *embeddings* gerados a partir de traços ortográficos, morfossintáticos e lexicais. Para o português, Santos e Guimaraes (2015) apresentam uma rede neural que combina *word embeddings* e traços ortográficos, porém, até onde sabemos, não há trabalhos explorando mais a fundo a influência de traços linguísticos em redes neurais. Além disso, testamos uma representação *word embedding* contextual, diferente dos autores citados. Desse modo, uma das contribuições desta pesquisa é a apresentação de uma rede neural LSTM combinada com traços linguísticos baseados na ortografia, na morfologia e no léxico e a discussão dos resultados de combinação desses com *word embeddings* do BERT.

Outro ponto analisado é a distribuição contextual de cada categoria de entidade no corpus Harem (SANTOS; CARDOSO, 2006). As distribuições contextuais de cada categoria de entidade são distintas, o que significa que determinados contextos aparecem mais frequentemente com uma(s) catego-

ria(s) do que com outras. Nesse sentido, Seok et al. (2016) mostram que os *embeddings* treinados na tarefa de REN são capazes de capturar similaridades entre os membros de uma mesma categoria de entidade. Com isso, uma das etapas desta pesquisa focou em testar as representações de palavras geradas a partir do BERT, explorando a similaridade entre elas para observar se são bons preditores de classificação no REN.

Por fim, as medidas de avaliação obtidas em determinados corpora têm se aproximado daquelas alcançadas por anotadores humanos, que varia entre entre 95% e 97%. Todavia, no Harem, principal corpus de avaliação disponível para REN em português, o estado-da-arte é de 78,67, obtido por Souza, Nogueira e Lotufo (2020). Portanto, ainda são necessárias pesquisas que busquem investigar métodos de REN no português, na tentativa de avançar o estado-da-arte na língua, além de fornecer recursos que possam ser disponibilizados publicamente para a comunidade.

1.4 Organização da pesquisa

No Capítulo 2, serão apresentados os pressupostos teóricos que embasam esta pesquisa. Será introduzida uma definição de Entidade Nomeada e uma discussão sobre as problemáticas na acepção do termo. Além disso, serão apresentadas as fundamentações linguísticas que motivaram a escolha do modelo adotado, assim como o corpus e métricas utilizados para avaliação de desempenho do modelo.

O Capítulo 3 é dedicado a uma breve revisão de trajetória do Reconhecimento de Entidades Nomeadas no português. Desse modo, um dos temas principais será o HAREM, o primeiro evento voltado para o REN, além das Coleções Douradas geradas a partir dele. O capítulo também contempla alguns trabalhos sobre REN e a recente adoção de modelos neurais.

Por sua vez, o Capítulo 4 traz conceitos básicos sobre os métodos de aprendizado profundo utilizados nesta pesquisa. Será fornecida uma introdução teórica a respeito de Redes Neurais, Redes Neurais Recorrentes, Transformer e BERT e seus respectivos métodos de treinamento.

No Capítulo 5, apresenta-se os modelos implementados, os hiperparâmetros estipulados para as redes neurais e os métodos de treinamento, assim como os experimentos realizados e bibliotecas auxiliares.

No Capítulo 6, faz-se uma discussão a respeito dos resultados obtidos pelos modelos, comparando o desempenho de cada abordagem testada e, ao final, será fornecido um quadro dos trabalhos mais recentes no REN do português. Por fim, serão discutidas as conclusões finais e trabalhos futuros no Capítulo 7.

2 | Pressupostos teóricos

Antes de iniciar a etapa de desenvolvimento do sistema de REN, é preciso escolher um corpus de desenvolvimento, um modelo de classificação e a métrica de avaliação. Este capítulo apresenta os principais conceitos subjacentes ao modelo adotado, o embasamento teórico linguístico de pesquisa e os materiais e medidas para avaliação do sistema.

2.1 A definição de Entidade Nomeada

Mesmo sendo o núcleo da tarefa de REN, o termo "Entidade Nomeada" não apresenta definição inteiramente consistente entre os trabalhos na área. Em seu surgimento, Grishman e Sundheim (1996) definem a tarefa de REN como a identificação de todas as pessoas, locais e organizações em um texto. Desse ponto de vista, as entidades nomeadas são tipos específicos de nomes próprios. Contudo, novas categorias foram sendo incluídas na tarefa, como expressões temporais (datas e horários) e cifras (valores em dinheiro e porcentagens).

Com isso, outras definições de EN surgiram na literatura, trazendo perspectivas distintas a respeito do objetivo da tarefa. Nas próximas subseções, introduz-se a definição de EN de Nadeau e Sekine (2007), uma das mais citadas em trabalhos, seguida de uma discussão crítica a respeito de sua contribuição. Por fim, será apresentada a proposta de definição adotada por nós.

2.1.1 Definição de base filosófica

Na definição de Nadeau e Sekine (2007), são consideradas Entidades Nomeadas os nomes cujo referente é um designador rígido, como proposto pelo filósofo Saul Kripke (1982). Para esse autor, um designador rígido é um nome que designa o mesmo objeto x em todos os mundos possíveis em que x exista. Kripke postula um mundo possível como dado pelas condições descritivas que associamos a ele (KRIPKE, 1982, p. 44), sendo assim algo estipulado em vez de dado *a priori*.

Para o filósofo, os nomes próprios constituem designadores rígidos, uma vez que o referente de um nome próprio permanece o mesmo, independente das descrições atribuídas a ele. O autor argumenta que o ex-presidente americano Nixon poderia não ter se tornado presidente dos EUA em 1970, porém "Nixon" ainda se referiria ao mesmo indivíduo. Isso seria possível porque podemos nos referir rigidamente a Nixon e estipular que estamos falando sobre o que poderia ter acontecido com ele, dadas certas circunstâncias. Com isso, Kripke exclui da definição de designador rígido as descrições definidas, tais como "o presidente dos EUA" e "o rei da França", visto que o referente nessas expressões não é sempre o mesmo em todas as situações possíveis.

Além dos nomes próprios, estão inclusas na definição de designador rígido algumas categorias da natureza, como nomes de substâncias (água e ouro, por exemplo) e espécies. O argumento para isso são as chamadas *propriedades essenciais*, uma propriedade do objeto que seria encontrada em qualquer mundo possível em que ele exista. Por exemplo, para a água a propriedade essencial seria a composição H_2O, que é o que a torna água.

2.1.2 Discussão sobre a proposta

A definição de Entidade Nomeada de Nadeau e Sekine (2007) parte do conceito de designador rígido, que inclui os nomes próprios e outras expressões que satisfazem a condição de unicidade pressuposta. Com isso, ela lida com determinadas entidades temporais e numéricas que não estavam inclusas na proposta inicial de REN (GRISHMAN; SUNDHEIM, 1996). Um exemplo são as expressões temporais definidas, como "21 de dezembro de 1965". Em

outros casos, a acepção de designador rígido como delimitador das entidades nomeadas é perdido por motivações práticas, como em "25 de setembro", que não possui ano definido. Em contrapartida, determinados nomes comuns considerados designadores rígidos, como ouro e água, não são, em geral, reconhecidos como ENs. Isso mostra uma divergência entre o que é um designador rígido por definição e o que está sendo adotado como Entidade Nomeada na aplicação prática.

Segundo, Marrero et al. (2013) apontam que há nomes próprios que podem se referir a mais de um objeto no mundo, como é o caso dos nomes de pessoas. A partir disso, conclui-se que um nome próprio isolado não necessariamente é um designador rígido, a menos que o referente seja fixado em relação a determinado contexto (caso de Nixon, que foi presidente dos EUA em 1970). Existem também os casos mais complexos, em que não é óbvio qual seria o referente para um nome próprio. Por exemplo, qual é o referente de "Apple"? Seria o conjunto de todas as filiais da empresa espalhadas pelo mundo? Ou algo abstrato, como as propriedades essenciais que tornam a empresa o que ela é? E se for isso, quais seriam essas propriedades? Essa é uma discussão que não cabe aprofundar aqui, mas mostra as dificuldades de aplicação do conceito de designador rígido para a definição de Entidade Nomeada.

2.1.3 Conclusões

Um dos principais objetivos do REN é extrair palavras informativas de textos. Os nomes próprios são ótimos candidatos para isso, já que através deles é possível identificar, por exemplo, de quem é a autoria de um texto, o lugar em que foi escrito, os agentes envolvidos na narrativa, entre outros. Além disso, é comum que as pesquisas na Web busquem por entidades, fazendo com que o REN seja importante para recuperar documentos relevantes naquela busca.

A dificuldade de adotar a definição de Entidade Nomeada como sendo os nomes próprios em geral é estabelecer os limites de busca em um texto, uma vez que praticamente qualquer coisa pode ter um nome próprio. Categorias como pessoa, local e organização são exemplos bem aceitos no REN, mas outras categorias, tais quais disciplinas (como "Teoria da Computação"),

fabricantes e modelos de automóveis (“BMW”, “Fusca”, etc.) e marcas (como “Havaianas”), são mais incomuns.

Conclui-se que não parece ser possível estabelecer uma definição única para o termo Entidade Nomeada. Como argumentam Marrero et al. (2013), cada trabalho vai considerar as entidades relevantes dentro do escopo e objetivos de sua pesquisa. Dito isso, o interesse aqui está na investigação dos nomes próprios como uma categoria linguística. Assim, diferente da acepção defendida por Kripke, foi adotada uma perspectiva pragmática, que assume que a referência de um nome próprio é atribuída somente em uso, ou seja, dentro de um contexto. Com isso, mesmo que um nome próprio possa ter mais de uma referência, é possível fixá-la pelo contexto.

2.2 Modelização da tarefa

Uma etapa fundamental para o desenvolvimento de um sistema de REN é a escolha de um modelo. Um modelo é uma representação simplificada da realidade baseado em um conjunto de pressupostos teóricos que pode ser aplicado para interpretar uma amostra de dados. Na área de Linguística Computacional, um modelo tem a função de representar determinada característica linguística de modo que possa ser processado por um computador. Há diversas formas de representar palavras em modelos computacionais, entre elas estão as representações lógicas, simbólicas e probabilísticas.

Nesta pesquisa, foi adotada uma abordagem probabilística, baseada na distribuição das palavras. Os modelos probabilísticos representam a língua em função da probabilidade de ocorrência de palavras em determinados contextos em um corpus. Nas seções seguintes, serão apresentados em mais detalhes alguns conceitos importantes subjacentes nesses modelos.

2.2.1 Modelo de Linguagem

No Processamento de Língua Natural, um modelo de linguagem refere-se ao conhecimento de um sistema sobre o que constitui uma palavra possível, quais palavras têm probabilidade de co-ocorrer e em que sequência (INDURKHYA;

DAMERAU, 2010, p. 340). É possível formulá-lo como uma distribuição de probabilidade $P(W)$ sobre uma sequência de palavras W que reflete quão frequentemente W ocorre como uma sequência.

Um dos modelos de linguagem mais simples é o n-gramas. Dado um vocabulário finito de palavras V, a função de distribuição de probabilidade p do vocabulário V é satisfeita por:

$$\sum_{x \in V} p(x) = 1 \tag{2.1}$$

Uma forma de obter a função p é através de um corpus de treinamento para aprender as estimativas de máxima verossimilhança e computar as probabilidades das sentenças no corpus. Assim, dada uma sequência de palavras $(x_1, ..., x_m)$, a função de probabilidade condicional das palavras é calculada pela regra da cadeia, como expressa a Equação 2.2.

$$p(x_1, ..., x_m) = \prod_{i=1}^{m} p(x_i | x_1, ..., x_{i-1}) \tag{2.2}$$

Pela Equação 2.2, a probabilidade de ocorrência de uma palavra é dada por todas as palavras precedentes a ela, o que é chamado de *histórico*. Na prática, é impossível estimar esses valores, já que a maioria dos históricos são únicos ou ocorrem apenas algumas vezes (INDURKHYA; DAMERAU, 2010, p. 344). Desse modo, essa probabilidade é computada como uma aproximação, em que se assume que uma palavra depende somente das n palavras imediatamente precedentes a ela (Equação 2.3). Para exemplificar, em um modelo de bigramas ($n = 2$), a probabilidade de ocorrência de uma palavra x_i é condicionada somente pelas duas palavras anteriores a ela.

$$p(x_1, ..., x_m) \approx \prod_{i=1}^{m} p(x_i | x_{i-(n-1)}, ..., x_{i-1}) \tag{2.3}$$

O modelo descrito é unidirecional e *forward* (tradução, para frente), já que percorre a sequência de palavras da esquerda para a direita, observando apenas as palavras precedentes. Também é possível computar um modelo *backwards* (tradução, para trás), que percorre a sequência em reverso, computando as

probabilidades condicionais da direita para a esquerda, prevendo, portanto, as palavras com base no contexto futuro.

Os modelos de linguagem estatísticos, como n-gramas, têm dificuldades de lidar com a esparsidade de dados, que aumenta conforme o tamanho do corpus de treinamento cresce. Isso faz com que o número de sequências de palavras possíveis aumente exponencialmente. Na tentativa de lidar com os problemas de esparsidade, Bengio et al. (2003) introduzem o primeiro modelo de linguagem neural. Nos modelos neurais, a entrada é uma representação distribuída, um vetor numérico, em vez de uma representação simbólica. Comparados aos modelos de linguagem estatísticos, esses não necessitam de suavização, conseguem lidar com históricos muito mais longos e podem generalizar sobre contextos de palavras similares (JURAFSKY; MARTIN, 2018, p. 137). Os modelos de linguagem neurais podem ser treinados utilizando redes neurais *feedforward*, como o proposto por Bengio et al. (2003), ou recorrentes, mais utilizados atualmente.

2.2.2 Representação Vetorial de Palavras

De acordo com Turian, Ratinov e Bengio (2010), uma representação de palavra é um objeto matemático associado a cada palavra, frequentemente um vetor. A representação vetorial se mostrou uma maneira eficaz de codificar palavras em modelos que só operam com entradas numéricas, como as redes neurais.

Um método muito utilizado para gerar representações de palavras é chamado de *word embedding*, uma representação vetorial induzida através de modelos de linguagem neurais. Esse modelo gera representações de palavras a partir de um vocabulário V, reduzindo uma matriz de contextos de ocorrência de cada palavra em um vetor numérico denso. Em oposição a uma representação vetorial esparsa, que contém muitos valores igual a zero, um vetor denso apresenta poucos valores em zero. Em *word embeddings*, cada dimensão vetorial corresponde a um traço da palavra, com o intuito de capturar propriedades semânticas e sintáticas úteis (TURIAN; RATINOV; BENGIO, 2010).

Collobert et al. (2011) apresentam um dos primeiros métodos para se obter vetores n-dimensionais para representação de palavras gerados a partir de dados não-anotados induzidos por redes neurais. Outros métodos de *word embeddings* surgiram a partir daí, como Word2Vec (MIKOLOV et al., 2013), GloVe (PENNINGTON; SOCHER; MANNING, 2014) e FastText (BOJANOWSKI et al., 2017).

Esses são modelos de representação livres de contexto. Isso significa que cada palavra do vocabulário é representada por um único vetor, independente do contexto em que aparece. Para exemplificar, "manga" é representa pelo mesmo vetor nos contextos "eu comi uma *manga*" e "a *manga* da camiseta", mesmo que o significado das palavras seja diferente em cada exemplo. Desse modo, esses modelos não são capazes de desambiguar palavras homógrafas (isto é, graficamente homônimas) ou diferentes sentidos de uma mesma palavra (polissemia).

Já os modelos mais modernos, como BERT (DEVLIN et al., 2018), GTP (RADFORD et al., 2018) e ELMo (PETERS et al., 2018), são contextuais, ou seja, a representação de cada palavra é baseada no contexto em que ela se encontra. Em comparação aos modelos livres de contexto, esses geram representações mais acuradas de palavras, com sensível melhora de desempenho nas tarefas em que são aplicados. Nesta pesquisa, foi adotado como modelo de representação de palavras o BERT. Diferente dos outros modelos citados, que são unidirecionais (GTP) ou concatenam modelos unidirecionais de direções opostas (ELMo), o BERT codifica a representação de cada palavra com base no contexto à direita e à esquerda conjuntamente.

2.2.3 Codificação de traços

Com o advento das redes neurais, a inserção de traços manuais em inúmeras tarefas acabaram sendo dispensáveis. Todavia, trabalhos recentes (GHADDAR; LANGLAIS, 2018; CHIU; NICHOLS, 2016; LIU; YAO; LIN, 2019) têm mostrado que a combinação de *word embeddings* com outros traços pode ser benéfica para o REN. Quando esses traços são valores categóricos, é preciso convertê-los em numéricos para utilizá-los em redes neurais. Existem diversas maneiras

de fazer isso. A seguir, serão apresentadas duas alternativas de codificação de traços categóricos.

Vetor one-hot

Na representação *one-hot*, o primeiro passo é converter os valores dos traços em índices. Em seguida, cada índice é transformado em um vetor binário, em que a posição do traço será 1 e os demais índices, 0. Abaixo está um exemplo de codificação *one-hot* para nível de escolaridade.

$$\begin{array}{ccccc} \text{ensino fundamental} & \rightarrow & 0 & \rightarrow & [1\ 0\ 0] \\ \text{ensino médio} & \rightarrow & 1 & \rightarrow & [0\ 1\ 0] \\ \text{ensino superior} & \rightarrow & 2 & \rightarrow & [0\ 0\ 1] \end{array}$$

Vetor numérico

Os traços também podem ser representados por vetores numéricos densos, como em *word embeddings*. Para isso, da mesma forma que em *one-hot*, os valores dos traços precisam ser convertidos em índices. Cada índice é associado a um vetor denso de dimensão d, um parâmetro a ser estabelecido *a priori*. Esses vetores podem ser inicializados aleatoriamente e treinados em uma tarefa específica. Diferente da codificação *one-hot*, que marca somente a presença de um traço, cada dimensão do vetor numérico representa determinada característica da palavra.

2.3 Embasamento teórico

A utilização de modelos baseados em distribuição de palavras é recente na área de PLN, contudo está embasada em uma ideia antiga, defendida por inúmeros pesquisadores na área de Linguística. Na próxima seção, serão apresentados os pressupostos teóricos motivadores para a escolha desses modelos no Reconhecimento de Entidades Nomeadas e suas aplicações computacionais.

2.3.1 Evidências linguísticas

As Entidades Nomeadas constituem uma classe muito heterogênea em termos de ortografia e morfologia, composta de nomes advindos de diversas línguas ("Apple", "Samsung", "Citroën", "Xiaomi", etc.), com inúmeras grafias concorrentes (como "Alan" e "Allan"), entre outros. Desse modo, não é confiável modelar um sistema de classificação baseado somente na forma superficial de entidades. Ter uma base de dados contendo uma lista de ENs também não seria suficiente, uma vez que elas são parte de uma classe aberta, o que significa que sempre existem itens não previstos. Assim, é necessário recorrer a outros aspectos linguísticos para ajudar na classificação de Entidades Nomeadas.

Pensando nisso, voltamo-nos para a análise de pistas contextuais que pudessem indicar a categoria de uma entidade. Observou-se que a distribuição dos tipos de entidades não é uniforme, ou seja, há contextos que favorecem a ocorrência de um tipo de entidade. Uma posição que pode conter forte indicador do tipo de entidade é a imediata esquerda. Alguns exemplos são fornecidos a seguir.

(A) A Sra. Laura é muito inteligente.

(B) Os alunos não gostam do Prof. Carlos.

(C) O banco Bradesco aumentou o valor das taxas.

(D) O Brasil é banhado pelo oceano Atlântico.

(E) Há um congestionamento na rodovia Castelo Branco.

Nos exemplos acima, as palavras sublinhadas nas sentenças atuam como modificadores restritivos, em que um nome usado para denominação genérica (como "oceano" ou "professor") é combinado ao nome próprio para especificá-lo. Isso ocorre, entre outros, por causa de associações entre propriedades semânticas de palavras, como a seleção e a atribuição de determinadas características.

Os modificadores restritivos acima funcionam bem para distinguir tipos de entidade, uma vez que esses nomes genéricos denominam subcategorias de entidades conectados com categorias superiores em uma hierarquia. Nas

sentenças, "senhora" e "professor" são indicativos de pessoa, enquanto "oceano" e "rodovia" ocorrem com lugares e "banco" com organização. Isso significa que é esperado na sentença (A) nomes próprios como "Maria", "Ana" e "Isabel", mas não "Atlântico", com a interpretação de oceano, exceto talvez em contextos fictícios, como livros de fantasia, em que objetos inanimados podem ser humanizados.

Desse modo, há contextos de ocorrência que estão semanticamente associados a determinados tipos de entidades. Por causa disso, esses modificadores ajudam na desambiguação entre entidades com a mesma forma superficial. Isoladamente o nome "Castelo Branco" pode se referir a inúmeros tipos de entidade, mas o modificador "rodovia" precedendo a entidade mostra que, nesse caso, ele está se referindo a um local.

Os exemplos apresentados mostram apenas um caso de co-ocorrência de palavras fortemente associadas entre os inúmeros existentes nas línguas naturais. Já foi observado por diversos linguistas que a combinação entre elementos linguísticos não é arbitrária, mas feita pela seleção de membros de determinadas categorias ordenados sequencialmente (JAKOBSON, 1954; HARRIS, 1954), sendo que certas combinações de palavras são mais prováveis do que outras, mesmo que ambas sejam sintaticamente aceitáveis. A ideia de distribuição semântica foi sistematicamente explorada na área da Semântica Distribucional.

2.3.2 Semântica Distribucional

Na década de 1950, diversos linguistas discutiam a respeito de análises semânticas baseadas na distribuição contextual de palavras em textos. A hipótese central era a de que palavras ocorrendo em contextos similares tendem a ter um significado semelhante, o que ficou conhecido como Hipótese Distribucional. Entre os principais nomes da Semântica Distribucional estão John Rupert Firth e Zellig Harris.

Em *"A Synopsis of Linguistic Theory"*, Firth escreve sua citação mais célebre: *"you shall know a word by the company it keeps"*[1] (FIRTH, 1957,

[1]Tradução livre: "Conhecerás uma palavra por aquelas que a acompanham".

p. 11), que posteriormente embasaria uma das ideias-chave na Semântica Distribucional. Ao tratar do significado, o autor adota um perspectiva baseada no contexto, distinguindo dois níveis de análise: da palavra (colocação) e da gramática (coligação). No nível da palavra, o significado é dado pelas colocações habituais em que uma palavra ocorre, isto é, as palavras que a acompanham em determinada ordem. No nível gramatical, o significado é dado em termos de categorias de palavras e a inter-relação dessas em coligações, ou seja, as relações entre as categorias gramaticais entre si, por exemplo, substantivos com verbos, com adjetivos e assim por diante.

Do mesmo modo, Harris (1954) afirma que diferenças de significado se correlacionam com diferenças na distribuição de itens, ou seja, palavras com significados muito distantes ocorrem em contextos habituais distintos. Para ele, a distribuição de um elemento pode ser compreendida como a soma de seus ambientes, sendo esse um vetor contendo os elementos co-ocorrendo com o primeiro em posições particulares. Essa sistematização de distribuição pode ser aplicada não somente para palavras individuais, mas também para outros níveis de análise linguística (na fonologia, na morfologia e outros).

Depois da década de 50, a Semântica Distribucional não recebeu muita atenção na Linguística, mas a disciplina não desapareceu, sendo incorporada por outras áreas de pesquisa, como no Processamento de Língua Natural e na Ciência Cognitiva, em que suas ideias influenciaram em grande parte devido a sua aplicação prática.

2.3.3 Aplicação Computacional

No âmbito do PLN, a Semântica Distribucional é aplicada, principalmente, na geração de modelos de linguagem. Esses modelos são centrais em tarefas de processamento linguístico, sendo usados em corretores textuais, reconhecimento de voz e geração de texto, além de serem utilizados como função auxiliar no treinamento de inúmeras tarefas.

Esses modelos também são usados na representação semântica de palavras[2], como *word embeddings*. Como essas representações são vetores numéricos, é possível calcular o grau de similaridade entre palavras, por meio de métodos como a similaridade de cossenos. Essa medida é dada pelo produto escalar normalizado entre dois vetores. Ela atua como uma métrica de similaridade porque tende a ser alta quando os dois vetores têm grandes valores nas mesmas dimensões e baixa em casos de vetores com zeros em diferentes dimensões (JURAFSKY; MARTIN, 2018, p. 103).

A Figura 2.1 traz uma representação visual das dez palavras mais similares a "*girl*".

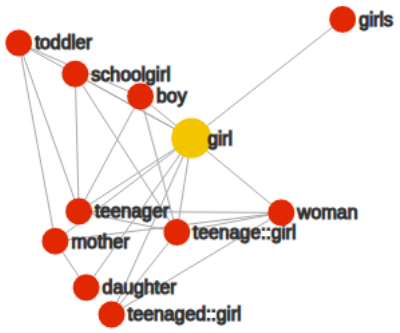


Figura 2.1: Representação de similaridade da palavra "*girl*" obtido por um modelo Word2Vec (MIKOLOV et al., 2013) no corpus Google News. A visualização foi gerada a partir do site `http://vectors.nlpl.eu/explore/embeddings/en/`.

Na Figura 2.1, as linhas indicam associações de similaridade com a palavra *girl*. No grafo, a palavra mais similar a "*girl*" (garota) é "*boy*" (garoto). Outras palavras que aparecem no grafo também têm significado associado com a palavra *girl*, como *girls* (garotas), *woman* (mulher) e *teenager* (adolescente). Análises como essa corroboram a hipótese de que palavras ocorrendo em contextos semelhantes tendem a ter um significado parecido.

Além dessa aplicação, os *word embeddings* podem ajudar a melhorar o desempenho de tarefas usando redes neurais. Tendo em vista que esses

[2]Estou pontuando somente a representação de palavras, mas o mesmo pode ser feito no nível ortográfico, morfológico e sintático.

modelos são custosos para treinar, é possível salvar *word embeddings* pré-treinados e disponibilizá-los em repositórios para serem utilizados em outras tarefas.

2.4 Avaliação da tarefa de REN

A avaliação dos sistemas de REN é feita através da comparação do resultado obtido por um modelo e um corpus anotado com as respostas esperadas. Existem dois tipos de corpus anotado: dourado e prateado. Um corpus dourado é aquele contendo anotação manual, ou seja, feito e/ou revisado por humanos, em geral especialistas na área de conhecimento da tarefa a ser resolvida. Já um corpus prateado é anotado automaticamente através de técnicas computacionais. Desenvolver um corpus dourado é muito mais custoso e demorado do que um prateado, portanto, o número de corpora dourado é bem menor, principalmente em determinadas tarefas.

