Universidade Federal da Bahia
Instituto de Matemática e Estatística

Bacharelado em Ciência da Computação

RECONHECIMENTO DE ENTIDADES NOMEADAS EM DOMÍNIOS ESPECÍFICOS ATRAVÉS DE ALINHAMENTO DE SUBESPAÇOS

Victor Soares Cardel

TRABALHO DE GRADUAÇÃO

Salvador
junho de 2021

VICTOR SOARES CARDEL

RECONHECIMENTO DE ENTIDADES NOMEADAS EM DOMÍNIOS ESPECÍFICOS ATRAVÉS DE ALINHAMENTO DE SUBESPAÇOS

Este Trabalho de Graduação foi apresentado ao Bacharelado em Ciência da Computação da Universidade Federal da Bahia, como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Marlo Viera dos Santos e Souza

Salvador
junho de 2021

RESUMO

O presente trabalho buscou investigar o uso de adaptação de domínio para auxiliar a tarefa de reconhecimento de entidades em domínios específicos no contexto do português brasileiro. Buscou-se atingir esse objetivo ao implementar e aplicar da técnica de adaptação denominada alinhamento não supervisionado de subespaços, conjuntamente com a análise experimental quantitativa e qualitativa dos resultados experimentais.

A questão chave que guiou a realização dos experimentos está relacionada com o limite para o erro de generalização, proposto na literatura de adaptação de domínio (KOUW; LOOG, 2019). Um dos principais termos desse limite é a medida da divergência entre domínios, então é natural se perguntar se um método qu procure diminuir essa divergência seria uma boa estratégia de adaptação. Em outras palavras: Existe um método que procure diminuir as divergências entre os domínios? Se esse método existe, a diminuição da divergência está associada com um aumento da performance dos modelos adaptados? Foram essas perguntas que procuraram ser respondidas com a realização dos experimentos.

Palavras-chave: Aprendizado de Máquina; Adaptação de Domínio; Alinhamento de Subespaços; Reconhecimento de Entidades Nomeadas

ABSTRACT

The present work seeks to investigate the use of domain adaptation techniques to assist the task of recognizing named entities in specific domains in the context of Brazilian Portuguese. We sought to achieve this objective by carrying out the implementation and application of the domain adaptation technique called subspace alignment, together with the quantitative and qualitative analysis of the experimental results.

The main question that guided the experiments is related to the bound for the adaptation error, proposed in the domain adaptation literature (KOUW; LOOG, 2019). One of the main terms of this limit is the measure of divergence between domains, so it is natural to ask whether applying an adaptation method that seeks to reduce this divergence would be a good adaptation strategy. In other words: is there a method that can reduce the divergence between domains, and if so, is the decrease in the divergence associated with an improvement in performance of the adapted models? Those are the questions that we tried to answer with the experimental results.

Keywords: Machine Learning Domain Adaptation Subspace Alignment Named Entity Recognition

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

PCA: Principal Component Analysis
LSTM: Long Short Term Memory
CRF: Conditional Random Fields
REN: Reconhecimento de Entidades Nomeadas
EI: Exatração de Informações
MUC: Messaging Undertanding Conferente
Darpa: Defense Advanced Research Projects Agency

Capítulo 1

INTRODUÇÃO

1.1 CONTEXTO

O reconhecimento de entidades nomeadas (REN) é a tarefa de identificar e classificar referências a entidades, como pessoas, organizações e expressões numéricas, usualmente em dados textuais (GRISHMAN; SUNDHEIM, 1995; NADEAU; SEKINE, 2007). REN nesse contexto é considerada uma sub-tarefa da extração de informações (EI), que consiste na extração de dados estruturados a partir de uma fonte de dados não estruturados (WILKS, 1997).

Uma importante motivação para aplicar métodos de REN em textos deriva do fato de que muita informação é expressa por entidades nomeadas. Por exemplo, se o objetivo for construir uma aplicação que faça uso de análise de sentimentos em avaliações de produtos, podemos processar os parágrafos em busca de entidades para descobrirmos a quem os sentimentos fazem referência. A seguinte sentença ilustra essa ideia:

```
O novo Iphone 6 é muito bom, apesar de um pouco caro.
```

Se conseguirmos identificar que a entidade presente na sentença é "Iphone 6", e que ela é a única entidade presente nessa sentença, sabemos que as expressões "muito bom" e "um pouco caro" são direcionadas à entidade "Iphone 6".

Outra aplicação em que REN demonstra sua utilidade é a análise automática da literatura científica acerca de certo tema, como biomedicina por exemplo. Um sistema de REN poderia ser aplicado sobre um conjunto de documentos relacionados (também chamado *corpus* na área de processamento de linguagem natural) para extrair as entidades contidas ali. Com essa informação, um especialista poderia inferir rapidamente (sem precisar ler todos os documentos) o tema central, apenas ao ver quais entidades (proteínas específicas, nomes de medicamentos) estão contidos no texto.

Devido a sua importância, muitos trabalhos foram realizados com o intuito de avançar o estado de arte para a tarefa de REN. A metodologia dominante consiste na aplicação de modelos supervisionados de aprendizado de máquina (NADEAU; SEKINE, 2007), de

forma que é vital a existência de um *corpus* com dados anotados. Uma arquitetura muito popular é uma rede neural profunda LSTM (*long short-term memory*) para extração de *features*, seguida de uma camada CRF (*conditional random fields*) que realiza a classificação (LAMPLE et al., 2016). Outras abordagens também foram e são rotineiramente utilizadas, como métodos semi-supervisionados ou não supervisionados. Mesmo com o avanço das pesquisas nesses dois últimos métodos, ainda é essencial a existência de *corporas* (uma coleção de *corpus*) anotados para o funcionamento da grande maioria dos modelos de REN, tendo em vista a prevalência do uso de modelos supervisionados.

Ainda em relação aos *corporas*, verifica-se que a grande maioria está inserida no contexto da língua inglesa, e ainda que existe um número considerável de *corporas* para o chinês (NADEAU; SEKINE, 2007). Mudando o foco para o contexto do português brasileiro, as pesquisas seguem a mesma tendência da utilização de métodos supervisionados de aprendizado de máquina, com um foco cada vez mais acentuado em redes neurais profundas e utilização de modelos de linguagem para modelar as características de entrada. Um exemplo dessa configuração são os trabalhos de (CASTRO; SILVA; SOARES, 2018), (SANTOS et al., 2019) e (SOUZA; NOGUEIRA; LOTUFO, 2019). Nesse contexto, existem *corporas* anotados especificamente para a tarefa de REN em português brasileiro, como o HAREM (SANTOS; CARDOSO, 2007), LeNER-BR (ARAUJO et al., 2018) ,GeoCorpus (AMARAL et al., 2017) e Cojur.

Para entedermos os desafios que a tarefa de REN nos apresenta, precisamos refletir um pouco sobre como poderíamos modelar o problema. Dessa maneira, é possível pensar em REN como um problema de duas partes: identificação e classificação. Ou seja, identifica-se que um dado segmento do texto é uma entidade, e posteriormente classifica-se essa entidade em algum tipo (como pessoa, ou organização). Se desejamos classificar, necessitamos primeiro definir quais são as classes, e essas naturalmente mudam de acordo com o domínio que estamos considerando. A modelagem clássica da tarefa de REN inclui classes genéricas, como pessoa, organização e expressões temporais, como definido pela MUC-6 (GRISHMAN; SUNDHEIM, 1995), e é seguindo essa modelagem que os maiores conjuntos de dados disponíveis foram construídos. Esses conjuntos são de domínios relativamente formais e bem estudados, como textos jornalísticos, e são derivados de conferências da área, como o Harem e CoNLL 2003 (SANG; MEULDER, 2003).

Considerar domínios mais específicos em contraste com os domínios discutidos acima introduz um novo nível de complexidade ao problema, o que se deve principalmente a dois fatores. O primeiro fator é derivado da observação da necessidade de dados anotados para a boa performance dos modelos supervisionados de REN. Anotar domínios específicos é um processo altamente dependente de especialistas do domínio em questão, e apesar do processo de anotação muitas vezes ser essencial para a melhor compreensão do fenômeno que estamos lidando, nem sempre existem anotadores prontamente disponíveis para realizar a tarefa.

O segundo fator diz respeito a natureza do domínio considerado. Diferentes domínios podem necessitar de diferentes classes, com diferentes níveis de glanularidade. Além disso, mesmo que as classes consideradas sejam as mesmas entre domínios, modelos treinados para obter uma boa performance em um conjunto de dados não necessariamente obterão a mesma performance em outro domínio, como discutido em (KOUW; LOOG, 2019).

Para ilustrar o desafio de REN para domínios específicos, vamos voltar ao exemplo do sistema de análise automática de literatura científica. Digamos que o nosso modelo de REN foi treinado no Harem, ou seja, as entidades observadas por ele incluem:

```
Reforma
Protestante
...
Idade
Média
...
cerca
de
600
km
```

Agora vamos comparar esse pequeno trecho do Harem com o seguinte extrato de entidades biomédicas:

```
PuB1
peri-kappa
Interferon
3-Amino-4-methyl-pentan-1-ol
```

Se aplicarmos um modelo treinado no Harem a um *corpus* de artigos de biomedicina, não é razoável esperar que a performance deste modelo seja boa, já que a natureza do que é considerado uma entidade muda completamente entre os dois domínios.

Diversos métodos foram explorados para abordar esses problemas. Um exemplo é a utilização de *self-training* em *corporas* em que se tem uma quantidade pequena de dados anotados, como o trabalho realizado por (SOARES, 2019). *Self-training* é uma abordagem de aprendizado de máquina em que se treina um classificador na pequena quantidade de dados iniciais, de maneira que esse classificador inicial possa ser aplicado para anotar as entidades que ainda não possuem rótulo, as quais são posteriormente utilizadas como exemplos na próxima iteração de treino.

Outra possível abordagem é aplicar uma das técnicas derivadas da área de pesquisa chamada adaptação de domínio. A configuraçao de um experimento de adaptação de domínio consiste em um domínio fonte e um domínio alvo, em que um classificador é treinado no domínio fonte e espera-se que ele obtenha uma boa performance no domínio alvo. Existem dois tipos de adaptação de domínio: não supervisionado e supervisionado. A forma não supervisonada assume que não existe nenhum rótulo no domíno fonte, de modo que o objetivo é tomar proveito da grande quantidade de dados provenientes do domínio fonte, mas garantindo que o classificador treinado nesses dados obtenha uma boa performance no domínio alvo (BEN-DAVID et al., 2010). Essa é uma estratégia que nos permite lidar com os casos onde temos poucos dados anotados, pois se certas propriedades forem obedecidas é possível adaptar com poucas informações acerca dos rótulos.

Mesmo que os dados em ambos os domínios estejam anotados, como é o caso de adaptação de domínio supervisionada, é possível que os rótulos tenham sido gerados através de diferentes distribuições de probabilidade para o domínio fonte e o domínio alvo. Dependendo do nível de discrepância, é provável que haja uma perda considerável de performance quando o modelo é aplicado ao conjunto alvo. Casos como esse ocorrem muito na área de visão computacional, em que imagens de treino e de teste podem ser obtidas de diferentes câmeras. Mas não é difível imaginar que seja um fenômeno comum em qualquer processo de anotação de *corpus* que envolva mais de um anotador.

Existem diversas maneiras de se realizar adaptação de domínio. É possível organizar os trabalhos da área de acordo com a estratégia utilizada para atacar o problema, como feito por (KOUW; LOOG, 2019). Adaptação de domínio baseada em *features* é uma dessas estratégias, e têm seu foco na manipulação da representação dos dados do domínio fonte para que essa informação possa ser aproveitada de alguma forma pelos classificadores, cujo objetivo é generalizar para um certo domínio alvo. Manipular nesse contexto usualmente significa achar um mapeamento de *features* do domínio fonte para *features* do domínio alvo. A intuição por trás dessa abordagem é que esse mapeamento diminui a discrepância entre os domínios, permitindo assim que um classificador treinado nas *features* transformadas generalize para o domínio alvo.

Explorar a aplicação de tais estratégias de aproximação parece ser uma rota interessante de investigação, pois os métodos nos permitem investigar a relação entre grandezas importantes para a adaptação de domínio. Uma dessas grandezas é a divergência entre domínios, que é utilizada para quantificar a diferença entre dois domínos sob um certo aspecto, como a distância geométrica de suas representações ou a diferença entre suas distribuições de probabildade.

Uma das técnicas de adaptação de domínio que nos permite tal exploração é o alinhamento de subespaços, cuja ideia é projetar os domínios fonte e alvo em subespaços de menor dimensão, e posteriormente alinhar as bases desses subespaços (FERNANDO et al., 2014). O classificador é treinado nas *features* transformadas, sendo possível utilizar os dados do domínio fonte como um conjunto de treino que em tese preserva a performance nas *features* transformadas do domínio alvo. Existem ainda extensões dessa ideia, como basear o alinhamento em instâncias invariáveis entre os domínios(ALJUNDI et al., 2015).

Com base em tudo que foi discutido até então, surgem algumas perguntas, as quais tentarei responder no curso do trabalho. Aplicar adaptação de domínio realmente melhora a performance de modelos que devem ser aplicado entre domínios para a tarefa de REN? Em relação a divergência entre domínios, qual o efeito do alinhamento sobre essa grandeza, e como essa variação está relacionada (se estiver) com o aumento ou diminuição da performance dos modelos adaptados? O alinhamento de subespaços também nos permite averiguar algumas propriedades dos subespaços gerados, de maneira que nos permitam verificar quais dessas prorpriedades se mantêm ou não quando os subespaços dos diferentes domínios são gerados?

A próxima seção segue a discussão com os trabalhos relacionados, os quais foram compilados durante a etapa de revisão bibliográfica.

1.2 TRABALHOS RELACIONADOS

Duas áreas da literatura foram exploradas para a realização deste trabalho. O primeiro corpo da literatura analisado foi o de REN, tanto em sua modalidade clássica quanto em domínios específicos. Em um primeiro momento, não foi feito qualquer restrição sobre a linguagem para a qual os modelos foram elaborados. Boa parte dos trabalhos lidos nesta primeira etapa eram para a língua inglesa, mas também incluiam outras línguas, como o chinês e o espanhol.

Em relação a REN clássica, a grande maioria das estratégias utilizadas para efetuar a tarefa envolviam aprendizado de máquina, com a modelagem da tarefa como classificação de sequências. Um modelo muito utilizado para efetuar essa classificação de sequências é o CRF, o qual foi utilizado como classificador no trabalho de (AMARAL; VIEIRA, 2014). Usualmente, o CRF é utilizado em conjuntura com outro modelo utilizado para obtenção de vetores de característica mais robustos. Os trabalhos (TORRALBA; EFROS, 2011) e (LING; WELD, 2012) utilizam uma rede neural *perceptron* e um modelo tópico juntamente com o CRF, respectivamente.

Como a tarefa de reconhecer entidades nomeadas é altamente dependente do contexto em que a palavra se insere, redes neurais recorrentes são um conjunto de modelos muito explorado para elaborar estratégias de classificação. Os trabalhos (SANTOS; GUIMARAES, 2015) e (CHIU; NICHOLS, 2016) utilizam uma rede neural recorreente para tratar o contexto, mas a tendência nos últimos anos é a utilização de uma rede biLSTM para obter os vetores de característica contextuais.

Essa tendência é demonstrada pelos trabalhos de (YADAV; SHARP; BETHARD, 2018), (BHARADWAJ et al., 2016), (DONG et al., 2016), (HUANG; XU; YU, 2015) e (MAI et al., 2018). Esses trabalhos procuram utilizar um modelo de linguagem e uma rede biLSTM para obter os vetores de característica, os quais são em seguida passados para o CRF para realizar a classificação.

Em relação a REN para o português, os trabalhos de (SANTOS et al., 2019) e (CASTRO; SILVA; SOARES, 2018) utilizam a arquitetura LSTM-CRF no Harem, enquanto os trabalhos de (LOPES; TEIXEIRA; OLIVEIRA, 2019), (SOARES, 2019), e (ARAUJO et al., 2018) utilizam a mesma arquitetura para tratar de domínios específicos no português, como o próprio LeNER-BR e domínios clínicos.

O segundo corpo de literatura explorado foi o de adaptação de domínio. Dois artigos nortearam essa revisão, e ambos possuiam o objetivo de compilar e apresentar a teoria que fundamenta o ato de adaptar entre domínios. O primeiro destes foi o artigo de (KOUW; LOOG, 2019), o qual apresenta um resumo compreensivo de boa parte do que foi proposto na área de adaptação de domínio, além de propor uma nomenclatura em comum e um sistema de classificação dos trabalhos da área. O segundo trabalho foi realizado por (BEN-DAVID et al., 2010), e introduz o conceito de erro de generalização, central para a realização deste trabalho.

Por fim, dois artigos foram utilizados como base para propor a técnica de alinhamento não supervisionado de subpesçaos. (FERNANDO et al., 2013) utilizam o alinhamento não supervisionado para a tarefa de classificação de imagens, enquanto o trabalho de (BJERVA; KOUW; AUGENSTEIN, 2019) procura utilizar a mesma estratégia para a

tarefa de REN, mas se limitando a verificar se a aplicação oferecia alguma melhora na performance dos modelos.

Com base nesse estudo da literatura, surgiu a ideia de explorar adaptação de domínio em REN para domínios específicos, e não apenas verificando os resultados experimentais em buscas de uma melhor performance, mas também investigando se as previsões teóricas da área de adaptação de domínio estão de acordo com os resultados experimentais obtidos.

As próximas seções encerram este capítulo ao delimitar a hipótese, justificativa e objetivos deste trabalho, além de uma estrutura geral da discussão que se segue. Todos os conceitos citados nesta seção serão discutidos com mais profundidade nos capítulos 2 e 3, de referencial teórico e adaptação de domínio respectivamente.

1.3 HIPÓTESE

A principal hipótese assumida por este trabalho é que o alinhamento de subespaços é capaz de promover a diminuição da divergência entre domínios ao aproximar as representações de ambos, o que permite treinar um classificador no domínio fonte que se adpte ao domínio alvo. Além disso, também se assume que conjuntos de dados relacionados (como os explorados aqui) mantém algumas propriedades, como as direções de maior variância. Essas considerações serão exploradas com mais detalhes nos próximos capítulos.

1.4 JUSTIFICATIVA

Compreender a utilização da técnica de alinhamento de subespaços na tarefa de REN para o português nos permite verificar a validade da teoria de adaptação de domínio nesse contexto, e tal compreensão é obtida através da comparação entre as previsões teóricas e dos resutados obtidos dos experimentos. Além disso, analisar as respostas dos modelos adaptados pode nos fornecer intuições sobre as mudanças que um modelo adaptado realiza na sua estratégia de anotação, e se essas mudanças fazem sentido no contexto da melhoria de performance na tarefa de REN.

Não há falta de aplicações que podem ser favorecidas por melhores sistemas de REN. Sendo uma sub-tarefa da tarefa de EI, toda aplicação que se beneficie de EI se beneficia de melhorias na tarefa de REN. Uma lista não exaustiva dessas aplicações inclui motores de busca, busca de moléculas em interações químicas, sistemas de respostas à perguntas, recuperação de informação e extração de relações (YADAV; BETHARD, 2019; NADEAU; SEKINE, 2007; SHARNAGAT, 2014). É importante reforçar ainda a vantagem de se buscar melhorias utilizando uma técnica como alinhamento de subespaços, vantagem essa que se deve principalmente ao fato de que é possível dispensar a obtenção de novos dados anotados para o domínio que se deseje trabalhar, desde que as condições necessárias sejam respeitadas.

1.5 OBJETIVO GERAL

Este trabalho tem como objetivo investigar a aplicação de adaptação de domínio na tarefa de REN em domínios específicos para o português.

1.6 OBJETIVOS ESPECÍFICOS

1. Levantar o estado da arte da tarefa de REN e REN em domínios específicos com foco no português brasileiro.

2. Levantar o estado da arte acerca das teorias e técnicas existentes na área de adaptação de domínio.

3. Propor um método com base na literatura estudada que permita aplicar adaptação de domínio em REN para domínios específicos em português.

4. Implementar o método proposto.

5. Validar o método proposto através da realização de experimentos.

1.7 ESTRUTURA

A organização do trabalho é como se segue: o capítulo 2 procura abordar o referencial teórico necessário para a compreensão do trabalho como um todo, tratando da tarefa de REN e aprendizado de máquina. O capítulo 3 é inteiramente dedicado à teoria de adaptação de domínio e à classificação das direntes maneiras de se realizar essa adaptação, com discussões detalhadas de cada abordagem. O capítulo 4 discute os diferentes conjuntos de dados considerados nos experimentos, a natureza de suas entidades e uma análise dos autovalores obtidos através da aplicação do alinhamento de subespaços. Os próximos dois capítulos elaboram os dois experimentos realizados, o primeiro baseado em reprentações não contextuais e o segundo em representações contextuais. O último capítulo fornece uma pequena revisão do que foi feito, além de uma discussão das conclusões e propostas de trabalhos futuros.

Capítulo

2

REFERÊNCIAL TEÓRICO

2.1 RECONHECIMENTO DE ENTIDADES NOMEADAS

Para compreendermos a importância da tarefa de REN é fundamental entedermos o que exatamente se considera uma entidade, e qual a importância de extrair e classificar essas ocorrências no texto. Para responder o que é uma entidade, é útil analisarmos alguns exemplos:

- João Carlos Albuquerque
- 26 de fevereiro de 2016
- Rua Jogo do Carneiro 2132
- Decreto Lei 2848/40

O que cada uma dessas sentenças possui em comum? a reposta é que cada uma delas faz referência a alguma entidade real ou abstrata, e usualmente essa é uma entidade específica ao invés de um conjunto de entidades. O conceito de designador rígido, introduzido por S. Kripke, pode ser utilizado para formalizar o conceito de entidade nomeada, e seu uso ocorre na literatura de REN (NADEAU; SEKINE, 2007; GRISHMAN; SUNDHEIM, 1995). Segundo Kripke, um designador rígido referencia a mesma entidade em todos os mundos possíveis.

