UNIVERSIDADE FEDERAL DO CEARÁ

CAMPUS DE QUIXADÁ

CURSO DE GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

NATANAEL DA SILVA ARAÚJO

RECONHECIMENTO DE ENTIDADES NOMEADAS EM TEXTOS DE BOLETINS DE OCORRÊNCIAS

QUIXADÁ

2019

NATANAEL DA SILVA ARAÚJO

RECONHECIMENTO DE ENTIDADES NOMEADAS EM TEXTOS DE BOLETINS DE OCORRÊNCIAS

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação do Campus de Quixadá da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Ciência da Computação.

Orientador: Prof. Dr. Regis Pires Magalhães

Coorientadora: Profª. Drª. Ticiana Linhares Coelho da Silva

QUIXADÁ

2019

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca Universitária
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

A69r Araújo, Natanael da Silva.
Reconhecimento de entidades nomeadas em textos de boletins de ocorrências / Natanael da Silva Araújo. – 2019.
41 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Quixadá, Curso de Ciência da Computação, Quixadá, 2019.
Orientação: Prof. Dr. Regis Pires Magalhães.
Coorientação: Profa. Dra. Ticiana Linhares Coelho da Silva.

1. Processamento de linguagem natural. 2. Reconhecimento de entidade nomeada. 3. Aprendizagem profunda. I. Título.

CDD 004

NATANAEL DA SILVA ARAÚJO

RECONHECIMENTO DE ENTIDADES NOMEADAS EM TEXTOS DE BOLETINS DE OCORRÊNCIAS

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação do Campus de Quixadá da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Ciência da Computação.

Aprovada em: __/__/___

BANCA EXAMINADORA

Prof. Dr. Regis Pires Magalhães (Orientador)
Universidade Federal do Ceará (UFC)

Prof[a]. Dr[a]. Ticiana Linhares Coelho da Silva (Coorientadora)
Universidade Federal do Ceará (UFC)

Prof. Dr. Marcos Antonio de Oliveira
Universidade Federal do Ceará (UFC)

A Deus,

A minha mãe e minha irmã, Francisca e Natalia.

AGRADECIMENTOS

Agradeço a Deus por seu Amor e Misericórdia, e por ter me dado sabedoria e força suficiente para concluir mais essa etapa da minha vida.

À minha mãe, Francisca, que sempre esteve comigo, que desde muito cedo assumiu também o papel de pai, nada que eu diga e faça pode retribuir o que ela fez (e faz) por e para mim.

Ao meu pai, Francisco, que durante o pouco tempo que passamos juntos aqui na terra sempre me incentivou a estudar. Hoje do céu, tenho certeza que está muito feliz por me ver concluindo minha graduação.

A minha irmã Natalia, que sempre me ajudou e aconselhou. Sou muito grato por tê-la como irmã!

Aos meus avós, Josefa e Lucas, que estão sempre me apoiando e incentivando.

A todos os meus amigos, que de alguma forma me incentivaram. Em especial o meu amigo Rodrigo Valentim, pelo seu grande apoio e incentivo para que eu começasse essa graduação. Sou muito grato a todos!

A professora Ticiana, por sua orientação, ensinamentos e paciência comigo durante esse trabalho, obrigado Ticiana!

Ao professor Regis, por sua orientação, ensinamentos e disponibilidade. Não só nesse trabalho, sou grato pelos seus ensinamentos. Obrigado, Regis!

Ao professor João Marcelo, por ter me orientado no meu primeiro artigo acadêmico. Obrigado professor, pelos ensinamentos, paciência e disponibilidade.

Ao professor Marcos Oliveira pelas suas valiosas contribuições nesse trabalho.

Aos demais professores do campus, em especial, Diana, Rainara, Paulo de Tarso, pelo seu esforço constante de tornar seus alunos não somente bons profissionais, mas boas pessoas.

A todos que compõe a UFC Quixadá, por tornarem o campus um dos melhores ambientes de aprendizagem que poderia ter. Agradeço a todos os funcionários do campus, em especial a Gerlyson, Dias, Natália, Gilmário, e o Venício.

A todos que contribuíram positivamente para a minha formação.