2.4.1 Formatos de anotação de corpora para REN

Para fazer a anotação de ENs em um texto, é preciso seguir diversos procedimentos, entre os quais está o tipo de etiquetagem. Uma abordagem muito utilizada nos primeiros trabalhos de REN foi a anotação por XML, sendo adotada em inúmeras competições importantes, como o MUC (GRISHMAN; SUNDHEIM, 1996) e o Harem (SANTOS; CARDOSO, 2006). XML é uma linguagem de marcação que estrutura informações através de etiquetas.

Tradicionalmente, somente a palavra ou palavras referentes a uma entidade eram etiquetadas no texto, delimitadas por < e >, como apresentado no exemplo abaixo. A etiqueta "EN" indica as fronteiras da entidade nomeada e "TIPO" se refere a etiqueta atribuída a ela.

> A <EN TIPO="LOCAL">Estatua da Liberdade< /EN> foi projetada pelo escultor francês <EN TIPO="PESSOA">Fredéric Auguste Bartholdi< /EN>.

Atualmente, é mais comum que os corpora de REN sejam anotados no formato BIO (Begin-Inside-Outside) (ver Tabela 2.1). Nele, todas as palavras

do corpus recebem uma etiqueta indicando sua categoria, sendo que as letras B e I são usadas para marcar, respectivamente, o início de uma entidade e uma ou mais palavras dentro de uma entidade. Essas letras são adicionadas antes da categoria da EN, como "B-Pessoa". Por sua vez, O é reservado para as palavras que não são ENs. Os corpora mais recentes tendem a aplicar esse esquema de anotação, uma vez que ele é baseado em unidades de palavra (*token*) em vez de sequências de palavras (*chunk*), como no exemplo anterior.

2.4.2 Corpus para REN do Português

No português, existem poucos corpora anotado para o REN. Entre os disponíveis, os mais utilizados são as Coleções Douradas (CD) do Harem, compostas por três corpus: Primeiro Harem, Mini Harem e Segundo Harem. As CDs do Harem são anotadas em formato XML e estão disponíveis gratuitamente na Linguateca. Utilizamos como material de treino e teste, respectivamente, o Primeiro Harem e o Mini Harem. Mais detalhes a respeito do Harem serão discutidos no Capítulo 3.

2.4.3 Avaliação de desempenho da tarefa

Ao longo dos anos, foram propostas diferentes métricas de avaliação para o REN. Dois tipos de métricas serão apresentados a seguir: a avaliação por entidade e por palavra. A avaliação por entidade é feita com base na correspondência por *chunk*, que pode ser composta por uma ou mais palavras. Como medida de desempenho da tarefa, adotamos o protocolo de avaliação do CoNLL (TJONG KIM SANG, 2002; TJONG KIM SANG; DE MEULDER, 2003). As métricas de avaliação propostas nessa conferência são baseadas nas medidas de precisão, cobertura e medida-F.

No CoNLL, a classificação de uma entidade é considerada correta somente se é uma correspondência idêntica daquela anotada no corpus. Assim, a precisão (p) é dada pela porcentagem de entidades classificadas corretamente pelo sistema e a cobertura (c) é a porcentagem de entidades presentes no corpus dourado encontradas pelo sistema. O cálculo de medida-f (f) é apresentado na Equação 2.4, em que $\beta = 1$.

$$f_{(\beta)} = \frac{(\beta^2 + 1) * p * c}{\beta^2 * (p + c)} \tag{2.4}$$

Já na avaliação por palavra, o acerto do modelo é computado para cada etiqueta atribuída. As medidas usadas são precisão, cobertura e medida-f. A precisão é a porcentagem de etiquetas atribuídas corretamente pelo modelo, enquanto a cobertura é a porcentagem de etiquetas encontradas pelo modelo contidas no corpus dourado. A medida-f é a média harmônica entre precisão e cobertura.

A Tabela 2.1 traz um exemplo de como é feita a avaliação de um sistema de REN com uma anotação BIO.

Texto	**Anotação dourada**	**Predição do modelo**
Steven	B-PES	B-PES
Jobs	I-PES	I-PES
foi	O	O
co-fundador	O	O
da	O	O
Apple	B-ORG	B-ORG
Inc.	I-ORG	O
.	O	O

Tabela 2.1: Exemplo de avaliação de um sistema de REN.

Observando a Tabela 2.1, o modelo obteve uma precisão de 0,5, a cobertura de 0,5 e a medida-f de 0,5, pelas métricas do CoNLL. Isso porque há duas entidades no texto (Steven Jobs e Apple Inc.), porém somente uma foi anotada corretamente pelo modelo. Na medida por palavra, a precisão é 0,75, a cobertura é 0,75 e a medida-f é 0,75, pois o cálculo é feito com base no número de palavras anotadas corretamente. Essa comparação ilustra a diferença de resultado entre as duas medidas, sendo que a métrica do CoNLL é mais rígida.

Foram adotadas as medidas de avaliação do COnLL, pois esse é o sistema mais utilizado para a tarefa. Desse modo, o desempenho final do modelo é obtido pelo *micro média* da precisão e da cobertura entre todas as entidades, isto é, em vez de obter as medidas individuais para cada categoria e, ao final,

fazer a média aritmética, obtém-se uma média com base no valor absoluto de acertos e erros, independente das categorias. A micro média é, em geral, mais alta do que a média aritmética.

3 O Reconhecimento de Entidades Nomeadas no Português

O Reconhecimento de Entidades Nomeadas é uma tarefa relativamente nova, principalmente considerando seu percurso na língua portuguesa. Nesse cenário, o Harem surge como a primeira iniciativa para impulsionar a pesquisa no REN, fornecendo as Coleções Douradas (CD), que até hoje são os corpora mais utilizados na avaliação da tarefa. Este capítulo fornecerá uma introdução sobre o Harem e suas CDs, além de outros recursos disponíveis para o REN do português e o cenário atual do estado-da-arte na tarefa.

3.1 Harem

A iniciativa do Harem (Avaliação de sistemas de Reconhecimento de Entidades Mencionadas) surgiu como uma conferência para discussão e desenvolvimento de sistemas de REN em língua portuguesa, semelhante as promovidas pelo CoNLL (TJONG KIM SANG, 2002; TJONG KIM SANG; DE MEULDER, 2003). O Harem (SANTOS; CARDOSO, 2007; MOTA; SANTOS, 2008) foi o primeiro evento voltado para o REN do português, criado a partir de uma avaliação conjunta promovida pela equipe do Linguateca[1]. Nas duas edições do evento, foram compiladas as Coleções Douradas (CD), Primeiro Harem e Segundo Harem, além do Mini Harem, organizado para uma edição especial do evento.

[1] https://www.linguateca.pt/

3.1.1 As Coleções Douradas do Harem

A primeira edição do Harem (SANTOS; CARDOSO, 2007) foi dividida em duas partes: Primeiro Harem e Mini Harem. A primeira ocorreu em 2005, reunindo 10 equipes de participantes de seis países diferentes para competir no evento. A CD do Primeiro Harem é uma coleção de textos anotados manualmente para o REN que foi utilizada para avaliar o desempenho dos sistemas submetidos. É composta de 129 documentos extraídos de textos de diversos gêneros, entre eles jornalístico, Web e oral, originários principalmente do português do Brasil e de Portugal.

No total, foram anotadas 5110 entidades distribuídas entre 10 categorias, sendo elas Pessoa, Local, Organização, Tempo, Obra, Acontecimento, Abstração, Coisa, Valor e Variado. As categorias são subdividas em tipos, somando 41 no total. A categoria Tempo, por exemplo, contém os tipos Data, Hora, Período e Cíclico. Essa divisão hierárquica entre categorias e tipos está presente em todas as CDs do Harem.

O Mini Harem ocorreu no ano seguinte, em que cinco dos participantes da edição anterior submeteram novamente seus sistemas para avaliação devido a atrasos nos prazos de submissão. Uma nova CD foi anotada para a ocasião, o Mini Harem, contendo 128 documentos extraídos do mesmo domínio do Primeiro Harem e 3758 entidades nomeadas.

Em 2008, foi realizada a segunda edição do Harem (MOTA; SANTOS, 2008), em que o desempenho de dez novos sistemas de REN foram avaliados. Para a nova CD, coletaram-se 129 documentos do português do Brasil e de Portugal, contemplando textos de diferentes gêneros, a citar publicações em blogs e textos jornalísticos. Os autores buscaram balancear mais o Segundo Harem, inserindo textos de domínios ausentes no primeiro, como artigos da Wikipédia. O corpus contém 7272 entidades categorizadas em Pessoa, Local, Organização, Tempo, Acontecimento, Abstração, Valor, Obra, Coisa e Outro.

A única distinção nas categorias do Primeiro e do Segundo Harem é a substituição de "Variado" por "Outro". Essa alteração foi, posteriormente, corrigida nas CDs anteriores. Já os tipos sofreram inúmeras alterações. Algumas dessas foram apenas mudanças na denominação, enquanto outras tiveram

maior impacto no esquema de classificação, por exemplo, a transferência de determinados tipos para outras categorias. Essas inconsistências na anotação acabaram prejudicando uma unificação entre as três CDs.

Uma novidade introduzida no Segundo Harem foi a tarefa ReRelEM (Reconhecimento de Relações de Entidades Mencionadas), em que os sistemas deviam encontrar relações entre entidades em um documento, o que resultou em uma mini CD contendo 12 documentos e 573 entidades. Apesar desse não ser o enfoque desta pesquisa, essa é uma tarefa importante na área de Extração de Informação que tem recebido pouca atenção, principalmente com relação ao português.

Na Tabela 3.1 está a distribuição de entidades por categoria em cada uma das CDs do Harem.

Categoria	**Primeiro Harem**	**Mini Harem**	**Segundo Harem**
Abstração	449	326	286
Acontecimento	128	63	300
Coisa	82	180	308
Local	1286	895	1311
Obra	222	130	449
Organização	956	622	961
Outro	40	14	79
Pessoa	1029	836	2036
Tempo	434	364	1189
Valor	484	328	353
Total	5110	3758	7272

Tabela 3.1: Distribuição das categorias de ENs nas CDs do Harem.

Verifica-se que a distribuição de membros por categoria não é equilibrada, havendo categorias com mais de 1000 itens e outras com menos de 100. Esse fator afeta diretamente a etapa de classificação, tendo em vista que é mais difícil para os modelos generalizarem traços nas categorias menores.

Para concluir, o Harem foi importante para o REN em língua portuguesa, fornecendo recursos gratuitos e fomentando as pesquisas iniciais nessa área. Até hoje, as CDs do Harem são os corpora mais utilizados, primeiro por serem um dos poucos anotados manualmente para REN, o que evidencia uma

carência na produção de materiais para a tarefa. Ademais, vale salientar que as CDs do Harem não são corpora grandes, um fator influenciador quando se trabalha com modelos de aprendizado profundo.

3.2 Outros corpora para REN

Além das Coleções Douradas do Harem, existem outros corpora de REN para o português que são menos conhecidos. Entre esses, o maior e mais popular é o WikiNER (NOTHMAN et al., 2013), um corpus multilíngue desenvolvido a partir de artigos da Wikipédia. No total, foram utilizados textos de nove línguas, incluindo o português. Esse corpus foi gerado a partir de anotação automática e rotula as entidades em quatro tipos: Pessoa, Local, Organização e Misto. O spacy[2], uma biblioteca de Python voltada para PLN, teve seus modelos do português treinados com o WikiNER.

Dois anos depois, Júnior et al. (2015) desenvolvem o Paramopama, um corpus gerado a partir de uma porção do WikiNER em português. Para melhorar o corpus, a anotação anterior foi revisada e corrigida manualmente. Além disso, a categoria Tempo foi incluída no Paramopama pelo método de anotação automática e revisão manual. Já voltado especificamente para a área de Direito, cita-se o LeNER (ARAUJO; TEÓFILO, 2018), um corpus composto somente de documentos legais extraídos de leis e decisões jurídicas. O LeNER foi anotado manualmente e categoriza as entidades em Pessoa, Local, Organização, Tempo, Legislação e Jurisprudência.

Os corpora referidos não são os únicos voltados para o REN do português, apenas uma amostra de alguns materiais acessíveis para a tarefa. A Tabela 3.2 compara os corpora apresentados de acordo com o tipo (dourado e prateado), total de entidades e total de palavras. O que está sendo considerado Harem I é concatenação das CDs Primeiro e Mini Harem.

[2] https://spacy.io/

Corpus	Tipo	Palavras	Entidades
Harem I	Dourado	80.000	8.868
Harem II	Dourado	100.000	7.272
Paramopama	Prateado	310.000	42.769
WikiNER	Prateado	2.830.000	330.286
LeNER	Dourado	318.073	44.513

Tabela 3.2: Comparação dos corpora do português.

3.3 As pesquisas no Harem

Mesmo após as duas edições do Harem, as pesquisas com base em suas CDs continuaram. Considerando que adotamos o Primeiro e o Mini Harem como corpora de treinamento e avaliação, respectivamente, esta seção se dedica a uma revisão de alguns trabalhos realizados com os corpora do Harem, desde os primeiros modelos testados até os mais atuais.

Os sistemas submetidos para o Primeiro Harem variaram entre baseados em regras e aprendizado de máquina. Na época, o sistema que obteve maior pontuação na tarefa de classificação foi o Palavras-NER (BICK, 2006), com medida-F de 58,3. O Palavras-NER é um modelo baseado em regras que integra o REN à tarefa de análise gramatical, recorrendo a regras morfossintáticas e listas para a classificação. Outro sistema de destaque naquela edição foi o SIEMES 2 (SARMENTO, 2007), um modelo baseado na similaridade ortográfica e um conjunto de regras contextuais, que obteve 53,3 de medida-F. Já no Segundo Harem, o sistema com melhor desempenho foi o Priberam (AMARAL, C. et al., 2008), com medida-F de 59,08, seguido do Rembrandt-2 (CARDOSO, 2008), com 58,08[3], dois sistemas baseados em regras.

Os resultados reportados no Harem não foram muito empolgantes, mostrando que o REN estava longe de estar resolvido no português. Outras pesquisas se basearam nas CDs do Harem no intuito de superar os resultados citados. Milidiú, Santos e Duarte (2008) utilizam um modelo *Entropy Guided Transformation Learning* (ETL), baseado em árvores de decisão para extrair

[3]Medidas obtidas com relação à avaliação relaxada de ALT. Para mais detalhes, ver Mota e Santos (2008).

regras de classificação. Os resultados foram testados no Primeiro e no Mini Harem, com desempenho de, respectivamente, 63,27 e 63,04. Com variações de modelos *Conditional Random Fields* (CRF), Daniela Oliveira Ferreira do Amaral et al. (2013) e Pirovani e Oliveira (2017) relataram medida-F de 48,43 e 57,8, respectivamente, no corpus do Segundo Harem.

Com o crescimento da área de aprendizado profundo, os modelos de redes neurais ganharam força no REN. Santos e Guimaraes (2015) foi um dos trabalhos pioneiros em aplicar redes neurais no Harem. Os autores propõem a CharWNN, uma rede neural baseada em Collobert et al. (2011) que concatena representações de palavra e de caractere. Com esse modelo, eles obtêm medida-F de 65,41 no Mini Harem.

Para avaliar o desempenho de redes neurais no Harem, Fernandes, Cardoso e Oliveira (2018) testam três arquiteturas (BiLSTM, BiLSTM-CNN e BiLSTM-Char) com dois modelos de *word embeddings* pré-treinados: Word2Vec (MIKOLOV et al., 2013) e Wang2Vec (HARTMANN et al., 2017), superando o resultado anterior em 4% com uma BiLSTM-CNN. Também valendo-se de redes BiLSTM, Castro, Silva e Silva Soares (2018) combinam representações de palavra e de caractere e Santos, Consoli et al. (2019) avaliam a combinação de *word embeddings* contextuais do tipo Flair (AKBIK; BLYTHE; VOLLGRAF, 2018) com não-contextuais.

Por fim, no trabalho mais recente desta revisão, Souza, Nogueira e Lotufo (2020) testam uma arquitetura BERT (DEVLIN et al., 2018): uma rede neural baseada em redes Transformer (VASWANI et al., 2017). Os autores pré-treinam o BERT em corpora do português, gerando o BERT$_{PT}$, e refinam o modelo no Primeiro Harem, passando por uma camada CRF para classificação final.

3.3.1 O estado-da-arte

O estado-da-arte é uma medida de referência com relação ao estado atual de pesquisa em determinada tarefa. No REN, essa referência é tomada com base na melhor medida-F obtida em determinado corpus. A seguir, serão comparados os resultados dos trabalhos em aprendizado profundo discutidos na Seção 3.3 para o corpus Mini Harem. Todos os modelos foram testados no

cenário total e no seletivo. No primeiro, são consideradas as 10 categorias de entidades e no segundo 5 (Pessoa, Local, Organização, Tempo e Valor). As Tabelas 3.3 e 3.4 apresentam as medidas de precisão (P), cobertura (C) e medida-F (F1) no cenários total e seletivo, respectivamente.

Sistema	P	M	F1
CharWNN (SANTOS; GUIMARAES, 2015)	67,16	63,74	65,41
BiLSTM-CNN (FERNANDES; CARDOSO; OLIVEIRA, 2018)	72,64	67,50	69,97
BiLSTM-CRF (CASTRO; SILVA; SILVA SOARES, 2018)	72,78	68,03	70,33
BiLSTM-CRF+FlairBBP (SANTOS; CONSOLI et al., 2019)	74,91	74,37	74,64
BERT$_{PT-LARGE}$-CRF (SOUZA; NOGUEIRA; LOTUFO, 2020)	**80,08**	**77,31**	**78,67**

Tabela 3.3: Comparação do estado-da-arte no Mini Harem no cenário total.

Sistema	P	C	F1
CharWNN (SANTOS; GUIMARAES, 2015)	73,98	68,68	71,23
BiLSTM-CNN (FERNANDES; CARDOSO; OLIVEIRA, 2018)	70,67	66,35	68,44
BiLSTM-CRF (CASTRO; SILVA; SILVA SOARES, 2018)	78,26	74,393	76,27
BiLSTM-CRF+FlairBBP (SANTOS; CONSOLI et al., 2019)	83,38	81,17	82,26
BERT$_{PT-LARGE}$-CRF (SOUZA; NOGUEIRA; LOTUFO, 2020)	**84,82**	**81,72**	**83,24**

Tabela 3.4: Comparação do estado-da-arte no Mini Harem no cenário seletivo.

Como mostram as Tabelas 3.3 e 3.4, o estado-da-arte no Harem é o trabalho de Souza, Nogueira e Lotufo (2020) nos dois cenários, seguido pelo trabalho Santos, Consoli et al. (2019), com uma diferença de 4% no cenário total e 1% no seletivo. Em geral, o cenário seletivo alcança resultados melhores do que o total, chegando a alcançar uma distância de quase 6%.

4 | Redes neurais

Os modelos de redes neurais estão em alta no momento, pois seu desempenho alcança o estado-da-arte em várias aplicações. Desde o *perceptron*, um dos primeiros modelos de rede neural, surgiram diversas arquiteturas e métodos de treinamento para esses modelos. Neste capítulo, apresenta-se uma breve introdução sobre as redes neurais e seu treinamento, além da apresentação de dois tipos de arquitetura, as Redes Neurais Recorrentes e o BERT.

4.1 Fundamentos das redes neurais

As redes neurais artificiais (RN) surgiram como modelos inspirados em uma representação computacional de um cérebro. O cérebro é composto por um grande conjunto de neurônios que se conectam através de sinapses para transmitir sinais entre si. Similarmente, as redes neurais artificiais são compostas de unidades ou nós conectados diretamente por links que propagam ativações entre as unidades (RUSSELL; NORVIG, 2016, p. 728). A força de conexão entre os nós é determinada por um peso associado aos links. A Figura 4.1 traz uma representação de uma rede neural *feedforward*, que será discutida a seguir.

As redes neurais são divididas em camadas, sendo elas a camada de entrada, as camadas escondidas e a camada de saída. Cada nó (círculo na Figura 4.1) na primeira camada recebe como entrada um traço extraído de uma amostra de dados, por exemplo uma imagem ou um documento, sendo esse um valor numérico. Nas camadas escondidas, os nós são computados como a soma ponderada de suas entradas, mais um valor numérico, chamado *viés* (do inglês, *bias*). Essas entradas são recebidas de cada nó da camada anterior.

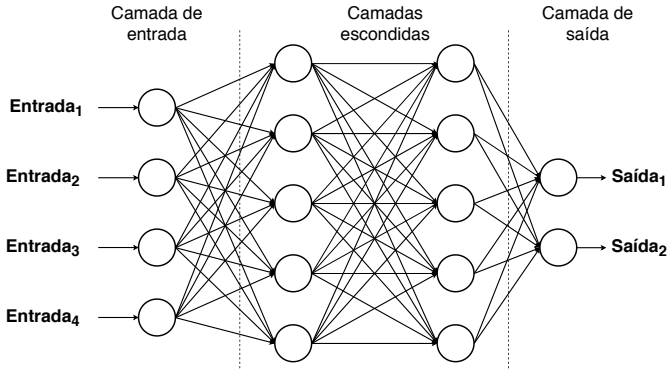


Figura 4.1: Representação de uma rede neural *feedforward*.

Para calcular a soma ponderada, um conjunto de entradas $[x_1, x_2, ..., x_n]$ é associado a um conjunto de pesos $[w_1, w_2, ..., w_n]$, inicializados aleatoriamente. Adotando uma notação convencional, as entradas e os pesos são representados por um *vetor*, um tipo de lista numérica. Assim, dado um vetor de entradas x, um vetor de pesos w e um viés b, computa-se a soma ponderada através do produto escalar[1] $(\cdot)$ entre x e w, como expresso na Equação 4.1.

$$z = x \cdot w + b \tag{4.1}$$

Sobre z, é aplicada uma função de ativação f, sendo essa não-linear, que determina um limiar de ativação em um certo intervalo, produzindo a saída do nó, como mostrado na Equação 4.2.

$$a = f(z) = f(x \cdot w + b) \tag{4.2}$$

Até este ponto, foram discutidas as computações com relação a um nó da camada. Na prática, o cálculo é feito paralelamente na camada inteira. Para isso, os vetores de pesos w de uma camada m são concatenados em uma matriz $W^{(m)}$ e os seus vieses são concatenados em um vetor $b^{(m)}$. Sendo assim, cada elemento $W_{ij}^{(m)}$ da matriz de pesos representa a conexão do nó i da camada $m-1$ para nó j da camada m. Na Figura 4.1, as matrizes de pesos W são ilustradas pelos links entre uma camada e outra. Portanto, a

[1] O produto escalar é obtido pela multiplicação entre dois vetores e a soma dos produtos resultantes.

saída de uma camada escondida, chamada de h, é dada pela Equação 4.3, aplicando uma operação de multiplicação entre uma matriz W e um vetor x.

$$h = f(Wx + b) \tag{4.3}$$

Na ultima etapa, a camada de saída recebe como entrada as saídas da ultima camada escondida $(h - 1)$ e calcula-se o produto escalar. Em determinados modelos, o viés não é incluído na camada final (JURAFSKY; MARTIN, 2018, p. 131). Em seguida, aplica-se uma função linear para converter os produtos em uma distribuição de probabilidades. Em tarefas com multi-classes, isto é, com mais de duas categorias, essa função é comumente uma *softmax* (Equação 4.4). Ela recebe como entrada um vetor numérico e o normaliza em uma distribuição de probabilidades de [0, 1].

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \tag{4.4}$$

Na Equação 4.4, $\vec{z}$ é um vetor $[z_1, ..., z_K]$ e $z_i \in \vec{z}$, em que K é o número de categorias na tarefa. Assim, cada categoria em $\vec{z}$ é associada a uma probabilidade de predição obtida pelo modelo. A soma de todas as probabilidades emitidas na camada de saída soma 1. Para se obter a predição final, escolhe-se a categoria com maior probabilidade. Esse passo-a-passo é chamado de passagem para frente (tradução de *forward pass*).

4.1.1 Treinamento de uma RN

O treinamento de uma rede neural é supervisionado, ou seja, é necessário que sejam fornecidos os valores alvos para serem comparados às predições. Esse treinamento é feito em reverso, de trás para frente, em que os erros do modelo são propagados e atualizados nos pesos e vieses associados aos nós. O erro é obtido por uma função de custo L, que calcula a diferença entre os valores alvos (Y) e os preditos pelo modelo $(\hat{Y})$. O objetivo do treinamento é alcançar o erro mínimo, encontrando os pesos que refletem a menor diferença entre Y e $\hat{Y}$. A escolha da função L vai depender do tipo de tarefa que se está resolvendo, binária ou multi-classes. Após aplicar a

função de custo, é preciso otimizar os pesos e vieses, definindo o ajuste de erro que será corrigido para atualizar os pesos. Para isso, usa-se um algoritmo chamado *error backpropagation* (RUMELHART; HINTON; WILLIAMS, 1985), que não cabe discutir aqui, dada sua complexidade[2].

Na ultima etapa, os pesos e os vieses são atualizados simultaneamente em valores que tentam diminuir o erro do modelo. Em uma rede neural, os pesos e vieses aprendidos durante o treinamento são chamados de *parâmetros* do modelo. Existem, além disso, os *hiperparâmetros*, que são parâmetros pré-estabelecidos para a configuração da rede. Na atualização dos pesos, um hiperparâmetro importante é inserido, chamado taxa de aprendizado (α), que influencia no quanto os parâmetros devem ser ajustados.

O processo de treinamento descrito é repetido iterativamente, recalculando e atualizando os parâmetros a cada iteração, até que o erro alcance um valor mínimo, o que significa que a rede convergiu. Na prática, esse valor pode nunca ser alcançado, podendo ficar oscilando entre um ponto e outro, então, para evitar um *loop* infinito, é estabelecido um número máximo de iterações, chamado de épocas, outro hiperparâmetro a ser pré-definido.

4.1.2 Hiperparâmetros em redes neurais

A modelização de uma RN envolve a seleção de diversos hiperparâmetros, como o número de épocas de treinamento e a taxa de aprendizado, já discutidos na Seção 4.1.1. A escolha dos hiperparâmetros tem grande influência para o desempenho do modelo, portanto é preciso refiná-los durante a etapa de desenvolvimento da rede neural, isto é, testá-los empiricamente.