Ademais, REN é uma etapa de pré-processamento fundamental que auxilia sistemas que necessitem de algum tipo de extração estruturada de informação. Um exemplo interessante é um sistema que deseje extrair que produtos foram mencionados em avaliações positivas em uma loja virtual, ou ainda descobrir se um *tweet* específico faz menção a uma pessoa, e ainda se essa pessoa está localizada em um local específico. Vamos agora discutir sobre a metodologia existente para abordar o problema de REN e suas particularidades.

Uma modelagem que se mostrou bem sucedida ao se tentar elaborar uma solução para o problema de REN é a marcação de sequência, do inglês *sequence tagging*. Para

resolver esse tipo de tarefa são utilizados os chamados modelos de sequência. Um modelo de sequência é um modelo que, dada uma sequência de palavras $W = (w_1, w_2, \ldots, w_k)$ com $k \in \mathbb{N}$ e um conjunto de marcações $M = \{m_1, m_2, \ldots, m_j\}$ com $j \in \mathbb{N}$, determina um mapeamento $F : W \rightarrow M$(JURAFSK, 2019).

Dada a necessidade de um conjunto de marcações de acordo com a modelagem acima, foi desenvolvido o conjunto BIO. Neste esquema, cada palavra do texto é marcado como estando no começo (B), dentro (I) ou fora (O) de uma entidade nomeada. Isso cobre a parte da identificação das EN, mas não a de classificação. Duas abordagens ocorrem na literatura neste caso: estender o conjunto de marcações, o que é mais comum, ou separar a tarefa de identificação e classificação.

Estender o conjunto de marcações para a tarefa de classificação consiste em associar a cada marcação a classe da entidade a ser detectada. Por exemplo, no caso de uma entidade pessoa chamada João Victor, a anotação seria (João;B-PER) (Victor;I-PER). Esse método aumenta a cardinalidade do conjunto de marcações de 3 para $2 * N + 1$, com N o número de classes a serem identificadas.

Separar ambas as tarefas consiste em utilizar o conjunto BIO de marcações para identificar as entidades, mas utilizar outra abordagem para classifica-las. Por exemplo, um determinado sistema pode utilizar um modelo de sequência, como o *conditional ramdom fields* (WALLACH, 2004), para identificar as entidades, e utilizar uma abordagem não-supervisionada que faz uso de um recurso externo (como um dicionário) para obter a distribuição de probabilidade sobre as possíveis classes das entidades segmentadas. Essa foi exatamente a estratégia utilizada por (RITTER et al., 2011).

2.2 RECONHECIMENTO DE ENTIDADES NOMEADAS PARA DOMÍNIOS ESPECÍFICOS: PARTICULARIDADES E DESAFIOS

Parte dos trabalhos de REN focam na identificação e classificação de classes genéricas em domínios bem estudados. No português, esse é o caso de métodos baseados nos dados como o HAREM, que possuem como fonte principal de dados textuais originados do gênero jornalístico e procuram anotar classes genéricas, como pessoa, tempo, valor e lugar.

Levando em consideração que é necessário um *corpus* anotado para treinar boa parte dos modelos de REN, os domínios específicos apresentam alguns desafios derivados dessa necessidade. O primeiro é o fato de que pode haver pouco material para se trabalhar, ou seja, o domínio não fornece material de treino suficiente para que os modelos de classificação possam convergir e produzir uma resposta satisfatória. A técnica de adaptação de domínio é uma das propostas para tentar resolver o problema da baixa quantidade de dados disponíveis para treino.

Mesmo que o *corpus* do domínio específico seja extenso o suficiente, ou que consigamos aplicar alguma técnica adaptativa num modelo treinado em outro domínio, existe o problema da granularização das classes consideradas. Muitos domínios específicos possuem um número grande de classes, algumas das quais possuem uma frequência muito baixa no conjunto de dados disponíveis. As classes também podem estar organizadas em algum tipo de hierarquia. Um exemplo seria a marcação *pessoa/homem/adolescente* para

um determinado indivíduo. Isso torna a tarefa de classificação de entidades nesses *corpus* altamente difícil, mesmo utilizando alguma técnica de adaptação.

2.3 APRENDIZADO DE MÁQUINA

2.3.1 Definição

Aprendizado de máquina é a área que concerne os chamados algoritmos adaptativos, ou seja, algoritmos que conseguem melhorar a sua performance em uma determinada tarefa (medida através de uma função de performance) através da observação dos dados que modelam aquela tarefa (GOODFELLOW; BENGIO; COURVILLE, 2016). Uma maneira comum de representar esses dados é através de vetores de característica. Uma característica é alguma propriedade individual de um fenômeno que pode ser mensurada. Podemos usar como exemplo um sistema de aprendizado de máquina que procure prever se uma determinada pessoa tem risco de desenvolver diabetes ao analisar características como altura, peso e idade. Desta forma, uma típica entrada para tal sistema seria codificada na forma de um vetor de características do tipo $(idade, altura, peso)$.

Uma tarefa rotineiramente resolvida por aprendizado de máquina é classificação. Nesta tarefa, temos um conjunto de observações de entrada $X = \{x_1, x_2, ..., x_n\}$, onde $x_i \in R^D$, com D a dimensão do nosso vetor de características, e outro conjunto de classes ou rótulos $Y = \{y_1, y_2, ..., y_m\}$, e nos interessa encontrar um mapeamento $f : R^D \to Y$. Colocado de outra forma, desejamos atribuir um rótulo ou classe y_i a um vetor de características que representa uma instância de nossos dados. Um algoritmo adaptativo tenta achar uma função aproximativa: $\hat{f}(x, W) = y$ com W um conjunto de parâmetros atualizados a medida que os dados rotulados são observados.

A especificação da função a ser aproximada varia de acordo com o tipo de modelo que está sendo empregado, além da estratégia de "pontuação", codificada pela função de perda. A função de perda é uma função que atribui uma "penalidade"caso o modelo erre o resultado da classificação ou uma "recompensa"caso contrário. Esse conceito é importante pois a princípio o que todo modelo de aprendizado de máquina possui em comum é a minimização de uma determinada função de perda . Em relação a sua definição, a função de perda é dependente da função do modelo, ou seja, a função $\hat{f}(x, W) = \hat{y}$. A título de exemplo, uma função de perda comum é a 0-1, definida como:

$$L(y, \hat{y}) = \begin{cases} 0, & \text{se } \hat{y} = y \\ 1, & \text{se } \hat{y} \neq y \end{cases}$$

Ou seja, as respostas do nosso modelo devem ser tais que façam com que se acerte o máximo possível, consequentemente minimizando a função de perda como definida acima. Generalizando, o modelo que desejamos é m', em que:

$$m' = \min_m L(Y, \hat{Y})$$

A estratégia para minimizar a função também é variável. Em redes neurais por exemplo, é comum a utilização do chamado algorítmo de gradiente descendente. Esse

algorítmo procura calcular o gradiente cujo sentido aponta para a direção em que a função decresce mais rapidamente. Os pesos W são então modificados através da utilização de um *bias* e do gradiente. O *bias* é utilizado para que a modificação dos pesos não seja exagerada e o modelo dê um passo muito grande no processo de minimização.

A sucessiva iteração desse algoritmo a medida que se observam os dados faz com que se caminhe para uma região de mínimo local, onde não é mais possivel otimizar o resultado. Gradiente descendente e suas modificações são a base da maior parte dos algorítmos de aprendizado de máquina.

Os conceitos discutidos acima são importantes por serem comuns a grande maioria dos algoritmos de aprendizado de máquina, os quais são discutidos no decorrer do trabalho. Falaremos a seguir de outros conceitos importantes na área de aprendizado de máquina, que se referem aos conjuntos de dados que são observados para que os modelos adaptem e verifiquem os seus resultados.

2.3.2 Partição dos Dados e Hiperparâmetros

Em experimentos de aprendizado de máquna, é usual particionar o conjunto de dados em três: treino, desenvolvimento e teste. Essa separação é necessária pois o modelo ajusta seus parâmetros para os dados que ele observa, ou seja, se testarmos o modelo nos mesmos dados em que ele treinou, ele terá uma performance muito melhor do que realmente obteria no mundo real.

O conjunto de treino é aquele o qual o modelo irá observar, ou seja, comparar as anotações com sua função de perda, ajustando os parâmetros de acordo. O conjunto de desenvolvimento é aquele em que podemos verificar a performance do modelo com o intuito de ajustar seus hiperparâmetros. Um hiperparâmetro é um parâmetro do modelo que não varia com o treino, dependendo da alteração manual de seus valores. Um exemplo é o *bias* citado acima. Diferentes modelos podem especificar diferentes hiperparâmetros dependendo da estrategia utilizada para classificar.

Por último, temos o conjunto de teste. Nesse conjunto é onde verificamos experimentalmente os resultados do nosso modelo, ou seja, verificamos sua performance como esperamos encontrar em uma configuração não artificial de aplicação. É interessante garantir que todos os conjuntos estejam razoavelmente balanceados, ou seja, possuam aproximadamente a mesma proporção de classes ocorrendo em seus exemplos. Caso contrário, o modelo será enviesado pelo conjunto de treino, e pode não ser capaz de generalizar suas conclusões para o conjunto de teste.

Uma pergunta que surge naturalmente é como dividir o seu conjunto de dados entre essas três partições. Existem diversas propostas para particionar o conjunto de dados, baseadas em porcentagens do conjunto de dados alocado para cada partição. Por exemplo, é comum usar o particionamento 70/15/15, significando 70% para treino, 15% para desenvolvimento e 15% para teste. Usualmente se aloca mais dados para o treino do que para teste e desenvolvimento, com os exatos valores variando de acordo com a necessidade do modelo. Há ainda estratégias que não utilizam o conjunto de desenvolvimento, apenas os de treino e teste.

Dada a importância do conceito de vetores de característica para boa parte dos al-

gorítmos de aprendizado de máquina, vamos agora elaborar um pouco mais esse conceito e suas implicações.

2.3.3 Vetores de Característica

Os dados observados pelos modelos de aprendizagem de máquina estão codificados na forma de vetores de característica. Como já mencionado, uma característica é um aspecto mensurável de um determinado fenômeno de interesse. Um vetor de características é portanto um vetor $x \in \mathbb{R}^D; D \in \mathbb{N}; D >= 1$, onde cada componente representa uma característica.

Por consequência, um vetor de características pode ser interpretado como um ponto do espaço vetorial definido pelo nosso conjunto de vetores de característica. A maneira de se obter tais vetores e a interpretação do que seria uma característica varia dependendo do contexto, de forma que precisamos discutir um pouco mais essas nuances.

A primeira interpretação de um vetor de características é aquela onde cada componente c_i pode ser considerada uma variável aleatória que mede um aspecto distinto do fenômeno estudado. Um exemplo concreto seria um vetor $(altura, idade)$ usadas como entrada para um algoritmo que retorna 1 se a pessoa possui problemas cardíacos e 0 caso contrário. Esse tipo de característica costuma ser projetada de forma manual por um especialista do domínio de aplicação.

A outra interpretação não atribui um significado específico a um componente individual do vetor de características. O significado desses vetores só pode ser compreendido quando se leva em consideração o contexto em que o vetor se insere, ou seja, quando se compara esse vetor com os outros vetores do mesmo espaço vetorial. Um exemplo desse tipo de abordagem são os vetores de característica extraídas pelos modelos de linguagem, como Word2Vec (MIKOLOV et al., 2013).

Mas como toda essa discussão se traduz para o estudo computacional da linguagem e a tarefa de REN? Em processamento de linguagem natural, vetores de característica podem representar documentos, sentenças, palavras e caracteres, com a glanuralidade da representação dependente do problema que se deseja resolver. Alguns métodos representam mais de um nível ao mesmo tempo (como palavras e caracteres), posteriormente realizando a sua composição.

Existem métodos que conseguem extrair representações vetoriais que codificam informações de diferentes níveis linguísticos, como morfológico, sintático e semântico. O funcionamento desse tipo de técnica se baseia na chamada hipótese distribucional, ou seja, que palavras que aparecem juntas no mesmo contexto com grande frequência possuem significados semelhantes (JURAFSKY; MARTIN, 2019). Um dos modelos que toma proveito dessa hipotese é o já mencionado Wor2Vec é um método que explora essa hipótese para derivar um espaço vetorial onde palavras que aparecem com frequência na mesma vizinhança estão mais próximas nesse espaço vetorial. A noção de proximidade pode ser quantificada por alguma medida de orientação ou distância, como similaridade de cosseno ou distância euclidiana.

Como já comentado, componentes dos vetores de característica derivadas por métodos como Word2Vec não são tão interpretáveis como uma componente que mede um fenômeno

diretamente quantificável. O cerne da questão é que os padrões codificados por essas caracterísicas são utilizados pelos algoritmos de aprendizado de máquina para ajustar seus parâmetros, consequentemente melhorando sua performance na tarefa que se deseja resolver. A figura 2.1 demonstra um processo comum relacionado à representação através de *features*.

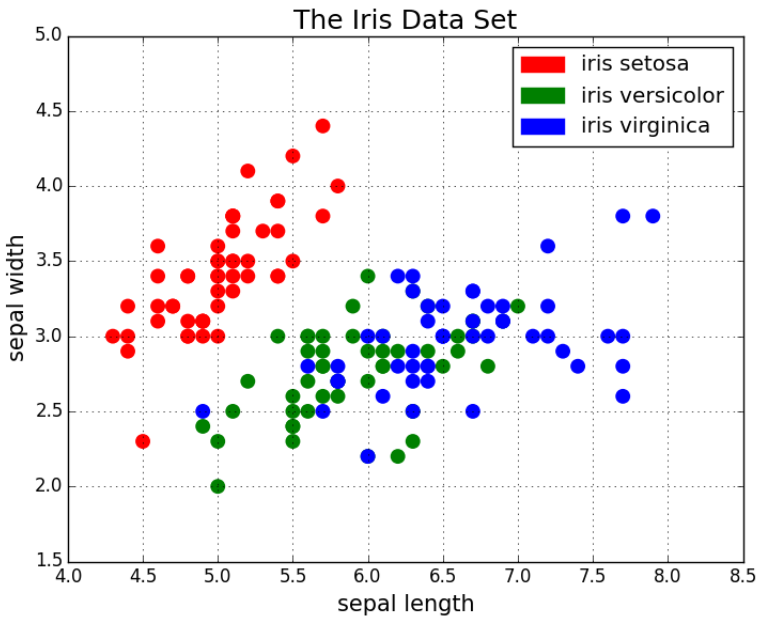

Figura 2.1: Demonstração gráfica dos vetores de característica do conjunto Iris para as características de comprimento e largura da sepas respectivamente.

Fonte: http://www.pybloggers.com/2015/09/my-first-time-using-matplotlib/

O conjunto de dados ilustrado é o Iris, atribuido ao estatístico Ronald Fisher (FISHER, 1936). Nesses dados estão catalogados as medidas de diversas flores, como o comprimento e largura de suas pétalas, juntamente com a espécie de cada uma delas. Vemos que o vetor de características é do tipo $(comprimentoSepa, larguraSepa)$. A imagem mostra os vetores de característica pertencentes a três classes distintas, a saber: Setosa, Versicolor e Virginica. Um modelo que utilizasse como estratégia a distância entre os pontos, por exemplo, obteria sucesso ao diferenciar instâncias da Setosa, mas teria muito mais dificuldades de diferenciar entre Versicolor e Virginica, pois os pontos dessas duas classes encontram-se muito próximos entre si em alguns casos.

Uma maneira de resolver esse problema seria retornar uma probablidade para cada classe, ao invés de se basear unicamente nas distância entre os pontos. Dessa forma, um modelo de classificação pode ser interpretado como um estimador da probablidade condicional sobre as classes: $P(y|x, W)$. Tendo em vista essa necessidade, veremos agora modelos que retornam esse tipo de resposta, denominados redes neurais.

2.3.4 Redes Neurais

Redes neurais são um conjunto de técnicas de aprendizagem de máquina comumente utilizadas para problemas em visão computacional (KHAN et al., 2018) e processamento de linguagem natural (GOLDBERG, 2017). Sua projeção é inspirada no funcionamento do cérebro de organismos vivos, o qual possui unidades (neurônios) que se comunicam através de canais os quais trocam sinais entre si (sinápses).

Da mesma maneira que um cérebro humano é capaz de aprender ao experimentar o mundo ao seu redor, redes neurais são capazes de aprender, em um sentido mais restrito, quando a ela são apresentadas instâncias dos dados do nosso conjunto de interesse.

No contexto de um problema de classificação, uma rede neural é um modelo matemático que tem por objetivo construir uma distribuição de probabilidades sobre o conjunto Y de classes. Essa distribuição é obtida ao se aplicar uma série de transformações sobre o vetor de entrada, até que se obtenha o vetor final representando a distribuição de probabilidade sobre as classes.

Uma transformação individual pode ser representada pela fórmula:

$$u = \sigma(Wx)$$

onde W é uma matriz m x n de parâmetros denominados pesos. É essa matriz que é atualizada pelo algorítmo de gradiente descendente para melhorar a performance do modelo. A função σ é uma função não linear denominada função de ativação, como a sigmóide. Essa função é fixada como não linear pois caso contrário estaríamos aplicando somente uma transformação linear, e consequentemente não seríamos capazes de resolver problemas que não são linearmente separáveis.

A função de ativação pode ser linear, mas em muitos casos a resposta do nosso problema não varia linearmente com os dados de entrada. Podemos exemplificar essa ideia pensando num problema simplificado que poderia ser linearmente separável, como determinar a probabilidade de um paciente desenvolver diabetes. Digamos que a resposta dependa somente de uma variável: o peso do paciente. Neste tipo de problema, é razoável assumir que quanto maior o peso maior a probabilidade de desenvolver diabetes, então uma função de ativação linear poderia ser utilizada. Se pensarmos em problemas mais complexos, como a previsão de tempo, precisamos introduzir no modelo algum tipo de não-linearidade para que tênhamos chances de prever os resultados corretamente, simplesmente porque a resposta não varia linearmente com as variáveis do vetor de características.

Se assumirmos que $|x| = k$, o vetor u possui tamanho m, e cada componente u_i de u é resultado da sigmóide aplicada ao produto vetorial entre W_i e x. Em símbolos:

$$u_i = \sigma(W_i x)$$

O resultado de cada operação acima é chamado camada. Se obtém o resultado final através da sucessiva obtenção das camadas, até que se chegue na camada final. A função aplicada sobre a última camada é normalmente uma normalização, para que o resultado possa ser interpretado como uma distribuição de probabilidade. Essas operações podem ser representadas graficamente pelo esquema abaixo:

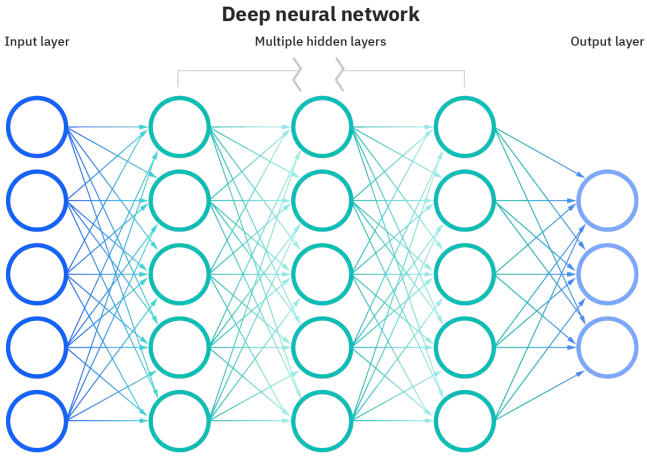

Figura 2.2: Interpretação gráfica de uma rede neural

Fonte: https://www.ibm.com/cloud/learn/neural-networks

Cada pilha de círculos é um vetor obtido pelas transformações especificadas acima. Cada aresta que liga um dos círculos é um peso individual. O modelo retratado acima é o chamado percéptron multicamada. Nesta abordagem, o tamanho do vetor de entrada é fixo, e cada predição não leva em consideração as predições de instâncias precedentes ou posteriores, ou seja, não leva em conta o contexto em que a observação se insere. Discutiremos a seguir arquiteturas que procuram trabalhar com essas variáveis. Agora que compreendemos o conceito de rede neural, podemos partir para definir métodos que utilizam essa técnica para tratar problemas que nos interessam relativos ao processamento de linguagem natural, e consequentemente à tarefa de REN.

2.3.4.1 Linguagem, Contexto e Arquiteturas Recorrentes Linguagem é um fenômeno inerentemente temporal (JURAFSK, 2019). Isso significa que a linguagem é interpretada como uma sequência que se desenrola no tempo, de forma que a predição de um modelo que trate de um problema linguístico deve levar em consideração o contexto que a observação atual se insere. Percéptrons como os discutidos na seção anterior possuem uma limitação nesse sentido: a predição não leva em consideração as predições anteriores. Redes neurais recorrentes (*recurrent neural networks, ou RNN*) surgem para resolver essa limitação.

A única diferença entre uma RNN e uma percéptron é que a computação da camada escondida (camada entre a entrada e a saída de uma rede neural) depende do valor das camadas escondidas obtidas no passo anterior. A inserção da camada escondida do passo anterior na computação da camada atual serve como uma espécie de memória que auxilia a tomada de decisão levando em consideração o contexto passado. Para implemenar essa mudança, precisamos de um nova matriz de pesos U que conecta a camada escondida anterior à nova camada. A seguinte fórmula é utilizada para expressar esse conceito matematicamente:

$$h_t = f(Uh_{t-1} + Wx_t)$$

Em que a função f é alguma função não linear, h_t é a camada escondida atual e h_{t-1} é a camada escondida anterior. É importante ressaltar que não há limites para a janela de contexto utilizada, já que a cada nova computação a camada escondida codifica informação de todas as camadas escondidas anteriores.