“A força mais potente do universo é a fé.”

(Madre Teresa de Calcutá)

RESUMO

Classificar entidades em um determinado texto é um problema desafiador no Processamento de Linguagem Natural. Uma técnica comum que lida com esse problema é o Reconhecimento de Entidades Nomeadas, em inglês é chamado de *Named Entity Recognition* (NER). As técnicas de *Deep Learning* são amplamente aplicadas nas tarefas NER, porque requerem pouca engenharia de recursos e estão livres de recursos específicos do idioma, aprendendo recursos importantes a partir de combinações de palavras ou caracteres treinadas em grandes quantidades de dados. No entanto, essas técnicas exigem uma enorme quantidade de dados para treinamento. Este trabalho propõe o *Human Named Entity Recognition with Deep learning* (Human NERD), que aborda esse problema incluindo o humano no ciclo. O Human NERD é uma ferramenta interativa para ajudar o usuário nas tarefas de classificação NER, desde a criação de um conjunto de dados massivo até a criação / manutenção de um modelo NER de *Deep Learning*. A estrutura Human NERD permite a verificação rápida do reconhecimento automático de entidades nomeadas e a correção de erros. Ele leva em consideração as correções do usuário, e o modelo de *Deep Learning* aprende e desenvolve essas ações. Sua interface gráfica (GUI) permite uma correção rápida usando ações de arrastar e soltar do usuário. O presente trabalho também propõe dois modelos NER, um utilizando o *framework Spacy* e o outro a biblioteca *Keras*, em que as predições realizadas por ambos se complementam.

Palavras-chave: Processamento de linguagem natural. Reconhecimento de entidade nomeada. Aprendizagem profunda.

ABSTRACT

Classifying entities in a given text is a challenging problem in Natural Language Processing (NLP). A common technique that deals with this problem is Named Entity Recognition (NER). Deep Learning techniques have been widely applied to NER tasks because they require little resource engineering and are free of language-specific resources, learning important resources from combinations of words or characters trained in large amounts of data. However, these techniques require a huge amount of data for training. However, these techniques require a huge amount of data for training. This work proposes Human NERD (stands for Human Named Entity Recognition with Deep learning) which addresses this problem by including humans in the loop. Human NERD is an interactive framework to assist the user in NER classification tasks from creating a massive dataset to building/maintaining a deep learning NER model. Human NERD framework allows the rapid verification of automatic named entity recognition and the correction of errors. It takes into account user corrections, and the deep learning model learns and builds upon these actions. The interface allows for rapid correction using drag and drop user actions. The present work also proposes two NER models, one using the Spacy framework and the other the Keras library, both complement each other.

Keywords: Natural language processing. Named entity recognition. Deep learning.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

BOs	Boletins de Ocorrências
CNN	*Convolutional Neural network*
CRF	*Classical Receptive Field*
FN	*False Negatives*
FP	*False Positives*
HOM	Homicídio
Human NERD	*Human Named Entity Recognition with Deep learning*
LATROC	Latrocínio
LEG_DEF_TER	Legítima Defesa de Terceiros
LOC	Localização
LTSM	*Long Short Term Memory*
MISC	Misto
NER	*Named Entity Recognition*
NLP	*Natural Language Processing*
ORG	Organização
PER	Pessoa
RNN	*Recurrent Neural Networks*
SSPDS	Secretaria de Segurança Pública e Defesa Social
TN	*True Negatives*
TP	*True Positives*
TRIPLO_HOM	Triplo Homicídio

SUMÁRIO

1 INTRODUÇÃO

A Secretaria de Segurança Pública e Defesa Social (SSPDS) apresenta relatórios mensais e anuais sobre todas as ocorrências que foram registradas. Tais relatórios são construídos com informações descritas nos Boletins de Ocorrências (BOs), como nome da vítima, nome do praticante da ação criminosa e o tipo da arma. Atualmente, essas informações são extraídas por técnicos da SSPDS. É uma tarefa demorada e exaustiva, que requer dos técnicos muito tempo para analisar os BOs. Tal processo requer pessoas especializadas no assunto.