A arquitetura da rede neural, como o número de camadas escondidas e de nós por camada, é um hiperparâmetro a ser testado. A função de ativação também pode variar, o que significa alterações no limiar de ativação de um nó. Entre as funções comumente usadas estão a sigmoide, a *Rectified Linear Unit*

[2]Para mais detalhes da implementação do algoritmo, veja-se o Capítulo 18 de *Artificial Intelligence: A Modern Approach* (RUSSELL; NORVIG, 2016).

(ReLU) e a tanh. A função sigmoide varia entre $[0, 1]$, a tanh entre $[-1, 1]$ e a ReLU é dada por $max(x, 0)$, sendo x o valor do nó.[3].

Além disso, existem os métodos de otimização que podem ser aplicados durante o treinamento, como Adam (KINGMA; BA, 2014), que utiliza uma taxa de aprendizado adaptativa. Já as técnicas de regularização ajudam a evitar *overfitting*, isto é, quando um modelo se ajusta muito bem aos dados de treinamento, mas perde a capacidade de generalização. Uma dessas técnicas consiste em atribuir uma taxa para eliminar nós aleatoriamente durante o treinamento, chamada de *dropout* (HINTON et al., 2012).

Os hiperparâmetros supracitados são os mais comuns em redes neurais em geral, mas pode haver outros. Desse modo, a etapa de desenvolvimento de uma RN requer diversos testes de combinação entre hiperparâmetros, antes da etapa de treinamento final para avaliação. É difícil saber se a combinação escolhida é a melhor opção, uma vez que a alteração de um hiperparâmetro pode afetar outros que já haviam sido refinados. Uma forma de evitar testar todas essas combinações é adotar hiperparâmetros já testados na literatura em redes neurais semelhantes na mesma tarefa.

4.2 Redes Neurais Recorrentes

Uma rede neural recorrente (RNR) é qualquer rede neural que contém um ciclo em sua rede de conexões. Assim, qualquer rede em que o valor de um nó é diretamente ou indiretamente dependente de saídas precedentes como uma entrada. A Figura 4.2 traz uma representação de uma rede recorrente.

Semelhante a uma rede neural *feedforward*, a RNR recebe como entrada um vetor x, que é multiplicado por uma matriz de pesos W, seguido de uma função de ativação para computar h na camada escondida. Mas, em redes neurais recorrentes, os elementos em uma sequência são processados um por vez, introduzindo a noção de tempo (t). Assim, a diferença da RNR está no nó de recorrência (linha azul tracejada na Figura 4.2), que adiciona como entrada na computação da camada escondida no passo t o valor da camada escondida

[3]Para uma discussão mais profunda a respeito de cada função de ativação, ver Jurafsky e Martin (2018, p. 124).

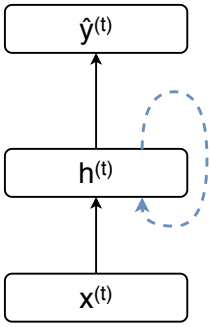


Figura 4.2: Representação simplificada de uma rede neural recorrente. A linha tracejada azul representa uma conexão de recorrência na camada escondida.

do passo anterior $(t-1)$. Desse modo, dado um vetor $[x^{(1)}, x^{(2)}, ..., x^{(T)}]$, a cada passo t, os nós com recorrência recebem uma entrada $x^{(t)}$ e o valor do camada escondida precedente $h^{(t-1)}$. Isso significa que o estado escondido produzido no passo $t-1$ influencia a saída $\hat{y}^{(t)}$, pois o valor da camada escondida em t é afetado pelos estados precedentes, semelhante ao que ocorre em um modelo de linguagem (cf. Capítulo 2).

Por sua capacidade de processamento temporal, as RNRs são muito utilizadas na classificação de dados sequenciais, como processamento de texto e fala e sequenciamento de DNA. Um dos problemas enfrentados por elas é o chamado *vanishing gradients*, caracterizado pela perda de informações relevantes que estão muitos passos atrás no tempo t. Isso faz com que as RNRs tenham dificuldades de lidar com análises de dependência de longa distância.

4.2.1 Long short-term memory

Long short-term memory (LSTM) é um tipo de rede neural recorrente que foi introduzida por Hochreiter e Schmidhuber (1997) para tentar lidar com o problema de *vanishing gradients* das RNRs. A ideia central em LSTMs é esquecer informações consideradas irrelevantes, mantendo somente aquelas que podem ajudar na predição final. Devido a sua capacidade de filtrar informações contextuais relevantes, as LSTMs foram por muito tempo um dos modelos mais adotados em tarefas de PLN.

A LSTM possui uma estrutura em cadeia, assim como a RNR, que contém blocos chamados de células de memória. O componente principal da célula de memória é um nó, chamado de *estado da célula*, que armazena as informações. Por sua vez, as informações são reguladas por unidades neurais, chamadas de *portão*. Os portões são compostos de uma camada *feedforward*, seguido de uma função de ativação sigmoide. Eles são multiplicados na camada a ser regulada para controlar o fluxo de informações nela. Assim sendo, se o valor de um nó é próximo de 1, a informação é adicionada à célula, enquanto valores próximos de zero são removidos.

Na arquitetura proposta por Hochreiter e Schmidhuber (1997) existem dois portões, o portão de entrada e o portão de saída. O primeiro adiciona as informações úteis no estado da célula e o segundo seleciona quais informações da célula serão enviadas como saída. A Figura 4.3 mostra uma representação de uma célula em uma LSTM.

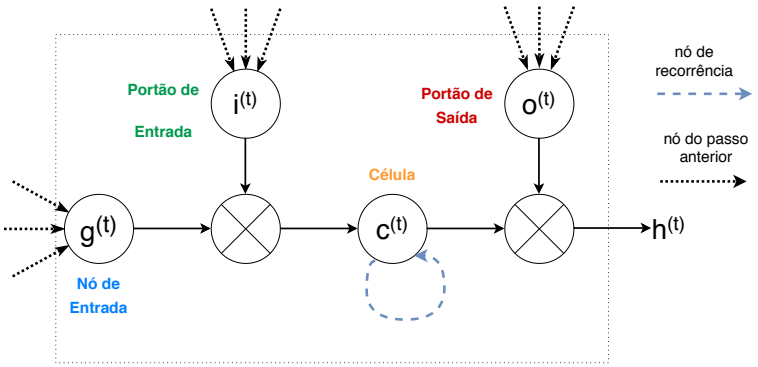


Figura 4.3: Representação simplificada de uma célula de memória na LSTM de Hochreiter e Schmidhuber (1997).

A célula de memória recebe uma entrada $x^{(t)}$ e o estado escondido do passo anterior $h^{(t-1)}$ e computa o valor do nó em $g^{(t)}$. Em seguida, o valor do portão de entrada $i^{(t)}$ é multiplicado a $g^{(t)}$ para selecionar as informações, que são atualizadas no estado da célula $c^{(t)}$. O último passo é multiplicar o estado interno $c^{(t)}$ pelo valor do portão de saída $o^{(t)}$ para obter o estado da camada escondida $h^{(t)}$.

4.2.2 LSTM bidirecional

Na rede neural LSTM apresentada, os estados escondidos $h^{(t)}$ representam as informações contextuais fornecidas até o momento precedente a t, portanto o contexto a esquerda da sequência é usado para calcular a predição. A esse modelo chamamos de *forward*. Contudo, o contexto a direita da palavra também pode conter informações relevantes para a classificação da palavra. Esse contexto pode ser capturado através de um modelo *backward*, que percorre a sequência da direita para a esquerda.

Ao combinar um modelo *forward* e um *backward*, obtém-se uma rede neural bidirecional, capaz de combinar as saídas das duas redes em uma única representação que captura informações à esquerda e à direita. A Figura 4.4 representa uma rede neural LSTM bidirecional (BiLSTM).

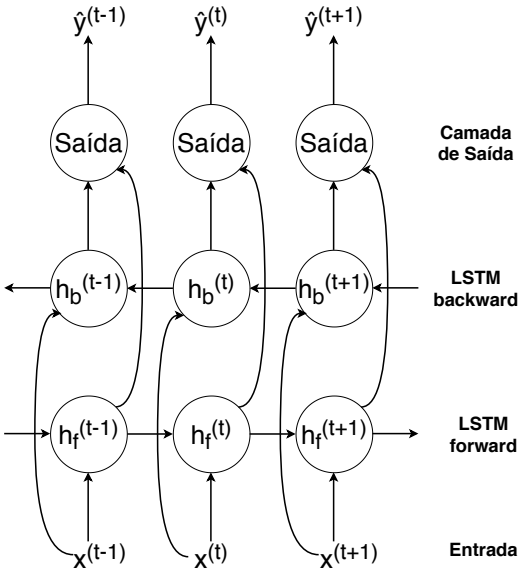


Figura 4.4: Representação de uma rede neural BiLSTM.

A cada passo t, os estados escondidos da LSTM *forward* e *backward* são combinados através de uma operação como concatenação, multiplicação ou soma. Essas saídas são usadas como entrada para uma camada de pós-processamento, por exemplo, uma rede neural *feedforward* com ativação *softmax*, para obter as predições do modelo.

4.3 Transformer

Transformer (VASWANI et al., 2017) é uma rede neural com arquitetura *encoder-decoder* baseada em mecanismos de atenção. A rede recebe como entrada uma sequência de palavras, codifica-as em representações nas camadas de atenção e as decodifica em palavras novamente. À medida que o modelo processa cada palavra (cada posição na sequência de entrada), um mecanismo de atenção computa a importância de palavras em outras posições para codificar a palavra atual. Inúmeros modelos surgiram a partir da rede Transformer, entre eles é possível citar BERT (DEVLIN et al., 2018) e GTP (RADFORD et al., 2018). Nas subseções seguintes, será apresentada brevemente a arquitetura do Transformer, com enfoque no mecanismo de atenção[4].

4.3.1 Arquitetura do Transformer

A rede Transformer é dividida em dois módulos: *encoder* e *decoder*, traduzido como codificador e decodificador. Tanto o primeiro quanto o segundo módulo são compostos de uma pilha de seis camadas idênticas. No módulo *encoder*, cada camada é dividida em duas subcamadas, sendo a primeira um mecanismo chamado *"Multi-Head Self-Attention"* e a segunda uma rede neural *feedforward*. Já nas camadas *decoder*, além das duas subcamadas citadas, há uma terceira subcamada de atenção que processa os dados vindos da camada *encoder*.

Por sua arquitetura *encoder-decoder*, o Transormer foi inicialmente proposto para a tarefa de tradução automática. O módulo *encoder* recebe como entrada uma sequência de palavras em determinada língua, por exemplo português, codifica as palavras em representações contextuais e as transfere para o módulo *decoder*. Nele, as representações são processadas passo-a-passo e suas saídas são passadas por uma camada de saída em que as predições do modelo são emitidas, ou seja, é gerada a tradução da sentença para uma língua alvo.

[4]Para mais detalhes a respeito da arquitetura interna do Transformer, ver o artigo "The Illustrated Transformer" de Jay Alammar, disponível em http://jalammar.github.io/illustrated-transformer/

4.3.2 Codificação de posição sequencial

Diferente das RNRs, o Transformer não lida com a ordenação das palavras em seu processamento. Para superar esse problema, utiliza-se um vetor de posição, que contém informações sobre a posição da palavra na sequência. O vetor de posição tem o mesmo tamanho do *word embedding* das palavras, para que os valores sejam somados.

4.3.3 Mecanismo de Atenção

O mecanismo de atenção calcula a pontuação de uma palavra em relação às demais palavras em uma sequência, ou seja, determina o quanto o modelo deve focar em cada posição da sequência ao codificar tal palavra. Considere a sentença "O pai não viu a filha, porque ela se escondeu". Ao codificar "ela", é importante identificar que esse pronome está associado com "a filha", portanto concorda com o sintagma em número e gênero. Esse tipo de informação contextual é relevante na tradução de uma língua para outra.

A função de atenção do Transformer pode ser vista como um mapeamento do peso de cada palavra $[x_1, ..., x_n]$ em uma sequência para a representação da palavra x_i. Na camada de atenção, cada palavra é convertida em três vetores, chamados *key* (k), *value* (v) e *query* (q). Esses vetores são apenas abstrações associadas a cada palavra, utilizados no cálculo de atenção. Para obtê-los, multiplica-se o vetor de uma palavra e três matrizes W^Q, W^V e W^K, iniciadas com valores aleatórios que são ajustados durante o treinamento.

Na prática, em vez de utilizar os vetores, usa-se as matrizes Q, K e V, que concatenam os vetores de todas as palavras em uma sequência. A função de atenção é dada na Equação 4.5, em que d_k é a dimensão de k.

$$Atenção(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{4.5}$$

O primeiro passo é calcular a pontuação de cada palavra em uma sequência em relação a determinada palavra. Para isso, multiplicam-se as matrizes Q e K e dividi-se o produto pela raiz quadrada de d_k. Em seguida, aplica-se a função *softmax* para normalizar os valores entre 0 e 1. Por fim, multiplica-se

V pelos pesos obtidos na *softmax* para determinar a importância de cada posição.

4.3.4 Múltiplos núcleos de atenção

Em vez de um único núcleo de atenção, a camada de atenção do Transformer, chamada *MultiHead Self-Attention*, é composta de oito núcleos de atenção. Assim, são projetadas paralelamente oito matrizes W^Q, W^K e W^V distintas, inicializadas aleatoriamente, então calcula-se a atenção de cada núcleo, como se viu anteriormente. Os vetores gerados são concatenados e multiplicados por uma matriz W^0 para obter o resultado final da camada. A ideia de utilizar inúmeros núcleos de atenção é que cada um dos núcleos esteja focado em diferentes palavras da sentença, o que melhora a codificação da palavra.

4.4 BERT

Bidirectional Encoder Representation from Transformers (DEVLIN et al., 2018), abreviado como BERT, é uma rede neural que gera representações bidirecionais de palavras, ou seja, baseado no contexto à esquerda e à direita. O BERT é composto de múltiplas camadas de codificadores Transformer (VASWANI et al., 2017). Existem duas arquiteturas do modelo, distintas pelo número de camadas Transformer, número de núcleos de atenção e tamanho da camada escondida. O BERT$_{BASE}$ é composto de 12 camadas, 12 núcleos de atenção e tamanho 768 na camada escondida. Já o BERT$_{LARGE}$ conta com 24 camadas, 16 núcleos de atenção e tamanho 1024 na camada escondida.

4.4.1 Representação da Entrada

O BERT pode receber como entrada uma sentença[5] ou um par de sentenças concatenados em uma sequência. As palavras da entrada são tokenizadas em WordPieces (WU et al., 2016) e convertidas em *word embeddings*. A tokenização WordPiece recebe esse nome porque quebra as palavras ausentes

[5] Aqui, considera-se "sentença" uma sequência de palavras, sem associação com a definição linguística.

no vocabulário em tokens, inserindo o símbolo ## nas partes não iniciais, por exemplo “meni” e “##na”.

Além disso, no início da sequência é inserido um token especial [CLS], usado na etapa de classificação para concatenar as representações individuais de palavras em uma sentença. Para lidar com entradas compostas por um par de sentenças, outro token especial ([SEP]) delimita a fronteira das sentenças. Em adição a isso, treina-se um vetor de segmentos, indicando se a palavra pertence à sentença A ou à B. Por fim, assim como o Transformer, o BERT não captura a ordem de palavras, portanto necessita de um vetor de posições, nesse caso computando a posição absoluta da palavra na sentença. A entrada final do BERT é dada pela somatória dos vetores de *word embedding*, de posição e de segmento, como mostrado na Figura 4.5.

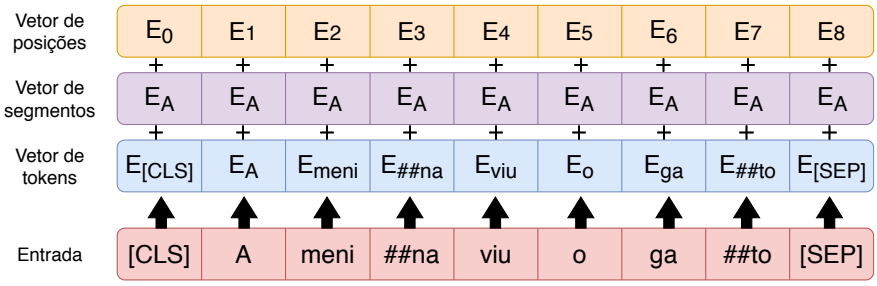


Figura 4.5: Representação da entrada do BERT.

4.4.2 Treinamento do BERT

São necessárias duas etapas de treinamento no BERT: o pré-treinamento e o refinamento (do inglês, *fine-tuning*). O pré-treinamento é feito em corpora não-anotados em duas tarefas distintas: *Masked Language Model* (MLM) e *Next Sentence Prediction* (NSP). Por sua vez, o refinamento é treinado em corpora anotados para uma tarefa específica, como REN, tradução automática, etc. A Figura 4.6 traz uma representação das etapas de pré-treinamento e de refinamento do modelo.

Um dos diferenciais do BERT está em sua etapa de pré-treinamento. Nela, o modelo é otimizado ao mesmo tempo em duas tarefas, uma delas um modelo de linguagem (ML) bidirecional. Em MLs bidirecionais anteriores,

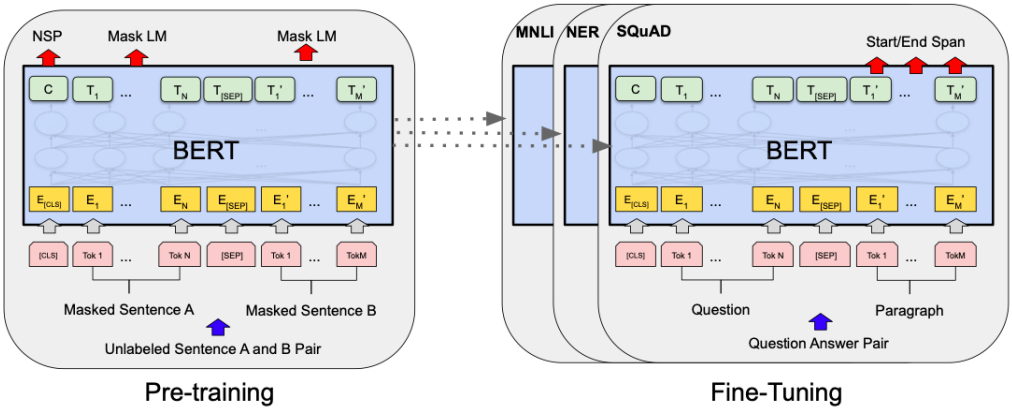


Figura 4.6: Representação da etapa de treinamento do BERT extraído de Devlin et al. (2018). À esquerda, a etapa de pré-treinamento e à direita, o refinamento.

eram necessários dois modelos, um *forward* e outro *backward*, para lidar com o contexto à esquerda e à direita em uma sequência. Diferentemente, o BERT faz isso utilizando somente um modelo de linguagem, chamado de *Masked Language Model* (traduzindo, modelo de linguagem mascarado) com capacidade de observar todas as palavras em uma sequência para predizer qual está faltando. As subseções seguintes fornecem uma descrição das duas tarefas envolvidas no pré-treinamento do BERT.

Masked Language Model

Devlin et al. (2018) utilizam um modelo de linguagem com máscaras, em que a tarefa é mascarar alguns tokens da entrada, substituindo-os pelo token especial [MASK], e predizer quais são eles com base nos demais tokens da sentença. Em seus experimentos, os autores mascaram aleatoriamente 15% dos tokens da entrada. Se o token da posição i é escolhido, ele é substituído em 80% dos casos por [MASK], em 10% por um token aleatório e nos outros 10% ele permanece o mesmo. Essas condições foram estabelecidas como forma de diminuir a disparidade entre o pré-treinamento e o refinamento, já que o token [MASK] só aparece na primeira etapa.

Next Sentence Prediction

A tarefa de predição da próxima sentença é voltada para a relação entre sentenças, calculando a probabilidade de uma sentença B estar imediatamente à direita de A. Para isso, ao selecionar um par de sentenças para treinamento, a chance de B seguir A é de 50%, recebendo a etiqueta "IsNext", enquanto nos outros 50%, B é selecionado aleatoriamente no corpus, sendo etiquetado como "NotNext".

4.4.3 Aplicações do BERT

Como já mencionado, o BERT pré-treinado pode ser refinado em uma tarefa específica. Nessa abordagem, treina-se o modelo em um corpus anotado, otimizando os pesos aprendidos. As predições são obtidas por uma camada de saída para classificação na tarefa em questão, que recebem como entrada as saídas do BERT. O refinamento é bem mais barato em termos computacionais do que o pré-treinamento, já que os pesos do BERT já passaram por treinamento, necessitando somente de ajustes para se adaptar na tarefa.

Além da opção acima, é possível utilizar o BERT como *word embeddings* em outros modelos, extraindo as representações de palavras geradas na etapa de pré-treinamento. Essa abordagem é conhecida como *feature-based*, uma técnica de transferência de aprendizado. Em vez de refinar o BERT em uma tarefa específica, retreinando os seus pesos, os traços do modelo pré-treinado são extraídos e podem alimentar outro modelo de classificação, mantendo os pesos congelados, sem refinamento.

Devlin et al. (2018) apresentam os resultados do BERT na abordagem *feature-based* para o REN, extraindo os traços gerados no pré-treinamento e passando por uma rede BiLSTM. Os autores fazem os experimentos com o BERT_{BASE}, extraindo traços de diferentes camadas, como, por exemplo, somente a última ou as quatro finais, entre outras. Os resultados reportados por eles se comparam àqueles obtidos pelo modelo refinado, mostrando que as representações geradas pelo BERT são acuradas mesmo sem refinamento.

5 | Metodologia

Este capítulo compreende uma descrição dos modelos de REN propostos e das ferramentas utilizadas para a implementação dos algoritmos, além de quaisquer recursos dos quais nos valemos para experimentação e avaliação de resultados.

5.1 Corpus

O corpus usado nesta pesquisa foi o Harem I, dividido em Primeiro Harem (PH) e Mini Harem (MH). O primeiro deles serviu para o treinamento dos modelos e, o segundo, para a avaliação. Essa é a mesma metodologia adotada em outros trabalhos de REN do português, como em Souza, Nogueira e Lotufo (2020), Santos, Consoli et al. (2019) e Santos e Guimaraes (2015), o que torna possível a comparação entre os resultados obtidos nessas pesquisas.

5.1.1 Anotações do Harem

No Capítulo 3, dissemos que o Harem classifica as Entidades Nomeadas em categorias e tipos. No total são 10 categorias: Pessoa, Local, Organização, Tempo, Valor, Obra, Acontecimento, Abstração, Coisa e Outro, sendo que cada uma apresenta um determinado número de tipos. Classificações refinadas como essa geralmente são mais difíceis para os sistemas de REN, além de serem bem incomuns na literatura. Seguindo abordagens anteriores, foram removidas todas as etiquetas de tipos durante o pré-processamento. Além disso, as categorias do Harem foram convertidas em formato BIO (cf. Capítulo 2),

como listadas na Tabela 5.1, totalizando 21 etiquetas, duas por categoria mais a de não-entidades.

Categoria	**etiquetas**
Pessoa	B-PES e I-PES
Local	B-LOC e I-LOC
Organização	B-ORG e I-ORG
Tempo	B-TEM e I-TEM
Valor	B-VAL e I-VAL
Acontecimento	B-ACO e I-ACO
Coisa	B-COI e I-COI
Obra	B-OBR e I-OBR
Abstração	B-ABS e I-ABS
Outro	B-OUT e I-OUT
Não-entidade	O

Tabela 5.1: As categorias do Harem convertidas em formato BIO.

As CDs do Harem apresentam a possibilidade de mais de uma classificação em contextos em que uma entidade pode pertencer a mais de uma categoria. No corpus, as etiquetas concorrentes são concatenadas em uma *string* separadas por "|". Na sentença (I), por exemplo, "Bombeiros" pode ser classificado como Pessoa ou Organização no Harem, pois pode se referir a um grupo de pessoas ou uma instituição organizacional. Nesses casos, selecionou-se somente a primeira etiqueta que aparecia na sequência.

(I) Caros amigos dos **Bombeiros**. → (PESSOA|ORGANIZAÇÃO)

Quando há mais de uma interpretação sobre as fronteiras de identificação de uma entidade, o Harem é marcado com as etiquetas <ALT> e </ALT>, indicando que as entidades entre elas são alternativas de anotação. Nesses casos, a categoria da entidade pode mudar. Para exemplificar, na sentença (II), o Harem oferece duas possibilidades de identificação de EN.

(II) Lutou contra a **Ditadura de João Franco**|**Ditadura** e **João Franco**.

Na primeira acepção, "Ditadura de João Franco" é considerada uma única entidade, classificada como Abstração, enquanto na segunda "Ditadura"

e "João Franco" são consideradas entidades distintas, classificadas como Abstração e Pessoa, nessa ordem. Nos casos de alternância, somente a primeira alternativa de entidade anotada na sequência foi considerada, no exemplo acima seria "Ditadura de João Franco".

5.1.2 Pré-processamento do corpus

Inicialmente, o texto é tomado como uma sequência de caracteres, chamada de *string*, não havendo qualquer distinção de fronteira entre palavras. A tokenização é uma técnica de segmentação que quebra uma sequência de caracteres estabelecendo o limite de cada palavra, chamada tecnicamente de *token*, já que nem sempre corresponde à definição linguística de palavra. Na tokenização adotada, considerou-se como um token uma sequência alfabética (como "amigo"), alfanumérica ("$C0_2$", por exemplo) ou não-alfanumérica ("."; "%", etc.). Após o pré-processamento, obtiveram-se 95679 tokens no PH e 64702 no MH e um total de 4982 entidades para o primeiro e 3624 no segundo.

Os corpora foram salvos em planilhas no formato CSV. Como ilustração, um trecho do Primeiro Harem é apresentado na Tabela 5.2.

documento	**token**	**etiqueta**
Doc14	O	O
Doc14	Uacari	B-ABS
Doc14	-	I-ABS
Doc14	Branco	I-ABS
Doc14	(	O
Doc14	nome	O
Doc14	científico	O
Doc14	:	O
Doc14	Cacajau	B-ABS
Doc14	calvus	O
Doc14	)	O
Doc14	é	O
Doc14	um	O
Doc14	macaco	O

Tabela 5.2: Um excerto do Primeiro Harem após o pré-processamento.

5.2 Modelos implementados

Nesta pesquisa, foram testados dois modelos para o REN: o BERT na abordagem *fine-tuning* e uma BiLSTM com *word embeddings* extraídos do BERT combinados a traços manuais. Em ambos os modelos, utilizou-se a arquitetura BERT_{BASE}, com os modelos pré-treinados BERT-Multilíngue (DEVLIN et al., 2018) e BERT-Português (SOUZA; NOGUEIRA; LOTUFO, 2020), disponíveis na biblioteca Transformers, desenvolvida pelo Hugging Face[1] para a linguagem Python[2].