A tarefa de classificação de sequências apresenta uma melhora considerável sob esta estratégia, já que a informação sobre uma classificação anterior afeta o resultado da classificação atual. A imagem abaixo ilustra graficamente uma RNN:

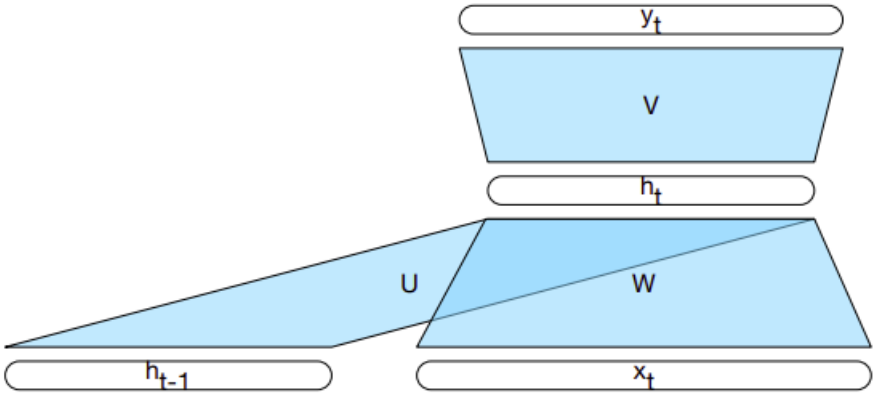

Figura 2.3: Rede neural recorrente

Fonte: Fonte:https://web.stanford.edu/ jurafsky/slp3/9.pdf

Uma extensão da RNN muito utilizada para a classificaçao de sequências em tarefas linguísticas é a LSTM bidirecional (HUANG; XU; YU, 2015). Uma RNN bidirencional leva em consideração tanto o contexto passado (assim como a tradicional) quanto o contexto futuro. Para obter o contexto futuro, uma outra RNN é treinada no *input* invertido até o *input* atual. Denotemos a camada escondida obtida do contexto passado como h_t^f (f de *foward*) e a camada escondida obtida do input invertido como h_t^b (b de *backward*). Podemos elaborar um vetor de contexto total aplicando alguma operação sobre as camadas *backward* e *foward*, como a concatenação:

$$h_t = h_t^b \oplus h_t^f$$

Isso permite ao modelo fazer inferências baseadas tanto no futuro quanto no passado. RNNs tradicionais também apresentam dificuldade em levar em consideração o contexto de observações distantes, pois a informação se perde no tempo através do cálculo sucessivo de novas camadas escondidas. A LSTM resolve esse problema ao introduzir maneiras de esquecer informações desnecessárias e lembrar das essenciais para que se classifique a observação atual.

Esse objetivo é alcançado através da utilização dos chamados *gates*, juntamente com uma camada de contexto (ao invés de utilizar somente a camada escondida). Existem três

gates: o *forget gate*, o *add gate* e o *output gate*. A tarefa do *forget gate* e do *add gate* é construir o próximo vetor de contexto, enquanto o *output gate* tem o objetivo de calcular a nova camada escondida com base no contexto construído pelo *add* e *forget gates*.

forget gate tem a tarefa de apagar informações que não são mais necessárias. As próximas equações ilustram esse comportamento:

$$f_t = \sigma(U_f h_{t-1} + W_f x_t)$$
$$k_t = c_{t-1} \odot f_t$$

a função σ tem a propriedade de levar seus valores para 1 ou 0, consequentemente valores não tão importantes são efetivamente apagados, enquanto valores relevantes são mantidos na camada f_t. f_t é então multiplicado (multiplicação de componesntes) com o contexto anterior, o que tem o efeito de apagar as informações irrelevantes, gerando k_t.

Após determinanos k_t, geramos a camada escondida para o *input* atual, assim como fazíamos normalmente com a RNN tradicional:

$$g_t = f(U_g h_t + W_g x_t)$$

O próximo *gate* é o *add*, que segue a mesma lógica do *forget gate*, mas aplicado sobre g_t, ou seja, selecionando a informação a ser mantida na camada do *input* atual.

$$i_t = \sigma(U_i h_{t-1} + W_i x_t)$$
$$j_t = g_t \odot i_t$$

Com k_t e j_t, podemos calcular o próximo contexto:

$$c_t = k_t + j_t$$

E, finalmente, usamos o *output gate* e o contexto atual c_t para calcular a camada escondida atual:

$$o_t = \sigma(U_o h_{t-1} + W_o x_t)$$
$$h_t = o_t \odot f(c_t)$$

Note que os pesos são específicos para cada *gate*, de forma que o modelo precisa aprender pesos separados para as tarefas de esquecer, adicionar e calcular a camada escondida atual. Essas propriedades, conjuntamente com a bidirecionalidade, fazem da LSTM um modelo poderoso para a classificação de sequências.

No que concerne a tarefa de REN, temos duas opções em relação a camada escondida h_t. Podemos passar essa camamda por uma função *softmax* e obter as probabilidades para as classes B,I e O, ou podemos passar essa camada que codifica contexto para outro modelo de classificação, que consequentemente terá uma performance melhor ao utilizar uma representação mais robusta. Um modelo de classificação muito usado em conjunção com as representações da LSTM é o CRF (*conditional random fields*) (HUANG; XU; YU, 2015).

O CRF é um modelo baseado em grafos que implementa contexto e dependência entre as predições. Essa etapa é útil para eliminar transições inválidas na classificação, como um O seguido de um I. A arquitetura LSTM CRF é portanto a utilização da LSTM para obter uma representação contextual para cada palavra do vocabulário, a qual é subsequentemente alimentada para o CRF, que irá predizer sua classe eliminando transições não permitidas e levando em conta as predições anteriores.

2.3.5 Resumo do Capítulo

Neste capítulo, fizemos a discussão do conteúdo necessário para entender a discussão levada a cabo no restante do trabalho. Definimos com mais detalhes o conceito de entidade nomeada e a sua relevância para uma série de tarefas que necessitam de algum tipo de extração estruturada de informações. Além disso, enumeramos as particularidades e desafios encontradas ao tentar resolver REN para domínios específicos.

Em seguida, explicamos os conceitos relativos a área de aprendizado de máquina, fundamentais para compreender os experimentos realizados. Dentre esses, é de fundamental importância para o presente trabalho a discussão acerca dos vetores de característica, redes neurais e a arquitetura LSTM-CRF.

O próximo capítulo procura elaborar outro campo de suma importância para a discussão, que é o corpo teórico da área de adaptação de domínio.

Capítulo

3

ADAPTAÇÃO DE DOMÍNIO

Adaptação de domínio é a tarefa de treinar um classificador em um domínio fonte D_S no qual se assume a existência de uma grande quantidade de dados anotados, de maneira que esse classificador obtenha uma boa performance em um domínio alvo D_T, em que se considera haver pouca ou nenhuma informação. Fazer com que um classificador generalize entre conjuntos de dados é desafiador mesmo que os conjuntos pertençam ao mesmo domínio, devido ao efeito de amostra polarizada. Colocado de outra forma, diferentes domínios apresentam distribuições de probabilidade distintas sobre as classes, de forma que um modelo que tente estimar essa distribuição em um dos domínios não fará a predição com a mesma performance no domínio alvo.(HECKMAN, 1979).

Para além do processo de amostras polarizadas, o que se considera como entidade varia sensivelmente entre domínios. Para ilustrar essa ideia, podemos relembrar o nosso exemplo de um sistema que detecta entidades biomédicas. Tais entidades são nomes de proteínas, genes e medicamentos. Se contrastarmos isso com domínios mais tradicionais, como os jornalísticos, em que é comum entidades nomeadas que designam pessoas, a exemplo de "João Ubaldo Ribeiro", fica evidente que um modelo treinado em um desses domínios não obteria a mesma performance ao tentar detectar entidades no outro.

Na prática, os domínios não podem ser muito diferentes para que seja possível adaptar um classificador. Uma maneira de formalizar esse fenômeno é considerar que diferentes domínios representam diferentes distribuições de probablidade para o mesmo conjunto Y de classes, ou seja, temos $P_S(x, y) \neq P_T(x, y)$. Diferentes métodos de adaptação de domínio assumem diferentes suposições acerca das distribuições de probabilidade.

Uma maneira encontrada pela literatura de entender melhor o fenômeno de adaptação de domínio foi tentar limitar o chamado erro de generalização. A próxima seção dedica-se a explicar essa ideia, além dos conceitos relacionados, além dos tipos de adaptação de domínio catalogados. Também será abordado o método utilizado neste trabalho, que é o alinhamento de subespaços não supervisionado.

3.1 UM LIMITE PARA O ERRO DE GENERALIZAÇÃO

Nesta seção, vamos discutir as definições necessárias para compreender o conceito de erro de generalização. Discutiremos também um teorema que fornece um limite para esse erro que envolve a divergência entre os domínios

3.1.1 Função de Erro

Uma função de erro ou função de risco, denotada e, definida sobre um espaço de hipóteses H, é a função $e = E(l(h(x), y))$. Nessa expressão h é um classificador e y o rótulo de x. Ou seja, $e(h)$ é o valor esperado da função de perda 0-1 ao aplicar o classificador h no conjunto de *features*. Quanto melhor o classificador h, mais próximo de zero se encontra $e(h)$.

3.1.2 Erro de Generalização

O erro de generalização é definido como: $e_r = e(\hat{h_s}) - e(h_t^*)$. Em que $\hat{h_s}$ é o classificador estimado no domínio fonte e h_t^* é o classificador ótimo no domínio alvo. Deseja-se que o erro do classificador estimado seja exatamente igual ao erro do classificador ótimo, de forma que e_r seja 0. Na prática, $e_r >= 0$, pois $e(\hat{h_s}) >= e(h_t^*)$. A intuição por trás do cálculo do erro de generalização é que se esse valor for muito alto, não é possível generalizar entre conjuntos de dados.

3.1.3 Hipótese Conjunta Ideal

A hipótese conjunta ideal é um dos três termos que limitam o erro de generalização. Essa grandeza mede o erro do classificador que comete a menor quantidade de erros quando aplicado ao domínio fonte e ao domínio alvo:

$$e_{S,T}^* = min_{h \in H}[e_S(h) + e_T(h)]$$

.

A ideia aqui é que se temos um classificador ótimo para os dois domínios que apresente um erro muito alto, é impossível realizar o processo de adaptação.

3.1.4 Limitando o Erro de Generalização

O erro de generalização é limitado por três termos: a hipótese conjunta ideal, a medida de dissimilaridade entre o domínio fonte e o domínio alvo e um termo que depende da complexidade do classificador. Formalmente (BEN-DAVID et al., 2010):

$$e_r \leq 2e_{S,T}^* + D_{H\Delta H} + C$$

Onde $D_{H\Delta H}$ e C são a medida de dissimilaridade entre os domínios e um termo que depende da cardinalidade N dos domínios e da complexidade do classificador.

Existem duas observações importantes a serem feitas em relação a esse limite. A primeira é que ele não incorpora nenhuma estratégia específica de adaptação, ou seja, ele não assume nada sobre a natureza dos classificadores utilizados (se são baseados em redes neurais, métodos probabilísticos etc). Usando esse limite como base, é possível propor suposições que permitam diminuir o erro de generalização. Uma suposição comum é assumir que apesar de termos $P_S(x, y) \neq P_T(x, y)$, vale que $P_S(y|x) = P_T(y|x)$.

A segunda observação diz respeito a suposição que foi feita neste trabalho. (GONG et al., 2016b) afirmam: Se assumirmos que exista uma transformação t tal que $p_S(t(x)|y) = p_T(t(x)|y)$, é possível limitar o erro de generalização utilizando a medida de divergência entre os domínios. Isso significa que o termo mais importante a ser manipulado é o valor dessa divergência. Foi a suposição de que existe essa transformação t na tarefa de REN para o português a assumida neste trabalho, de forma que o método utilizado teve o objetivo de diminuir a divergência entre os domínios, assumindo que isso traria diminuição do erro de generalização e consequentemente um aumento na performance do modelo.

A seguir, discutiremos os diferentes tipos de adaptação de domínio, as suposições assumidas por cada abordagem e o contexto onde cada uma pode ser aplicada. Abordaremos com mais detalhes a adaptação baseada em *features*, que serve como suporte teórico para o método utilizado neste trabalho, o alinhamento não supervisionado de subespaços.

3.2 TIPOS DE ADAPTAÇÃO DE DOMÍNIO

3.2.1 Abordagens Baseadas em Amostras

Abordagens baseadas em amostra procuram minimizar o risco no conjunto alvo através da observação de amostras no domínio fonte. Vamos ilustrar essa estratégia ao discutir um método chamado *data importance-weighting*. Para isso, vamos determinar primeiro como calcularíamos o risco do domínio alvo para um dado classificador h. Basta aplicarmos a fórmula do valor esperado:

$$e_T(h) = \sum_y \int_x l(x, y) * p_T(x, y)dx$$

O que fazemos com essa fórmula depende do tipo de suposição que assumimos ser válida. Duas suposições foram discutidas na literatura por (KOUW; LOOG, 2019). A primeira é a chamada mudança *a priori*, e a segunda é chamada mudança covariável. A mudança *a priori* assume $p_S(x|y) = p_T(x|y)$. Por outro lado, a mudança covariável assume $p_S(y|x) = p_T(y|x)$. A pergunta que surge naturalmente é: qual a utilidade dessas suposições?

Cada uma dessas suposições possui uma interpretação acerca da natureza do aprendizado levado a cabo pelo modelo. A mudança covariável é considerada como aprendizado causal, ou seja, predizer os efeitos (probabilidade dos rótulos) através da observação das causas (vetores de característica). Em constraste, a mudança a priori considera que o aprendizado é anti-causal, o que seria tentar prever as causas através dos efeitos.

Note que podemos decompor $p(x, y)$ da seguinte forma:

$$p(x, y) = p(x|y) * p(y) = p(y|x) * p(x)$$

Podemos também relacionar as distribuições do domínio fonte e do domínio alvo através da equação de risco equivalente:

$$e_T(h) = \sum_y \int_x l(x, y) * \frac{p_T(x, y)}{p_S(x, y)} * p_S(x, y)dx$$

Agora, assumindo a mudança covariável, decompomos as distribuições:

$$e_T(h) = \sum_y \int_x l(x, y) * \frac{p_T(y|x) * p_T(x)}{p_S(y|x) * p_S(x)} * p_S(x, y)dx$$

E como pela nossa suposição $p_T(y|x) = p_S(y|x)$, podemos cancelar esses termos, obtendo:

$$e_T(h) = \sum_y \int_x l(x, y) * \frac{p_T(x)}{p_S(x)} * p_S(x, y)dx$$

A partir da nossa suposição, simplificamos a fórmula do risco no domínio fonte, que só depende das distribuições marginais do domínio fonte e do domínio alvo, além da distribição do domínio fonte, a qual podemos estimar por assumirmos que há informação disponível. A razão $\frac{p_T(x)}{p_T(y)}$ é chamada pesos de importância. É possível observar que os pesos terão um alto valor se a amostra observada tiver uma alta probabilidade no domínio alvo e baixa probabilidade no domínio fonte. Os pesos afetam a classificação ao afetar o valor da função de perda.

O que gera a variedade nos métodos de abordagem baseada em amostras é o tipo de suposição feita e como estimar a razão $\frac{p_T(x)}{p_T(y)}$. É possível tentar estimar as distribuições marginais individualmente, fazer alguma suposição sobre sua natureza (se é uma distribuição normal, quais os valores dos parâmetros) ou ainda encarar os pesos de importância como uma variável a ser estimada por algum processo de otimização.

3.2.2 Abordagens Baseadas em Inferência

As abordagens baseadas em inferência são uma categoria diversa, com uma gama ampla de estratégias. O que elas possuem em comum é a inserção da estratégia de adaptação no procedimento de inferência. Isso contrasta com a abordagem baseada em amostra, que adapta sua classificação através da observação de exemplos individuais do domínio fonte.

Algoritmos robustos são aqueles para onde é garantido que a variação da função de custo quando ele é aplicado em partições do espaço de features é limitada. Se interpretamos o domínio fonte e o domínio fonte como partições distintas onde instâncias são retiradas e colocadas de um para o outro, um algorítmo robusto poderia ser treinado no domínio fonte e sua perda de performance garantidamente seria limitada. Essa é uma das técnicas utilizadas em adaptação de domínio baseada em inferência, e é a abordagem seguida por (MANSOUR; SCHAIN, 2014).

Outra estratégia é modelar o problema de adaptação como um processo de otimização *minimax*. Nessa abordagem, um adversário do classificador maximiza o risco com respeito a uma certa quantidade desconhecida. No nosso caso, essa quantidade é a distribuição $p(y|x)$ do domínio alvo. A ideia é que, ao tentar minimizar o risco dessa forma, o classificador estaria sendo treinado para adaptar para o pior caso possível (máxima incerteza).

O trabalho de (LIU; ZIEBART, 2014) propõe exatamente essa estratégia, com a adição de limitar a classe dos adversários àqueles cuja distribuição de probabilidade estimada tenha propriedades semelhantes à distribuição do domínio fonte. Caso essa limitação não seja imposta, corre-se o risco do adversário gerar sempre o pior classificador possível (um que sempre discorde do seu classificador). A fórmula abaixo codifica esse raciocínio, em que $H \cap_\Theta$ é o conjunto de hipóteses restrito como mencionado acima:

$$\hat{h} = \arg\min_{h \in H} \max_{H \cap_\Theta} \frac{1}{m} \sum_{j}^{m} l(h(z_j), g(z_j))$$

Por fim, discutiremos a estratégia de *self learning*. No caso de se possuir uma pequena quantidade de rótulos do domínio alvo, ou ser possível anotá-los com baixo custo, podemos treinar um classificador inicial nessa pequena quantidade de dados e aplicá-lo em todo o conjunto de treino para gerar os dados rotulados. É possível treinar um modelo que iterativamente anote e treine nos dados, com a performance do modelo melhorando a cada nova iteração. A vantagem dessa abordagem é precisar de poucos dados anotados, enquanto a desvantagem é a possível propagação dos erros que o classificador comete durante as sucessivas iterações de anotação e treino.

Na próxima seção trataremos das estratégias de adaptação baseadas em *features*. Detalharemos também o método no qual esse trabalho é baseado: alinhamento não supervisionado de subespaçoes.

3.2.3 Abordagens Baseadas em Features

Abordagens baseadas em *features* possuem como objetivo encontrar um mapeamento que transforme as *features* do conjunto fonte em *features* mais semelhantes às do conjunto alvo. A obtenção desse mapeamento tornaria possível treinar um modelo nas *features* transformadas que quando aplicado no domínio alvo obteria uma performance mais satisfatória. Tal abordagem se justificaria pelo limite da seção 3.1.4, pois fazer com que $p_S(t(x))) \approx p_T(x)$ (onde $t(x)$ é o mapeamento) diminuiria o valor da divergência entre os domínios, o que limitaria o erro de generalização e melhoraria a performance. Seguem essa linha de raciocínio os trabalhos (LONG et al., 2014), (FERNANDO et al., 2013) e (TZENG et al., 2015).

O ideial seria que o mapeamento fizesse com que $p_S(y|t(x))) \approx p_T(y|x)$, pois dessa forma treinar no domínio fonte garantiria que o modelo convergiria no domínio alvo, já que ele estimaria a distribuição de probabilidade correta para fazer a classificação. Acontece que não é garantido que o mapeamento gere essa propriedade, mesmo que ele aproxime as distribuições marginais. É necessário que certas suposições sejam válidas para afirmar que o mapeamento garante o alinhamento das distribuições condicionais também, como argumentam (GONG et al., 2016a) e (ZHANG et al., 2013).

Técnicas que se baseiam na obtenção de subespaços em comum para ambos os domínios são chamadas de mapeamento de subespaços, de modo que o alinhamento de subespaços é um caso especial do mapeamento de subespaços. Enquanto o alinhamento de subespaços codifica o fato de que diferentes domínios são gerados de acordo com diferentes distribuições apenas implicitamente, variações dessa estratégia tentam basear o alinhamento tanto nas distribuições de probabilidade que geram os dados quanto nos subespaços em comum(SUN; SAENKO, 2015).

Outros métodos assumem que as diferenças entre os domínios têm sua origem principalmente atribuída a ruídos indesejados derivados do método de obtenção das observações, como é o caso de conjuntos de dados de imagens. Para esses casos, é interessante achar uma representação invariável para ambos os domínios que elimine tais ruídos (TORRALBA; EFROS, 2011).

Outros trabalhos procuram obter as represetações empregando redes neurais, como *autoencoders* . *Autoencoders* são redes que procuram reconstruir as entradas que passam por suas camadas escondidas, normalmente após estas entradas passarem por algum tipo de restrição, como uma diminuição de sua dimensionalidade. Posteriormente, as camadas escondidas podem ser usadas como *features* de treino (BENGIO; COURVILLE; VINCENT, 2013).

A próxima seção discute com detalhes a estratégia de alinhamento de subespaços, além de toda o referencial teórico necessário para compreender a motivação e o funcionamento desse método.

3.2.4 Alinhamento de Subespaços

O alinhamento de subespaços é um método que procura, para dois domínios distintos, encontrar subespaços em comum e alinhá-los (aproximá-los). Esses subespaços seriam compostos pelas *features* cujos componentes codificam apenas as informações relevantes para a tarefa de REN, e não ruídos específicos de cada domínio. Isso permitiria que um classificador treinado num subespaço fonte tenha maior probabilidade de convergir quando aplicado ao subespaço do domínio alvo. Existem várias maneiras de derivar subespaços, dentre elas o alinhamento de subespaços.