Classificar esse tipo de dado é difícil até mesmo para um especialista no domínio. Existe uma abordagem em *Natural Language Processing* (NLP)(em português, Processamento de Linguagem Natural), conhecida por NER(em português, Reconhecimento de Entidade Nomeada), que pode solucionar esse problema. NER consiste na tarefa de identificar entidades nomeadas, a partir de textos de qualquer domínio e rotulá-las dentro de um conjunto de categorias pré-definidas, tais como nomes próprios, localização e organização (MOTA *et al.*, 2007).

Adaptar um modelo NER para um domínio que possui mais categorias do que o conjunto padrão de classes pré estabelecidas para este tipo de modelo, é um desafio (MA; XIA, 2014). Este trabalho propõe a ferramenta *Human NERD*. O Human NERD tem como principal fundamento manter o usuário envolvido no processo de construção de grandes conjuntos de dados de entidades nomeadas. A entrada para a estrutura Human NERD é um conjunto de documentos para anotar e um conjunto de rótulos NER. A saída é um conjunto de documentos anotados, um modelo de *Deep Learning* para reconhecimento de entidades e os valores das métricas de avaliação que podem aferir o custo operacional durante o tempo de anotação e o ganho em relação à precisão do modelo.

A cada ciclo de interação, o *Human NERD* sugere uma anotação de entidade, e o usuário como revisor pode aceitar ou rejeitar essa anotação. Ele também pode marcar um novo trecho do texto no documento com uma entidade que não foi sugerida pelo modelo NER. O *feedback* é usado para aperfeiçoar o modelo na próxima interação e enriquecer o conjunto de dados (FOLEY *et al.*, 2018).

NER pode ser utilizado em diversas abordagens com domínios diferentes. O trabalho de (LIU *et al.*, 2019) procura por entidades nomeadas em textos de relatórios médicos. Em (FOLEY *et al.*, 2018), é utilizada a mesma técnica de aproveitamento do *feedback* do usuário como revisor, entretanto, busca entidades de outro domínio. A ferramenta Human NERD proporciona um ambiente genérico que pode ser usado em outros domínios, mas para este

trabalho a proposta é deixar eficiente o processo de extração de entidades nomeadas em BOs policiais com a união das predições de um modelo construído utilizando o *framework Spacy* e o outro a biblioteca *Keras*.

1.1 Objetivos

Nesta seção, serão apresentados os objetivos deste trabalho.

1.1.1 Objetivo geral

O objetivo geral deste trabalho é fornecer um *framework* interativo, denominado Human NERD, para auxiliar o usuário nas tarefas de classificação NER, desde a criação de um conjunto de dados massivo até a construção/manutenção de um modelo NER de *Deep Learning*.

1.1.2 Objetivos específicos

a) Desenvolver a ferramenta Human NERD com suporte para algoritmos de redes neurais profundas, utilizados para o processamento de linguagem natural.
b) Construir um modelo NER de *Deep Learning* usando o Human NERD com um desempenho satisfatório para reconhecimento de entidades nomeadas em BOs.
c) Comparar o desempenho entre um modelo NER construído com o *Spacy* e outro modelo NER construído com o *Keras*.

1.2 Organização

Este trabalho está organizado da seguinte forma: no Capítulo 2, são apresentados os termos e conceitos necessários para o entendimento do trabalho. No Capítulo 3, são retratados cinco trabalhos relacionados com o presente. O Capítulo 4, contém os procedimentos metodológicos que foram seguidos durante a realização deste trabalho. O Capítulo 5 apresenta os resultados. Por fim, o Capítulo 6 apresenta as conclusões finais do trabalho.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo, serão apresentados os principais conceitos relacionados a este trabalho.

2.1 *Named Entity Recognition*

NER ou Reconhecimento de Entidade Nomeada é a tarefa de identificar e classificar palavras ou frases em um texto de acordo com classes definidas para o modelo NER (NADEAU, 2007). As classes convencionais incluem Pessoa, Localização, Organização e mais comumente um tipo diverso para incluir vários outros tipos que não necessariamente estão nesse conjunto de classes convencionais (KONKOL *et al.*, 2015).