O BERT-Multilíngue (BERT-ML) foi pré-treinado em artigos da Wikipédia disponíveis em 104 línguas distintas, entre elas, o português. Como o número de documentos em cada língua variou muito, os autores utilizaram uma técnica de suavização durante o pré-treinamento para balancear os pesos. O BERT-ML conta com um vocabulário compartilhado de 110 mil WordPieces. Por sua vez, no pré-treinamento do BERT-Português (BERT-PT) utilizou-se o brWac (WAGNER FILHO et al., 2018), um corpus de português brasileiro contendo 2.68 bilhões de tokens extraídos de documentos da internet. O seu vocabulário WordPiece contém 230 mil unidades.

Com a popularização do BERT, várias bibliotecas do Python forneceram implementações prontas para o treinamento desses modelos em tarefas específicas. Na abordagem *fine-tuning*, adotamos a implementação do BERT para classificação de palavras da Hugging Face na versão baseada em Pytorch[3]. Essa arquitetura é composta de um modelo pré-treinado BERT seguido de uma camada linear com ativação *softmax*. Para cada token, a predição final é dada pela etiqueta cujo modelo atribui maior probabilidade, chamada de função *argmax*. O BERT foi testado com os modelos BERT-ML e BERT-PT. A Figura 5.1 traz uma ilustração do modelo para a tarefa de REN.

Nossa ideia inicial era inserir traços linguísticos diretamente na arquitetura do BERT, treinando conjuntamente as representações *word embeddings* contextuais e os traços linguísticos na tarefa de REN, na intenção de melhorar o

[1] https://huggingface.co/transformers/
[2] https://www.python.org/
[3] https://pytorch.org/

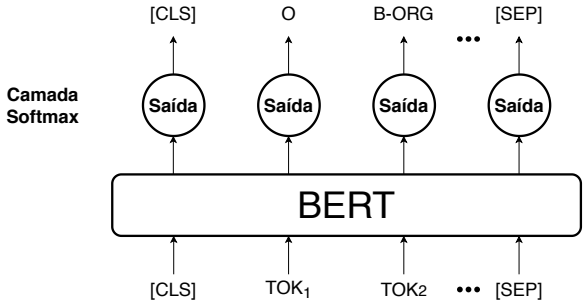


Figura 5.1: Representação do BERT para o REN.

desempenho do modelo. Contudo, isso não foi possível, pois a implementação do BERT para classificação de palavras disponibilizada pela biblioteca Transformer não permite modificar a estrutura interna da rede, isto é, inserir outros tipos de traços para serem treinados pelo BERT. Para isso, seria necessário reimplementar partes consideráveis do código manualmente, o que significaria um trabalho que vai além do escopo desta pesquisa.

Desse modo, optamos por utilizar outra arquitetura de rede neural, uma BiLSTM, implementada a partir da biblioteca Keras[4]. A rede BiLSTM aceita como entrada outros tipos de traços para a representação de palavras, permitindo a combinação entre *word embeddings* e traços manuais. Além disso, essa rede neural mantém determinadas semelhanças com o BERT, como o processamento contextual de uma sequência de texto. Portanto, testou-se como segundo modelo uma BiLSTM combinando *word embeddings* contextuais do BERT e traços linguísticos.

A abordagem *feature-based* foi utilizada para extrair os *word embeddings* pré-treinados do BERT-PT. Um dos pontos negativos nesse método é que os traços extraídos do BERT não podem ser treinados, o que significa que os *word embeddings* não foram refinados ao REN. Para melhorar a representação de palavras, os traços do BERT foram concatenados com traços ortográficos, lexicais e morfossintáticos. A combinação de *word embeddings* e traços manuais já provou trazer benefícios no desempenho do REN em outras pesquisas, a citar Chiu e Nichols (2016), Ghaddar e Langlais (2018) e Song et al. (2020). Na Figura 5.2, é dada a arquitetura da BiLSTM proposta.

[4] https://keras.io/

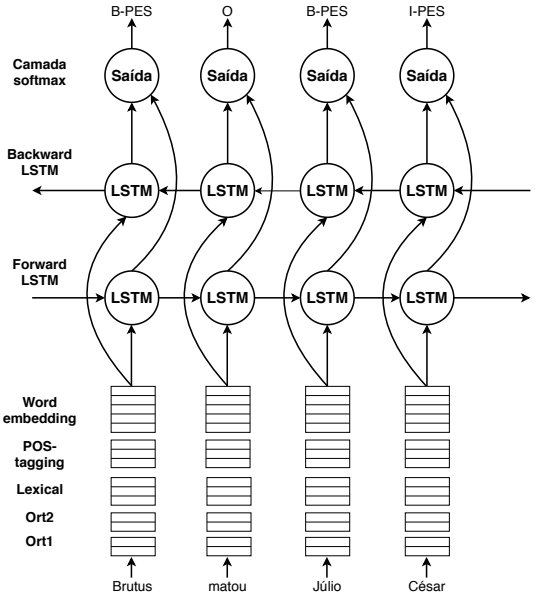


Figura 5.2: Rede neural BiLSTM implementada para o REN.

Para cada palavra na sequência, são extraídos cinco vetores: a representação *word embedding* extraída do BERT-PT, dois traços ortográficos, um morfossintático e um lexical. Esses traços são utilizados como entrada da BiLSTM para extrair suas informações. A cada passo, a saída da BiLSTM é decodificada por uma camada linear com ativação *softmax* para convertê-la em uma distribuição de probabilidade de cada categoria. Na última etapa, aplica-se uma função *argmax* para obter a predição do modelo.

Nas subseções seguintes, discutiremos em detalhes a implementação de cada um dos traços da BiLSTM.

5.2.1 Word embeddings

Os *word embeddings*, já discutidos no Capítulo 2, são muito eficientes em gerar representações baseadas na distribuição de palavras. Por serem *word embeddings* contextuais, optamos usar como um dos traços da palavra a representação gerada pelo BERT-PT pré-treinado.

Na etapa de desenvolvimento da BiLSTM, testou-se a extração de *word embeddings* do BERT-ML e do BERT-PT. Além disso, avaliamos a extração de

traços da última camada do BERT, das duas últimas e das quatro, combinando-as através da operação de soma ou concatenação. Os experimentos foram rodados com uma BiLSTM somente com os traços *word embeddings* BERT para representação das palavras.

Os parâmetros da rede neural foram os mesmos descritos na Seção 5.4, com exceção das épocas treinamento e do número de camadas do BERT, que era o parâmetro a ser refinado. A BiLSTM foi treinada por 5 épocas e validada por 5 rodadas distintas. Em cada rodada, o corpus de treinamento, o Primeiro Harem, foi embaralhado para evitar viés e dividido entre treino (90%) e validação (10%). O corpus de validação serve para avaliar o desempenho e erro do modelo em exemplos não-vistos durante o treinamento. O corpus de avaliação foi o Mini Harem.

A Tabela 5.3 traz a média da medida-F obtida nas 5 rodadas para o BERT-ML e o BERT-PT.

Camadas Extraídas	**Medida-F**	
	BERT-PT	BERT-ML
última	65,57	61,93
soma-2	69,65	66,57
soma-4	69,64	66,81
concatena-2	69,92	65,56
concatena-4	**70,80**	65,89

Tabela 5.3: Resultados dos experimentos com os *word embeddings* do BERT.

Os resultados da Tabela 5.3 mostram que o BERT-PT fornece um desempenho melhor na tarefa em todas as condições testadas. Com relação ao número de camadas, a BiLSTM obteve o melhor resultado pela concatenação das quatro camadas finais do BERT-PT, o que condiz com os resultados reportados por Devlin et al. (2018). Esse resultado diverge entre o BERT-PT e o BERT-ML, já que o segundo apresenta melhor resultado com a operação de soma. De qualquer modo, mesmo os melhores resultados do BERT-ML não superaram aqueles obtidos pelo BERT-PT.

A partir dos resultados, adotamos como modelo de *word embeddings* o BERT-PT concatenando as 4 camadas finais, o que totaliza uma dimensão

vetorial de *word embeddings* de 3072, considerando que a saída de cada camada BERT tem dimensão 768. Durante o treinamento da BiLSTM para REN, os pesos da camada *word embedding* BERT ficaram congelados, isto é, não foram refinados para a tarefa de REN.

5.2.2 POS-tagging

Part-of-speech tagging (abreviação, POS-tagging) é a tarefa de atribuir uma etiqueta morfossintática, por exemplo verbo e pronome, para cada palavra em uma sequência. A utilização de POS-tagging em modelos supervisionados para o REN é muito comum na literatura. Nosso intuito em utilizar esse traço foi que a rede neural LSTM pudesse extrair informações relevantes sobre a distribuição morfossintática das ENs, considerando a capacidade de processamento temporal desse modelo.

Cada token em uma sequência de texto foi classificado com a etiqueta morfossintática correspondente por meio do etiquetador morfossintático da biblioteca nlpnet[5] (FONSECA; ROSA, 2013). Esse etiquetador é uma rede neural similar à de Collobert et al. (2011), treinado no corpus Mac-morpho (ALUISIO et al., 2003). Os autores reportam sua acurácia em 96,48%. O nlpnet distingue 30 etiquetas morfossintáticas[6], listadas na Tabela 5.4.

Para representar o traço POS-tagging, as etiquetas na Tabela 5.4 foram convertidas em vetores numéricos através de uma camada *embedding* do Keras. Primeiro, cada uma das etiquetas foi convertida em um índice de 1–30, por exemplo, V correspondendo a 3, N a 5, etc. Na camada *embedding* esses índices são associados a vetores, inicializados com valores aleatórios de acordo com uma distribuição uniforme [-0,5, 0,5]. Esses vetores têm dimensão d, um hiperparâmetro pré-estabelecido. Os testes realizados para ajustar esse parâmetro serão discutidos na Seção 5.4, sobre os hiperparâmetros das RNs.

Os valores da camada *embedding* são ajustados durante o treinamento da BiLSTM. Desse modo, cada categoria morfossintática é associada a uma representação *embedding*, cujos pesos serão treinados com relação à tarefa de

[5] http://nilc.icmc.usp.br/nlpnet/

[6] Uma descrição detalhada a respeito da anotação pode ser encontrada em http://www.nilc.icmc.usp.br/macmorpho/macmorpho-manual.pdf

ADJ	ADV
ADV-KS	ADV-KS-REL
ART	CUR
IN	KC
KS	N
NPROP	NUM
PCP	PDEN
PREP	PREP+ADV
PREP+ART	PREP+PROADJ
PREP+PRO-KS	PREP+PRO-KS-REL
PREP+PROPESS	PREP+PROSUB
PROADJ	PRO-KS
PRO-KS-REL	PROPESS
PROSUB	PU
V	VAUX

Tabela 5.4: Lista de etiquetas categorizadas pela nlpnet.

REN. Explicando de outro modo, a rede neural irá modelizar a associação entre a ocorrência de determinada categoria morfossintática e as categorias do Harem, por exemplo, a ocorrência de "V" (isto é, verbo) e a etiqueta "O" (não-entidade).

5.2.3 Traços ortográficos

Consideramos dois traços ortográficos para a representação da entrada: o tipo e o formato da palavra. O traço de tipo distingue quatro valores: alfabético (ALFA), alfanumérico (ALFA-NUM), numérico (NUM) e não-alfanumérico (NÃO-ALNUM). O traço de formato categoriza as palavras em: todos os caracteres maiúsculos (UP), somente o primeiro caractere maiúsculo (1-UP), todos os caracteres minúsculos (LOW) e misto (MISC), que abrange qualquer outra combinação, como "YouTube" e "spaCy".

Cada um dos traços ortográficos é alimentado em uma camada *embedding*, que inicializa aleatoriamente uma matriz com distribuição uniforme [-0,5, 0,5], e tem os pesos ajustados durante o treinamento da rede neural. Cada linha da matriz é um vetor com dimensão d, associado a um valor do traço. A

ideia é a mesma do POS-tagging, sendo essa modelizar a associação entre a ocorrência dessas categorias e as etiquetas do corpus.

5.2.4 Traço lexical

Além dos traços ortográficos, POS-tagging e *word embeddings*, incluiu-se um traço lexical. Chiu e Nichols (2016) apontam que a utilização de léxico externo ajuda no REN. Uma forma de fazer isso é através de *gazetteers*, ou seja, inventários contendo exemplos para cada tipo de entidade, extraídos de fontes externas. Por exemplo, uma base de dados classificando nomes de pessoas e de lugares. Assim, as palavras contidas em *gazetteers* são buscadas no texto e, quando encontrada uma correspondência, a entidade é marcada com a etiqueta fornecida naquele *gazetteer*. Em redes neurais, traços baseados em léxico externo são, em geral, convertidos em vetores *one-hot* mapeando a presença de entidades em um texto.

Uma das limitações dos *gazetters* é que eles são independentes de contexto, portanto incapazes de desambiguar entidades pelo contexto de ocorrência. Como exemplo, isoladamente, "Castelo Branco" pode se referir ao nome de um presidente, uma rodovia, etc. Outro ponto é que dificilmente essas bases de dados contém entidades raras, que são as mais difíceis de classificar. Para tentar lidar com esse problema, Ghaddar e Langlais (2018) propõem um método de gerar *word embeddings* de entidades, extraindo o seu contexto a partir da Wikipédia e usando-os como traços para o REN com redes neurais.

Inspirados no trabalho de Ghaddar e Langlais (2018), utilizamos uma abordagem lexical que categoriza as entidades pelo contexto em que elas ocorrem, através de uma busca em listas de palavras-chave que podem indicar uma categoria de entidade específica, semelhante aos exemplos discutidos no Capítulo 2. A Tabela 5.5 traz uma descrição dessas listas.

Cada uma das listas definidas é uma subcategoria de uma categoria de EN, como indicado na Tabela 5.5. Nas demais categorias (Tempo, Valor, Abstração e Outro) não foram encontradas palavras contextuais que pudessem ajudar na classificação. As palavras classificadoras podem estar contidas no

Categoria	Subcategoria	Exemplos	Palavras
Pessoa	parentesco	prima, pai	62
	pronome de tratamento	doutor, senhor	39
Local	logradouro	rua, avenida	45
	topônimo	floresta, montanha	25
Organização	organização	farmácia, mercado	79
Coisa	produto	computador, carro	64
	coisa	próton, galáxia	41
Acontecimento	evento	festival, palestra	26
Obra	obra	livro, filme	26
Total			407

Tabela 5.5: Listas de palavras por categoria.

início de uma EN ou na fronteira imediadamente à esquerda. No total há 407 palavras classificadoras. Alguns exemplos são fornecidos abaixo.

(A) Machado de Assis é o autor do **livro** Dom Casmurro. → obra

(B) O **navio** Titanic afundou em 1912. → coisa

(C) A **Floresta** Amazônica representa mais da metade das florestas tropicais remanescentes no planeta. → topônimo

(D) **Festival** de Cannes é um festival de cinema criado em 1946. → evento

Nas sentenças (A) e (B), "livro" e "navio" são classificadores ocorrendo à esquerda das entidades, pertencentes às categorias Obra e Coisa, em ordem. Já nos casos (C) e (D), os classificadores "floresta" e "festival" fazem parte da EN, ocorrendo no início de cada uma. Tivemos que incluir essa opção porque, em muitos tipos de entidades, a primeira palavra do nome próprio é um modificador categórico, um substantivo genérico usado para a denominação da categoria, como em "Banco do Brasil", "Monte Everest", etc.

Como não possuíamos *gazetteers* para a busca de entidades, a aplicação do traço lexical dependeu de um classificador treinado para a identificação de

ENs. Primeiramente, convertemos as etiquetas dos corpora em binárias, transformando todas as categorias de entidades em uma única (EN) em oposição de não-entidades (O), e treinamos um *Conditional Random Fields* (CRF) no Primeiro Harem. O CRF é um modelo de aprendizado supervisionado que calcula a predição de uma etiqueta considerando o contexto vizinho, semelhante às redes neurais recorrentes. Os detalhes do treinamento de identificação estão no Apêndice B. Ao fim do treino, o CRF aprendeu a classificar um token em EN ou O com uma medida-F de 93,23% no MH.

Esse classificador CRF foi aplicado sobre os corpora PH e MH para identificar os tokens de entidades nomeadas. Através de uma função de busca, as fronteiras de uma EN foram anotadas com o esquema BIO pela detecção de uma sequência de tokens classificados como EN pelo CRF. Assim, ao encontrar uma entidade no texto, verifica-se se algum dos classificadores nas listas ocorreu no contexto da EN, sendo esse à esquerda da entidade ou o primeiro token. Caso encontrado um classificador, a entidade é anotada com a subcategoria correspondente, como parentesco, logradouro, etc.

Para cada token em uma sequência, esse traço foi codificado como um vetor *one-hot* de dimensão $d = 9$, representando as subcategorias de ENs, isto é, as listas na Tabela 5.5. Cada subcategoria pode receber o valor de 1, indicando a presença daquele traço, ou 0, caso contrário. Somente um traço pode estar ativo por token, ou seja, só pode haver um valor 1 no vetor. A Tabela 5.6 traz um exemplo da aplicação do traço lexical.

texto	O	Dr	José	Fiorin	é	formado	na	Universidade	de	São	Paulo	.
parentesco	0	0	0	0	0	0	0	0	0	0	0	0
pronome	0	0	1	1	0	0	0	0	0	0	0	0
logradouro	0	0	0	0	0	0	0	0	0	0	0	0
topônimo	0	0	0	0	0	0	0	0	0	0	0	0
organização	0	0	0	0	0	0	0	1	1	1	1	0
produto	0	0	0	0	0	0	0	0	0	0	0	0
coisa	0	0	0	0	0	0	0	0	0	0	0	0
evento	0	0	0	0	0	0	0	0	0	0	0	0
obra	0	0	0	0	0	0	0	0	0	0	0	0

Tabela 5.6: Exemplo da matriz *one-hot* de mapeamento do traço lexical em uma sentença.

Na Tabela 5.6, as palavras "'dr" e "universidade" são classificadores nas subcategorias pronome de tratamento e organização, respectivamente. O

traço lexical para cada token da entidade "'José Fiorin" é um vetor em que a subcategoria indicativa de pronome de tratamento está ativa, recebe 1. Para "Universidade de São Paulo" ocorre o mesmo, só que agora o traço ativo é organização. Diferente dos traços POS-tagging e ortográficos, o traço lexical não é treinado durante a rede neural. Ele é utilizado apenas como forma de reforçar para a BiLSTM que há uma associação entre determinado traço e uma categoria de EN, por exemplo, a presença de um pronome de tratamento e da classe Pessoa.

5.3 Preparação das entradas para as RNs

Seguindo a metodologia de Souza, Nogueira e Lotufo (2020), está sendo considerada como uma sequência de entrada das redes neurais um documento do corpus. Como discutido no Capítulo 3, o Harem é dividido por documentos em vez de sentenças. Ao utilizar o documento como entrada, as redes neurais têm um contexto extenso para extrair informações para a representação da palavra.

5.3.1 Entrada do BERT

Cada documento do corpus pré-processado é composto de tuplas de um token e a etiqueta correspondente. Antes de alimentar os tokens como entrada do BERT, é preciso pré-processá-los. O primeiro passo foi converter os tokens em WordPieces com o tokenizador da biblioteca Transformers. Além disso, no início e no final de cada sequência foram inseridos, nessa ordem, os tokens especiais [CLS] e [SEP], discutidos no Capítulo 4.

Isso gera um desalinhamento entre os tokens e as etiquetas, assim um pré-processamento semelhante foi aplicado à sequência de etiquetas para realinhá-los. Para isso, utilizou-se uma etiqueta espúria "X" para mapear os tokens iniciados por "##", ausentes antes da tokenização. Também foram inseridas três etiquetas "CLS", "SEP" e "PAD" para mapear os tokens especiais [CLS], [SEP] e [PAD]. O token [PAD] é usado para preencher posições na

normalização do tamanho da entrada (Seção 5.3.3). Na Tabela 5.7 está uma representação do formato da entrada do BERT.

[CLS]	Jim	Hen	##son	era	um	ti	##tere	##iro	[SEP]
CLS	B-PES	I-PES	X	O	O	O	X	X	SEP

Tabela 5.7: Representação da entrada do BERT.

Durante o treinamento, as etiquetas espúrias são treinadas como categorias a serem aprendidas pelo BERT. Entre elas, a etiqueta X serve para mascarar os pedaços não-iniciais dos tokens, senão haveria mais de uma predição para a mesma palavra, como no caso de "Henson" e "titereiro" acima. Isso quer dizer que o modelo aprende a classificar como EN somente o primeiro WordPiece de cada palavra.

5.3.2 Entrada da BiLSTM

No caso da rede neural BiLSTM, o mesmo pré-processamento descrito na Seção 5.3.1 foi aplicado para a extração dos traços *word embeddings*, POS-tagging, ortográficos e lexical. Isso foi necessário porque os *word embeddings* do BERT foram diretamente concatenados aos demais traços. Para que não houvesse um desalinhamento entre eles, optou-se por inserir as etiquetas espúrias em todos os traços. A Tabela 5.8 é uma representação do mapeamento de traços para a entrada da BiLSTM. Nela, são usadas representações categóricas para cada traço, pela ordem: *word embeddings*, POS-tagging, tipo ortográfico, formato ortográfico, lexical e etiqueta. A abreviação "'s/" significa "sem contexto".

[CLS]	Jim	Hen	##son	era	um	ti	##tere	##iro	[SEP]
CLS	NPROP	NPROP	X	VAUX	ART	N	X	X	SEP
CLS	ALFA	ALFA	X	ALFA	ALFA	ALFA	X	X	SEP
CLS	1-UP	1-UP	X	LOW	LOW	LOW	X	X	SEP
CLS	s/	s/	X	s/	s/	s/	X	X	SEP
CLS	B-PES	I-PES	X	O	O	O	X	X	SEP

Tabela 5.8: Representação da entrada da BiLSTM.

5.3.3 Tamanho da entrada

Em redes neurais como LSTM e BERT, é esperado que as sequências de entrada sejam normalizadas para o mesmo tamanho. Desse modo, foi definido um hiperparâmetro para os modelos: o tamanho máximo S da sequência de entrada. Após o pré-processamento, todas as entradas foram limitadas ao tamanho máximo estabelecido. Sequências menores que S foram estendidas até o tamanho máximo pela inserção do token especial [PAD]. Já as maiores que S foram quebradas em períodos de tamanho S contando um passo P para indicar o início da sequência seguinte.

Por exemplo, considerando $S = 5$ e $P = 3$, a sentença "Jim Henson era um titereiro" seria redimensionada como mostrado na Tabela 5.9, dividindo-se em três períodos. Desse modo, parte da informação contida no período anterior é recuperada no seguinte.

	1	2	3	4	5
período 1	[CLS]	Jim	Hen	##son	era
período 2	##son	era	um	ti	##tere
período 3	ti	##tere	##iro	[SEP]	[PAD]

Tabela 5.9: Separação de texto em períodos, considerando $S = 5$ e $P = 3$.

5.3.4 Representação da entrada e saída

O BERT recebe como entrada uma sequência de WordPieces, ou mais precisamente, índices do vocabulário WordPiece, e devolve como saída as representações *word embeddings*. Na rede neural BiLSTM, os *word embeddings* extraídos do BERT são concatenados aos vetores dos demais traços (POS-tagging, lexical e ortográficos), formando um único vetor para representar cada token. Já o conjunto de etiquetas, composto daquelas listadas na Tabela 5.1 mais as espúrias, foi representado como índices de 0-24.

5.3.5 Pós-processamento

Durante o treinamento, cada um dos períodos é uma entrada para as redes neurais. Contudo, na etapa de avaliação, o mesmo token t_i pode aparecer em mais de um período da entrada, o que significa mais de uma predição para aquele token, como é o caso de "era" na Tabela 5.9. Para resolver esse problema, a predição de cada token é tomada do período em que o mesmo se encontra em uma posição mais central. Esse valor é obtido pelo mínimo entre o número de tokens à direita de t_i (n_d) e à esquerda (n_e) multiplicado por S, mais 0,01 para evitar um valor em zero, de acordo com a Equação 5.1.

$$score = \min(n_d, n_e) + 0,01 \times S \tag{5.1}$$

Além disso, as etiquetas espúrias foram desconsideradas na avaliação, ou seja, consideram-se somente as predições obtidas para o primeiro WordPiece de cada palavra, aqueles que não são marcados com a etiqueta "X". Seguimos essa abordagem por Devlin et al. (2018) para a implementação do BERT na abordagem *feature-based*.

5.4 Hiperparâmetros dos modelos

No BERT, adotamos os mesmos hiperparâmetros indicados por Souza, Nogueira e Lotufo (2020): 50 épocas de treinamento com *batch* tamanho 16, otimização AdamW (LOSHCHILOV; HUTTER, 2019) com $\beta_1 = 0,9$ e $\beta_2 = 0,999$, *weigth decay* de 0,1 e taxa de aprendizado inicial de 1e-5 com um decaimento linear e *warmup* nos primeiros 100 passos. O tamanho máximo de $S = 512$ com passo $P = 128$.

Na BiLSTM, os hiperparâmetros foram refinados por meio de testes. A arquitetura base da BiLSTM é composta de duas LSTMs, uma *forward* e uma *backward*. Sobre as saídas da camada BiLSTM foi aplicado um *dropout* de 0,5 para evitar *overfitting*. O traço *word embedding* é composto de quatro camadas BERT, o que dá uma dimensão de 3072. Esse traço é considerado *default* na arquitetura da RN, enquanto os demais (POS-tagging, lexical e ortográficos) são opcionais.

A BiLSTM foi otimizada com Adam (KULLBACK; LEIBLER, 1951), sendo $\beta_1 = 0{,}9$, $\beta_2 = 0{,}999$, $\epsilon =$ 1e-7 e taxa de aprendizado inicial de 1e-3 com uma taxa de decaimento de 1e-5. Após experimentos, definimos $S = 128$ e $P = 64$, que mostraram trazer melhor desempenho para o modelo. Testamos a BiLSTM com essa arquitetura de base para ajustar os demais hiperparâmetros. Isso significa que somente o traço *word embedding* foi utilizado nos experimentos.

Da arquitetura da rede, um dos hiperparâmetros testado foi o número de nós da camada de saída de cada LSTM. Avaliamos um intervalo entre [128, 768], com passo 128, treinando o modelo no Primeiro Harem e avaliando no Mini Harem. A BiLSTM foi treinada por 5 épocas e validada em 5 rodadas distintas. Os resultados na Tabela 5.10 foram obtidos pela média de precisão, cobertura e medida-F das cinco rodadas.