O alinhamento de subespaços é um método que depende que existam dados anotados tanto no domínio fonte quanto no alvo. Isso pode parecer redundante, pois se já temos rótulos no domínio alvo, para que adaptar? Existem casos onde o conjunto de treino e o conjunto de teste, apesar de serem rotulados com as mesmas classes, foram gerados através de diferentes distribuições de probabilidade. Isso pode ocorrer quando um processo de anotação de *features* envolve mais de um anotador ou ainda que os rótulos tenham sido obtidos através de metodologias diferentes. De qualquer forma, é necessário um método de adaptação de domínio para que o modelo obtenha a performance ótima.

No caso de adaptação de domínio para REN, dois conjuntos de dados podem ser anotados com os mesmos rótulos, mas a probabilidade de uma palavra ser associada a um rótulo é diferente quando se muda de um domínio para o outro, pois o que se considera como entidade nomeada também muda. O HAREM, por exemplo, associa considera o nome de pessoas como entidades nomeadas, enquanto o GeoCorpus não o faz.

Esse também foi o método escolhido para testar as hipóteses desse trabalho, pois a sua aplicação permite explorar a existência ou não de subespaços em comum para a tarefa de REN para o português, além da sua ligação direta com o conceito de divergência entre domínios, que é central para a teoria de adaptação de domínio.

3.2.4.1 Transformações Lineares de Matrizes Simétricas: Sabemos que toda matriz $A \in R^{n \times m}$ pode ser interpretada como um operador linear $f : R^m \rightarrow R^n$ através da equação $\vec{y} = A\vec{x}$. Geometricamente, podemos interpretar uma transformação linear como uma rotação, translação ou ainda um escalonamento do espaço por algum valor em determinada direção. Matrizes diagonais determinam transformações de interesse para a atual discussão. Matriz diagonal é uma matriz em que $(a_{ij}) = 0$ para todo $i \neq j$. Vamos considerar a matriz diagonal Λ cujos elementos da diagonal são $\lambda_1, \lambda_2, ..., \lambda_n$. Sejam $e_1, e_2, ..., e_n$ os vetores canônicos de R^n, o efeito de uma matriz diagonal sobre o vetor e_i é simplesmente o seu escalonamento por λ_i, ou seja, a seguinte equação é verdadeira para todo natural $i >= n$:

$$(1)\ \Lambda e_i = \lambda_i e_i$$

Essa equação é denimonada equação dos autovalores, ou seja, os **autovetores** de uma matriz diagonal são os vetores da base do espaço vetorial, e os **autovalores** associados a esses autovetores são os elementos da diagonal definidos pela matriz Λ. Os autovetores podem ser pensados como as direções afetadas pela transformação somente por um escalonamento, e não por uma rotação. A proporção no qual a direção é escalonada é determinada pelos autovalores.

Uma matriz simétrica é aquela que coincide com sua transposta, ou seja, $A = A^T$. A seguinte equação é verdadeira para toda matriz simétrica A:

$$A = \Phi \Lambda \Phi^T$$

Essa transformação tem uma interpretação geométrica interessante. As matrizes Φ e Φ^T são matrizes de rotação. No caso de uma matriz de rotação, sua inversa é a sua prória transposta, ou seja, $\Phi^T = \Phi^{-1}$. Isso significa que se Φ for uma rotação de um ângulo θ ao redor de um eixo, Φ^T é simplesmente a rotação ao redor do mesmo eixo na direção oposta pelo mesmo ângulo θ. A matriz $\Phi \Lambda \Phi^T$ portanto tem o efeito de rotacionar o espaço, escaloná-lo, e depois rotáciona-lo para a posição original. É possível ilustrar esse conceito mostrando a matriz que determina uma escalonamento do espaço na direção do eixo de 45^o, como faz a figura 3.1:

Na figura, os vetores indicados pela cor vermelha são os originais, enquanto os azuis são os mesmos vetores após sofrerem a transformação. No canto inferior esquerdo, vemos vetores que sofreram uma rotação, e a direita vetores que foram somente escalonados pela transformação.

Já que Λ só consegue escalonar sem efetuar a rotação de vetores paralelos aos vetores canônicos, é necesśario rotacionar o espaço de forma obter essa configuração. Esse é o papel de Φ^T. Subsequentemente, a matriz Λ escalona por um fator de 2 os vetores paralelos ao eixo x, e por um fator de 1 os vetores paralelos ao eixo y. Após essa transformação,

$$A = \Phi\Lambda\Phi^T = \begin{pmatrix} 1.5 & 0.5 \\ 0.5 & 1.5 \end{pmatrix} \qquad \Lambda = \begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix} \qquad \Phi(45^0) = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix}$$

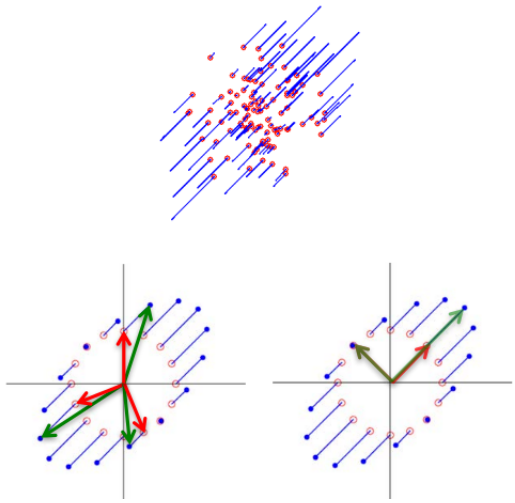

Figura 3.1: Demonstração do efeito da matriz de transformação A sobre os vetores do plano

Fonte: https://ocw.mit.edu/courses/brain-and-cognitive-sciences

os vetores são novamente rotacionados para sua posição original, obtendo-se o resultado final indicado pela figura 3.1.

A próxima pergunta a ser respondida é: quais os autovetores dessa transformação? Colocado de outra maneira, quais são as direções especiais que são apenas escalonadas, e não rotacionadas? Para resposnder a esse questionamento, é necessário solucionar a seguinte equação para os valores de x_i:

$$\Phi\Lambda\Phi^T x_i = a_i x_i$$

multplicando por Φ^T à esquerda em ambos os lados:

$$\Phi^T\Phi\Lambda\Phi^T x_i = (\Phi^T a_i)x_i$$

Isso elimina Φ à esquerda, pois $\Phi\Phi^T = I$, com I a matriz identidade. Logo, obtemos:

$$(2)\ \Lambda\Phi^T x_i = (\Phi^T a_i)x_i$$

Que é exatamente a equação dos autovalores para a matriz diagonal, como visto anteriormente. Comparando as equações (1) e (2), chegamos a conclusão que $e_i = \Phi^T x_i$. Multplicando por Φ em ambos os lados, obtemos a expressão para os autovetores da transformação $\Phi\Lambda\Phi^T$:

$$x_i = \Phi e_i$$

Ou seja, os autovetores da transformação $\Phi\Lambda\Phi^T$ são os vetores canônicos rotacionados por Φ, ou seja, são as próprias colunas da matriz Φ! Já vimos que o autovalor associado a cada um desses autovetores é λ_i, ou seja, os autovalores para cada autovetor x_i é o respectivo valor λ_i em Λ.

Chegamos portanto às seguintes importantes conclusões: para toda matriz simétrica A, podemos escrever $A = \Phi\Lambda\Phi^T$. Essa decomposição é chamada decomposição em valores singulares de A. Determinamos também que os autovetores de A são as colunas de Φ, e os autovalores os elementos da diagonal de Λ. Vamos agora unir os conceitos discutidos nessa seção com a matriz de covariância e distribuições normais multivariadas, o que leva naturalmente ao método de redução de dimensionalidade chamado análise de componentes principais.

3.2.4.2 Distribuição Normal Multivariada Uma maneira de interpretar dados codificados por vetores de alta dimensão é pensar numa distribuição de probabilidade que gera esses dados. Assumindo que temos um vetor de dimensão n, podemos pensar em cada componente do vetor como o valor assumido por uma variável aleatória X independente, de forma que a probabilidade do vetor é o valor da distribuição conjunta de probabilidade $p(x_1, x_2, ..., x_n)$, cujo valor é determinado pela equação $\prod_i^n p(x_i)$. Variáveis aleatórias nesse formato são denominadas independente e identicamente distribuídas (hipótese iid) (MURPHY, 2021).

Se fizermos a suposição que a nossa distribuição geradora é uma distribuição normal, ou seja, $D = N(\mu, \sigma^2)$, podemos interpretar a matriz de covariância como um operador linear que transforma uma distribuição normal de variância unitária em uma distribuição com variância e covariância arbitrárias. Primeiro, vamos ilustrar a distribuição de pontos que sigam uma distribuição normal de variância unitária e média nula:

A figura ilustra o conceito para duas dimensões para facilitar a visualização. Vemos que temos uma densidade de pontos maior perto da média,e a medida que nos afastamos desse valor central os pontos se tornam mais esparsos. A chave para entender a interpretação da matriz de covariância é respoender a seguinte pergunta: Se nossa distribuição geradora fosse uma distribuição normal de média nula e variância arbitrária, como os pontos estariam distribuídos? É a matriz de covariância que determina a transformação dos pontos que seguem uma distribuição normal isotrópica para uma arbitrária, portanto devemos definir seu conceito agora.

A covariância entre duas variáveis aleatórias quaisquer pode ser interpretada como o grau de correlação entre essas variáveis. Alta covariância significa um alto grau de interdependência estatística. Variáveis independentes portanto possuem covariância nula. A covariância entre duas variáveis aleatórias quaisquer é calculada de acordo com a seguinte fórmula:

$$COV(x, y) = \frac{1}{n}\sum_{1}^{n}(x_i - \mu_x)(y_i - \mu_y)$$

Em que μ_x e μ_y são os valores médios de X e Y, respectivamente. Se considerarmos uma sequencia de m vetores colunas de n dimensões, $x^{(1)}, x^{(2)}, ..., x^{(m)}$, cada vetor pode

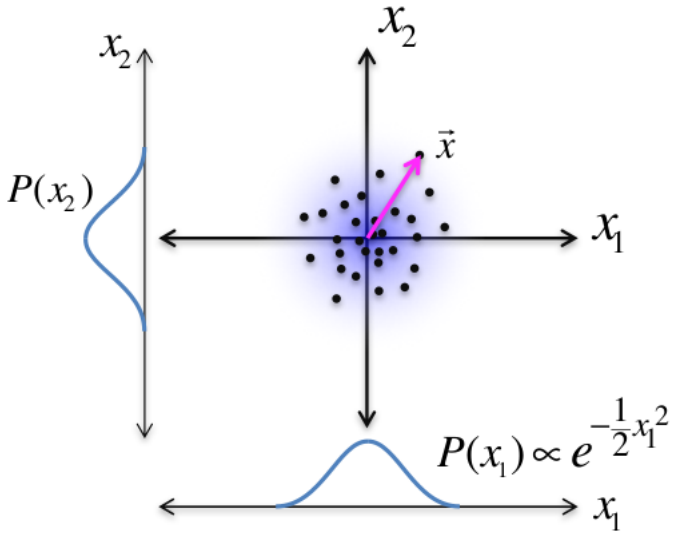

Figura 3.2: Distribuição normal multivariada de média variância unitária em duas dimensões.

Fonte: Fonte:https://ocw.mit.edu/courses/brain-and-cognitive-sciences

ser interpretado como n observações de uma variável aleatória. Com isso montamos a nossa matriz M de observações, de dimensões nxm:

$$M = \begin{bmatrix} x_{11} & x_{12} & ... & x_{1m} \\ x_{21} & x_{22} & ... & x_{2m} \\ ... & ... & ... & ... \\ x_{n1} & x_{m1} & ... & x_{nm} \end{bmatrix}$$

Definimos então C, a nossa matriz de covariância , onde $c_{ij} = COV(M^{(i)}, M^{(j)})$. A observação mais importante relativa a nossa matriz de covariância é que ela é simétrica, pois $COV(X, Y) = COV(Y, X)$. Nós já vimos que toda matriz simétrica pode ser interpretada como uma transformação linear decomposta em três matrizes: uma diagonal de escalonamento e duas de rotação. Portanto, nossa matriz C de covariância pode ser escrita da seguinte maneira, através da sua decomposição em valores singulares:

$$C = \Phi \Lambda \Phi^T$$

.

Mas qual o efeito dessa matriz sobre nosso conjunto de observações? Como já mencionado, um vetor no qual se aplica essa tranformação é mapeado para o vetor que seria observado para uma variável aleatória cuja distribuição possua variância determinada pela matriz Λ, ou seja, a matriz de escalonamento. A imagem 3.3 mostra a transformação de uma distribuição normal isotrópica para uma distribuição normal de variância arbitrária utilizando a matriz de covariancia:

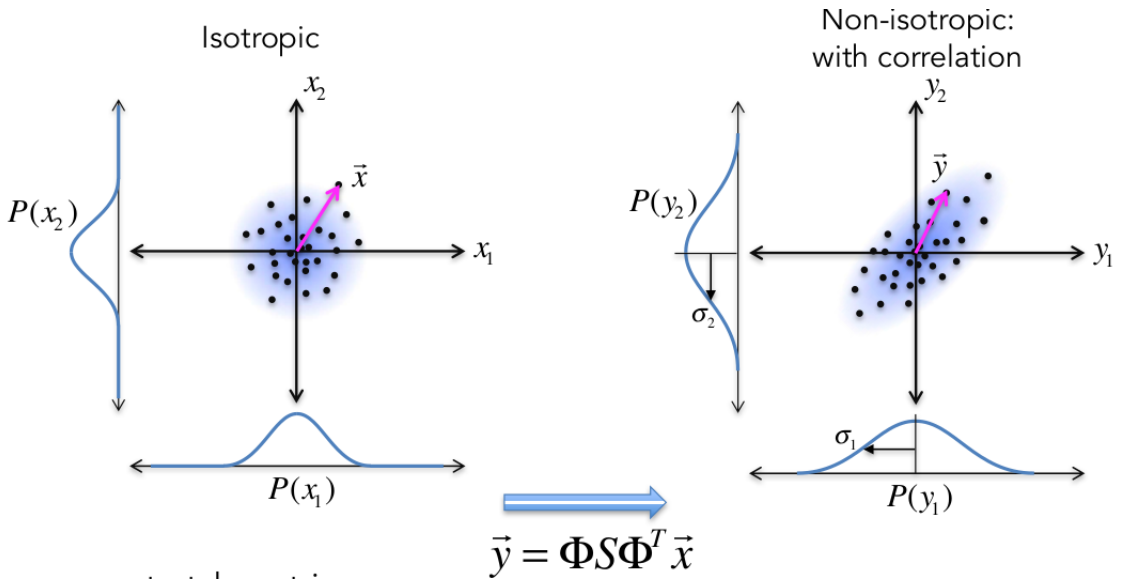

Figura 3.3: Efeito da matriz de covariância sobre a distribuição normal isotrópica.

Fonte: Fonte:https://ocw.mit.edu/courses/brain-and-cognitive-sciences

O interessante dessa interpretação é que podemos determinar os autovetores e os autovalores da matriz de covariância. Os autorevetores, como já demonstrado, são as colunas da matriz Φ, e são as direções de maior variância da nossa distribuição normal. Já os autovalores são os elementos diagonais de Λ, sendo os valores da variância de cada dimensão. Em geral, matrizes compostas de vetores de n dimensões possuem n autovetores unitários distintos. A imagem 3.4 mostra os autovetores obtidos para um conjunto de observações em duas dimensões:

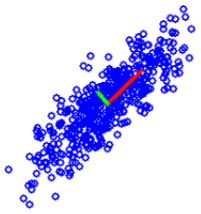

Figura 3.4: Autovetores obtidos através do PCA aplicado em um conjunto de observações em duas dimensões.

Fonte: Fonte:https://ocw.mit.edu/courses/brain-and-cognitive-sciences

Finalmente, com a teoria que desenvolvemos até então, podemos explicitar o funcionamento da análise de componentes principais.

3.2.4.3 Análise de Componentes Principais A análise de componentes principais opera sob a noção de que se for necessário reduzir a dimensionalidade de um conjunto de observações, a melhor maneira de fazê-lo é projetando os pontos em um espaço vetorial de dimensão $d < D$, com D a dimensão original, utilizando como colunas da matriz de projeção os d autovetores com os maiores autovalores associados a eles. Isso projetaria os pontos nas regiões de maior variância, preservando o máximo possível de informação.

A imagem 3.5 ilustra a intuição por trás do PCA. Se temos um conjunto de observações em duas dimensões, a melhor estratégia de redução de dimensionalidade é projetar os pontos na reta definida pelo autovetor da matriz de covariância. Já em três dimensões, se quisermos reduzir para duas dimensões podemos projetar os pontos no plano definido pelos dois autovetores da matriz de covariância, ou mesmo na reta definida pelo autovetor de maior autovalor associado. A ideia pode ser naturalmente estendida para qualquer número de dimensões D, em que a dimensão do conjunto de observações reduzidas é um número $d < D$.

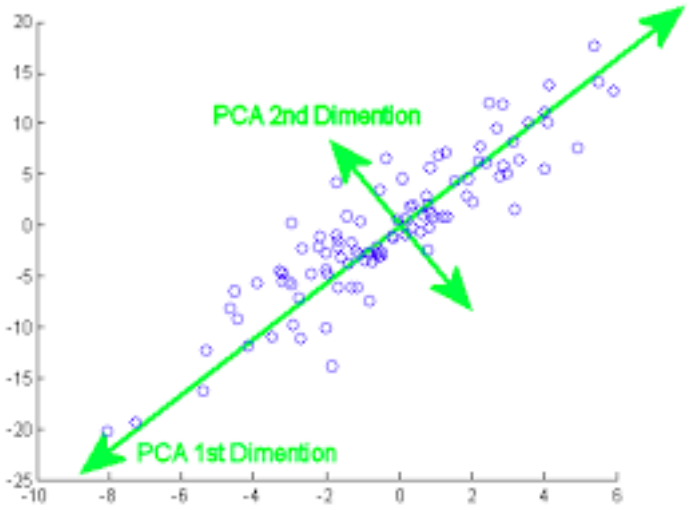

Figura 3.5: Projeção das observações no autovetor de maior variância.

Fonte: Fonte:https://www.dezyre.com/data-science-in-python-tutorial/principal-component-analysis-tutorial

Se considerarmos X_S e X_T as matrizes $D \times d$ contendo os d autovetores associados aos d maiores autovalores, a operação de projetar as *features* originais num conjunto de menor dimensionalidade é dada pela seguinte equação:

$$y = x^T X_S$$

Em que x possui dimensão D e y possui dimensão d. X_S e X_T são portanto as bases dos espaços vetoriais reduzidos. A projeção dos vetores originais em seus respectivos subespaços é o que codifica o fato de que eles foram selecionados de acordo com diferentes distribuições de probabilidade no domínio fonte e no domínio alvo, pois as matrizes de covariância de cada domínio são distintas. Mas como representamos a mudança do domínio

fonte para o domínio alvo? É nessa etapa que entra o alinhamento de subespaços, através do cálculo de uma transformação linear que aproxima os subespaços projetados por X_S e X_T.

3.2.4.4 Alinhamento de Subespaços consiste no cálculo de uma transformação linear, representada por uma matriz, que transforme um vetor expresso pelos vetores da base de X_S em vetores expressos pelos vetores da base de X_T. Encontrar essa transformação linear implica resolver o seguinte problema de minimização:

$$M^* = \arg\min_M ||X_S M - X_T||_F^2$$

Isto é, a matriz M^* que mais se aproxime de expressar os vetores projetados por X_S como vetores projetados por X_T. Na equação acima, $||.||_F^2$ é norma matricial de Frobenius, definida como a raíz da soma dos quadrados dos elementos de uma matriz. A seguinte derivação nos permite achar uma expressão em termos de X_S e X_T para efetuar o cálculo de M.

$$\begin{aligned} M^* &= \arg\min_M ||X_S M - X_T||_F^2 \\ &= \arg\min_M ||X_S^T X_S M - X_S^T X_T||_F^2 \\ &= \arg\min_M ||M - X_S^T X_T||_F^2 \\ &= X_S^T X_T \end{aligned} \tag{3.1}$$

(BJERVA; KOUW; AUGENSTEIN, 2019)

Ou seja, o novo sistema de coordenadas é definido pela operação $X_S X_S^T X_T$. O classificador é treinado nas *features* do domínio fonte transformadas e aplicado sobre as *features* do domínio alvo projetadas por X_T.

A imagem 3.6 demonstra graficamente o funcionamento do alinhamento não supervisionado de subespaços. Note que a divergência na figura, originalmente possuindo o valor de ΔD_1, quando aproximada apresenta uma divergência menor de valor ΔD_2.

3.2.5 Resumo do Capítulo

Este capítulo teve o propósito de definir e discutir com maior nível de detalhes os conceitos relativos ao campo de adaptação de domínio. Dentre as ideias discutidas, é de grande relevância para esse trabalho o limite para o erro de generalização, e mais especificamente a possibilidade de limitar essa quantidade através da medida de divergência entre os domínios.

Para que possamos manipular a divergência, precisamos de métodos que se proponham a diminuí-la. Foi com esse objetivo que discutimos as teorias de decomposição em valores singulares e a interpretação da matriz de covariância, pois essas ideias levam ao método utilizado nos experimentos, que é o alinhamento de subespaços.