Saída LSTM	**Precisão**	**Cobertura**	**Medida-F**
128	67,37	70,25	68,77
256	68,22	71,06	69,61
512	**68,70**	**71,58**	**70,11**
768	68,58	71,38	69.95

Tabela 5.10: Desempenho da BiLSTM com relação à dimensão da camada de saída.

A medida-F mostra que o melhor desempenho foi obtido com camada de saída tamanho 512 (70,11%), tendo um desempenho inferior com 768. Desse modo, estabelecemos que a camada de saída de cada LSTM (*forward* e *backward*) seria composta de 512 nós.

Também testamos o tamanho da camada *embeding* para os traços ortográficos e POS-tagging. Nesse experimento, treinamos a BiLSTM com os traços POS-tagging e ortográficos, além do traço *word embedding*, variando o tamanho da camada *embedding* entre [25, 100], com passo 25. Realizamos um treinamento por 5 épocas no PH, avaliando no MH. Não houve validação. Seguem os resultados na Tabela 5.11.

Como mostra a Tabela 5.11, com dimensão 100 o desempenho do modelo cai, portanto não testamos valores acima. Os resultados para dimensão 25 (70,54) e 75 (70,82) ficaram bem próximos, com uma diferença maior na

Dimensão	Precisão	Cobertura	Medida-F
25	69,25	71,88	70,54
50	67,09	70,83	68,91
75	**69,56**	**72,13**	**70,82**
100	68,09	71,05	69,54

Tabela 5.11: Resultado para o teste da dimensão *embedding* dos traços.

precisão. Mesmo que seja uma melhora pequena, não houve custo em manter uma camada *embedding* maior, portanto estabelecemos $d = 75$ para os traços ortográficos e o POS-tagging.

Ademais, o número de épocas de treinamento foi ajustado. A BiLSTM foi treinada por 75 épocas, monitorando o erro e a acurácia através do corpus de treino e de validação, correspondente a uma porção de 10% do primeiro. A Figura 5.3 apresenta o histórico de erro e acurácia durante esse treinamento.

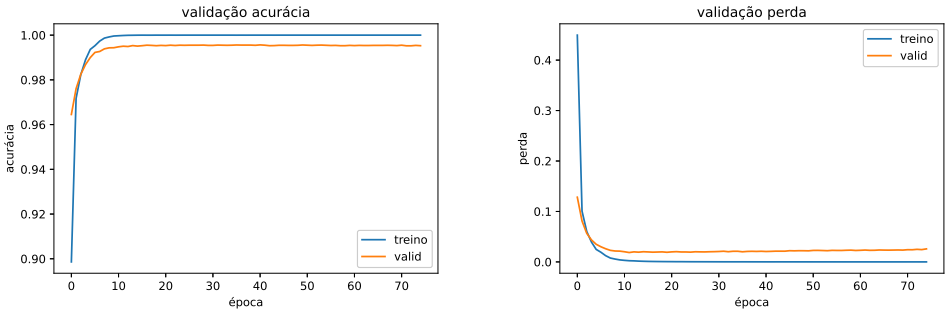


Figura 5.3: Monitoramento de erro e acurácia durante o treinamento. À esquerda, a acurácia em relação ao número de épocas; à direita, o erro.

Em mais ou menos 15 épocas, a acurácia do corpus de treino alcança 1,0, isto é, acerta todas as classificações do corpus. A acurácia na validação, todavia, se estabiliza em 0,995, mais ou menos, variando para cima ou para baixo durante o treinamento. Já a perda começa a subir após cerca de 25 épocas para o conjunto de validação.

A Tabela 5.12 compara os resultados de avaliação de desempenho no MH em um rodada aleatória com 75, 50 e 25 épocas. Em 50 épocas, o modelo alcançou melhor resultado.

Épocas	Precisão	Cobertura	Medida-F
25	72,00	72,46	72,23
50	**73,27**	**73,23**	**73,25**
75	72,88	73,21	73,05

Tabela 5.12: Resultados da BiLSTM em relação ao número de épocas.

Após o refinamento dos hiperparâmetros, chegamos aos valores especificados na Tabela 5.13.

Hiperparâmetro	Selecionado	Intervalo testado
camadas LSTM	2	-
saída da LSTM	512	[128, 768]
taxa de aprendizado	1e-3	-
dropout	0,5	-
épocas	50	[25, 75]
batch	32	-
número de camadas BERT	4	[1, 4]
dimensão *embedding* (traços)	75	[25, 100]
dimensão *embedding* (BERT)	3072	-

Tabela 5.13: Hiperparâmetros da rede neural BiLSTM.

5.5 Treinamento e avaliação

Tanto o BERT quanto a BiLSTM foram treinados no PH e testados no MH. Durante o treinamento, a ordenação dos documentos do PH foi embaralhada aleatoriamente e dividiu-se o corpus em duas partes: treino (90%) e validação (10%). Nos experimentos, testamos todas as combinações entre os traços POS-tagging, ortográficos e lexical para validar a influência de cada um no desempenho.

Ambos os modelos foram testados em dois cenários distintos: o cenário total, considerando as dez categorias do Harem, e o seletivo, em que os modelos são treinados somente nas categorias Pessoa, Local, Organização, Tempo e Valor. Para a validação do desempenho, utilizamos *10-fold validation*,

uma técnica usada para evitar um resultado enviesado pelo treinamento. Esse método consiste em treinar o mesmo modelo por 10 rodadas diferentes, embaralhando aleatoriamente o corpus de treinamento a cada rodada. Assim, cada uma das rodadas receberá um conjunto de treino distinto, evitando o viés dos dados e fornecendo um resultado mais confiável. O desempenho é obtido pela média das 10 rodadas testadas. As métricas de avaliação de desempenho são aquelas propostas pelo COnLL: precisão, cobertura e medida-F, como descritas no Capítulo 2.

A plataforma *Google Colaboratory*[7] (Colab) serviu como ambiente de desenvolvimento para a implementação das redes neurais. O Colab é uma plataforma que permite escrever e rodar códigos em Python de forma *online*, disponibilizando acesso gratuito a GPUs (*Graphics Processing Unit*), processadores potentes que aceleram a execução de sistemas que realizam grandes conjuntos de cálculos, como as redes neurais.

5.6 Experimento com *word embeddings*

Como resultados secundários, apresentamos alguns experimentos com as representações *word embeddings* geradas pelo BERT. No Capítulo 2, pontuou-se que uma das aplicações de representações *word embeddings* é a comparação de similaridade entre palavras ocorrendo em contextos semelhantes. Para *embeddings* contextuais, como é o caso do BERT, a obtenção dessa medida não é usual, uma vez que a representação de uma palavra difere de acordo com o contexto em que ela ocorre, portanto é possível haver mais de uma representação por palavra. O objetivo desse teste, porém, foi exatamente analisar as representações de cada palavra com base no contexto de ocorrência, de modo que esse quesito não foi um problema.

Nesse experimento, foram extraídos os *word embeddings* BERT das Entidades Nomeadas contidas no Primeiro Harem. Na primeira etapa, o PH foi dividido em sentenças em vez de documentos. As fronteiras da sentença foram identificadas por ponto final (.), interrogação (?), exclamação (!) e

[7] https://colab.research.google.com/

ponto-e-vírgula (;), obtendo-se um total 4556 sentenças. A tokenização das palavras não foi alterada.

Em seguida, extraiu-se o *word embedding* de cada token etiquetado como EN no corpus com auxílio da biblioteca bert-as-service[8]. Utilizamos o modelo BERT-PT pré-treinado, extraindo as representações somente da última camada. Com relação aos tokens quebrados em mais de um WordPiece, somente a primeira representação foi importada. Como as sentenças são mais curtas do que os documentos, adotou-se um contexto máximo $S = 100$, sem necessidade de P, já que nenhuma sentença ultrapassou esse tamanho.

O resultado desse passo-a-passo foi um dicionário associando cada token de EN a sua representação BERT *embedding*. Tendo em vista que o mesmo token pode ocorrer mais de uma vez no corpus, foram anotadas numerações indicando a ordem de ocorrência daquele token no corpus. Suponha que "Brasil" ocorreu vinte vezes, então existem vinte entradas da palavra no dicionário etiquetadas com números de 1-20, ("Brasil-1", "Brasil-2", etc.), cada qual com sua representação respectiva.

Por fim, implementamos uma função de similaridade por cosseno[9] (Algoritmo 1), que recebe como entrada duas representações *word embeddings* e calcula o valor de similaridade entre elas.

Algoritmo 1 Similaridade entre cossenos

função SIMILARIDADE($vetor_1$, $vetor_2$)
    $numerador = vetor_1 \cdot vetor_2$
    $V_1normalizado = \sqrt{\sum_i^n} v_i^2$, for $v_i \in vetor_1$
    $V_2normalizado = \sqrt{\sum_i^n} v_i^2$, for $v_i \in vetor_2$
    $denominador = V_1normalizado \times V_2normalizado$
    $similaridade = numerador \div denominador$
    retorna *similaridade*
fim função

Durante os experimentos, dois algoritmos foram usados na análise dos *word embeddings*: K-Means e T-SNE, o primeiro para indução de *clusters* e

[8] https://bert-as-service.readthedocs.io/en/latest/
[9] Ver Jurafsky e Martin (2018, p. 104).

o segundo para visualização. Ambos serão discutidos em mais detalhes nas subseções a seguir.

5.6.1 K-Means

K-means (MACQUEEN et al., 1967) é uma técnica de aprendizado não-supervisionado usado para a clusterização de dados, isto é, agrupamento de exemplos em subcategorias de acordo com as características compartilhadas entre eles. Um dos parâmetros a ser definido no K-means é o número k de centroides ou categorias que se deseja no final do treinamento. Centroide é um local representando o centro de um agrupamento (do inglês, *cluster*). Um ponto é considerado parte de determinado agrupamento se ele se encontra mais próximo do seu centroide de que dos demais.

O K-means recebe como entrada um conjunto de dados representados por vetores n-dimensionais. Inicialmente, k centroides são escolhidos aleatoriamente e os pontos de dados mais próximos de cada centroide são atribuídos àquele agrupamento. Então, iterativamente, novos centroides são computados com o objetivo de minimizar a variância nos grupos, recalculando novos agrupamentos até o modelo convergir, o que ocorre quando os agrupamentos formados param de mudar ou o número máximo de iterações é atingido.

5.6.2 T-SNE

T-distributed Stochastic Neighbor Embedding (T-SNE) é um algoritmo para visualização de dados multidimensionais proposto por Maaten e Hinton (2008). O algoritmo aplica uma técnica de redução de dimensionalidade não-linear que modela as dimensões de um objeto em pontos de duas ou três dimensões por similaridade e dissimilaridade entre os pontos.

Na primeira etapa, o T-SNE constrói uma distribuição de probabilidades sobre pares de objetos n-dimensionais, por exemplo *word embeddings*, de forma que os objetos similares recebem alta probabilidade, enquanto os dissimilares têm uma baixa probabilidade. A similaridade entre os objetos é calculada como a probabilidade condicional com a qual um objeto A escolheria o objeto B como seu vizinho. Em seguida, o T-SNE projeta uma distribuição de

probabilidades semelhante sobre os pontos em um mapa de baixa dimensão, 2D por exemplo, e tenta minimizar a divergência de Kullback-Leibler (KULLBACK; LEIBLER, 1951) entre as duas distribuições com respeito às localizações dos pontos no mapa de alta e de baixa dimensão.

5.7 Recursos auxiliares

Outros algoritmos e ferramentas auxiliaram na análise de dados, dos gráficos e no desenvolvimento dos experimentos, todos implementados em Python. Além das bibliotecas Keras e Transformers para implementação das redes neurais, as bibliotecas Seaborn[10] e Matplolib[11] serviram para visualização gráfica. Os algoritmos K-Means e T-SNE foram implementados a partir da biblioteca sckit-learn[12] (PEDREGOSA et al., 2011).

Entre os recursos gráficos utilizados estão os modelos boxplot e matriz de confusão. Como a interpretação dessas representações requer conhecimentos específicos em estatística e aprendizado de máquina, as subseções a seguir serão voltadas para uma breve explicação a respeito de cada uma.

5.7.1 Boxplot

Em estatística descritiva, boxplot é um modelo gráfico para a representação da dispersão em um conjunto amostral de uma variável quantitativa. O boxplot é baseado nas medidas de posição dos quartis. Essas são obtidas a partir de um conjunto de dados ordenados do menor para o maior, sendo que o segundo quartil ou mediana (MD) se encontra na posição central, o primeiro quartil ($Q1$) corresponde aos primeiros 25% dos dados (menores valores) e o terceiro quartil ($Q3$) aos 75% (valores maiores).

Considerando o conjunto amostral $[1, 3, 5, 5, 8, 12, 15, 16]$, com $n = 8$ valores ordenados, a mediana corresponde à posição 4,5 ($0{,}5 \times (n + 1)$). Como essa posição não existe nos dados, faz-se a média entre os valores na posição 4

[10] https://seaborn.pydata.org/
[11] https://matplotlib.org/
[12] https://scikit-learn.org/stable/.

e 5, obtendo um mediana de 6,5 $((5+8)/2)$. Já o $Q1$ seria igual à 4 $((3+5)/2)$ e o $Q3$ seria 13,5 $((12+15)/2)$.

O boxplot é construído a partir de um retângulo, cujos limites inferior e superior são as posições do $Q1$ e $Q3$. A Figura 5.4 ilustra um gráfico boxplot.

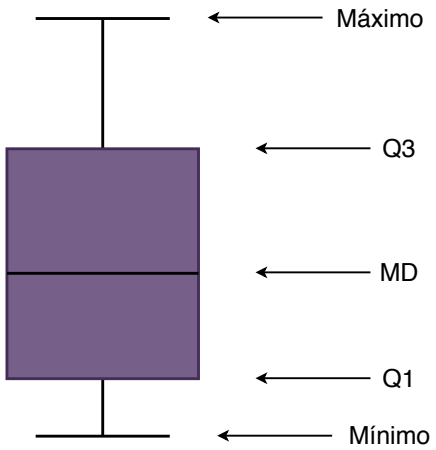


Figura 5.4: Representação de um boxplot.

A linha que corta o retângulo é a posição da mediana. Já as hastes "Máximo" e "Mínimo" são os pontos com o maior e o menor valor na amostra de dados, com exceção dos valores discrepantes ou *outliers*. Os *outliers* são obtidos pela diferença entre $Q3$ e $Q1$ multiplicada por 1,5. O produto dessa operação é subtraído de $Q1$ e somado à $Q3$. Se um ponto ultrapassar essas barreiras, ele é considerado um *outlier*.

5.7.2 Matriz de confusão

A matriz de confusão é uma tabela utilizada para a visualização do desempenho de um algoritmo de classificação. Suponha-se que um modelo tenha sido treinado para classificar cães e gatos em um corpus contendo 90 imagens, 42 gatos e 48 cachorros. A Tabela 5.14 representa as predições obtidas pelo modelo (linhas) versus as etiquetas esperadas (colunas) para cada categoria.

O modelo computou 33 acertos para a categoria gato e 43 para cachorro (diagonal principal na Tabela 5.14). Esse acerto é chamado de Verdadeiro Positivo (VP), isto é, a classificação prevista para um exemplo relevante é

		Etiqueta	
		gato	cachorro
Predição	gato	33	5
	cachorro	9	43

Tabela 5.14: Classificação da predição versus o esperado de um modelo.

igual àquela esperada. Considerando a categoria de gato, por 9 vezes o modelo classificou como "cachorro" imagens que, na verdade, eram de gatos. Quando um modelo prevê que um exemplo não pertence a determinada categoria, quando na verdade ele pertencia, chamamos esse erro de Falso Negativo (FN). Em contrapartida, 5 imagens de cachorro foram erroneamente classificadas como gato, erro chamado de Falso Positivo (FP), nos casos em que o modelo prediz um exemplo como membro de uma categoria da qual ele não faz parte.

A Tabela 5.14 é um exemplo de uma matriz de confusão para duas categorias, mas o mesmo conceito pode ser aplicado em mais. Essa é uma representação útil porque, a partir dela, podemos inferir a porcentagem de erros e acertos de um modelo e de confusão na classificação entre as classes.

6 Resultados

Um dos principais objetivos desta pesquisa foi a proposta de um modelo de classificação para as Entidades Nomeadas que se baseasse em informações extraídas do contexto. Ao longo do caminho, foram explorados algumas abordagens para o REN, inicialmente focando na elaboração de regras manuais e, depois, com os modelos de redes neurais. Neste capítulo, apresentaremos os resultados obtidos com os modelos implementados e discutiremos cada um deles. Ao final, fornecemos um quadro comparativo de trabalhos no REN do português.

6.1 Similaridade entre as entidades

Uma das características que tornam o BERT um modelo *word embedding* tão poderoso é sua capacidade de representação de palavra contextualizada. Nesta seção, são fornecidos alguns experimentos explorando essas representações extraídas do Harem. No primeiro teste, foram avaliadas as distinções de representação de uma única palavra de acordo com seu contexto de ocorrência.

A Figura 6.1 apresenta uma visualização bidimensional da distribuição espacial dos *word embeddings* de cada contexto de ocorrência da entidade "Santos" no PH, gerada utilizando o algoritmo T-SNE, com perplexidade 5, inicialização PCA e 3500 iterações. Cada ocorrência foi anotada com um número indicando a sua ordenação no corpus. Os contextos correspondentes podem ser verificados no Apêndice A.

Observando a Figura 6.1, percebe-se a formação de pequenos grupos, que indicam maior similaridade entre aqueles contextos. Um deles, formado por

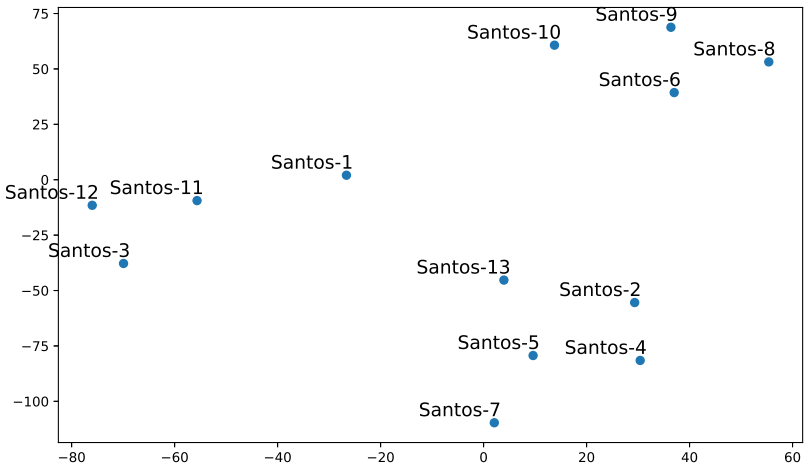


Figura 6.1: Representação visual da distribuição contextual de Santos.

13-2-5-4-7, contém ocorrências de "Santos" em contextos com interpretação locativa, por exemplo "Alameda Santos" e "para Santos", exceto por 7 que se refere a uma pessoa. Em muitos deles, a entidade foi precedida pela preposição "em", forte indicativo de locativo, que pode ser o motivo pelo qual esses contextos apresentaram mais similaridade entre si.

Já no aglomerado 10-9-6-8, os contextos são sobre uma única pessoa de nome "Machado dos Santos". Isso indica que a proximidade desses contextos está associada com o fato de que todos eles se referem à mesma entidade. Outras palavras compartilhadas entre esses contextos foram "professor" e "reitor", duas palavras com significados associados.

O aglomerado composto de 12-11-3 também se refere a entidades do tipo Pessoa, sendo elas "Santos Simões", nos contextos 12 e 11, e "Santos Silva", em 3. É interessante que, neste caso, "Santos" é o primeiro nome da entidade, enquanto no aglomerado anterior era parte do sobrenome. O único contexto que não apresentou muita similaridade com os demais foi o 1, também se referindo a uma pessoa chamada "Arulemo Santos Novaes".

Essa análise sobre a associação por similaridade entre os *word embeddings* da entidade "Santos" nos permitiu classificá-la de acordo com o tipo de contexto de ocorrência, por exemplo locativo, e o tópico contextual, a quem o nome estava se referindo.

Em um segundo experimento, selecionamos dois dos contextos de ocorrência da entidade "Santos", um relacionado a uma pessoa (contexto 1) e o outro a um lugar (contexto 13), e calculamos a similaridade por cosseno com cada uma das demais representações de entidades extraídas do PH. A Figura 6.2 traz as 10 representações mais similares a cada um desses dois contextos.

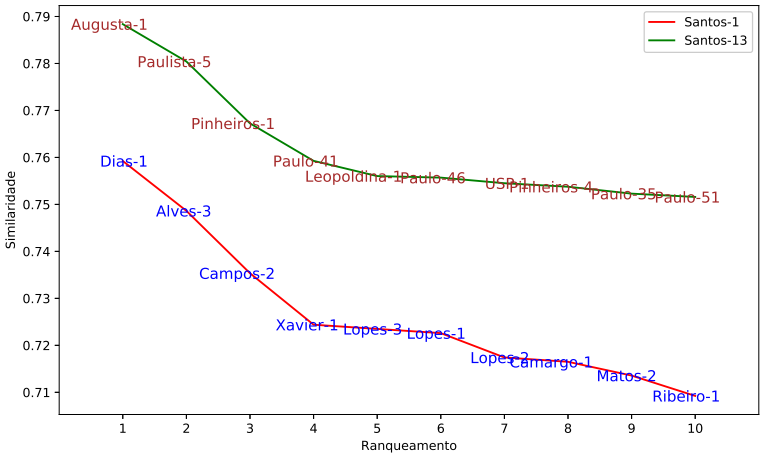


Figura 6.2: Entidades mais similares a "Santos" em dois contextos distintos.

Entre as representações mais similares ao contexto de Santos-13 (locativo) estão "Augusta", "Paulista", "Pinheiros" e "Paulo", ou seja, entidades que se referem a nomes de bairros, avenidas e cidades. Já na acepção de pessoa, as palavras mais similares são "Dias", "Alves", "Campos" e "Xavier", todos partes de nomes de pessoas e, o que mais chama a atenção, referentes a sobrenomes, assim como é "Santos", em geral.

Como último teste, extraímos 100 representações de entidades aleatórias do PH e treinamos um modelo de clusterização K-means para agrupá-las por similaridade. Como hiperparâmetro do algoritmo, foi pré-estabelecido $k = 7$ agrupamentos. Nesse teste, foi feita uma média dos vetores correspondentes a um único token, então cada ponto corresponde à representação da palavra em vez do contexto.

A Figura 6.3 traz os agrupamentos gerados pelo K-means. Cada um dos pontos é marcado com uma cor que indica a qual agrupamento ele pertence e os pontos bejes maiores são as posições dos centroides.

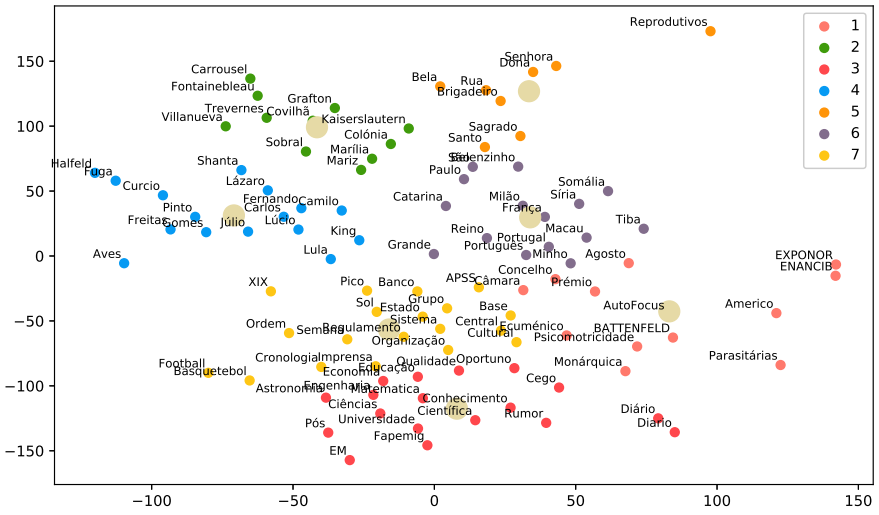


Figura 6.3: Agrupamentos de 100 entidades do Harem geradas com o K-means.

Analisando os agrupamentos obtidos, vemos que alguns deles capturaram bem as similaridades semânticas entre os membros. Por exemplo, o agrupamento número 4 tem inúmeros nomes de pessoas, como “Freitas”, “Gomes” e “Júlio”. Por sua vez, o número 6 apresenta diversos nomes de lugares, entre os quais “Portugal”, “Milão” e “França”. Outro agrupamento interessante é o número 3, composto de nomes associados a disciplinas, como “Matemática” e “Economia”, além de palavras como “Universidade”, “Pós” e “Conhecimento”.

Outros agrupamentos são mais mistos, com menor relação semântica entre os termos. Para citar um, o grupo 5 tem palavras como “Santo” e “Sagrado”, bem próximas, e outras como “Rua” e “Brigadeiro”, que já não tem muita relação com as outras duas.

O intuito dessas análises foi explorar um pouco mais de perto as representações geradas pelo BERT, fornecendo exemplos que permitissem observar as associações por similaridade que podem ser extraídas delas. Com isso, foi

possível analisar as variações de representação para uma mesma entidade de acordo com o contexto e como essas representações se associam entre si. Os agrupamentos que obtivemos corroboram a hipótese da Semântica Distribucional sobre a proximidade de distribuição entre palavras estar correlacionada com similaridades semânticas. Já do ponto de vista computacional, vimos que o BERT é capaz de capturar e representar traços sobre o contexto de maneira que, mesmo que cada vetor de palavra seja único, existem similaridades entre eles que nos permitem associá-los com base em sua distribuição.

6.2 Resultados do BERT

Nesta seção, serão discutidos os resultados do BERT na abordagem *fine-tuning*. Todos os resultados foram avaliados em relação aos cenários total (10 categorias) e seletivo (5 categorias). O desempenho do modelo é calculado pela média dos resultados obtidos na etapa de validação em 10 rodadas. Na Tabela 6.1, seguem os resultados do BERT-ML e do BERT-PT.

Modelo	Cenário	Precisão	Cobertura	Medida-F
BERT-ML	total	72,29	72,85	72,41
	seletivo	78,16	78,18	78,17
BERT-PT	total	**74,11**	**76,35**	**75,22**
	seletivo	**80,97**	**81,23**	**81,09**

Tabela 6.1: Medidas de desempenho do BERT.