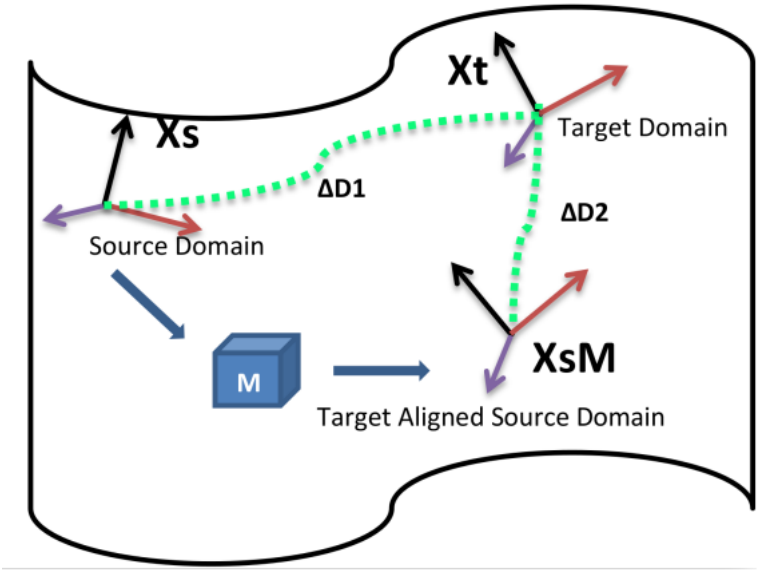

Figura 3.6: Ilustração do alinhamento não supervisionado de subespaços.

Fonte: Fonte: (FERNANDO et al., 2013)

Com o intuito de compreender com mais profundidade os dados que utilizamos durante os experimentos, o próximo capítulo aborda com mais detalhes as características dos conjuntos relativas às entidades nomeadas.

Capítulo 4

CONJUNTOS DE DADOS

Quatro conjuntos de dados foram utilizados durante o curso dos experimentos: HAREM, GeoCorpus, LeNER-Br e Cojur. O HAREM foi considerado o domínio fonte, e é composto por uma gama de textos derivados de diversos estílos. O GeoCorpus é um *corpus* de entidades geólogicas, e o LeNER-BR e Cojur são ambos *corpus* de entidades jurídicas, como lei e decreto de lei.

A primeira tarefa foi converter os dados desses conjuntos em um formato em comum para que eles pudessem ser consumidos pelos modelos de aprendizagem de máquina utilizados para realizar a classificação. Com esse obejetivo, o formato escolhido foi o BIO, ou seja, uma palavra possuia um rótulo associado que fazia parte do conjunto $\{B, I, O\}$. O rótulo B indica que uma palavra encontra-se no começo de uma entidade nomeada, o rótulo I indica que a palavra está dentro da entidade, mas não no começo, e o rótulo O indica que a palavra não faz parte de nenhuma entidade. Esse formato foi definido pela conferência CoNLL (SANG; MEULDER, 2003), e é muito utilizado em configurações onde se utiliza modelos de aprendizado de máquina para resolver o problema de REN.

O formato preciso do arquivo que contém os dados era de uma palavra por linha, onde cada palavra possuia o rótulo associado a ela separado por uma espaço. Ademais, duas quebras de linhas seguidas separavam um *batch* de outro, em que um *batch* é uma sequencia de observações utilizada como unidade de treino. O seguinte trecho do Harem ilustra essa configuração:

```
A O
Reforma O
Religiosa O
e O
Política O
na O
Inglaterra O

Henrique B-
```

```
VIII I-
O O
...
```

Vemos que a expressão "Henrique VIII"é marcada como uma entidade nomeada através da atribuição dos rótulos B e I, e como nenhuma outra palavra no exemplo faz parte de uma entidade, a elas são atribuídos os rótulos O. Por fim, Obsevamos também as duas quebras de linha seguidas separando um *batch* de outro.

A divisão dos conjuntos foi da forma 70/15/15, através de um algoritmo que recebia como entrada o arquivo contendo todas as observações do conjunto e retornava os arquivos de treino, desenvolvimento e teste. Para garantir uma boa distribuição da ocorrência das entidades em cada arquivo, foi promovida também o embaralhamento dos *batchs* antes de efetuar o particionamento.

A seguir, vamos descrever a estrutura e o papel de cada conjunto de dados nos experimentos realizados, a começar pelo Harem.

4.1 HAREM

O HAREM (SANTOS; CARDOSO, 2007) é um conjunto para a tarefa de reconhecimento de entidades nomeadas para o português que se encontra distribuído em duas versões, o primeiro e o segundo HAREM. Os documentos que compõem esse conjunto não são selecionados de um domínio específico, de maneira que suas origens advém de gêneros textuais diversos. Nesse trabalho foi utilizado como *corpus* o segundo HAREM, pois a organização desse conjunto se propôs corrigir e aperfeiçoar alguns aspectos da primeira versão.

Dez classes distintas são utilizadas nesse conjunto de dados para classificar as entidades nomeadas, são elas: ORGANIZAÇÃO, PESSOA, LOCAL, COISA, TEMPO, VALOR, OBRA, ACONTECIMENTO, ABSTRAÇÃO e OUTRO. A classe Fora não se encontra naturalmente dentro do conjunto, mas foi acrescentada como rótulo para classificar as palavras que não são entidades. Cada categoria possui também inclusa um certo número de subcategorias, a lista comleta das categorias e subcategorias pode ser referenciada no link: ⟨https://www.linguateca.pt/aval_conjunta/HAREM/harem_ing.html⟩.

O HAREM, precisou ser convertido de seu formato original, em XML, para o formato mencionado no começo do capítulo, o BIO. Uma entidade no formato original do Harem está envelopada por uma etiqueta contendo atributos como categoria, subcategoria e identificador. A entidade Europa, por exemplo, encontra-se no seguinte formato:

```
<EM ID="id" CATEG="LOCAL" TIPO="HUMANO">Europa</EM>
```

Nesse caso específico, Europa deveria ser convertida para o formato "Europa B". Para realizar a conversão, foi desenvolvido um algoritmo que processava o conjunto no formato XML e retornava o conjunto no formato BIO. Após passar pelo processo de conversão, o conjunto de dados possui um total de 84271 linhas, além de 6593 instâncias de entidades

Tamanho das Entidades	Sentenças	Entidades	Entidades por Sentenças	Classes
2,07	6593	6593	2,62	11

Tabela 4.1: Características do Harem

nomeadas. No Harem é relativamente rara a presença de entidades que não possuam rótulos I, ou seja, que seja composta somente por uma palavra. Esse fato constrasta com conjuntos de dados como o Geocorpus, que possuem diversos exemplos de entidades que seguem essa configuração.

A tabela 4.1 demonstra o valor médio de algumas características quantitavas do Harem, e nos permitirá fazer a comparação do Harem com o outros *corpus* usados como domínios fontes.

4.2 LENER-BR

O LeNER-Br (ARAUJO et al., 2018) é um *corpus* de domínio específico para a tarefa de REN, em que o foco é direcionado para entidades do meio jurídico, como legislação e jurisprudência. Apesar disso, o Lener inclui como entidades instâncias pertencentes a classes tradicionais, como PESSOA e LOCAL. São um total de sete classes, sendo elas JURISPRUDÊNCIA, ORGANIZAÇÃO, PESSOA, TEMPO, LOCAL e LEGISLAÇÃO.

O arquivo processado final do LeNER possui um total de 328464 linhas, significativamente maior do que o Harem, além de 12248 entidades nomeadas encontradas no texto. A tendência dessas entidades é a de possuírem um grande número de rótulos I, ou seja, são entidades longas. Outra característica marcante é a presença de numerais e abreviações, derivados da numeração das leis no meio jurídico. O seguinte trexo exemplifica uma típica entidade pertencente ao LeNER-BR:

```
art B-
. I-
178 I-
, I-
II I-
, I-
do I-
CPC I-
```

Isso implica um desafio maior em adaptar do Harem para o Lener, pois há poucas instâncias semelhantes ocorrendo dessa forma no Harem.

A tabela 4.2 demonstra as características do Lener, assim como foi feito com o Harem. O que chama a atenção ao comparar as duas tabelas é o tamanho médio das entidades, o qual é maior no Lener. Esse fato se deve principalmente às entidades do tipo LEGISLAÇÃO, as quais usualmente são compostas por múltiplas palavras. Entidades como PESSOA e LOCAL possuem um tamanho menor, o que acaba diminuindo o tama-

Tamanho das Entidades	Sentenças	Entidades	Entidades por Sentenças	Classes
3,63	13937	12248	0,87	8

Tabela 4.2: Características do Lener

Tamanho das Entidades	Sentenças	Entidades	Entidades por Sentenças	Classes
1,6	6379	4835	0,75	13

Tabela 4.3: Características do GeoCorpus

nho médio. Por fim, já que o Lener é um *corpus* maior, naturalmente temos a presença de mais entidades do que no Harem.

4.3 GEOCORPUS

Outro conjunto de dados de domínio específico é o Geocorpus (AMARAL et al., 2017). Esse *corpus* têm o intuito de servir como base de dados de entidades geológicas, como rios, rochas e diferentes eras. O GeoCorpus não possui incluido em suas categorias entidades normalmente consideradas na tarefa de REN, como pessoa e valor, optando por anotar apenas o que seriam entidades nomeadas geológicas.

O Geocorpus é também o possuidor do maior número de classes dentre os *corpus* considerados nos experimentos, sendo o seu número 13 no total. Ao fim do processamento, o arquivo contendo o *corpus* no formato BIO possui 174772 linhas e 4835 entidades nomeadas, muitas das quais são formadas por apenas uma palavra, ou seja, possuem associadas a si apenas um rótulo B.

Um trecho do GeoCorpus ajudará a ilustrar o formato das entidades consideradas:

```
no O
decorrer O
do O
Cenozóico B-
e O
no O
Quaternário B-
```

A dificuldade para adaptar do Harem para o GeoCorpus se encontra no tamanho médio das entidades, as quais são menores no GeoCorpus, e na ausência completa da anotação de entidades que são consideradas como tal no Harem. Por tal motivo, um modelo treinado no Harem pode ser levado a anotar entidades que não deveriam ser anotadas quando se efetua a transferência para o GeoCorpus.

Analisando a tabela 4.3, vemos que o fenômeno do GeoCorpus possuir muitas entidades marcadas apenas com o rótulo “B”é quantificado pelo tamanho médio das entidades, que é apenas 1,6.

Tamanho das Entidades	Sentenças	Entidades	Entidades por Sentenças	Classes
3,04	9486	8617	0,9	9

Tabela 4.4: Características do Cojur

4.4 COJUR

O último *corpus* de domínio específico a ser discutido é o Cojur. De maneira semelhante ao LeNER-BR o Cojur é um conjunto de entidades jurídicas, mas diferentemente do LeNER, que considera entidades como pessoa, tempo e local, o Cojur se assemelha ao GeoCorpus ao considerar somente entidades relevantes para o contexto jurídico. São 8 classes, dentre as quais estão inclusas LEI, DLEI (decreto de lei) e DEC (decreto).

Todos os *corporas* utilizados nos experimentos sofrem do fenômeno de desbalanceamento de classes, mas o Cojur possui essa característica acentuada pela presença majoritária de entidades do tipo decreto. São 334224 linhas no arquivo final, além da presença de 8618 entidades. O seguinte trexo ilustra esse aspecto do Cojur:

```
Decreto B-
no I-
92.689 I-
...
Decreto B-
no I-
92.693 I-
, O
de O
19 O
de O
maio O
de O
1986 O
```

As possíveis dificuldades de adaptar para o Cojur são semelhantes às das existentes no GeoCorpus, ou seja, a falta de entidades tradicionais no *corpus*, as quais estão presentes no Harem, apesar de diferentemente do GeoCorpus o Cojur possuir entidades maiores. A tabela 4.4 demonstra as características quantitativas do Cojur, como feito com os outros conjuntos.

4.5 AUTOVALORES

Como já mencionado, os autovalores derivados da decomposição da matriz de covariância, juntamente com os autovetores, nos dão as direções de maior variância do conjunto de observações. Quanto maior o autovalor associado a um autovetor, mais importante é aquela direção para preservar a variância dos dados originais após a projeção pelo PCA. É

interessante traçar um gráfico de autovalores para cada conjunto de dados com o intuito de obtermos uma intuição de quando exatamente os autovalores começam a diminuir, de maneira que possamos tomar uma decisão em relação a dimensão do subespaço de projeção.

As imagens da figura 4.1 foram obtidas através da projeção dos autovalores em escala logarítmica. Vamos analisar o padrão que esses valores assumem para cada conjunto, e verificar se podemos inferir a dimensão ótima do subespaço de projeção. É possível observar que apesar dos valores individuais de cada autovalor diferirem entre um conjunto e outro, o que é reflexo das diferentes matrizes de covariância, o comportamento dos autovalores é muito semelhante. Isso parece indicar que as direções de maior variância se preservam entre os conjuntos.

Em outras palavras, sabemos que cada autovalor está associado a um único autovetor, e é o valor do autovalor que determina a magnitude desse vetor. Para cada conjunto, vemos que valores muito semelhantes de autovalores seguem o mesmo padrão de comportamento, como indicado pela figura 4.1. É por isso que podemos inferir que as bases de cada subespaço são muito semelhantes, ou seja, as direções de maior variância são praticamente idênticas entre conjuntos. Esse fenômeno provavelmente se deve ao fato de que a única mudança entre uma matriz de covariância e outra é a remoção ou inserção de colunas, as quais representam uma determinada palavra do vocabulário. Inserir a informação das classes nas colunas provavelmente permitiria codificar mais informação nas matrizes de covariância, mas então o método já não poderia ser chamado de não-supervisionado.

Seria razoável inferir que a dimensão ótima para aplicar o método se encontra na faixa de 80, o que preservaria a maior parte das direções relevantes. Na prática que aconteceu foi que cada conjunto apresentou um valor ótimo para o número de dimensões dos subespaços, como será discutido com detalhes nas seções dos experimentos.

4.6 RESUMO DO CAPÍTULO

Neste capítulo, discutimos as características dos conjuntos de dados utilizados nos experimentos, com foco em alguns aspectos quantitativos que nos fornecem intuições acerca da distribuição e natureza das entidades. Ademais, discutimos os gráficos de autovalores, e através da análise desses gráficos concluímos que as direções de maior variância são as mesmas entre conjuntos, com a diferença entre elas sendo apenas a magnitude das direções, definida pelos autovalores.

Estamos portanto prontos para entrar na descrição dos experimentos e resultados obtidos por esse trabalho, tópicos esses que serão discutidos nos capítulos 5 e 6.

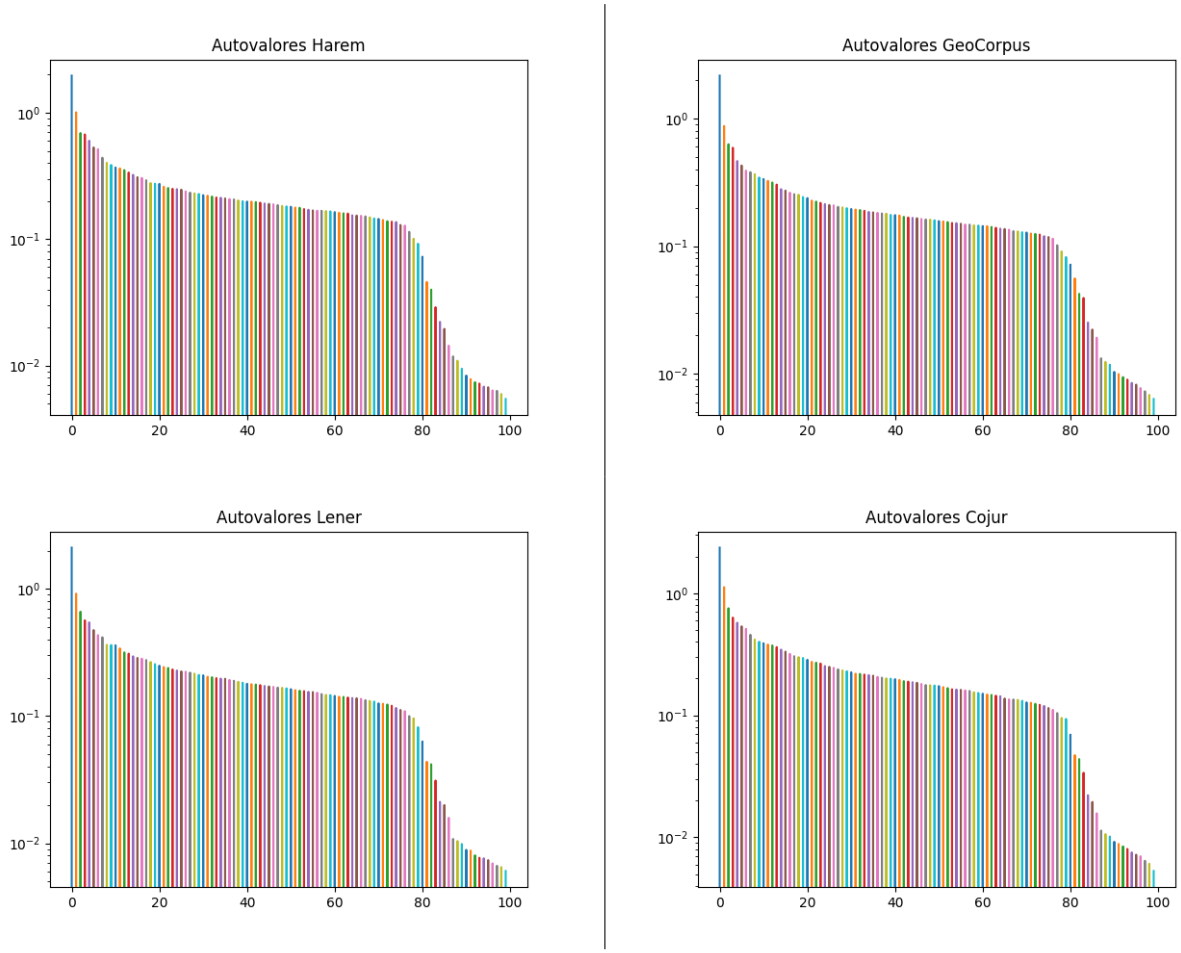

Figura 4.1: Gráficos dos autovalores. São 100 autovalores, um para cada dimensão. O eixo y demonstra o valor de cada um dos 100 autovalores, que são enumerados no eixo x.

Capítulo 5

ADAPTAÇÃO DE DOMÍNIO PARA REN BASEADO EM REPRESENTAÇÕES NÃO CONTEXTUAIS

Esse capítulo discute a arquitetura, metodologia e resultados do experimento baseado em representações não contextuais. Tais representações são ditas não contextuais pois são derivadas diretamente de um modelo de linguagem Glove (PENNINGTON; SOCHER; MANNING, 2014) ao invés de uma LSTM, de forma que as representações não levam em consideração o contexto. Em suma, o método de alinhamento de subespaços foi aplicado sobre os vetores de característica derivados do Glove, sendo a motivação desse experimento poder comparar essa adaptação com a variação de aplicar o alinhamento sobre a saída da LSTM. Essa comparação nos permitirá verificar se devemos especificar o momento de adaptação dos vetores de característica de acordo com o modelo utilizado para classificar, pois os vetores de característica efetivamente utilizados para a classificação pelo CRF são as representações que codificam contexto derivadas da LSTM. A figura 5.1 ilustra a estratégia utilizada nos experimentos deste capítulo.

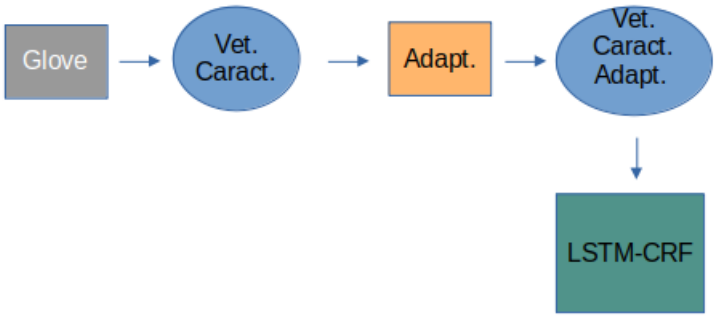

Figura 5.1: Estratégia do experimento baseado em representações não contextuais

Na figura, "adapt." representa o alinhamento de subespaços que gera nossos vetores de características adaptados, os quais são passados para o modelo LSTM-CRF para realizar a classificação usando os rótulos BIO.

5.1 ARQUITETURA UTILIZADA

A arquitetura utilizada para realizar o experimento foi a mesma do artigo de (LAMPLE et al., 2016), com a única diferença sendo a remoção da função de ativação aplicada sobre a saída da LSTM. Ou seja, os vetores de característica da entrada representando uma palavra são passados por uma LSTM bidirecional, e a camada derivada dessa computação (que codifica contexto) é passada para o CRF, o qual realiza a classificação através dos rótulos B, I e O. A imagem 5.2 ilustra a arquitetura utilizada pelo artigo.

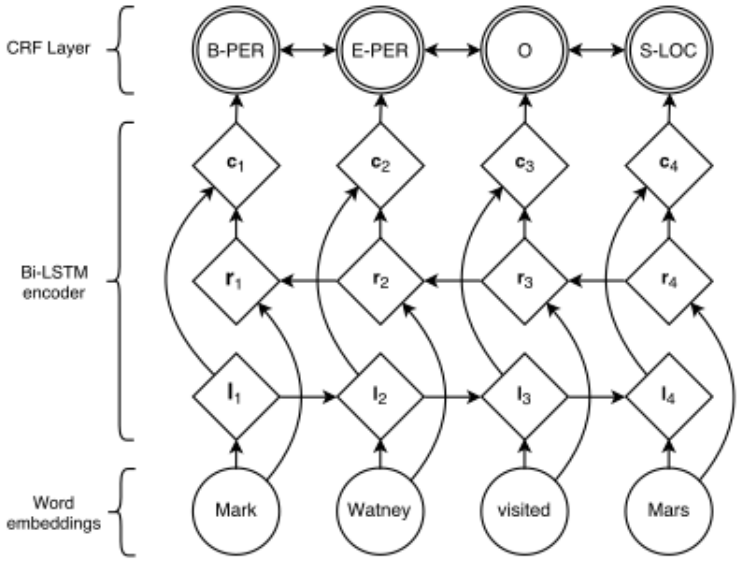

Figura 5.2: Arquitetura utilizada Por Lample et.al.

Fonte: (LAMPLE et al., 2016)

A LSTM possui a seguinte implementação:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i)$$

a equação acima define o *add gate*, o qual é utilizado para determinar a informação que será inserida no contexto atual. O *add gate* depende do *embedding* x_t, da camada escondida anterior h_{t-1} e do contexto anterior c_{t-1}, a aplicação do símbolo σ indica a funçao sigmoide.

A equação da camada de contexto é como se segue:

$$c_t = (1 - i_t) \odot c_{t-1} + i_t \odot \tanh(W_{xc}x_t + W_{hc}h_t + b_c)$$

E finalmente, usando c_t podemos definir o *output gate* e em sequência a camada escondida h_t:

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o)$$
$$h_t = o_t \odot c_t$$

Note que, como a LSTM é bidirecional, duas redes como as especificadas acima são aplicadas, mas uma com orientação da esquerda para a direita e outra com orientação da direita para a esquerda, de maneira que para obter a camada escondida final é necessário concatenar as representações obtidas através da aplicação das duas redes, também chamadas de *foward* e *backward*. Cada rede possui seu próprio conjunto de pesos e parâmetros.

A camada escondida h_t é passada para o CRF, que computa para cada sequeência de observações $(x_1, x_2, ..., x_n)$ um possível *score*, que depende de uma sequência de rótulos $(y_1, y_2, ..., y_n)$, vamos portanto denotar o *score* como $S(X, Y)$. A ideia do CRF durante o treino é maximizar a probabilidade da sequência de rótulos correta, de forma que na etapa de classificação a resposta do seguinte problema de minimização seja a sequência correta de rótulos:

$$Y^* = \arg\max_{Y} S(X, Y)$$

Ou seja, a sequência de rótulos para X cujo *score* seja o maior possível entre todas as sequências.

5.2 METODOLOGIA

A arquitetura discutida acima foi treinada utilizando o algorítmo do gradiente descendente, com os parâmetros atualizados a cada nova iteração. Foi utilizada uma taxa de aprendizado de 0,01, além de um *dropout* de 0,5 na saída da LSTM. Prosseguindo, a camada escondida foi configurada para possuir 100 dimensões, a mesma dimensão da entrada.

O conjunto elegido como conjunto fonte foi o Harem, enquanto o GeoCorpus, LeNER-BR e Cojur assumiram o papel de conjuntos alvo. O motivo dessa escolha advém do fato de que o Harem foi construído a partir de um número maior de gêneros textuais do que os outros conjuntos, os quais possuem uma gama mais específica de entidades anotadas. Para cada um dos conjuntos, resultados iniciais foram coletados ao aplicar o Harem sem adaptação em seus conjuntos de desenvolvimento, o que permitiu verificar a variação de performance ao aplicar o modelo adaptado.

O único hiperparâmetro do alinhamento de subespaços é a dimensão d dos subespaços, consequentemente é necessário uma maneira de estimar esse valor para cada conjunto de dados. Como os vetores de entrada possuíam 100 dimensões, qualquer dimensão entre 1 e 99 seria uma dimensão válida para ser testada. Como testar todas as possibilidades levaria muito tempo, determinou-se que as dimensões seriam testadas de 10 em 10, ou seja, foram 9 valores de dimensões testados, variando de 10 à 90.

Para cada valor de d, foram calculadas as divergências entre o Harem e o conjunto alvo aproximado. Esses resultados foram comparados com os valores das divergências originais com o intuito de verificar se as divergências entre domínios realmente diminuem e, mais importante, se há uma correlação entre a diminuição das divergências e o aumento da performance dos modelos adaptados. Por fim, para cada conjunto alvo, as dimensões que obtiveram maior sucesso em relação a performance são escolhidas para realizar uma avaliação quantitativa por classes. Em sequência, foi realizada também uma avaliação

Tabela 5.1: Harem aplicado no Geocorpus.

GEOCORPUS	precisao	recall	f-measure	variação
FORA	0,83	0,91	0,87	-0,13
EPC	0,88	0,94	0,91	-0,08
IDA	0,74	0,85	0,79	-0,18
OTR	0,22	0,36	0,27	-0,13
CAR	0,5	0,67	0,57	-0,24
UES	0,54	0,7	0,61	-0,3
EON	0,41	0,59	0,48	-0,52
SIL	0,21	0,34	0,26	-0,61
BAS	0,71	0,83	0,77	0,02
CON	0,32	0,49	0,39	-0,15
PRD	0,51	0,68	0,58	-0,42
ERA	0,71	0,83	0,77	-0,17
OGN	0,67	0,8	0,73	-0,27
TOTAL	0,56	0,69	0,62	-0,25

qualitativa para verificar o que exatamenteo o modelo aprendeu ao adaptar de um domínio para o outro, e se essas adaptações realmente foram razoáveis no contexto da classificação de entidades nomeadas.

5.3 RESULTADOS

Todos os resultados foram obtidos através da *micro averaging* dos rótulos B, I e O para cada classe, em que o cálculo da precisão, *recall* e *F-measure* totais foram obtidos através da média de cada um desses *scores* para cada classe.

5.3.1 Aplicação do Modelo sem Adaptação

As tabelas 5.1, 5.2 e 5.3 dizem respeito a aplicação do Harem sem nenhum tipo de adaptação nos conjuntos alvo. Isto é, o modelo foi treinado sobre o Harem no conjunto Glove original de 100 dimensões e posteriormente aplicado nos conjuntos alvo. Para obter a informação de qual seria a performance ótima esperada de um modelo, um modelo foi treinado e aplicado no próprio conjunto de dados, como numa tarefa de classificação tradicional. Essas performances base foram comparadas com a performance dos modelos treinados no HAREM e não adaptados, de forma a verificar o quanto a performance decaiu em decorrência de treinar em um modelo distinto. A comparação entre a performance base e a performance dos modelos não adaptados está armazenada na coluna "variação"nas tabelas. A variação demonstra o quanto a *f-measure* do modelo não adaptado decaiu em relação em relação ao modelo tradicional.

Tabela 5.2: Harem aplicado no LeNER-BR.

LENER	precisao	recall	f-measure	variação
FORA	0,9	0,95	0,92	-0,07
JURISPRUDENCIA	0,46	0,63	0,53	-0,4
ORGANIZACAO	0,65	0,79	0,71	-0,22
PESSOA	0,56	0,72	0,63	-0,35
TEMPO	0,71	0,83	0,77	-0,14
LOCAL	0,56	0,72	0,63	-0,31
LEGISLACAO	0,32	0,48	0,38	-0,6
TOTAL	0,6	0,73	0,66	-0,3

Tabela 5.3: Harem aplicado no Cojur.

COJUR	precisao	recall	f-measure	variação
FORA	0,58	0,73	0,65	-0,35
LEI	0,45	0,62	0,52	-0,39
DEC	0,44	0,61	0,51	-0,38
LCOMP	0,67	0,8	0,73	-0,22
DLEI	0,37	0,49	0,42	-0,44
DLEG	0,5	0,67	0,57	-0,29
MP	0,6	0,75	0,67	-0,26
EC	0,75	0,86	0,8	-0,2
TOTAL	0,54	0,69	0,61	-0,31

5.3.2 Variação das Divergências

Com o intuito de determinar se a diminuição da divergência entre dois domínios encontra-se relacioada com uma melhora de performance, como indica o limite da seçao 3.1.4, é ncessário verificar se a divergência realmente diminui a partir do processo de projeção e aproximação dos subespaços.

Três divergências foram escolhidas para avaliação, cada uma captando um aspecto de divergência entre domínios. A primeira é simplesmente a distância entre as centroides dos conjuntos de dados, o que representaria uma distância geométrica entre ambos. As outras duas divergências são uma medida da discrepância das distribuições de probabildades que geram os conjuntos, são elas a divergência de Kullback-leibler (KL) (JOHNSON; SINANOVIC, 2001) e a divergência de Jensen Shannon (JS) (FUGLEDE; TOPSOE, 2004). A única diferença entre essas duas medidas é o fato de que a divergência de Jason Shannon é simétrica, enquanto a Kullback-leibler não possui essa propriedade.

Os gráficos da figura 5.3 demonstram o valor da divergências KL para o Geocorpus, LeNER-BR e Cojur respectivamente de acordo com d. A reta constante no gráfico indica o valor da divergência original, com $d = 100$, para que se possa acompanhar visualmente a oscilação da divergência para os diferentes valores de d.

As imagens 5.4 e 5.5 fazem o mesmo para as divergências JS e Centróide respectivamente.

5.3.3 Resultados dos Modelos Adaptados

As dimensões que obtiveram a melhor performance foram selecionadas para realizar uma avaliação quantitativa por classe, em que a performance do modelo para cada classe foi avaliada em relação às métricas de performance, *recall* e *F-measure*. Adicionalmente, foi realizada uma avaliação qualitativa ao se analisar os arquivos de resultados e verificar o que exatamente o modelo mudou em sua estratégia de classificação na tentativa de adaptar. As tabelas 5.4, 5.5 e 5.6 indicam os resultados para o Geocorpus, LeNER-BR e Cojur, com d assumindo os valores de 80, 80 e 60 respectivamente.

5.3.4 Discussão

Um dos questionamentos relativos aos experientos era se a aplicação de alinhamento de subespaços realmente seria capaz de diminuir as divergências entre os domínios. Analisando as figuras 5.3, 5.4 e 5.5 é possível verificar que, para as três medidas de divergência consideradas, a tendência é a diminuição dessa divergência quanto menor é a dimensão d do subespaço. Não fica claro porém se esse é um fenômeno derivado do alinhamento de subespaços como um todo (projeção e aproximação) ou apenas da projeção, pois pontos que se encontravam mais distantes em maiores dimensões podem ter se aproximado ao serem projetados em dimnsões menores.

A próxima pergunta a ser respondida é se a diminuição da divergncia está relacionada de alguma maneira com o aumento da performance. Ao vermos os resultados das tabelas 5.4, 5.5 e 5.6 aferimos que houve um aumento da performance geral do GeoCorpus, com um ganho de 0,02% na média dos *scores*. O LeNER não obteve nenhuma melhora em

Figura 5.3: Divergência KL para os Conjuntos

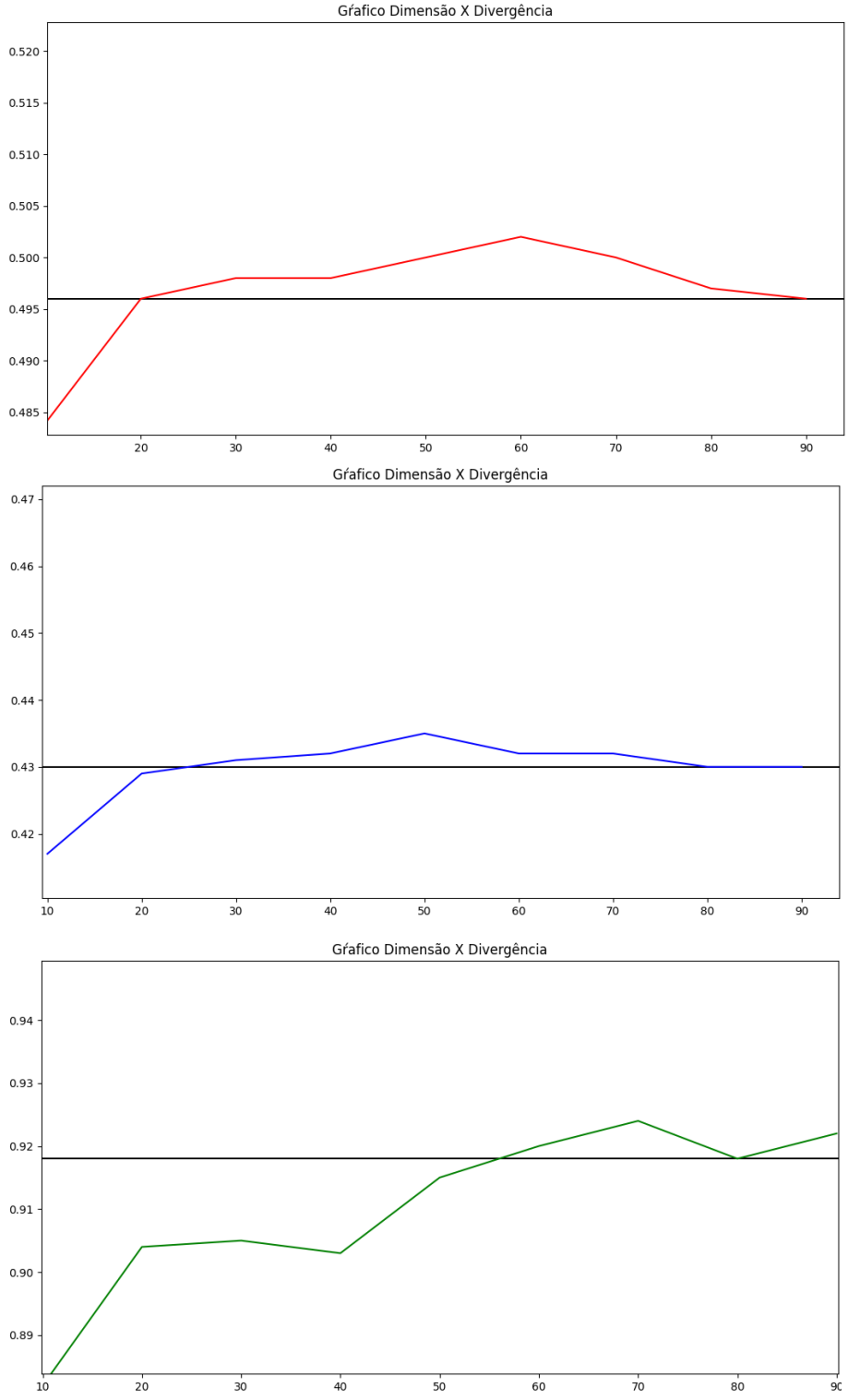

Figura 5.4: Divergência JS para os Conjuntos

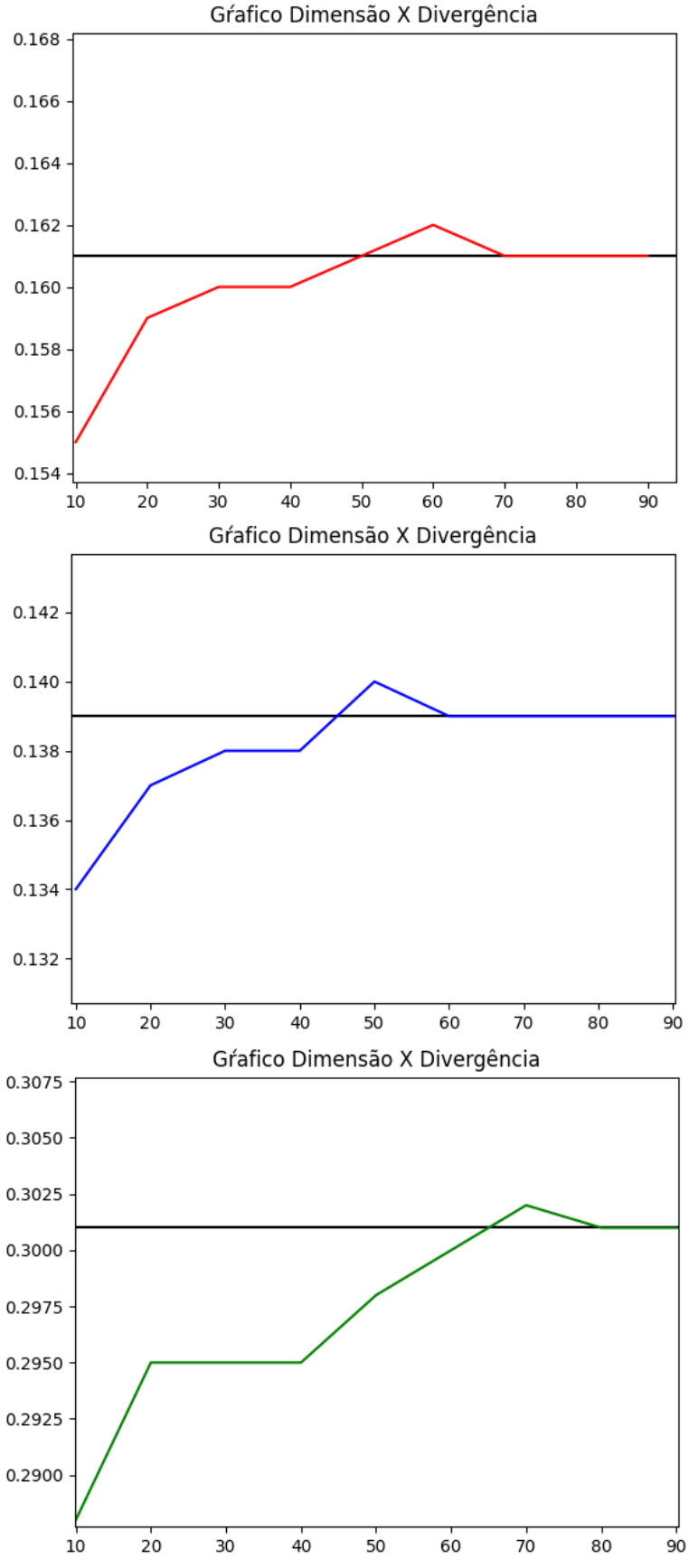

Figura 5.5: Divergência Centróide para os Conjuntos

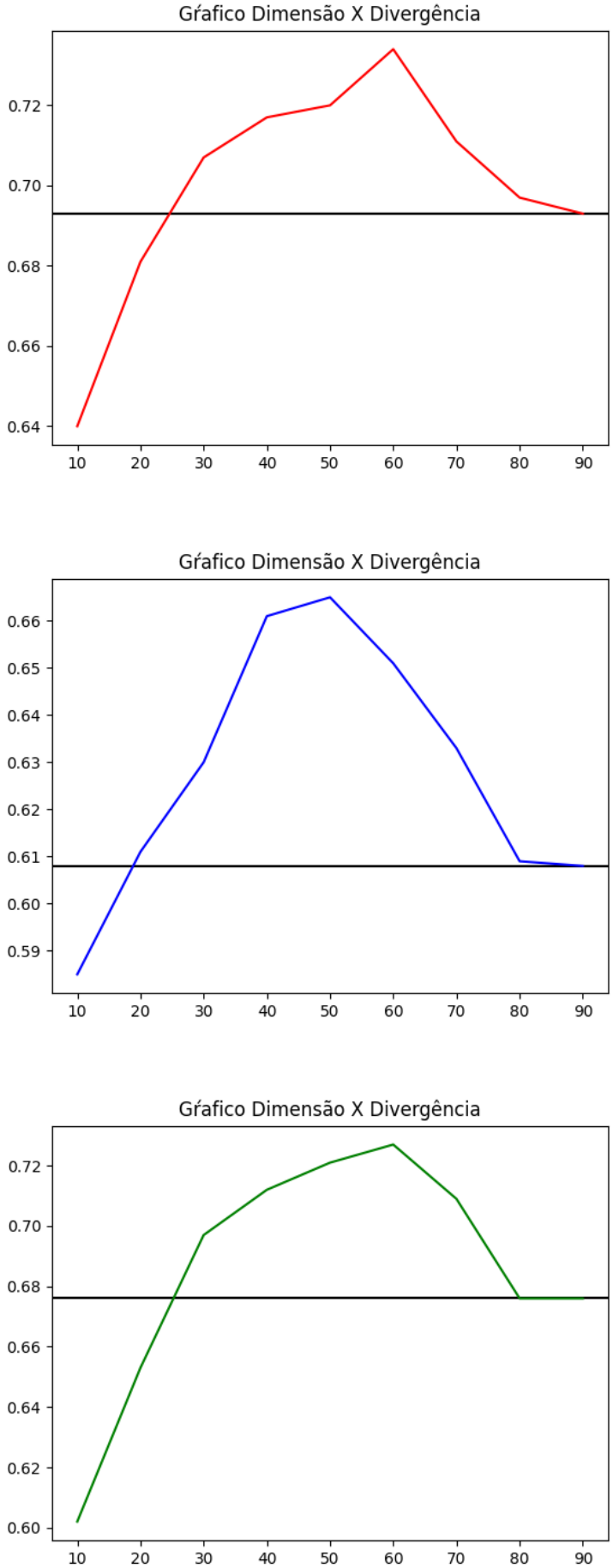

Tabela 5.4: Harem Aproximado Para o Geocorpus, com d = 80.

GEOCORPUS	precisao	recall	f-measure	variação
FORA	0,87	0,93	0,9	0,04
EPC	0,77	0,87	0,82	-0,01
IDA	0,82	0,9	0,86	0,04
OTR	0,19	0,32	0,24	-0,12
CAR	0,41	0,58	0,48	-0,09
UES	0,48	0,65	0,55	-0,07
EON	0,41	0,58	0,48	0,12
SIL	0,11	0,19	0,14	0
BAS	0,54	0,7	0,61	-0,06
CON	0,58	0,74	0,65	-0,03
PRD	0,56	0,72	0,63	0,02
ERA	1	1	1	0,35
OGN	1	1	1	0
TOTAL	0,59	0,72	0,65	0,02

Tabela 5.5: Harem Aproximado Para o Lener, com d = 80.

LENER	precisao	recall	f-measure	variação
FORA	0,91	0,95	0,93	0
JURISPRUDENCIA	0,45	0,62	0,52	0,02
ORGANIZACAO	0,69	0,82	0,75	0,02
PESSOA	0,57	0,73	0,64	-0,02
TEMPO	0,75	0,86	0,8	0
LOCAL	0,5	0,67	0,57	-0,05
LEGISLACAO	0,24	0,39	0,3	-0,02
TOTAL	0,59	0,72	0,65	0

Tabela 5.6: Harem Aproximado Para o Cojur, com d = 60.