Em ambos os cenários, o BERT-PT chegou a melhores resultados do que o BERT-ML, com uma diferença de 2,81 de medida-F no cenário total e 2,92 no seletivo, demonstrando que o BERT-PT fornece um melhor desempenho no Harem. Além disso, não surpreendentemente, os resultados no cenário seletivo foram melhores do que no total, com uma diferença de 5,76 no BERT-ML e 5,88 no BERT-PT. Um dos motivos para isso são as categorias com poucos membros, mais difíceis de classificar, que são desconsideradas no cenário seletivo. Uma das indicações disso é que a diferença entre o cenário total e o seletivo foi maior com relação à precisão do que quanto à cobertura, isto

é, os modelos passaram a acertar mais as classificações no cenário seletivo. Por último, a cobertura prevaleceu mais alta do que a precisão em todos os cenários, o que quer dizer que os modelos recuperam muitos dos exemplos de entidades, mas erraram na classificação das categorias.

6.2.1 Dispersão dos dados de validação

Quando uma amostra de dados não é muito grande, como neste caso, a média se torna uma medida pouco segura, pois é facilmente enviesada por valores extremos. Para fornecer confiabilidade nos resultados obtidos, avaliamos as medidas de dispersão para as 10 rodadas de validação em cada cenário. Os gráficos boxplot na Figura 6.4 representam a precisão, a cobertura e a medida-F obtidas na validação do BERT-ML para o cenário total. A linha cortando o retângulo é a mediana, o triângulo verde demarca a média e os pontos mínimo e máximo são indicados pelas hastes.

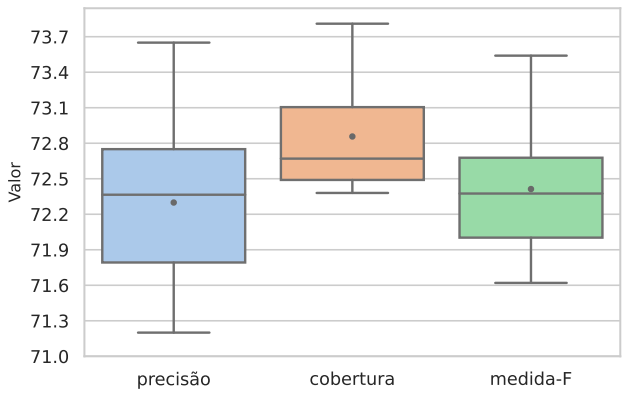


Figura 6.4: Dispersão das medidas de desempenho do BERT-ML no cenário total.

A medida-F do BERT-ML teve uma dispersão de 1,92, com valor máximo em 73,54 e mínimo de 71,62. A sua média ficou bem próxima da mediana (72,38), indicando pouca assimetria entre os valores mais extremos (máximo e mínimo). Quanto à precisão, a média apareceu um pouco abaixo da mediana, ou seja, foi influenciada por valores muito baixos, tendo uma dispersão entre 73,65 e 71,89. A cobertura foi a mais assimétrica entre as três medidas, com

uma distância de 0,19 entre a média e a mediana. Isso se deve a alguns valores muito altos no terceiro quartil que puxaram a média para cima, o que traz uma ideia enganadora sobre a distribuição da amostra. É possível perceber isso no boxplot pelo ponto máximo, que chega a 73,81, relativamente afastado da mediana (72,67).

Os gráficos de dispersão de desempenho na validação do BERT-ML para o cenário seletivo estão na Figura 6.5.

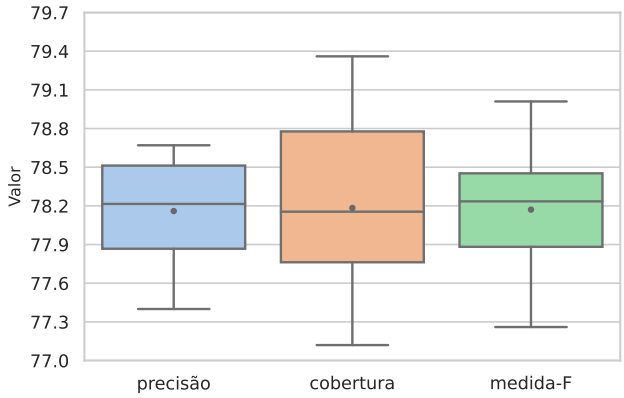


Figura 6.5: Dispersão das medidas de desempenho do BERT-ML no cenário seletivo.

Os resultados no cenário seletivo foram menos dispersos quando comparados aos do total, com as medianas e médias bem próximas nas três medidas. Na melhor rodada, o BERT-ML conseguiu 79,01 de medida-F e, na pior, 77,26, uma diferença de 1,75. Para a precisão, o valor máximo ficou bem baixo, não muito longe da média, enquanto o mínimo (77,40) se distanciou do ponto médio em 0,82, puxando a média para baixo. Desse modo, se mais rodadas fossem realizadas, há indicações de que a precisão poderia ser um pouco mais alta, já que esse pontos extremos influenciariam menos. Com a cobertura ocorre o oposto, com a mediana alguns décimos abaixo da média, portanto ao menos metade dos valores obtidos está abaixo da média.

Passando ao BERT-PT, os gráficos de dispersão no cenário total estão na Figura 6.6.

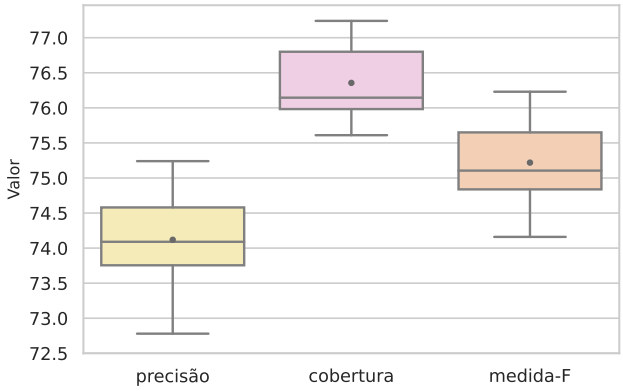


Figura 6.6: Dispersão das medidas de desempenho do BERT-PT no cenário total.

A primeira observação é que a dispersão no BERT-PT foi menor do que no BERT-ML, com as medianas e médias bem similares. Na medida-F, a dispersão vai de 74,16 até 76,23, com uma mediana de 75,10. Por outro lado, o BERT-PT apresentou bastante diferença entre as medidas de cobertura e precisão, com o valor máximo da precisão em 75,24 e o mínimo da cobertura em 75,61, isto é, o valor máximo alcançado na precisão nem chegou ao mínimo da cobertura. Essa discrepância aponta que o modelo cometeu muitos erros de classificação, apesar de ter reconhecido as Entidades Nomeadas. Assim como no cenário total do BERT-ML, neste caso também houve mais assimetria na cobertura, uma diferença de 2% entre a média e a mediana, assim ao menos metade dos valores ficaram abaixo da média.

Por fim, os gráficos de dispersão de desempenho para o cenário seletivo estão dados na Figura 6.7.

Os resultados neste cenário foram os com menor dispersão, principalmente considerando a cobertura, em contraste ao cenário total. A medida com mais assimetria foi a medida-F, com mediana de 80,94, 0,15 abaixo da média, e a com maior dispersão entre o ponto máximo e o mínimo foi a precisão.

Na melhor rodada, o BERT-PT alcançou 82,58 de medida-F, 82,93 de precisão e 82,23 de cobertura, representados pelos três pontos acima de cada medida, indicando que se trata de *outliers*, isto é, essa foi uma rodada discrepante comparada às demais do BERT-PT. Por causa dessa rodada, a

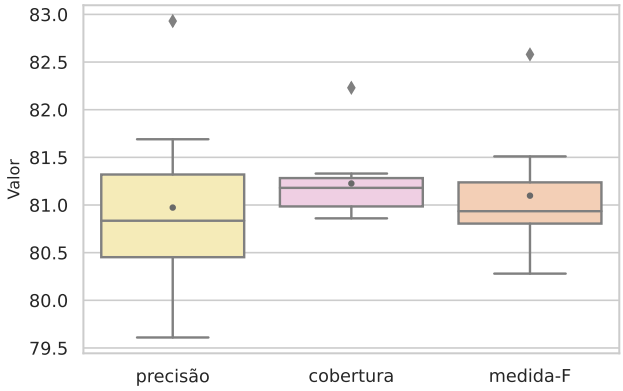


Figura 6.7: Dispersão de medidas de desempenho do BERT-PT no cenário seletivo.

média do modelo ficou um pouco acima do ponto médio, enviesando os dados da amostra. Desconsiderando essa rodada, a medida-F média ficaria em 80,93 e não 81,09 como calculamos pela amostra completa.

Como comentários finais temos que, via de regra, o BERT-PT teve resultados menos dispersos do que o BERT-ML, mostrando maior estabilidade de desempenho. Em ambos os modelos, a cobertura no cenário total se mostrou mais assimétrica, com tendência de média maior do que mediana, portanto muitos dos valores obtidos nas rodadas de validação ficaram abaixo da média. Os cenários seletivos obtiveram resultados mais estáveis, com as médias de precisão e cobertura mais próximas, o que é um comportamento desejável em um bom modelo de classificação.

6.2.2 Análise por categoria de entidade

Nesta seção, serão analisados mais de perto os resultados dos modelos por categoria de entidade, comparando os dois cenários e modelos. Selecionou-se uma rodada do conjunto de validação que tivesse a medida-F mais próxima da média para tentar evitar valores enviesados. Em todo o caso, essas medidas são apenas uma observação individual de uma rodada. Na Tabela 6.2, está o desempenho do BERT-ML por categoria e na Tabela 6.3, do BERT-PT. A última coluna (N) se refere ao número de entidades no MH.

Categoria	Precisão	Cobertura	Medida-F	N
ABS	41,25	48,77	44,70	203
ACO	31,73	57,89	40,99	57
COI	58,14	44,12	50,17	170
LOC	83,43	83,43	83,43	875
OBR	43,86	52,36	47,73	191
ORG	64,03	67,45	65,69	599
OUT	6,25	14,29	8,70	14
PES	79,93	78,19	79,05	830
TEM	90,78	87,50	89,11	360
VAL	76,18	79,69	77,89	325
micro média	72,75	73,57	72,99	3624

Tabela 6.2: Resultados por categoria do BERT-ML no cenário total.

Como já esperado, o desempenho entre as categorias variou muito, com tendência de aquelas com ao menos 300 ocorrências (PES, LOC, ORG, TEM, VAL) se sair melhor. Entre elas, a categoria Tempo alcançou 89,11 no BERT-ML e 91,81 no BERT-PT, o maior resultado em ambos os casos. Um dos motivos para isso, além da quantidade de ocorrências nessa categoria, é a regularidade de determinadas expressões temporais, tais como "século XIX" e "dia 16 de novembro de 2020", que facilitam a identificação desse grupo pelo modelo. Como evidência, temos a alta precisão alcançada nessa categoria, acima de 90% nos dois casos, mostrando que os modelos tiveram muita certeza da classificação.

Em contrapartida, a classe Outro obteve o pior desempenho, com apenas 8,70 de medida-F no BERT-ML e 8,51 no BERT-PT. No Harem, Outro representa todos os exemplos que não se encaixaram em nenhuma outra categoria formando, portanto, um grupo extremamente heterogêneo, o que dificulta na classificação. Isso, combinado ao número de membros da categoria, apenas 40 no treinamento e 14 na avaliação, a tornaram a mais difícil para classificação.

Especificamente em relação ao BERT-PT, podemos observar que as medidas de precisão e cobertura por categoria estão mais equilibradas quando comparadas ao BERT-ML, com exceção de Obra e Outro. No primeiro, a

Categoria	Precisão	Cobertura	Medida-F	N
ABS	44,74	50,25	47,33	203
ACO	41,03	56,14	47,41	57
COI	48,23	40,00	43,73	170
LOC	83,80	82,17	82,98	875
OBR	45,52	69,11	54,89	191
ORG	67,45	71,62	69,47	599
OUT	6,06	14,29	8,51	14
PES	86,47	83,13	84,77	830
TEM	93,39	90,28	91,81	360
VAL	79,01	78,77	78,89	325
micro média	74,02	76,27	75,13	3624

Tabela 6.3: Resultados do BERT-PT por categoria no cenário total.

cobertura superou a precisão em 23,16 e, no segundo, em 8,23. Tais discrepâncias são um indicativo de que o modelo confundiu essas categorias com outras na etapa de classificação. Entre as classes menores, Abstração, Acontecimento e Obra melhoraram quando comparadas aos resultados do BERT-ML, principalmente com relação à precisão, o que pode ser efeito dessa rodada específica ou reflexo de uma melhora nessas categorias.

Para comparação, as Tabelas 6.4 e 6.5 trazem os resultados de desempenho dos modelos no cenário seletivo.

Categoria	Precisão	Cobertura	Medida-F	N
LOC	82,43	81,49	81,95	875
ORG	62,96	68,11	65,44	599
PES	81,61	78,07	79,80	830
TEM	91,25	86,94	89,05	360
VAL	75,68	77,54	76,60	325
micro média	78,24	78,09	78,16	2989

Tabela 6.4: Resultados por categoria do BERT-ML no cenário seletivo.

As medidas obtidas por categoria no cenário seletivo ficaram parecidas com aquelas do cenário total, isto é, o treinamento em somente cinco categorias

Categoria	Precisão	Cobertura	Medida-F	N
LOC	84,60	84,11	84,36	875
ORG	68,37	71,45	69,88	599
PES	82,62	81,33	81,97	830
TEM	93,08	89,72	91,37	360
VAL	78,70	81,85	80,24	325
micro média	80,88	81,23	81,05	2989

Tabela 6.5: Resultados do BERT-PT por categoria no cenário seletivo.

não melhorou tanto o desempenho individual em cada uma. É claro que este representa somente o resultado de uma rodada, o que não nos permite falar com certeza sobre a melhora ou piora em determinada categoria.

Entre as cinco categorias, a com o pior desempenho nos dois modelos foi Organização, mesmo contendo 956 ocorrências no treinamento e 599 na avaliação, sendo uma categoria com mais exemplos do que Tempo e Valor. A dificuldade de generalização de alguns tipos de entidades já foi notada em outras pesquisas, conforme Jiang, Banchs e Li (2016) e Augenstein, Derczynski e Bontcheva (2017), que apontam para Organização como a categoria mais difícil na tríade Pessoa, Local e Organização. A diferença entre a precisão e a cobertura nessa categoria mostra que há algum viés fazendo com que o modelo consiga identificar esses termos, mas falhe na tarefa específica de classificação.

Atribuímos o fato, principalmente, à dificuldade de extração de informações contextuais de organizações, já que é mais raro encontrar contextos específicos para se referir a esses termos, diferente dos casos de Pessoa e Local, que frequentemente vêm precedidos de modificadores (por exemplo "cidade", "jogador", etc.), ainda mais quando se considera um corpus fortemente jornalístico, como o Harem.

A partir dessa análise, podemos concluir que dois fatores principais parecem ter afetado a classificação de entidades pelo BERT: o número de exemplos e a heterogeneidade da categoria. Categorias com poucos exemplos de treinamento e muito heterogêneas, tanto no nível lexical quanto no contextual, foram as mais difíceis de classificar. A heterogeneidade lexical se refere às

categorias que frequentemente contêm, por exemplo, termos estrangeiros, palavras raras ou inventadas, siglas, substantivos comuns, etc.

Já comparando os cenários total e seletivo, como não fizemos uma análise estatística do desempenho por categoria, não há como afirmar se houve uma melhora de desempenho nas categorias ou se as diferenças entres eles é devido à desconsideração das categorias menores. Observando aquelas rodadas específicas, os resultados mantiveram distribuições similares na classificação. Uma coisa que podemos afirmar é que muitos dos erros dos modelos estavam concentrados nas categorias menores, como suspeitávamos, que apresentaram os piores desempenhos.

6.2.3 Resultados do BERT no Harem

O BERT já havia sido testado no REN do português em outra pesquisa, já comentada no Capítulo 3. Souza, Nogueira e Lotufo (2020), os autores do BERT-PT, avaliaram o desempenho do BERT_{BASE} no Harem com a abordagem *fine-tuning*, utilizando os modelos multilíngue e português. Como os nossos modelos foram testados com os mesmos hiperparâmetros que os dos autores, faremos uma breve comparação de resultados nesta seção. As medidas obtidas em ambos os cenários estão na Tabela 6.6.

Trabalho	**Modelo**	**Cenário total**			**Cenário seletivo**		
		P	C	F1	P	C	F1
Souza et al.	BERT-ML	72,97	73,78	73,37	77,35	79,16	78,25
	BERT-PT	78,36	77,62	77,98	83,22	82,85	83,03
Nosso	BERT-ML	72,29	72,85	72,41	78,16	78,18	78,17
	BERT-PT	74,11	76,35	75,22	80,97	81,23	81,09

Tabela 6.6: Comparação dos nossos resultados no BERT aos de Souza, Nogueira e Lotufo (2020).

Os resultados de Souza, Nogueira e Lotufo (2020) superaram os nossos em ambos os modelos. No BERT-ML, eles obtiveram medida-F de 73,37 no cenário total e 78,25 no seletivo em comparação a 72,41 e 78,17, desta pesquisa. No cenário seletivo, a diferença foi pequena, apenas 0,08. Já no

BERT-PT, o contraste entre os resultados foi maior, com 2,76 no cenário total e 1,94, no seletivo.

Determinadas distinções nos métodos de implementação e avaliação entre os trabalhos podem ser responsáveis por essas diferenças. Primeiro, Souza, Nogueira e Lotufo (2020) aplicam uma técnica de estabilização para lidar com as desproporções entre o número de categorias do corpus, aplicando um viés maior sobre a categoria de não-entidades. Na etapa de avaliação, os autores aplicam um pós-processamento que mascara as transições de etiquetas inválidas no esquema BIO, como "I" precedido por "O", substituindo-as por "O". Isso tem como efeito um aumento de precisão, com um concomitante custo de cobertura.

Desse modo, essas diferenças de desempenho são esperadas. Além do mais, os nossos resultados seguem nas mesmas direções, com padrões semelhantes nos cenários total e seletivo e de um modelo para o outro. Também é preciso considerar que o Harem é um corpus difícil para o REN, pois tem um número de categorias maior do que outros. Por isso, os resultados obtidos pelo BERT chamam a atenção, já que, sem o auxílio de outros recursos, alcançam desempenho muito bom, principalmente no cenário seletivo.

Para terminar, também pudemos observar o quanto o desempenho da mesma arquitetura de rede neural pode variar se técnicas distintas são aplicadas sobre elas, o que é uma das maiores preocupações quando se trabalha com esses modelos, já que mesmo a diferença em um hiperparâmetro pode gerar resultados bastante distintos.

6.3 Resultados da BiLSTM

Assim como o BERT, o desempenho da BiLSTM foi validado por 10 rodadas. Antes da combinação com os traços, testamos a BiLSTM com as configurações *default*, o que significa que somente os *word embeddings* extraídos do BERT foram usados para a representação das palavras. A média de validação da rede está na Tabela 6.7.

Somente com as representações do BERT, a BiLSTM alcançou medida-F de 73,06 no cenário total e 79,68 no seletivo. Com uma rede neural que

Cenário	Precisão	Cobertura	Medida-F
total	72,80	73,32	73,06
seletivo	81,24	78,19	79,68

Tabela 6.7: Validação de desempenho da BiLSTM.

combinava representações de palavra e de caractere, Santos e Guimaraes (2015) alcançaram, respectivamente, 65,41 e 71,23. Esse resultado mostra o quanto os *word embeddings* do BERT são eficientes na representação de palavras, mesmo quando eles não são refinados em uma tarefa específica. Assim como no BERT, o desempenho da BiLSTM no cenário seletivo foi melhor, superando em 6,62 o cenário total.

As medidas obtidas na BiLSTM foram comparáveis às da abordagem *fine-tuning*, ficando apenas um pouco abaixo nos dois cenários. Na tentativa de melhorar esses resultados, foram testadas todas as combinações dos traços manuais propostos com os *word embeddings* BERT para alimentar a BiLSTM.

6.3.1 Avaliação dos traços

Seguindo o procedimento anterior, foi feita uma avaliação em 10 rodadas para medir a influência dos traços POS-tagging, lexical e ortográficos no desempenho da BiLSTM. Ela foi treinada com cada uma das combinações de traços. Seguem na Tabela 6.8 as médias de validação dos traços para o cenário total.

Traços	Precisão	Cobertura	Medida-F
todos os traços	**73,56**	**74,19**	**73,87**
lexical	72,40	73,04	72,72
ortográficos	73,02	73,91	73,46
POS-tag	73,14	73,72	73,43
ortográficos+lexical	72,58	73,99	73,28
ortográficos+POS-tag	72,82	74,04	73,43
lexical+POS-tag	73,13	73,64	73,38

Tabela 6.8: Validação dos traços no desempenho da BiLSTM no cenário total.

A combinação de todos os traços foi a estratégia que obteve melhor desempenho na BiLSTM, com medida-F média de 73,87. O segundo melhor resultado foi com os ortográficos (73,46), seguido do POS-tagging (73,43), isto é, os traços ajustados na camada *embedding* durante o treinamento da rede. No caso do traço POS-tagging, esse resultado ressalta a vantagem em aprender representações morfossintáticas junto ao treinamento no REN.

O traço com os termos lexicais foi aquele que apresentou menor influência no desempenho da BiLSTM (72,72). De fato, esse traço pode até mesmo ter prejudicado o desempenho individual de outros, já que quando combinado com os ortográficos e o POS-tagging, o desempenho médio da BiLSTM caiu de 73,46 e 73,43 para, em ordem, 73,27 e 73,38.

Talvez isso seja porque esse é o único traço não-treinável na rede neural, o que pode ter inserido algum viés errôneo que prejudicou as predições do modelo. De qualquer modo, quando todos os traços são combinados, o desempenho do modelo sobe em 0,41 décimos em relação ao segundo melhor resultado (ortográficos), portanto a abordagem mais vantajosa.

Observando-se os resultados para o cenário seletivo, na Tabela 6.9, não são percebidas muitas diferenças entre as combinações. A combinação de todos os traços obteve medida-F média de 80,25, 0,03 acima da combinação dos traços ortográficos com os termos lexicais, com 80,22. No quadro geral, os resultados ficaram bem parecidos, mas podemos ver que os traços ortográficos parecem estar relacionados com resultados mais altos, seja sozinhos ou combinados com outros. Por sua vez, os traços POS-tagging e lexical apresentaram resultados médios um pouco abaixo dos demais.

Concluindo, a comparação entre os cenários total e seletivo revelaram uma mudança da influência dos traços. No cenário total, a combinação de todos os traços obteve os resultados mais altos, com uma distância maior para o segundo melhor caso. Já no seletivo, a diferença entre as combinações é bem pequena, com medidas equiparáveis tanto na precisão quanto na cobertura. Entre os traços, os ortográficos parecem ter tido mais peso nesse cenário. Analisando o cenário total, vemos que esse traço também aparece como um dos com melhor desempenho, o que mostra que ele teve importância nos dois cenários. Para terminar, uma possível explicação para a diferença entre os

Traços	Precisão	Cobertura	Medida-F
todos os traços	81,69	78,84	80,25
lexical	81,15	77,87	79,47
ortográficos	81,43	78,96	80,17
POS-tag	81,16	78,34	79,72
ortográficos+lexical	81,68	78,81	80,22
ortográficos+POS-tag	81,65	78,72	80,16
lexical+POS-tag	80,43	78,86	79,61

Tabela 6.9: Validação dos traços no desempenho da BiLSTM no cenário seletivo.

cenários é que os traços manuais tenham tido mais influência nas categorias menores, retiradas no cenário seletivo. Daqui por diante, adotaremos a arquitetura da BiLSTM com todos os traços (BiLSTM+traços) em contraste com a arquitetura *default* (BiLSTM).

6.3.2 Resultados dos traços na BiLSTM

Após validar a combinação de traços, focamos na comparação entre os resultados da BiLSTM na arquitetura base e da BiLSTM+traços para estabelecer se o uso de traços manuais melhorou o desempenho do modelo. Primeiramente, treinamos a BiLSTM sem e com traços e comparamos o histórico do erro e da acurácia obtidos no corpus de validação dos modelos. A Figura 6.8 contrasta o erro e a acurácia da BiLSTM e da BiLSTM+traços nessa rodada aleatória considerando o cenário total.

Em ambos os modelos, o erro começa praticamente igual. Porém, em mais ou menos 5 épocas de treinamento, eles se separam, com a BiLSTM+traços tendo um decaimento maior no erro e depois se mantendo mais estável, enquanto que na BiLSTM sem traços o erro começa a subir em mais ou menos 20 épocas. Do mesmo modo, a acurácia da BiLSTM com traços ficou mais alta do que a da BiLSTM sem traços. Assim sendo, a BiLSTM com os traços forneceu mais estabilidade no erro e melhor acurácia. Portanto, esse é um modelo com melhor desempenho no Harem.

É preciso ressaltar que as ondulações nas linhas do erro e da acurácia indicam instabilidades nas classificações, o que ocorre devido a um viés dos

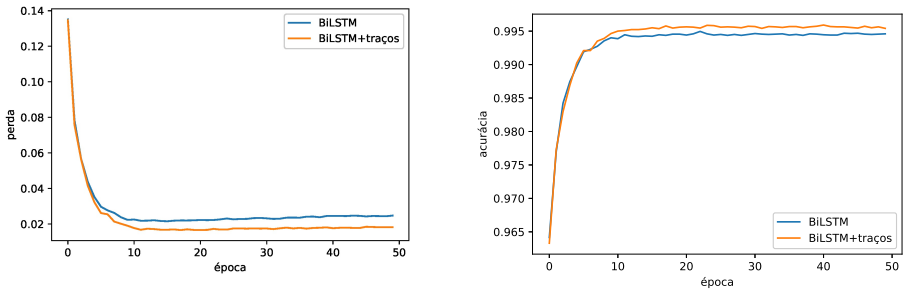


Figura 6.8: Comparação entre erro e acurácia obtido pela BiLSTM e BiLSTM+traços. À esquerda, a validação da perda; à direita, a validação da acurácia.

modelos no conjunto de treinamento. Em outras palavras, o erro no corpus de validação aumenta porque a capacidade de generalização começa a diminuir. Na etapa de avaliação, isso pode prejudicar a classificação de entidades raras, não-vistas durante o treinamento, e favorecer aquelas já vistas.

Em seguida, comparamos os resultados obtidos nas avaliações da BiLSTM e BiLSTM+traços (Tabela 6.10).

Modelo	Cenário	Precisão	Cobertura	Medida-F
BiLSTM	total	72,80	73,32	73,06
	seletivo	81,24	78,19	79,68
BiLSTM+traços	total	**73,56**	**74,19**	**73,87**
	seletivo	**81,69**	**78,84**	**80,25**

Tabela 6.10: Desempenho da BiLSTM base e com traços.

Tanto no cenário total quanto no seletivo, a BiLSTM+traços obteve desempenho mais alto, com diferenças respectivas de 0,81 e 0,57 na medida-F. A inserção de traços representou um aumento nas três medidas, tendo maior impacto na cobertura da BiLSTM, com uma diferença de 0,87, no cenário total, e 0,65, no seletivo.

Para determinar se essa diferença pode ser considerada estatisticamente significativa, aplicamos o teste U de Mann-Whitney, um teste não-paramétrico, usado quando não se sabe a distribuição específica do conjunto de dados.

A nossa hipótese nula foi a de que não havia diferença significativa entre a medida-F da BiLSTM e da BiLSTM+traços. A rejeição dessa hipótese significa que há evidências significativas para a diferença na medida-F da BiLSTM+traços.

A fim de testá-la, utilizamos as amostras obtidas na validação das redes, em relação à medida-F, nos cenários total e seletivo. Obtivemos o valor-p com a implementação de Mann-Whitney da biblioteca scikit-learn, considerando $\alpha = 0{,}05$. Os resultados do teste estão na Tabela 6.11.

Cenário	**Valor-p**
total	0,000164
seletivo	0,040991

Tabela 6.11: Resultado do Teste U de Mann-Whitney para os traços da BiLSTM.

Em ambos os casos, os resultados mostram que a diferença entre a medida-F da BiLSTM e da BiLSTM+traços é estatisticamente significativa. Portanto, podemos afirmar que os traços linguísticos trouxeram melhorias para o desempenho da rede neural.

Como último teste, apresenta-se nas Figuras 6.9 e 6.10 uma comparação da dispersão da medida-F entre a BiLSTM e a BiLSTM+traços nos cenários total e seletivo.

Pela Figura 6.9, é possível observar que a média e a mediana da BiLSTM com traços estão quase iguais, então há pouca assimetria entre os valores extremos, com a melhor rodada em 74,35 e a pior em 73,46. Já na BiLSTM, a mediana se encontra um pouco acima da média, portanto valores muito baixos fizeram a média cair. Na melhor rodada, o modelo obteve 73,48, o que é quase o mínimo obtido na BiLSTM+traços.

Já no cenário seletivo (Figura 6.10), os resultados apresentaram mais assimetria nos dois modelos. A BiLSTM+traços teve algumas rodadas com valores altos, que enviesaram a média para cima em comparação ao ponto médio da amostra (80,06). Em contraste, na BiLSTM a média ficou abaixo da mediana (79,78), puxada por extremos inferiores nos dados. Por fim, o inter-

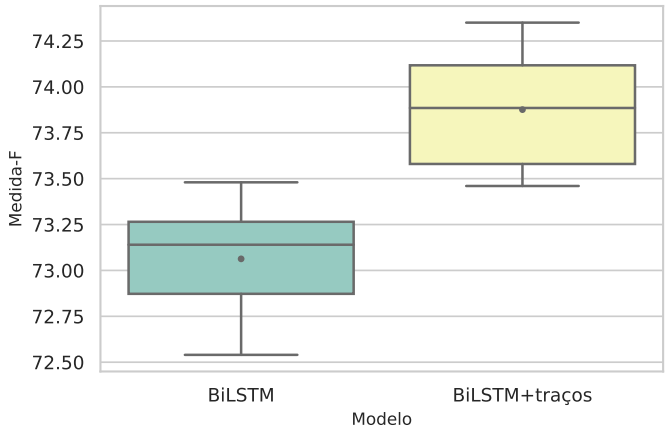


Figura 6.9: Dispersão da medida-F na BiLSTM e BiLSTM+traços para o cenário total.

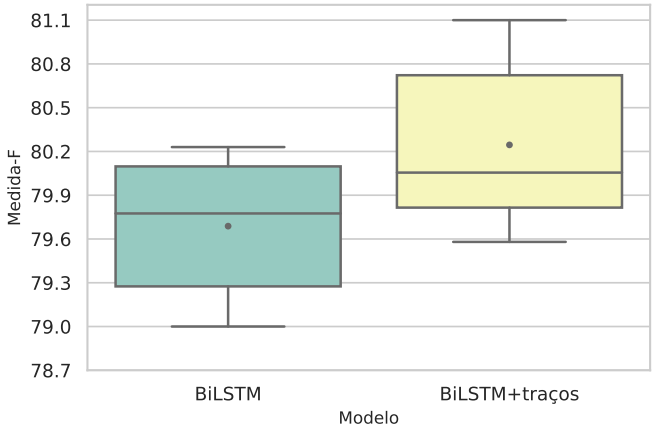


Figura 6.10: Dispersão da medida-F na BiLSTM e BiLSTM+traços para o cenário seletivo.

valo de dispersão das medidas foi pequeno, principalmente na BiLSTM+traços, mostrando maior estabilidade do modelo.

6.3.3 Análise por categorias

Seguindo o procedimento realizado no BERT, selecionamos a rodada de validação com desempenho mais próximo da média e comparamos os resultados por categoria obtidos pela BiLSTM e BiLSTM+traços na Figura 6.11.

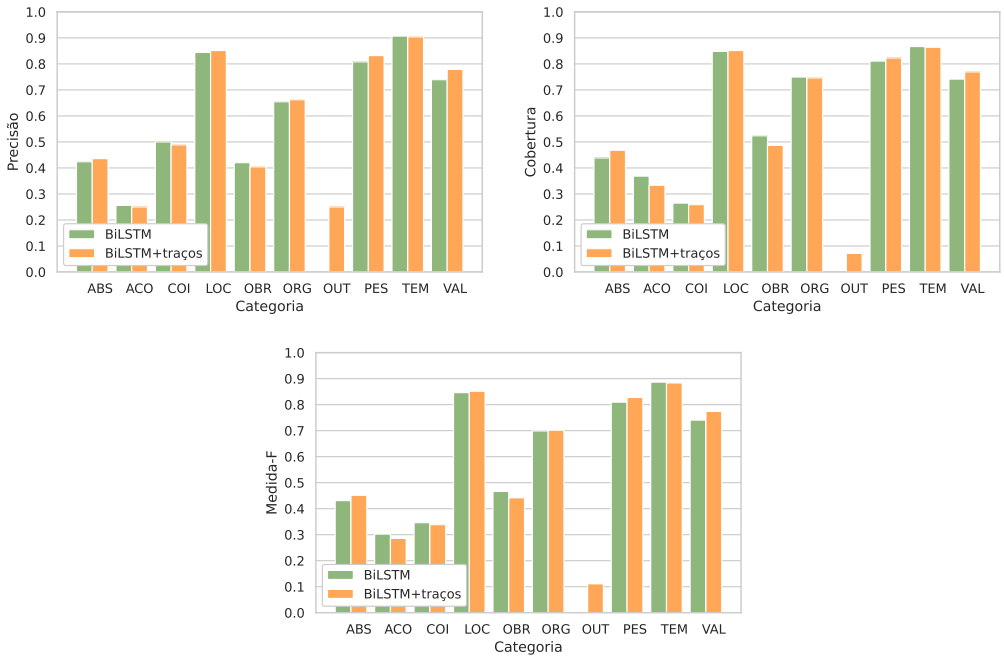


Figura 6.11: Comparação das medidas de desempenho entre BiLSTM e BiLSTM+traços no cenário total.

Do mesmo modo que no BERT, a categoria com o melhor desempenho em ambos os modelos foi Tempo, seguida de Local e Pessoa. No quadro geral, a BiLSTM+traços obteve medida-F superior em seis das dez categorias de entidade, ficando abaixo em Acontecimento, Coisa, Obra e Tempo. Considerando as categorias maiores, a diferença mais alta entre os resultados foi em Pessoa e Valor. Nas categorias menores, Abstração e Outro alcançaram uma medida-F mais alta na BiLSTM+traços, o que pode ser reflexo de melhoria obtido pelos traços manuais.

Observando, por exemplo, a categoria Outro, a medida-F na BiLSTM foi zero e na BiLSTM+traços cerca de 11%. Além disso, tanto a cobertura

quanto a precisão de Abstração foram mais altos na BiLSTM+traços. Por outro lado, Obra alcançou medidas mais altas com a BiLSTM.

Essas distinções podem ter associação com essas rodadas específicas, já que em cada rodada há variações nas medidas obtidas. Para tentar evitar esse viés, calculamos a medida-F média nas rodadas de validação para cada categoria de entidade nos dois modelos (Tabela 6.12).

Categoria	**Medida-F**	
	BiLSTM	BiLSTM+traços
ABS	44,35	45,24
ACO	28,79	29,86
COI	31,53	36,32
LOC	84,58	85,17
OBR	44,76	46,64
ORG	70,18	70,37
OUT	00,00	5,56
PES	81,17	82,04
TEM	87,41	87,21
VAL	74,51	76,43

Tabela 6.12: Comparação da média da medida-F entre os modelos BiLSTM.

As médias da validação estão bem próximas das medidas-F obtidas nas rodadas da Figura 6.11 para cada modelo, com algumas categorias acima da média e outras abaixo, nos dois casos. Na média, a medida-F em todas as categorias de entidades foi melhor com a BiLSTM+traços, com exceção de Tempo. Para citar um exemplo, a categoria Outro ficou com medida-F média de 0,0 na BiLSTM, enquanto com a inserção de traços, passou para 5,56. Outras categorias com diferenças de pelo menos 1% entre as médias foram Acontecimento, Coisa, Obra e Valor.

A mesma análise foi feita para o cenário seletivo, em que os resultados de uma única rodada de validação se encontram na Figura 6.12.

A partir dos gráficos podemos ver que nas duas BiLSTMs a precisão e a cobertura de cada categoria tiveram valores bem parecidos, isto é, não houve discrepância entre uma cobertura muito alta em contraste a uma precisão muito baixa ou vice-versa. Diferente do cenário total, neste as BiLSTMs

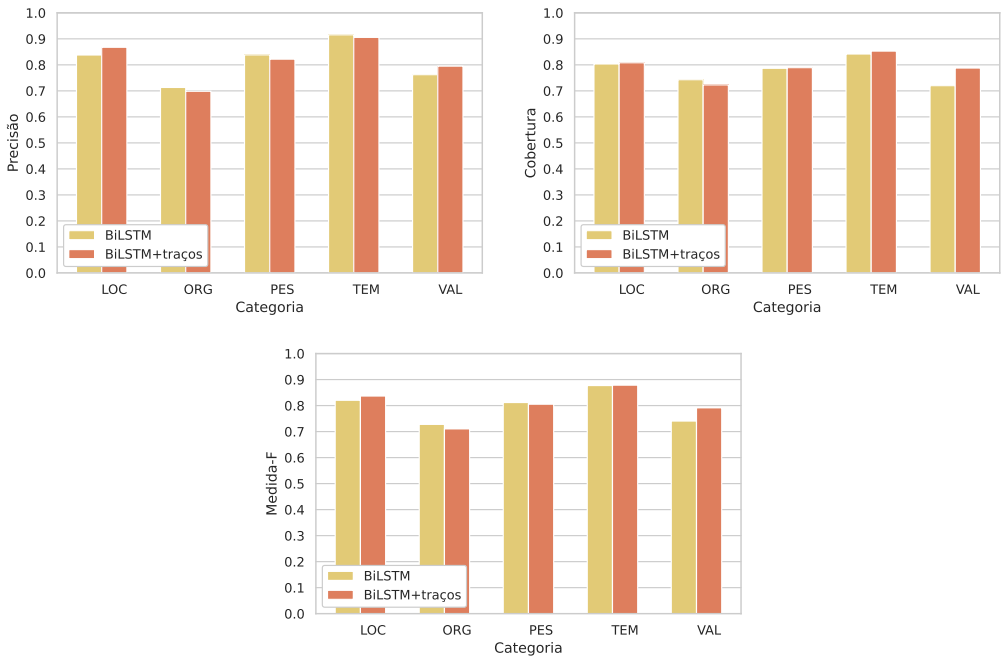


Figura 6.12: Comparação das medidas de desempenho entre BiLSTM e BiLSTM+traços no cenário seletivo.

obtiveram medidas equiparáveis em duas categorias, com menos de 1% de diferença. As distinções mais acentuadas se concentraram nas categorias Local, Organização e Valor.

Para finalizar, vemos a tendência de uma precisão maior do que a cobertura no cenário seletivo, que só transparece quando eliminamos as classes menores. O único que foge à regra é Organização. Na análise com o BERT, essa também havia sido a categoria mais difícil entre essas. Com já ressaltamos, isso pode estar associado a uma dificuldade de generalização da categoria Organização. Outro possível fator é a confusão com outras categorias, por exemplo, Local, uma vez que nomes de cidades e países, principalmente, podem ser classificados como Organização em determinados contextos.

6.3.4 Análise de erros

Nas seções anteriores, nossas análises focaram nos acertos da rede neural BiLSTM. Agora, voltamo-nos a uma breve análise sobre os erros do modelo,

com o intuito de investigar quais foram os principais deslizes e onde se concentram. Para isso, foi computada uma matriz de confusão para uma rodada aleatória no cenário total com a rede BiLSTM+traços, a Figura 6.13. A escala de cores na tabela está associada com o valor na célula, em que cores escuras indicam valores mais altos e, as claras, mais baixos. Na diagonal principal da matriz, estão os acertos para cada categoria.

Diferente das avaliações anteriores, baseadas no COnLL, as medidas nesta rodada foram obtidas com relação ao token, isto é, observando as correspondências entre etiqueta e predição para cada token em vez de por entidade, por isso os totais de ocorrências se distinguem dos apresentados anteriormente. Além disso, extraímos a anotação BIO, mantendo somente a parte da etiqueta correspondente à categoria.

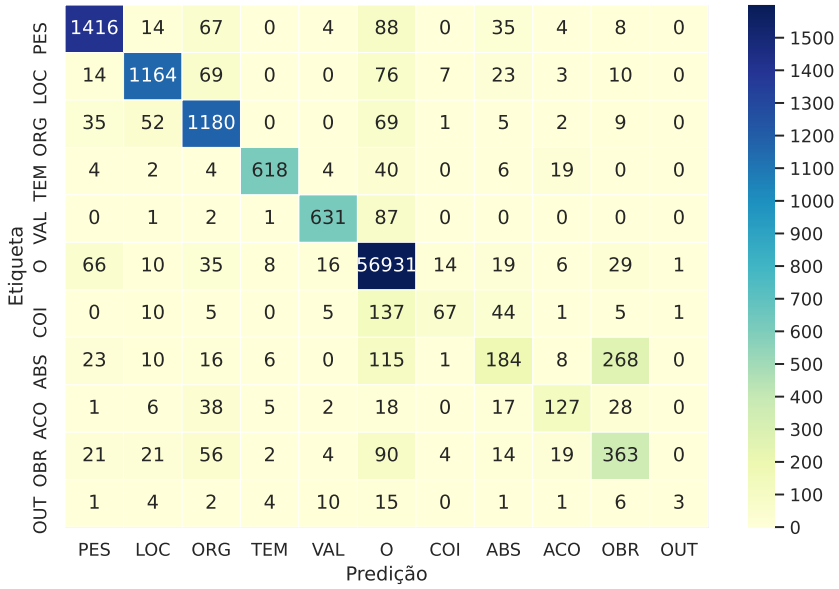


Figura 6.13: Matriz de confusão para a BiLSTM+traços.

Dos 1636 tokens de entidades do tipo Pessoa (primeira linha da matriz), a BiLSTM acertou 1416. Entre os erros, o mais frequente deles foi em *Outside* (O), a categoria de não-entidades. Se olharmos para a sexta coluna da matriz, encontramos todas as vezes em que o modelo previu um exemplo

como pertencente da categoria de não-entidades. A maioria dos erros na classificação de ENs está nessa categoria, isto é, havia uma entidade, mas o modelo a classificou como O.

Um dos casos em que esse tipo de confusão foi frequente é Coisa: dos seus 275 tokens, 137 foram classificados como não-entidade pela BiLSTM, o que representa metade do total e mais que do corresponde ao acerto, meros 67 tokens. Isso mostra que a BiLSTM nem mesmo identificou as entidades dessa categoria na maioria dos casos. Esse é um exemplo de erro do tipo Falso Negativo, isto é, o modelo previu que aqueles não eram exemplos de Coisa quando, na verdade, eram.

Uma explicação para isso é a natureza da categoria Coisa, composta de termos como: tipos de automóveis, por exemplo "Fusca", objetos celestes, como "Sol", raças de animais, como "*Golden Retriever*", ou seja, nomes usualmente tratados como genéricos, comuns, que podem ter aparecido com a classificação de não-entidades em outros contextos no corpus.

Passando para a categoria Abstração, vemos que, além da confusão com a categoria de não-entidades por 115 vezes, a BiLSTM a confundiu com Obra em 268 casos, tudo isso somando mais do que o correspondente ao acerto. Analisamos essa confusão de Abstração com Obra como fortemente associada à forma superficial, pois palavras consideradas Abstração, tal qual "Filosofia" e "Psicologia", aparecem com frequência em títulos de livros, classificados como Obra pelo Harem. O caso oposto, a troca de Obra por Abstração, é bem menos frequente. Isso indica que esse erro ocorre em contextos específicos em que há fortes indicações associadas à categoria Obra.

A confusão entre Abstração e Obra pelo modelo reflete no desempenho como uma baixa precisão da segunda categoria. O raciocínio é que o modelo previu que havia uma entidade Obra, quando na verdade tínhamos uma Abstração (erro do tipo Falso Positivo). Se retornarmos à Figura 6.11, vemos que a precisão da categoria Obra foi bem mais baixa do que a cobertura. Com os dados da matriz de confusão, sabemos que grande parte desses erros foi por causa dessa troca entre Abstração e Obra.

Por fim, já apontamos em seções anteriores sobre a dificuldade dos modelos na classificação de Organização. Com a matriz de confusão, podemos observar

mais de perto onde esses erros ocorreram. Seguindo a terceira linha da matriz, podemos ver que em 52 casos a BiLSTM classificou como Local entidades que pertenciam à Organização. O oposto ocorreu em 69 vezes, isto é, a BiLSTM classificou como organizações tokens que na verdade eram locais. Esse é um tipo de erro que já esperávamos, pois há contextos em que uma entidade pode ser ambígua entre um local e uma organização.

Abaixo são fornecidos alguns exemplos desses erros, em que a palavra em negrito representa a entidade classificada incorretamente pelo modelo.

- ... « Hora da Boa Vontade », na Rádio **Globo** do Rio de Janeiro. LOC → ORG
- ...a Alunorte, também no Pará, que é um insumo muito importante para o **Brasil**. LOC → ORG
- Pouco tempo depois de chegar a **Princeton**, Oates iniciou a obra Bellefleur. ORG → LOC
- ... após uma entrevista que Andréa fez em meados de 93 para o **Tokyo Journal**. ORG → LOC

No primeiro exemplo, "Globo" foi incorretamente classificada pela BiLSTM como Organização, quando era um Local. Esse erro é completamente justificável, já que "Globo", ou "Rádio Globo", é uma entidade típica de organização e, mesmo considerando um anotador humano, esse seria um contexto complexo. No Harem, optou-se pela sua colocação como Local por causa da posição de locativo do contexto da EN, mas esse foi um critério *ad-hoc*, já que mesmo nesse contexto caberia anotá-la como organização.

O segundo exemplo também é problemático, pois a classificação de "Brasil" como Organização é plausível nesse contexto, podendo se referir à economia do Brasil. É até surpreendente que o modelo a tenha classificado como organização em vez da classificação mais frequente como Local.

Na terceira sentença, vemos uma inconsistência na anotação comparando ao critério da primeira, porque "Princeton" está em uma posição com indicação locativa, mas é anotada pelo Harem como Organização. Sendo uma

universidade, Princeton é tipicamente uma organização, mas nesse contexto específico, pode haver dúvida na sua classificação.

Entre as três sentenças, a última é a única com a qual há poucas questões sobre a anotação de "Tokyo Journal" como Organização. Mas esse é um contexto interessante, já que temos as palavras "para o" precedentes à entidade, que é um tipo de construção que pode ser locativa, o que pode ter favorecido essa classificação. Outra possibilidade é que o modelo tenha usado um viés com a forma superficial, já que "Tokyo" pode ter ocorrido como Local.

Ainda a respeito de Organização, verifica-se que, no eixo da predição, há ocorrências de todas as categorias sendo preditas como organização, a categoria mais assinalada depois de não-entidades. Isso explica o porquê da precisão mais baixa nessa categoria e fornece evidências sobre a sua heterogeneidade, tendo em vista que, no momento em que um modelo começa a anotar qualquer categoria como Organização, sabemos que há evidências indicando essas predições, caso contrário ele tenderia a concentrar a sua probabilidade somente na mais frequente, por exemplo O. Em outras palavras, a categoria Organização é muito heterogênea e abrangente, o que a torna difícil de classificar, mesmo com muitos exemplos de treinamento.

Com essa discussão, foi possível observar mais de perto os padrões de erros cometidos pela BiLSTM+traços, visualizando algumas das trocas de classificação frequentes. Para o conjunto de categorias, um erro muito comum foi a classificação como não-entidade, o que consiste em uma falha na identificação. Li et al. (2020) chamam a atenção para esse problema de identificação, reforçando que é necessário estabelecer uma etapa precedente dedicada somente para a detecção de entidades, ignorando a classificação, para que os modelos se tornem mais robustos na classificação.

Analisando casos particulares de erros em Organização, vimos que contextos complexos geraram ambiguidade na classificação da entidade. Esse foi um problema que nos chamou a atenção, já que em alguns exemplos encontramos divergências de anotação no Harem. Esse tipo de questão é sempre uma preocupação quando se trabalha com um corpus anotado, já que exemplos com anotação inconsistente podem levar os modelos a cometer erros. Desse

modo, é possível atribuir alguns dos erros da BiLSTM aos contextos com ambiguidade de classificação.

6.4 Comparação de resultados

Após apresentar os resultados de cada um dos modelos testados, dedicamos esta seção à comparação com as pesquisas em redes neurais já discutidas no Capítulo 3. Retomamos os resultados fornecidos anteriormente junto aos nossos nas Tabelas 6.13 (cenário total) e 6.14 (cenário seletivo).

sistema	**precisão**	**cobertura**	**medida-F**
CharWNN (SANTOS; GUIMARAES, 2015)	67,16	63,74	65,41
BiLSTM-CNN (FERNANDES; CARDOSO; OLIVEIRA, 2018)	72,64	67,50	69,97
BiLSTM-CRF (CASTRO; SILVA; SILVA SOARES, 2018)	72,78	68,03	70,33
BiLSTM-CRF+FlairBBP (SANTOS; CONSOLI et al., 2019)	74,91	74,37	74,64
BERT$_{PT-LARGE}$-CRF (SOUZA; NOGUEIRA; LOTUFO, 2020)	**80,08**	**77,31**	**78,67**
BERT-ML	72,29	72,85	72,41
BERT-PT	74,11	76,35	75,22
BiLSTM	72,80	73,32	73,06
BiLSTM+traços	73,56	74,19	73,87

Tabela 6.13: Comparação de desempenhos no cenário total.

Os melhores resultados do Harem foram alcançados com o trabalho de Souza, Nogueira e Lotufo (2020), utilizando uma arquitetura BERT-PT$_{LARGE}$ com uma camada de saída CRF para decodificação das predições. Os autores alcançam o estado-da-arte tanto no primeiro quanto no segundo cenário. Antes do BERT, os modelos com mais êxito no corpus eram BiLSTMs.

Em relação aos modelos implementados nesta pesquisa, vemos que o BERT-PT também superou os demais. No cenário total, a diferença de

sistema	precisão	cobertura	medida-F
CharWNN (SANTOS; GUIMARAES, 2015)	73,98	68,68	71,23
BiLSTM-CNN (FERNANDES; CARDOSO; OLIVEIRA, 2018)	70,67	66,35	68,44
BiLSTM-CRF (CASTRO; SILVA; SILVA SOARES, 2018)	78,26	74,393	76,27
BiLSTM-CRF+FlairBBP (SANTOS; CONSOLI et al., 2019)	83,38	81,17	82,26
BERT$_{PT-LARGE}$-CRF (SOUZA; NOGUEIRA; LOTUFO, 2020)	**84,82**	**81,72**	**83,24**
BERT-ML	78,16	78,18	78,17
BERT-PT	80,97	81,23	81,09
BiLSTM	81,24	78,19	79,68
BiLSTM+traços	81,69	78,84	80,25

Tabela 6.14: Comparação de desempenhos no cenário seletivo.

medida-F entre o BERT-PT e a BiLSTM+traços, segundo melhor modelo, é de 1,35. Essa diferença é menor no cenário seletivo, com a BiLSTM+traços bem próxima do BERT-PT. De fato, se observarmos a precisão da BiLSTM e da BILSTM+traços, vemos que ambas superam o BERT-PT. É na cobertura que as arquiteturas diferem mais.

Ademais, a nossa BiLSTM, tanto com quanto sem os traços linguísticos, se saiu melhor do que o BERT-ML nos dois cenários. Portanto, mesmo que o *fine-tuning* tenha se mostrado uma melhor abordagem, uma rede neural BiLSTM com representações contextuais pode ser capaz de superar os seus resultados.

Comparando a BiLSTM+traços com a BiLSTM-CRF+FlairBBP de Santos, Consoli et al. (2019), temos resultados semelhantes no cenário total, com 73,87 e 74,64, respectivamente, uma pequena diferença para baixo no nosso modelo. Na arquitetura dos autores, eles usam uma camada de saída CRF, assim como Souza, Nogueira e Lotufo (2020), que é um decodificador

mais poderoso do que a camada *softmax*, porque computa as probabilidades condicionadas de associação entre as etiquetas em uma sequência.

Já comparada à BiLSTM-CRF de Castro, Silva e Silva Soares (2018), a nossa BiLSTM obtém um desempenho melhor nos dois cenários. Portanto, mais relevante do que a camada de saída parece ser o tipo de representação de palavra adotado. Os autores utilizam *word embeddings* clássicos, que se mostram piores do que o BERT, ainda mais em uma tarefa dependente de contexto, como é o REN.