COJUR	precisao	recall	f-measure	variação
FORA	0,61	0,76	0,68	-0,04
LEI	0,46	0,63	0,53	-0,08
DEC	0,66	0,8	0,72	0,08
LCOMP	0,5	0,67	0,57	-0,17
DLEI	0,03	0,06	0,04	-0,52
DLEG	0,42	0,59	0,49	-0,31
MP	0,09	0,17	0,12	-0,57
EC	0	0	0	-0,8
TOTAL	0,35	0,46	0,4	-0,3

sua performance geral, enquanto o Cojur apresentou uma piora considerável de 30%. Nos três casos, os valores de d estavam associados a um aumento ou constância do valor das divergências entre os domínios. Conclui-se portanto que provalvemente a melhora (ou piora) na performance não está diretamente associada com o aumento ou diminuição da divergência entre os domínios para as representações não contextuais.

5.3.5 Análise Qualitativa dos Resultados

Analisar os casos em que o modelo adaptado obteve sucesso e os que ele falhou em adaptar pode nos fornecer uma boa intuição do que exatamente foi alcançado na nossa tentativa de aproximar ambos os conjuntos de dados. Para o GeoCorpus, a classe ERA teve um aumento expressivo de 35% em seu *score* geral. Uma amostra do arquivo de teste que demonstra a classe ERA como ocorre no *corpus* é:

```
Paleozóico B-ERA
, O
Mesozóico B-ERA
e O
Cenozóico B-ERA
```

O modelo treinado no Harem sem adaptação nos forncece a resposta:

```
Paleozóico B-
, O
Mesozóico B-
e I-
Cenozóico I-
```

O excesso de rótulos I é um indicativo da natureza das entidades encontradas no Harem, ou seja, entidades maiores dos que as encontradas no GeoCorpus. A marcação fornecedida pelo modelo adaptado fornece evidências de que o modelo obteve sucesso em adaptar para a classe ERA:

```
Paleozóico B-
, O
Mesozóico B-
e O
Cenozóico B-
```

Essa é exatamente a marcação encontrada no conjunto de testes, por isso o aumento expressivo na performance da classificação da classe ERA. Em contraste com o sucesso da classe ERA, temos a classe OTR, a qual teve uma queda de 12% em performance. Um exemplo da ocorrência dessas entidades é:

```
mármores B-OTR
, O
calcifilitos B-OTR
e O
quartzitos B-OTR
```

As respostas dos modelos treinados no Harem, tanto adaptados quanto não adaptados, é simplesmente desconsiderar a ocorrência dessas entidades, como demonstra o trecho:

```
mármores O
, O
calcifilitos O
e O
quartzitos O
```

Note que nenhuma das palavras que fazem partes das entidades são capitalizadas, ou seja, um modelo treinado no Harem, tanto adaptado quanto não adaptado, teria dificuldades de considerar tais ocorrências como pertencentes a alguma entidade, justamente pela ausência de capitalização.

A ausência de melhora na performance geral do LeNER não implica que nenhuma classe melhorou, e sim que o balanço total entre os *scores* das classes que melhoraram e das que pioraram se cancelam. Uma das classes que obteve sucesso em sua classificação foi JURISPRUDÊNCIA, com um aumento de 2%. Vamos analisar um exemplo para tentar entender o que deu certo.

```
RICARDO B-PESSOA
LEWANDOWSKI I-PESSOA
ACO B-JURISPRUDENCIA
2821 I-JURISPRUDENCIA
A I-JURISPRUDENCIA
GR I-JURISPRUDENCIA
/ I-JURISPRUDENCIA
MT I-JURISPRUDENCIA
```

O fragmento acima demonstra a marcação correta da entidade jurídica acima. A resposta do nosso modelo não adaptado é como se segue:

```
RICARDO B-
LEWANDOWSKI I-
ACO I-
2821 B-
A O
```

```
GR B-
/ O
MT B-
```

O modelo não adaptado foi incapaz de tratar a passagem como uma entidade única, pois no Harem números são tratadas como entidades separadas das outras. O modelo adaptado nos fornece a seguinte respota:

```
 RICARDO B-
LEWANDOWSKI I-
ACO I-
2821 I-
A I-
GR B-
/ O
MT B-
```

Isto significa que o modelo foi capaz de compreender que o número 2821 fazia parte da entidade, mas ainda assim não obteve sucesso completo em tratar a sequência como uma única entidade.

Por fim, vamos refletir acerca da classe EC (emenda constitucional) presente no Cojur, a qual não teve uma instância sequer classificada corretamente.

```
Emenda B-EC
Constitucional I-EC
no I-EC
51 I-EC
```

A resposta do modelo adaptado descarta as palavras “Emenda”e ”Constituição”como pertencentes à entidade, além de marcar ”no”e “51”como entidades únicas.

```
Emenda O
Constitucional O
no B-
51 B-
```

Esse comportamento é um indicativo do porque a performance do modelo adaptado usando o Cojur como conjunto alvo obteve uma performance tão abaixo do esperado em todas as classes.

5.4 RESUMO DO CAPÍTULO

Este capítulo tratou da arquitetura, metodologia e resultados utilizados nos experimentos baseados em representações não contextuais. Pudemos verificar a performance dos modelos adaptados, além do melhor parâmetro d para cada conjunto de dados, além de analisar o comportamento das divergências a medida que variamos o valor desse parâmetro. Foi também realizada uma análise qualitativa, com o intuito de verificar qual a estratégia utilizada pelos modelos adaptados na classificação. O próximo capítulo se presta a fazer a mesma coisa para os experimentos baseados em representações contextuais.

Capítulo 6

ADAPTAÇÃO DE DOMÍNIO PARA REN BASEADO EM REPRESENTAÇÕES CONTEXTUAIS

Neste capítulo trataremos da arquitetura, metodologia e resultados do experiento baseado em representações contextuais. A única diferença relativa aos experimentos não contextuais é que a aproximação é feita sobre representações de palavras passadas por uma LSTM, ou seja, sobre representações que levam o contexto em consideração. A intuição por trás dessa estratégia é que a aproximação é feita sobre o espaço utilizado diretamente pelo CRF como conjunto de treino, que são as representações contextuais procedentes da LSTM. A figura 6.1 demonstra a estratégia utilizada nesse experimento, em contraste com o experimento baseado em representações não contextuais.

6.1 ARQUITETURA UTILIZADA

A arquitetura utilizada nos experimentos contextuais foi semelhante à explicitada na seção 5.1, com a única diferença entre as duas advindo das *features* de entrada que são passadas para o CRF. Nos experimentos contextuais, essas *features* eram a saída de uma LSTM que passavam por um *dropout* e já haviam sido transformadas pelo alinhamento de subespaços. Já nos experimentos contextuais, o CRF recebe como entrada a saída completa de 100 dimensões da LSTM, a qual passa pelo processo de adaptação logo antes de ser processada pelo CRF.

6.2 METODOLOGIA DO EXPERIMENTO

Durante o curso dos experimentos deste capítulo, foram mantidos os mesmos parâmetros e hiperparâmetros empregados no experimento baseado em representações não contextuais. Em seguida, treinou-se uma rede LSTM no Harem que foi usada como geradora de representações contextuais. Em outras palavras, enquanto nos experimentos anteriores o espaço de representações originais era derivado de um modelo GLOVE, nos experimentos baseados em representações contextuais as representações GLOVE foram passadadas primeiro para a LSTM treinada no Harem, obtendo-se as representações contextuais para

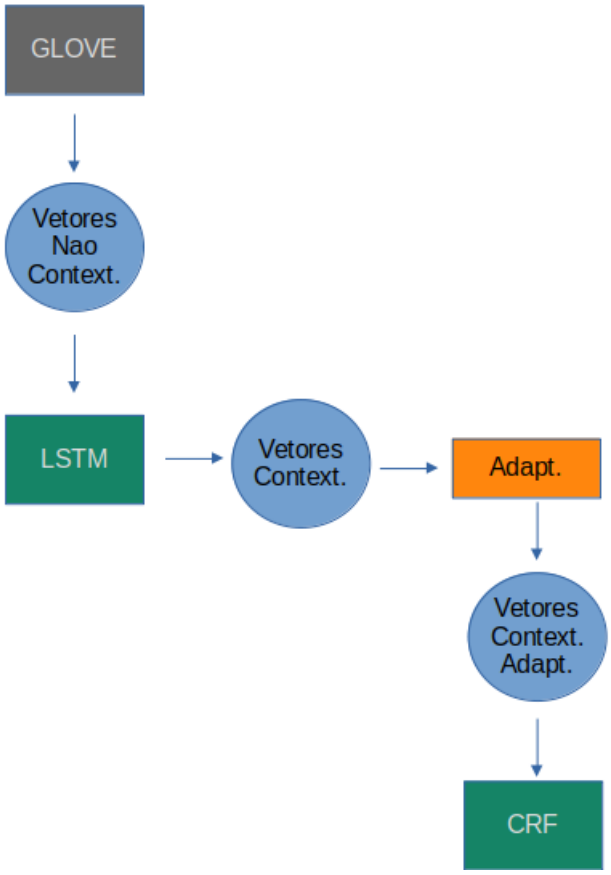

Figura 6.1: Estratégia do experimento baseado em representações contextuais

cada conjunto de dados. O processo de adaptação foi então realizado sobre os vetores codificadores de contexto, de maneira a treinar os modelos sobre esses vetores transformados.

Foi mantido o mesmo processo de cálculo das divergências originais entre o Harem e os conjuntos alvo e a comparação destas com as divergências dos eubespaços aproximados. Posteriormente, obtidos todos os modelos aproximados, foi repetido a mesma metodologia de verificar os melhores valores d de dimensao dos subespaços, com a realização de uma avaliação qualitativa e quantitava para os valores que obtiveram a melhor performance.

6.3 RESULTADOS

Assim como nos experimentos contextuais, todos os resultados foram obtidos através da *micro averaging* dos rótulos B, I e O para cada classe, em que o cálculo da precisão, *recall* e *F-measure* totais foram obtidos através da média de cada um desses *scores* para cada classe.

Tabela 6.1: Harem aplicado no GeoCorpus

GEOCORPUS	precisao	recall	f-measure
FORA	0,82	0,90	0,86
EPC	0,22	0,36	0,27
IDA	0,23	0,38	0,29
OTR	0,3	0,47	0,37
CAR	0,19	0,32	0,24
UES	0,11	0,20	0,14
EON	0,04	0,07	0,05
SIL	0,32	0,48	0,38
BAS	0,34	0,51	0,41
CON	0,35	0,52	0,42
PRD	0,1	0,19	0,13
ERA	0	0,00	0,00
OGN	0,00	0,00	0,00
TOTAL	0,23	0,34	0,27

Tabela 6.2: Harem aplicado no LeNER

LENER	precisao	recall	f-measure
FORA	0,89	0,94	0,91
JURISPRUDENCIA	0,45	0,62	0,52
ORGANIZACAO	0,52	0,68	0,59
PESSOA	0,82	0,90	0,86
TEMPO	0,44	0,61	0,51
LOCAL	0,62	0,76	0,68
LEGISLACAO	0,16	0,27	0,20
TOTAL	0,56	0,68	0,61

6.3.1 Aplicação do Modelo sem Adaptação

As tabelas 6.1, 6.2 e 6.3 mostram o resultado de um modelo treinado no Harem sem adaptação aplicado nos conjuntos alvo.

6.3.2 Variação das Divergências

As imagens 6.1, 6.2 e 6.3 demonstram a variação da divergência em função da dimensão do subespaço na circustância dos experimentos de representações contextuais.

6.3.3 Resultados dos Modelos Adaptados

De forma análoga à seção 5.3.3, as tabelas 6.4, 6.5 e 6.6 mostram os resultados dos modelos adaptados que obtiveram a melhor performance nos conjuntos alvos.

Tabela 6.3: Harem aplicado no Cojur

COJUR	precisao	recall	f-measure
FORA	0,65	0,79	0,71
LEI	0	0,00	0,00
DEC	0	0,00	0,00
LCOMP	0	0,00	0,00
DLEI	0	0,00	0,00
DLEG	0,5	0,67	0,57
MP	0	0,00	0,00
EC	0	0,00	0,00
TOTAL	0,14	0,18	0,16

Tabela 6.4: Harem Aproximado Para o GeoCorpus, com d = 10.

GEOCORPUS	precisao	recall	f-measure	variação
FORA	0,94	0,97	0,95	0,1
EPC	0,54	0,7	0,61	0,34
IDA	0,83	0,91	0,87	0,58
OTR	0,44	0,61	0,51	0,15
CAR	0,5	0,67	0,57	0,33
UES	0	0	0	-0,14
EON	0,52	0,68	0,59	0,54
SIL	0,35	0,51	0,42	0,03
BAS	0,03	0,05	0,04	-0,37
CON	0,05	0,1	0,07	-0,35
PRD	0,74	0,85	0,79	0,66
ERA	0,47	0,64	0,54	0,54
OGN	1	1	1	1
TOTAL	0,49	0,59	0,54	0,26

Tabela 6.5: Harem Aproximado Para o LeNER, com d = 90.

LENER	precisao	recall	f-measure	variação
FORA	0,93	0,96	0,94	0,03
JURISPRUDENCIA	0,31	0,48	0,38	-0,14
ORGANIZACAO	0,47	0,64	0,54	-0,05
PESSOA	0,63	0,77	0,69	-0,17
TEMPO	0,21	0,35	0,26	-0,25
LOCAL	0,52	0,69	0,59	-0,09
LEGISLACAO	0,22	0,37	0,28	0,08
TOTAL	0,47	0,61	0,53	-0,08

Figura 6.2: Divergência KL para os Conjuntos

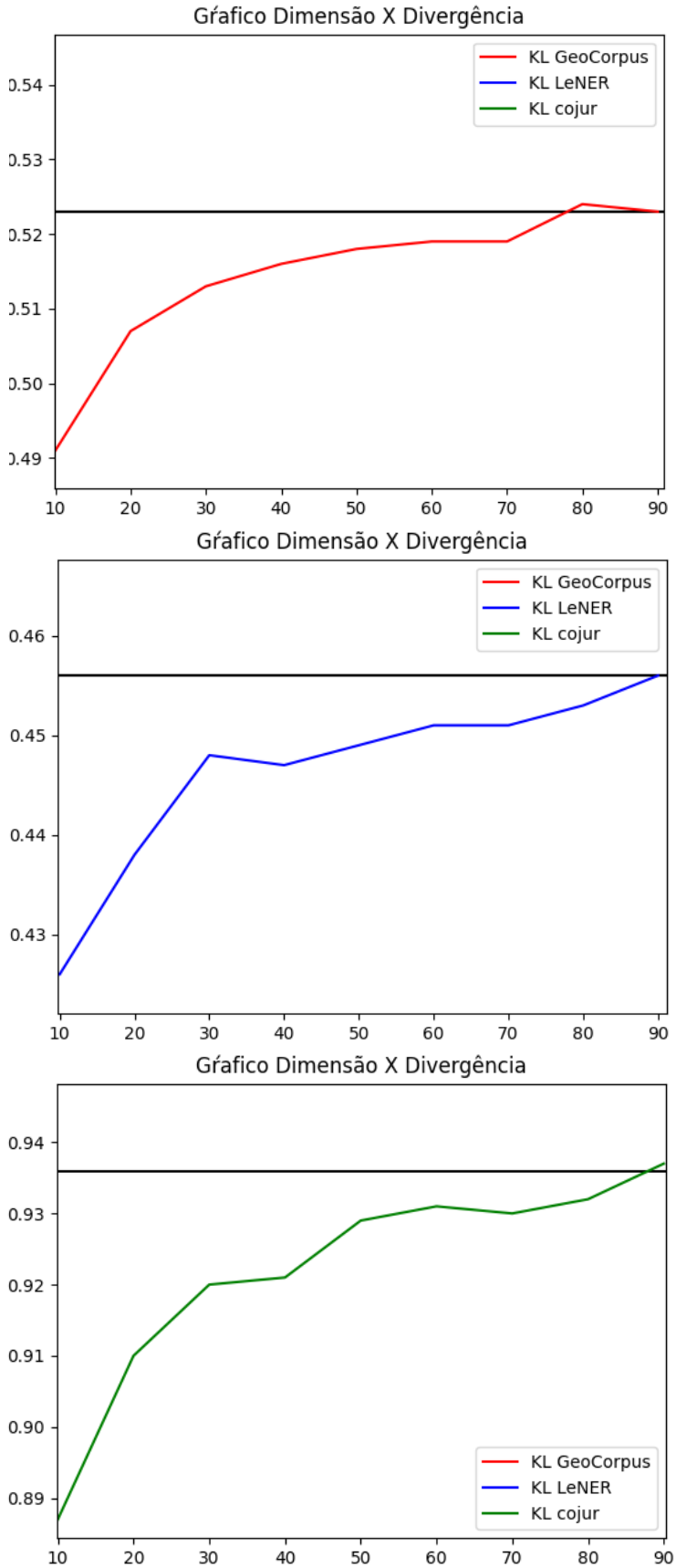

Figura 6.3: Divergência KL para os Conjuntos

Figura 6.4: Divergência JS para os Conjuntos

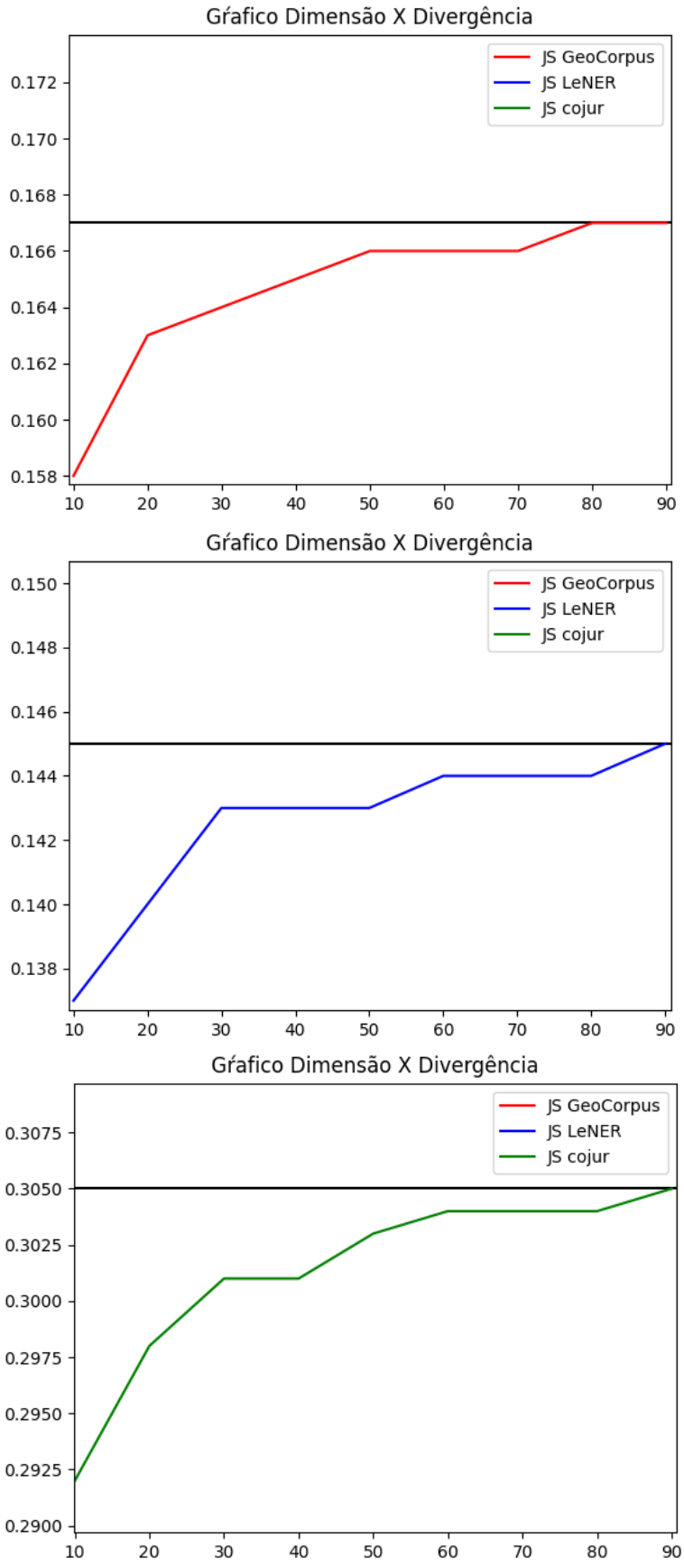

Figura 6.5: Divergência Centróide para os Conjuntos

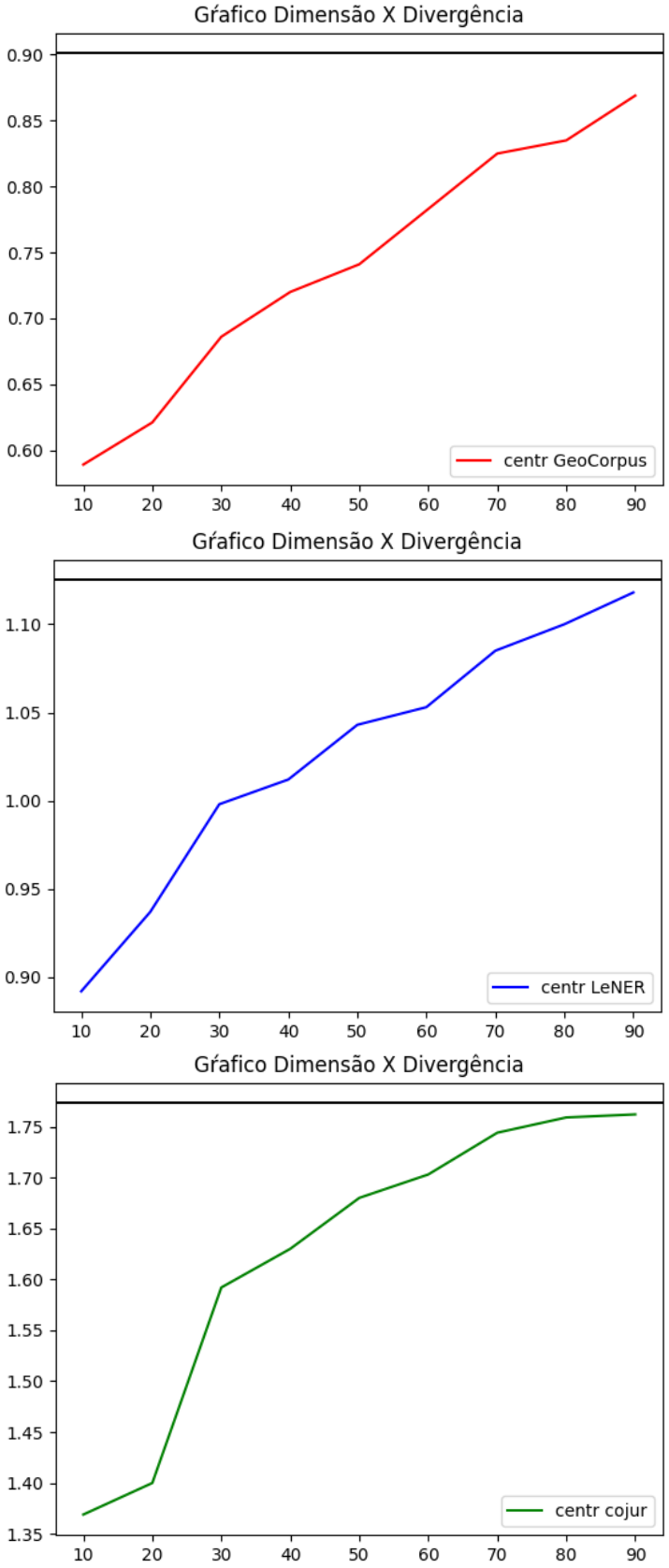

Tabela 6.6: Harem Aproximado Para o Cojur, com d = 20.

COJUR	precisao	recall	f-measure	variação
FORA	0,99	1	0,99	0,28
LEI	0,33	0,5	0,4	0,4
DEC	0,33	0,5	0,4	0,4
LCOMP	0,25	0,4	0,31	0,31
DLEI	0,33	0,5	0,4	0,4
DLEG	0,25	0,4	0,31	-0,26
MP	0	0	0	0
EC	0,25	0,46	0,32	0,32
TOTAL	0,34	0,46	0,39	0,23

6.3.4 Discussão

Comparando os resultados obtidos através da aproximação de subespaços usando representações contextuais e não contextuais, podemos notar algumas diferenças interessantes. Em relação às divergências entre domínios, se compararmos os gráficos em ambos os experimentos, conseguimos ver que apesar de ambos apresentarem a mesma tendência de queda quando a dimensão do subespaço de projeção diminui, as representações contextuais tendem a ser muito mais consistentes em seu comportamento.

Colocado de outra forma, para a maioria dos pontos dos gráficos do experimento contextual vale que se $d_1 \leq d_2 \rightarrow f(d_1) \leq f(d_2)$, em que $f(d)$ é o valor da divergncia. Esse comportamento parece indicar que as *features* contextuais, se comparadas com as *features* não contextuais, são mais capazes de codificar informações relevantes para a tarefa de REN, já que o método de aproximação aplicado sobre elas provavelmente implicará em uma redução das divergência entre os domínios.

Se compararmos os resultados dos modelos adaptados, vemos que dois modelos obtiveram um ganho expressivo de performance no GeoCorpus e no Cojur, em contraste com apenas um modelo para os experimentos contextuais, que foi o modelo aplicado no GeoCorpus. Em contrapartida, a queda de performance quando um modelo não adaptado foi aplicado nesses dois conjuntos foi muito maior do que nos experimentos não contextuais, com exceção do LeNER, cujas métricas se mantiveram mais ou menos constantes. Isso pode explicar o porquê do ganho tão expressivo de performance ao se aplicar um modelo adaptado no Cojur e no GeoCorpus.

O motivo dos modelos adaptados para o LeNER-BR não terem melhorado a sua performance se deve provavelmente ao fato de que o LeNER-BR possui uma estrutura textual muito específica dos textos jurídicos, que é muito diferente do tipo de texto encontrado no Harem. Em comparação, o Cojur, mesmo sendo um *corpus* jurídico como o LeNER-BR, apresentou melhoras por possuir uma estrutura textual mais parecida com a do Harem.

Outra diferença importante em relação aos experimentos contextuais foi a relação entre divergência e performance. Nos experimentos contextuais os modelos que obtiveram a melhor performance não eram os modelos que apresentavam a menor divergência, já nos

contextuais os modelos de menor dimensão, com $d = 10$ e $d = 20$ para GeoCorpus e Cojur respectivamente, foram os que apresentaram a melhor performance. Como as menores dimensões estão associadas experimentalmente com os menores valores de divergência, os experimentos parecem apontar que para a aproximação de vetores de característica contextuais, diminuir a divergência entre domínios pode ser uma estratégia viável de adaptação.

6.3.5 Análise Qualitativa dos Resultados

Vamos partir para a análise qualitativa dos resultados dos experimentos contextuais, o que nos permitirá entender as mudanças realizadas na estratégia de anotação adotada pelos modelo adaptados. Vemos que uma classe que obteve uma melhoria significativa de performance no GeoCorpus foi a classe IDA, que indica um certo tipo de período geológico. O modelo não adaptado era incapaz de classificar ocorrências dessa classe, como demonstra o trecho a seguir.

```
Serpukhoviano O
inferior O
- O
Bashkiriano O
médio O
e O
Bashkiriano O
médio O
- O
Moscoviano B-
```

Em que a única entidade marcada corretamente foi “Moscoviano”. O modelo adaptado por sua vez foi capaz de marcar todas as ocorrências de entidades do tipo IDA:

```
erpukhoviano B-
inferior O
- O
Bashkiriano B-
médio O
e O
Bashkiriano B-
médio O
- O
Moscoviano B-
```

Em contrapartida, houve classes que apresentaram uma piora em seus resultados, como é o caso da classe UES. Neste caso, o modelo adaptado desconsiderou praticamente todas as ocorrencias desta classe, pois enquanto a marcação original era:

```
Formação B-UES
Dardanelos I-UES
```

o modelo adaptado realizou a marcação:

```
Formação -O
Dardanelos -O
```

O LeNER-BR apresentou uma piora em seu *score* geral. Como demonstra tabela 6.5, as únicas classes que apresentaram alguma melhora foram LEGISLAÇÃO e FORA. O extrato do conjunto LeNER demonstra a classificação correta de uma entidade do tipo legislação:

```
 o O
artigo B-LEGISLACAO
71 I-LEGISLACAO
da I-LEGISLACAO
Lei I-LEGISLACAO
nº I-LEGISLACAO
8.666/93 I-LEGISLACAO
```

O modelo não adaptado apresenta a seguinte anotação para o mesmo trecho:

```
o O
artigo O
71 O
da O
Lei O
nº O
8.666/93 O
```

Por fim, o modelo adaptado nos forncese a seguinte reposta:

```
  o B-
artigo I-
71 I-
da O
Lei O
nº O
8.666/93 O
```

Vemos que a entidade começou a ser marcada, mas o processo não foi completado. Em outras palavras, o modelo não foi capaz de detectar que a entidade possuia mais componentes. Isso pode ser um comportamente determinado pelo tamanho das entidades presentes no Harem, que são, em média, menores do que as entidades encontradas no LeNER.

Por fim, o Cojur apresentou melhoras expressivas, com um ganho total de *score* de 23%. Um exemplo dessa melhora é a classe DLEI, que obteve um aumento de 40% para sua *f-measure*. O próximo extrato demonstra a classificação correta de um decreto de lei:

```
Decreto-Lei B-DLEI
no I-DLEI
5.452 I-DLEI
```

O modelo não adaptado não é capaz de classificar nenhuma das palavras pertencentes à entidade. Já o modelo adaptado faz um pequeno progresso ao identificar que a palavra "Decreto"é componente de uma entidade, mas não consegue marcar as outras palavras como pertencentes à mesma entidade.

```
Decreto-Lei B
no O
5.452 O
```

De maneira geral, a análise qualitativa nos mostra que a escolha do domínio fonte é de suma importância para o processo de adaptação, já que a natureza das entidades presentes nesse conjunto determina em grande parte a anotação do modelo adaptado no dominio alvo. Adicionalmente, O HAREM não é um conjunto em que se costuma obter performances altas, como observado em outros trabalhos (FREITAS et al., 2010). Esse fenômeno, por consequência, provavelmente influencia negativamente na performance dos modelos adaptados.

6.4 RESUMO DO CAPÍTULO

Este capítulo tratou da arquitetura, metodologia e resultados dos experimentos baseados em representações contextuais. Vimos que, em comparação com os experimentos baseados em representações não contextuais, os modelos adaptados seguindo a primeira abordagem obtiveram melhores resultados (variação positiva de performance pós adaptação). Em relação as divergências, vimos que o comportamento destas quando se variou d foi mais consistente, o que implica que os vetores de característica contextuais obtidos pós adaptação codificam informaçoes mais relevantes para a tarefa de REN. O próximo capítulo conclui o presente trabalho com uma discussão de tudo o que foi feito, além de uma proposta para os trabalhos futuros.

Capítulo 7

CONCLUSÕES

Este trabalho desenvolveu um estudo sobre a aplicação do alinhamento de subespaços para a tarefa de REN em domínios específicos na língua portuguesa. Tal investigação teve dois eixos principais, um relativo ao conceito de divergência entre os domínios e outro referente a relação entre a divergência e a performance dos modelos adaptados.

Os questionamentos que procuramos responder foram: existe algum método que possibilite a diminuição da divergência entre dois domínios? confirmada a existência de tal método, é possível verificar que existe uma relação entre a diminuição da divergência e o aumento da performance de modelos adaptados, como indicado pelo limite de generalização?

Vimos que o alinhamento de subespaços, aplicado quando o espaço de representação das palavras é construído usando informação contextual da sentença quanto não a utiliza, promovem uma diminuição de divergência proporcional ao tamanho do subespaço. Ou seja, quanto maior a dimensão de subespaço maior tende a ser a divergência entre os domínios. Tal fenômeno foi capturado de maneira mais clara pelas aproximações das *features* contextuais, como discutido no capítulo 6.

Já a relação entre performance e divergência não se mostrou tão transparente. O único *corpus* que apresentou melhora de performance nos experimentos que utilizam representações não contexuais foi o GeoCorpus, cuja divergência para a dimensão ótima ($d = 90$) era uma das maiores entre as divergências dos subespaços. Já nas respresentações contextuais, os *corpus* que apresentaram melhora estavam associados a dimensões com valores baixos de divergência. Tal relação parece indicar que para o caso da aproximação de representações contextuais, que são comummente utilizadas na tarefa de REN, diminuir a divergência entre os domínios pode ser uma estratégia viável de adaptação.

A análise qualitativa realizada nos permitiu compreender o que exatamente os modelos mudaram em suas estratégias de anotação para adaptar de um domínio para o outro. Vimos que enquanto algumas classes apresentam uma melhora considerável, outras perdem performance, de maneira que o que determina o *score* geral é o balanço entre classes que apresentam melhoras e classes que pioram. A partir da observação dos exemplos, vimos

também que a natureza das classes do domínio fonte são essenciais para a performance no domínio alvo, pois um modelo só pode adaptar em cima do que já foi observado. Se as entidades entre um domínio e outro divergem de maneira significativa, o processo de adaptaçao se torna praticamente inviável.

7.1 TRABALHOS FUTUROS

Algumas variações nos experimentos poderiam ajudar a investigar outros aspectos de adaptação de domínio aplicado à REN, além de auxiliar na elucidação dos já levantados. Seria interessante eleger outros conjuntos de dados como domínos fonte, com o intuito de observar se a performance dos modelos adaptados melhora ou piora.

Encontram-se presentes na literatura extensões do alinhamento de subespaços que procuram alinhar não somente as bases dos subespaços projetados, mas também as prórpias distribuições de probabilidade(SUN; SAENKO, 2015). Uma proposta de trabalho futuro é aplicar essa variação e explorar como ela afeta a diminuição da divergência entre domínios e as métricas de performance.

REFERÊNCIAS BIBLIOGRÁFICAS

ALJUNDI, R. et al. Landmarks-based kernelized subspace alignment for unsupervised adaptation. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. [S.l.: s.n.], 2015. p. 56–63.

AMARAL, D. et al. Processo de construção de um corpus anotado com entidades geológicas visando ren (building an annotated corpus with geological entities for ner)[in portuguese]. In: *Proceedings of the 11th Brazilian Symposium in Information and Human Language Technology*. [S.l.: s.n.], 2017. p. 63–72.

AMARAL, D. O. F. do; VIEIRA, R. Nerp-crf: uma ferramenta para o reconhecimento de entidades nomeadas por meio de conditional random fields. *Linguamática*, v. 6, n. 1, p. 41–49, 2014.

ARAUJO, P. H. L. de et al. Lener-br: A dataset for named entity recognition in brazilian legal text. In: SPRINGER. *International Conference on Computational Processing of the Portuguese Language*. [S.l.], 2018. p. 313–323.

BEN-DAVID, S. et al. A theory of learning from different s. *Machine learning*, Springer, v. 79, n. 1-2, p. 151–175, 2010.

BENGIO, Y.; COURVILLE, A.; VINCENT, P. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, IEEE, v. 35, n. 8, p. 1798–1828, 2013.

BHARADWAJ, A. et al. Phonologically aware neural model for named entity recognition in low resource transfer settings. In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, Texas: Association for Computational Linguistics, 2016. p. 1462–1472. Disponível em: ⟨https://www.aclweb.org/anthology/D16-1153⟩.

BJERVA, J.; KOUW, W. M.; AUGENSTEIN, I. Back to the future–sequential alignment of text representations. In: ASSOCIATION FOR THE ADVANCEMENT OF ARTIFICIAL INTELLIGENCE. *34rd AAAI Conference on Artificial Intelligence*. [S.l.], 2019. p. 1909–03464.

CASTRO, P. V. Q. de; SILVA, N. F. F. da; SOARES, A. da S. Portuguese named entity recognition using lstm-crf. In: SPRINGER. *International Conference on Computational Processing of the Portuguese Language*. [S.l.], 2018. p. 83–92.

CHIU, J. P.; NICHOLS, E. Named entity recognition with bidirectional lstm-cnns. *Transactions of the Association for Computational Linguistics*, MIT Press, v. 4, p. 357–370, 2016.

DONG, C. et al. Character-based lstm-crf with radical-level features for chinese named entity recognition. In: *Natural Language Understanding and Intelligent Applications*. [S.l.]: Springer, 2016. p. 239–250.

FERNANDO, B. et al. Unsupervised visual adaptation using subspace alignment. In: *Proceedings of the IEEE international conference on computer vision*. [S.l.: s.n.], 2013. p. 2960–2967.

FERNANDO, B. et al. Subspace alignment for adaptation. *arXiv preprint arXiv:1409.5241*, 2014.

FISHER, R. A. The use of multiple measurements in taxonomic problems. *Annals of eugenics*, Wiley Online Library, v. 7, n. 2, p. 179–188, 1936.

FREITAS, C. et al. Second harem: advancing the state of the art of named entity recognition in portuguese. In: EUROPEAN LANGUAGE RESOURCES ASSOCIATION. *quot; In Nicoletta Calzolari; Khalid Choukri; Bente Maegaard; Joseph Mariani; Jan Odijk; Stelios Piperidis; Mike Rosner; Daniel Tapias (ed) Proceedings of the International Conference on Language Resources and Evaluation (LREC 2010)(Valletta 17-23 May de 2010) European Language Resources Association*. [S.l.], 2010.

FUGLEDE, B.; TOPSOE, F. Jensen-shannon divergence and hilbert space embedding. In: IEEE. *International Symposium onInformation Theory, 2004. ISIT 2004. Proceedings.* [S.l.], 2004. p. 31.

GOLDBERG, Y. Neural network methods for natural language processing. *Synthesis lectures on human language technologies*, Morgan & Claypool Publishers, v. 10, n. 1, p. 1–309, 2017.

GONG, M. et al. adaptation with conditional transferable components. In: PMLR. *International conference on machine learning*. [S.l.], 2016. p. 2839–2848.

GONG, M. et al. Domain adaptation with conditional transferable components. In: PMLR. *International conference on machine learning*. [S.l.], 2016. p. 2839–2848.

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

GRISHMAN, R.; SUNDHEIM, B. Design of the muc-6 evaluation. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the 6th conference on Message understanding*. [S.l.], 1995. p. 1–11.

HECKMAN, J. J. Sample selection bias as a specification error. *Econometrica: Journal of the econometric society*, JSTOR, p. 153–161, 1979.

HUANG, Z.; XU, W.; YU, K. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*, 2015.

JOHNSON, D.; SINANOVIC, S. Symmetrizing the kullback-leibler distance. *IEEE Transactions on Information Theory*, 2001.

JURAFSK, J. H. M. D. *Speech and Language Processing*. [s.n.], 2019. Disponível em: ⟨https://web.stanford.edu/~jurafsky/slp3/⟩.

JURAFSKY, D.; MARTIN, J. Vector semantics and embeddings. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*, p. 94–122, 2019.

KHAN, S. et al. A guide to convolutional neural networks for computer vision. *Synthesis Lectures on Computer Vision*, Morgan & Claypool Publishers, v. 8, n. 1, p. 1–207, 2018.

KOUW, W. M.; LOOG, M. A review of adaptation without target labels. *IEEE transactions on pattern analysis and machine intelligence*, IEEE, 2019.

LAMPLE, G. et al. Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360*, 2016.

LING, X.; WELD, D. Fine-grained entity recognition. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. [S.l.: s.n.], 2012. v. 26, n. 1.

LIU, A.; ZIEBART, B. Robust classification under sample selection bias. *Advances in neural information processing systems*, v. 27, p. 37–45, 2014.

LONG, M. et al. invariant transfer kernel learning. *IEEE Transactions on Knowledge and Data Engineering*, IEEE, v. 27, n. 6, p. 1519–1532, 2014.

LOPES, F.; TEIXEIRA, C.; OLIVEIRA, H. G. Contributions to clinical named entity recognition in portuguese. In: *Proceedings of the 18th BioNLP Workshop and Shared Task*. [S.l.: s.n.], 2019. p. 223–233.

MAI, K. et al. An empirical study on fine-grained named entity recognition. In: *Proceedings of the 27th International Conference on Computational Linguistics*. [S.l.: s.n.], 2018. p. 711–722.

MANSOUR, Y.; SCHAIN, M. Robust domain adaptation. *Annals of Mathematics and Artificial Intelligence*, Springer, v. 71, n. 4, p. 365–380, 2014.

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

MURPHY, K. P. *Probabilistic Machine Learning: An introduction*. MIT Press, 2021. Disponível em: ⟨probml.ai⟩.

NADEAU, D.; SEKINE, S. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, v. 30, n. 1, p. 3–26, 2007.

PENNINGTON, J.; SOCHER, R.; MANNING, C. D. Glove: Global vectors for word representation. In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. [S.l.: s.n.], 2014. p. 1532–1543.

RITTER, A. et al. Named entity recognition in tweets: An experimental study. In: *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*. Edinburgh, Scotland, UK.: Association for Computational Linguistics, 2011. p. 1524–1534. Disponível em: ⟨https://www.aclweb.org/anthology/D11-1141⟩.

SANG, E. F.; MEULDER, F. D. Introduction to the conll-2003 shared task: Language-independent named entity recognition. *arXiv preprint cs/0306050*, 2003.

SANTOS, C. N. d.; GUIMARAES, V. Boosting named entity recognition with neural character embeddings. *arXiv preprint arXiv:1505.05008*, 2015.

SANTOS, D.; CARDOSO, N. *Reconhecimento de entidades mencionadas em português: Documentação e actas do HAREM, a primeira avaliação conjunta na área*. 2007.

SANTOS, J. et al. Assessing the impact of contextual embeddings for portuguese named entity recognition. In: IEEE. *2019 8th Brazilian Conference on Intelligent Systems (BRACIS)*. [S.l.], 2019. p. 437–442.

SHARNAGAT, R. Named entity recognition: A literature survey. *Center For Indian Language Technology*, 2014.

SOARES, S. Contextual representations and semi-supervised named entity recognition for portuguese language. 2019.

SOUZA, F.; NOGUEIRA, R.; LOTUFO, R. Portuguese named entity recognition using bert-crf. *arXiv preprint arXiv:1909.10649*, 2019.

SUN, B.; SAENKO, K. Subspace distribution alignment for unsupervised domain adaptation. In: *BMVC*. [S.l.: s.n.], 2015. v. 4, p. 24–1.

TORRALBA, A.; EFROS, A. A. Unbiased look at dataset bias. In: IEEE. *CVPR 2011*. [S.l.], 2011. p. 1521–1528.

TZENG, E. et al. Simultaneous deep transfer across domains and tasks. In: *Proceedings of the IEEE international conference on computer vision*. [S.l.: s.n.], 2015. p. 4068–4076.

WALLACH, H. M. Conditional random fields: An introduction. *Technical Reports (CIS)*, p. 22, 2004.

WILKS, Y. Information extraction as a core language technology. In: SPRINGER. *International Summer School on Information Extraction*. [S.l.], 1997. p. 1–9.

YADAV, V.; BETHARD, S. A survey on recent advances in named entity recognition from deep learning models. *arXiv preprint arXiv:1910.11470*, 2019.

YADAV, V.; SHARP, R.; BETHARD, S. Deep affix features improve neural named entity recognizers. In: *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*. [S.l.: s.n.], 2018. p. 167–172.

ZHANG, K. et al. Domain adaptation under target and conditional shift. In: PMLR. *International Conference on Machine Learning*. [S.l.], 2013. p. 819–827.