A partir desses resultados, é possível dizer que o BERT-PT, atualmente, é o melhor modelo para o REN testado no Harem. Mesmo com a arquitetura BASE e uma camada de saída *softmax*, o modelo chega a resultados promissores. Finalmente, trouxemos um quadro geral de trabalhos recentes no REN do português com redes neurais, comparando as arquiteturas mais adotadas e os resultados considerados o estado-da-arte com aqueles reportados em outras pesquisas, incluindo esta.

7 | Conclusões

O Reconhecimento de Entidades Nomeadas é uma tarefa difícil de ser resolvida, pois envolve o conhecimento sobre o funcionamento de uma língua, tal como a morfologia e a sintaxe, assim como o conhecimento acerca do mundo. Do ponto de vista linguístico, o REN está fortemente associado à identificação de propriedades morfossintáticas, como a concordância verbal, e de usos de palavras num texto, o que não é fácil de representar computacionalmente. Neste capítulo, apresentamos as principais conclusões sobre a tarefa de REN com base nos resultados obtidos durante esta pesquisa.

7.1 Principais contribuições desta pesquisa

Inicialmente, começamos a investigar determinados contextos linguísticos que favoreciam a ocorrência de um tipo de entidade em oposição aos demais, como os discutidos no Capítulo 2. Apesar de esse ser um critério bem preciso para a classificação de entidades, a utilização de um modelo baseado somente em contextos regrados mostrou-se muito restritiva, capturando poucos exemplos no corpus.

Por causa disso, optamos por tentar duas abordagens baseadas em aprendizado de máquina: BERT e BiLSTM. Na primeira etapa, o BERT-ML e o BERT-PT foram treinados no Harem na abordagem *fine-tuning*. Entre eles, o BERT-PT obteve melhores resultados, indicando que, em tarefas voltadas para uma língua específica, um modelo de *word embeddings* treinado nessa mesma língua obtém resultados mais acurados. Uma explicação para isso

é que os tokens de outras línguas contidos no inventário do BERT-ML não demonstram o mesmo poder preditivo para o REN em português.

Já na abordagem *feature-based*, foi testada uma BiLSTM com *word embeddings* do BERT. A principal desvantagem da BiLSTM em relação ao BERT é que os *word embeddings* não foram refinados para a tarefa específica de REN, diferente da abordagem *fine-tuning*, o que se reflete na queda de desempenho do modelo. Com isso, conclui-se que a generalização de representações *word embeddings* extraídas de outro domínio não funciona tão bem quando aplicada ao Harem. Talvez isso ocorra porque o corpus tem muitas entidades raras, não vistas na etapa de pré-treinamento do BERT, o que implica em representações que não captam as características dessas entidades.

Para obter representações mais acuradas, os *word embeddings* foram combinados a traços linguísticos, sendo que dois deles foram otimizados durante o treinamento da BiLSTM. Com a inserção dos traços, a medida-F da rede aumentou em relação ao modelo sem traços, alcançando uma diferença estatisticamente significativa. Esse resultado mostra que é possível utilizar conhecimento linguístico não explicitamente codificado no modelo para gerar representações que captam diferentes aspectos das palavras, complementando as representações *word embeddings* do BERT. Dessa forma, comprova-se que mesmo um modelo de rede neural de reconhecido poder nas tarefas de classificação textual pode se beneficiar da inserção de traços manuais.

Entre os traços linguísticos testados, aqueles treinados na rede neural BiLSTM (POS-tagging e ortográficos) tiveram maior impacto no desempenho, reforçando que representações treinadas no domínio específico da tarefa são mais eficientes. Já quanto aos traços lexicais, quando aplicados isoladamente, os resultados não foram tão bons. Uma de suas desvantagens por relação aos demais traços é o pequeno número de exemplos nas listas de classificadores, o que restringe o mapeamento das entidades contidas no Harem. Em adição a isso, os traços lexicais só foram aplicados a algumas categorias de entidades, o que se reflete em sua baixa cobertura.

Apesar da melhoria alcançada com a inserção dos traços linguísticos, a BiLSTM não superou o desempenho obtido pelo BERT-PT. Apontamos como sua maior limitação a falta de treinamento dos *word embeddings* na

tarefa de REN, uma vez que eles eram a principal forma de representação de palavra usada pelo modelo. Desse modo, a inserção dos traços linguísticos não foi suficiente para ajudar na classificação de determinadas entidades do Harem, provavelmente aquelas cuja representação *word embeding* já era pouco acurada.

Dos resultados gerais obtidos, observou-se que há diferenças significativas de desempenho de um mesmo modelo quando treinado com 5 ou com 10 categorias de entidades. Atribui-se essa diferença, em parte, aos poucos exemplos de treinamento das categorias menores (Abstração, Acontecimento, Obra, Coisa e Outro). Por sua vez, a análise de erros mostrou que houve confusão na classificação entre essas categorias, o que indica que elas ocorrem em contextos semelhantes, que não apresentam muitas informações relevantes para discernir entre uma categoria e outra.

Ademais, as coleções douradas do Harem são pequenas, contendo cerca de 180.000 palavras somando os corpora de treino e teste, o que foi um fator limitante nesta pesquisa, já que os modelos de redes neurais requerem grandes quantidades de dados de treinamento para obter seu melhor desempenho. Outra questão persistente sobre o Harem são determinados tipos de variações de anotação frequentes. A anotação manual de um corpus é sempre um trabalho árduo e requer uma equipe dedicada à tarefa. Por isso, é normal encontrar inconsistências entre os anotadores. Tais inconsistências, por sua vez, podem prejudicar o desempenho de avaliação dos modelos, ainda mais quando há tão poucos exemplos.

Por fim, pudemos testar a nossa hipótese inicial a respeito da distribuição contextual como um critério para classificação de entidades. Com os testes de associação por similaridade entre os *word embeddings* do BERT, obtivemos diversos aglomerados que correspondiam a categorias de entidades específicas, como nomes de pessoas e de lugares. Os bons resultados obtidos por nossos modelos também apontam para o poder das representações de traços baseadas em contexto para a classificação de entidades.

Todos esses elementos corroboram a hipótese da distribuição contextual como um fator importante no REN. Assim, fica claro o porquê de os modelos de *word embeddings* terem ficado tão populares na área de PLN, tendo em

vista sua capacidade de extração e representação de padrões contextuais dos dados que seres humanos teriam muita dificuldade de pré-estabelecer de antemão. Seria necessária uma análise minuciosa e muito tempo para identificar os padrões extraídos em horas por essas redes neurais, o que não significa a substituição do trabalho humano, mas uma demonstração de que esta é uma ferramenta de auxílio para que nós, pesquisadores, possamos nos concentrar no que é mais importante, isto é, a interpretação e análise linguística desses resultados.

7.2 Encaminhamentos Futuros

As pesquisas com redes neurais têm sempre procurado novas técnicas de treinamento voltadas a melhorar esses modelos. Nesse sentido, uma das possibilidades para pesquisas futuras seria avaliar a inserção de um traço lexical baseado em *word embeddings* de entidades nomeadas extraídos a partir de fontes externas (por exemplo, a Wikipédia), como proposto por Ghaddar e Langlais (2018), e testar tal traço na arquitetura da BiLSTM.

Outro caminho promissor seria o treinamento de *word embeddings* do BERT combinados a traços linguísticos para a tarefa de REN. Recentemente, Souza, Nogueira e Lotufo (2020) disponibilizaram em sua conta do GitHub[1] os códigos de implementação de seus modelos, o que pode facilitar a reimplementação do BERT com traços manuais a partir dos códigos fornecidos pelos autores.

[1] https://github.com/neuralmind-ai/portuguese-bert

Referências

AKBIK, Alan; BLYTHE, Duncan; VOLLGRAF, Roland. Contextual string embeddings for sequence labeling. In: PROCEEDINGS of the 27th International Conference on Computational Linguistics. 2018. p. 1638–1649.

ALUISIO, Sandra et al. An account of the challenge of tagging a reference corpus for brazilian portuguese. In: SPRINGER. INTERNATIONAL Workshop on Computational Processing of the Portuguese Language. 2003. p. 110–117.

AMARAL, Carlos et al. Adaptação do sistema de reconhecimento de entidades mencionadas da Priberam ao HAREM. *Desafios na avaliação conjunta do reconhecimento de entidades mencionadas: O Segundo HAREM. Linguateca*, 2008.

AMARAL, Daniela Oliveira Ferreira do et al. O reconhecimento de entidades nomeadas por meio de conditional random fields para a língua portuguesa. Pontifícia Universidade Católica do Rio Grande do Sul, 2013.

ARAUJO, Pedro H Luz de; TEÓFILO, E. de Campos, Renato RR de Oliveira, Matheus Stauffer, Samuel Couto, and Paulo Bermejo. Lener-br: a dataset for named entity recognition in brazilian legal text. In: INTERNATIONAL Conference on the Computational Processing of Portuguese (PROPOR), Canela, RS, Brazil. 2018.

AUGENSTEIN, Isabelle; DERCZYNSKI, Leon; BONTCHEVA, Kalina. Generalisation in named entity recognition: A quantitative analysis. *Computer Speech & Language*, Elsevier, v. 44, p. 61–83, 2017.

BENGIO, Yoshua et al. A neural probabilistic language model. *Journal of machine learning research*, v. 3, Feb, p. 1137–1155, 2003.

BICK, Eckhard. Functional aspects in portuguese ner. In: SPRINGER. INTERNATIONAL Workshop on Computational Processing of the Portuguese Language. 2006. p. 80–89.

BOJANOWSKI, Piotr et al. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, MIT Press, v. 5, p. 135–146, 2017.

CARDOSO, Nuno. Rembrandt-reconhecimento de entidades mencionadas baseado em relaçoes e análise detalhada do texto. *quot; Encontro do Segundo HAREM (Universidade de Aveiro Portugal 7 de Setembro de 2008)*, 2008.

CASTRO, Pedro Vitor Quinta de; SILVA, Nádia Félix Felipe da; SILVA SOARES, Anderson da. Portuguese named entity recognition using lstm-crf. In: SPRINGER. INTERNATIONAL Conference on Computational Processing of the Portuguese Language. 2018. p. 83–92.

CHIU, Jason PC; NICHOLS, Eric. Named entity recognition with bidirectional LSTM-CNNs. *Transactions of the Association for Computational Linguistics*, MIT Press, v. 4, p. 357–370, 2016.

COLLOBERT, Ronan et al. Natural language processing (almost) from scratch. *Journal of machine learning research*, v. 12, Aug, p. 2493–2537, 2011.

DERCZYNSKI, Leon et al. Results of the WNUT2017 shared task on novel and emerging entity recognition. In: PROCEEDINGS of the 3rd Workshop on Noisy User-generated Text. 2017. p. 140–147.

DEVLIN, Jacob et al. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

FERNANDES, Ivo; CARDOSO, Henrique Lopes; OLIVEIRA, Eugenio. Applying Deep Neural Networks to Named Entity Recognition in Portuguese Texts. In: IEEE. 2018 Fifth International Conference on Social Networks Analysis, Management and Security (SNAMS). 2018. p. 284–289.

FIRTH, John R. A synopsis of linguistic theory, 1930-1955. *Studies in linguistic analysis*, Basil Blackwell, 1957.

FONSECA, Erick; ROSA, João Luís G. Mac-morpho revisited: Towards robust part-of-speech tagging. In: PROCEEDINGS of the 9th Brazilian symposium in information and human language technology. 2013.

GHADDAR, Abbas; LANGLAIS, Philippe. Robust lexical features for improved neural network named-entity recognition. *arXiv preprint arXiv:1806.03489*, 2018.

GRISHMAN, Ralph; SUNDHEIM, Beth. Message understanding conference-6: A brief history. In: COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics. 1996. v. 1.

HARRIS, Zellig S. Distributional structure. *Word*, Taylor & Francis, v. 10, n. 2-3, p. 146–162, 1954.

HARTMANN, Nathan et al. Portuguese word embeddings: Evaluating on word analogies and natural language tasks. *arXiv preprint arXiv:1708.06025*, 2017.

HINTON, Geoffrey E et al. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

HOCHREITER, Sepp; SCHMIDHUBER, Jürgen. Long short-term memory. *Neural computation*, MIT Press, v. 9, n. 8, p. 1735–1780, 1997.

INDURKHYA, Nitin; DAMERAU, Fred J. *Handbook of natural language processing*. CRC Press, 2010. v. 2.

JAKOBSON, Roman. Dois aspectos da linguagem e dois tipos de afasia. *Lingüística e comunicação*, v. 13, 1954.

JIANG, Ridong; BANCHS, Rafael E; LI, Haizhou. Evaluating and combining name entity recognition systems. In: PROCEEDINGS of the Sixth Named Entity Workshop. 2016. p. 21–27.

JÚNIOR, Carlos Mendonça et al. Paramopama: a brazilian-portuguese corpus for named entity recognition. *Encontro Nac. de Int. Artificial e Computacional*, 2015.

JURAFSKY, Daniel; MARTIN, James H. *Speech and Language processing (draft)*. 2018. v. 19, p. 2019.

KINGMA, Diederik P; BA, Jimmy. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

KRIPKE, Saul. *Naming and Necessity*. Boston: Harvard University Press, 1982.

KULLBACK, Solomon; LEIBLER, Richard A. On information and sufficiency. *The annals of mathematical statistics*, JSTOR, v. 22, n. 1, p. 79–86, 1951.

LI, Jing et al. A survey on deep learning for named entity recognition. *IEEE Transactions on Knowledge and Data Engineering*, IEEE, 2020.

LIU, Tianyu; YAO, Jin-Ge; LIN, Chin-Yew. Towards improving neural named entity recognition with gazetteers. In: PROCEEDINGS of the 57th Annual Meeting of the Association for Computational Linguistics. 2019. p. 5301–5307.

LOSHCHILOV, Ilya; HUTTER, Frank. *Decoupled Weight Decay Regularization*. 2019. arXiv: 1711.05101 [cs.LG].

MAATEN, Laurens van der; HINTON, Geoffrey. Visualizing data using t-SNE. *Journal of machine learning research*, v. 9, Nov, p. 2579–2605, 2008.

MACQUEEN, James et al. Some methods for classification and analysis of multivariate observations. In: OAKLAND, CA, USA, 14. PROCEEDINGS of the fifth Berkeley symposium on mathematical statistics and probability. 1967. v. 1, p. 281–297.

MARRERO, Mónica et al. Named entity recognition: fallacies, challenges and opportunities. *Computer Standards & Interfaces*, Elsevier, v. 35, n. 5, p. 482–489, 2013.

MIKOLOV, Tomas et al. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

MILIDIÚ, Ruy Luiz; SANTOS, Cícero Nogueira dos; DUARTE, Julio Cesar. Portuguese corpus-based learning using ETL. *Journal of the Brazilian Computer Society*, Springer, v. 14, n. 4, p. 17–27, 2008.

MOTA, Cristina; SANTOS, Diana. *Desafios na avaliação conjunta do reconhecimento de entidades mencionadas: O Segundo HAREM*. Linguateca, 2008.

MURPHY, Kevin P. *Machine learning: a probabilistic perspective*. MIT press, 2012.

NADEAU, David; SEKINE, Satoshi. A survey of named entity recognition and classification. *Linguisticae Investigationes*, John Benjamins publishing company, v. 30, n. 1, p. 3–26, 2007.

NOTHMAN, Joel et al. Learning multilingual named entity recognition from Wikipedia. *Artificial Intelligence*, Elsevier, v. 194, p. 151–175, 2013.

PEDREGOSA, F. et al. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, v. 12, p. 2825–2830, 2011.

PENNINGTON, Jeffrey; SOCHER, Richard; MANNING, Christopher D. Glove: Global vectors for word representation. In: PROCEEDINGS of the 2014 conference on empirical methods in natural language processing (EMNLP). 2014. p. 1532–1543.

PETERS, Matthew E et al. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*, 2018.

PIROVANI, Juliana PC; OLIVEIRA, Elias de. CRF+ LG: A hybrid approach for the portuguese named entity recognition. In: SPRINGER. INTERNATIONAL Conference on Intelligent Systems Design and Applications. 2017. p. 102–113.

RADFORD, Alec et al. *Improving language understanding by generative pre-training*. 2018.

RUMELHART, David E; HINTON, Geoffrey E; WILLIAMS, Ronald J. *Learning internal representations by error propagation*. 1985.

RUSSELL, Stuart J; NORVIG, Peter. *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited, 2016.

SANTOS, Cicero Nogueira dos; GUIMARAES, Victor. Boosting named entity recognition with neural character embeddings. *arXiv preprint arXiv:1505.05008*, 2015.

SANTOS, Diana; CARDOSO, Nuno. A golden resource for named entity recognition in Portuguese. In: SPRINGER. INTERNATIONAL Workshop on Computational Processing of the Portuguese Language. 2006. p. 69–79.

_____. *Reconhecimento de entidades mencionadas em português: Documentação e actas do HAREM, a primeira avaliação conjunta na área.* 2007.

SANTOS, Joaquim; CONSOLI, Bernardo et al. Assessing the Impact of Contextual Embeddings for Portuguese Named Entity Recognition. In: IEEE. 2019 8th Brazilian Conference on Intelligent Systems (BRACIS). 2019. p. 437–442.

SARMENTO, Luís. O SIEMÊS e a sua participação no HAREM e no Mini-HAREM. *quot; In Diana Santos; Nuno Cardoso (ed) Reconhecimento de entidades mencionadas em português: Documentação e actas do HAREM a primeira avaliação conjunta na área Linguateca 2007*, 2007.

SEKINE, Satoshi; NOBATA, Chikashi. Definition, Dictionaries and Tagger for Extended Named Entity Hierarchy. In: LISBON, PORTUGAL. LREC. 2004. p. 1977–1980.

SEOK, Miran et al. Named entity recognition using word embedding as a feature. *Int. J. Softw. Eng. Appl*, v. 10, n. 2, p. 93–104, 2016.

SONG, Chan Hee et al. Improving neural named entity recognition with gazetteers. *arXiv preprint arXiv:2003.03072*, 2020.

SOUZA, Fábio; NOGUEIRA, Rodrigo; LOTUFO, Roberto. BERTimbau: pretrained BERT models for Brazilian Portuguese. In: 9TH Brazilian Conference on Intelligent Systems, BRACIS, Rio Grande do Sul, Brazil, October 20-23 (to appear). 2020.

TJONG KIM SANG, Erik F. Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In: PROC. Conference on Natural Language Learning. 2002.

TJONG KIM SANG, Erik F; DE MEULDER, Fien. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. PROCEEDINGS of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4. 2003. p. 142–147.

TURIAN, Joseph; RATINOV, Lev; BENGIO, Yoshua. Word representations: a simple and general method for semi-supervised learning. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. PROCEEDINGS of the 48th annual meeting of the association for computational linguistics. 2010. p. 384–394.

VASWANI, Ashish et al. Attention is all you need. In: ADVANCES in neural information processing systems. 2017. p. 5998–6008.

VITERBI, Andrew. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE transactions on Information Theory*, IEEE, v. 13, n. 2, p. 260–269, 1967.

WAGNER FILHO, Jorge A et al. The brwac corpus: A new open resource for brazilian portuguese. In: PROCEEDINGS of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018). 2018.

WU, Yonghui et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

YAMADA, Ikuya et al. LUKE: Deep Contextualized Entity Representations with Entity-aware Self-attention. *arXiv preprint arXiv:2010.01057*, 2020.

Apêndices

A | Tabelas de contextos

entidade	contexto
Santos-1	Seus primeiros diretores foram os então acadêmicos Wladimir da Prússia Gomes Ferraz (Presidente), Arulemo Santos Novaes e Jair Xavier Guimarães (Secretários).
Santos-2	E ele se alimentava bem, ele não bebia, ele não fumava, ele ginasticava, ele fazia excursões, nos levava para o Pico do Jaraguá, para Guarapiranga, que naquele tempo não era nada, para Santos;
Santos-3	Na parte de baixo tinha uma capela, que depois tiraram, e aqui em cima uma parede alta que já foi o Santos Silva que a deitou abaixo.
Santos-4	Foi uma insignificante mariposa, que batia as asas perto da boate , a pessoa que informou: Lu fora vista em Santos, devia estar em Santos, morar lá.
Santos-5	Foi uma insignificante mariposa, que batia as asas perto da boate, a pessoa que informou: Lu fora vista em Santos, devia estar em Santos, morar lá.
Santos-6	Este reitor ainda foi da comissão instaladora, assim como o Machado dos Santos.
Santos-7	Eu falo sinto bem, eu falo, acho gostoso de lembrar, as minhas filhas falam para mim: "Porque você não vai para o Silvio Santos e pede na Porta da Esperança?

Santos-8	E então, acabou por ir o Machado dos Santos e João de Deus Pinheiro, que foram reitores a seguir, foram os meus vice-reitores, precisamente porque eu não sou engenheiro, era preciso construir, é preciso saber os planos.
Santos-9	E depois o grande desenvolvimento foi com o professor Machado dos Santos, o João de Deus esteve pouco tempo, foi para ministro da Educação, e o Machado dos Santos foi nomeado reitor.
Santos-10	E depois o grande desenvolvimento foi com o professor Machado dos Santos, o João de Deus esteve pouco tempo, foi para ministro da Educação, e o Machado dos Santos foi nomeado reitor.
Santos-11	Éramos estes, depois veio o Santos Simões, que é o que está à frente da universidade em Guimarães e que foi substituir o Freitas do Amaral.
Santos-12	Mas foi uma sorte, ainda hoje somos muito amigos com o Santos Simões, ainda estamos em relação com ele e agora vai-se fundar uma unidade cultural em Guimarães, em Monção e em Guimarães.
Santos-13	R- Eu morava na Brigadeiro Luís Antônio, quase esquina com a Alameda Santos.

Tabela A.1: Contextos de ocorrência de entidade Santos.

B | Treinamento do CRF

A codificação do traço lexical foi feita com o auxílio de um classificador para identificação das Entidades Nomeadas. Para isso, utilizou-se um algorítimo *Conditional Random Fields* (CRF). O CRF é um modelo de aprendizado supervisionado que computa a probabilidade da etiqueta no passo t em relação a um conjunto de traços relevantes. Esse conjunto pode conter traços de palavras antecedentes ou futuras na sequência de dados.

Um dos motivos pelo qual o CRF é muito utilizado em tarefas sequenciais é que a predição do modelo é calculada sobre a probabilidade de uma sequência de etiquetas, em vez de ser calculada sobre cada etiqueta individualmente. Na etapa de decodificação do algorítimo, inúmeras (ou todas) as combinações de sequências de etiquetas são calculadas para obter a melhor sequência. Esse processo é feito pelo algorítimo de Viterbi (VITERBI, 1967).

B.1 Implementação

Uma série de traços foi considerada para a implementação do CRF. Os traços selecionados no modelo proposto estão apresentados na Tabela B.1. Foram considerados traços da palavra atual (p_i), da precedente (p_{i-1}) e da seguinte (p_{i+1}). As letras V e F na Tabela B.1 são abreviações para "Verdadeiro" e "Falso", respectivamente.

Os traços ortográficos (ALFA, NUM, NON-ALNUM, 1UP, UP, LOW, SPEC) são extraídos somente da palavra atual na sequência. Entre eles, o traço SPEC se mostrou relevante dado que os símbolos nesse conjunto estão fortemente associados com a ocorrência de ENs. Os traços p_i, p_{i-1} e

traço	descrição
p_i	a palavra atual
p_i-ALFA	V se a palavra atual é alfabética; F, caso contrário
p_i-NUM	V se a palavra atual é numérica; F, caso contrário
p_i-NON-ALNUM	V se a palavra atual não é alfanumérica; F, caso contrário
p_i-1UP	V se a primeira letra da palavra atual é maiúscula; F, caso contrário
p_i-UP	V se a palavra atual é maiúscula; F, caso contrário
p_i-LOW	V se a palavra atual é minúscula; F, caso contrário
p_i-SPEC	V se a palavra atual pertence ao conjunto {&, \$, ª,º,%}; F, caso contrário
p_i-POS	a etiqueta morfossintática da palavra atual
p_{i-1}-POS	a etiqueta morfossintática da palavra precedente
p_{i+1}-POS	a etiqueta morfossintática da palavra seguinte
p_{i-1}	a palavra no passo precedente
p_{i+1}	a palavra no passo seguinte

Tabela B.1: Traços usados no classificador CRF.

p_{i+1} correspondem a palavra atual, palavra precedente e palavra seguinte na sequência normalizadas em minúsculo, por exemplo "Brasil" → "brasil" e "ABC" → "abc". Para a extração dos traços morfossintáticos, indicados por POS, usou-se a nlpnet (FONSECA; ROSA, 2013).

O CRF foi treinado no Primeiro Harem e avaliado no Mini Harem, ambos convertidos para classificação binária, distinguindo entidades (EN) de não-entidades (O). Para cada token nos documentos, foram extraídos todos os traços descritos na Tabela B.1. O CRF foi implementado a partir do algorítimo da biblioteca scikit-learn-crf-suite[1], com os parâmetros *default* do modelo, exceto pelo coeficiente de regularização L_1 e L_2, que ficaram com $0,1$ e $0,1$, e o número de iterações, definido em 100.

[1] https://sklearn-crfsuite.readthedocs.io/en/latest/

B.2 Resultados

A Tabela B.2 traz os resultados do CRF no Mini Harem. A última coluna da tabela se refere ao número de tokens na categoria.

etiqueta	**precisão**	**cobertura**	**medida-F**	**referência**
EN	89,90	86,20	88,01	7563
O	98,18	98,72	98,45	57135
média	94,04	92,46	93,23	64698

Tabela B.2: Desempenho do CRF no Mini Harem.

A medida-F para a identificação de entidades ficou em $88,01$, enquanto a de não-entidades alcançou $98,45$. Na categoria EN, a precisão ficou mais alta do que a cobertura, em outras palavras, o modelo foi melhor na classificação de entidades, tendo mais dificuldades em recuperar alguns exemplos, classificando-os como não-entidades. É possível que isso tenha ocorrido nos casos de palavras comuns que são ENs, tais como os conectivos, por exemplo "'de" e "e", assim como "filosofia" e "estado" presentes em ENs. Os erros cometidos pelo modelo envolvendo alguns conectivos estão na Tabela B.3.

conetivo	**erros na classificação**	
	EN	O
e	32	27
de	60	25
do	17	11
da	16	8
total	125	71

Tabela B.3: Erros para alguns conectivos comuns em entidades.

Os dados apontam que o CRF cometeu mais erros classificando esses conetivos no caso de entidades (125) do que não-entidades (71). Como a proporção de tokens anotados como ENs representa somente cerca de 11,7% do corpus, esses erros tem um peso maior que para a categoria entidades.