NER baseia-se em duas atividades principais: A identificação de *tokens* em um texto não estruturado e a classificação desses *tokens* em tipos de entidades definidas de acordo com a peculiaridade do domínio (SPECK; NGOMO, 2014). Além do conjunto de entidades pré-definidas, é possível adicionar um conjunto extra de entidades nomeadas ao NER para adaptá-lo a um contexto específico. Por exemplo, no domínio de ocorrências policiais, o tipo da arma, nome da vítima, homicídio e entre outras são entidades próprias do domínio.

O principal conteúdo de um documento são os nomes das entidades. NER é um passo intermediário para extração e gerenciamento de informações mais ágil. Embora seja relativamente simples construir um sistema NER com desempenho razoável, ainda é desafiador adaptá-lo a um domínio especifico e obter ótimo desempenho (MA; XIA, 2014).

No entanto, este trabalho foca em criar um modelo NER para reconhecimento de entidades em BOs policiais com um bom desempenho e sem problemas de ambiguidade.

2.2 *Deep Learning*

Deep Learning ou Aprendizagem Profunda é uma subárea dentro do Aprendizado de Máquina, que é baseada em algoritmos que aprendem em múltiplos níveis de representações e formam complexas relações entre os dados. Aprendizagem profunda consiste no desafio de treinar redes neurais artificiais que executam o aprendizado de características de forma hierárquica, de tal maneira que as características no topo da hierarquia sejam formadas pela combinação de características que estão na raiz da hierarquia (DENG *et al.*, 2014).

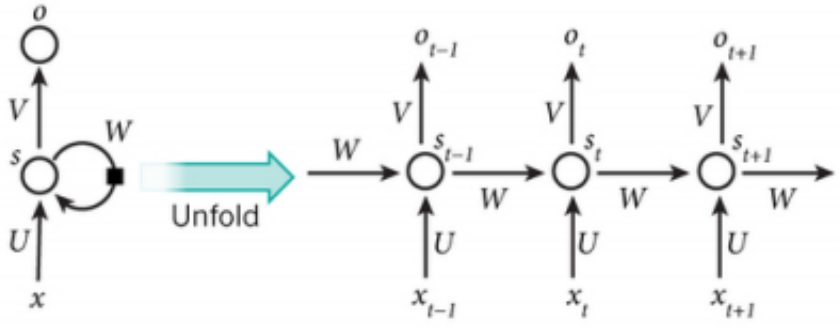
A expressão Redes Neurais referencia-se à neurobiologia, mas para (CHOLLET,

2018) apesar dessa semelhança entre os conceitos, redes neurais não são modelos do cérebro, mas sim um arranjo matemático para aprender representações a partir dos dados. Cada uma possui sua arquitetura específica, definindo o modo como suas unidades de processamento estão conectadas. A arquitetura influencia nos problemas que uma rede neural pode resolver, como reconhecimento de imagens, fala e processamento de linguagem natural (BEZERRA, 2016).

Existem variados tipos de arquiteturas para redes neurais. Para processamento de informação sequencial (textos, áudio e vídeo), as mais recomendadas são as Redes Neurais Recorrentes ou *Recurrent Neural Networks* (RNN). Para atividades em Processamento de Linguagem Natural, é importante a memória das RNN, que permite registrar que informações foram processadas até o momento (BEZERRA, 2016).

A Figura 1 apresenta o estado mais simples possível de uma RNN que possui apenas uma camada oculta, composta de uma única unidade, denotada por s. A unidade oculta contém uma conexão que a liga consigo mesma. Assim, s recebe sinais que foram propagados pela própria rede no passado em um instante de tempo, denotado por t. Desta forma, uma RNN pode mapear uma sequência de entrada com x_t elementos em uma sequência de saída com o_t elementos, com cada um dependendo de todos os x_t anteriores (para $t' \leq t$, onde t' e t são instantes de tempo). Os mesmos parâmetros U, V e W (matrizes) são usados em cada etapa de tempo (LECUN *et al.*, 2015).

Figura 1 – *Recurrent Neural Networks*, RNN.



Fonte: LeCun *et al.* (2015)

2.3 *Human in the Loop*

Apesar de que tenha havido um importante progresso no desempenho dos algoritmos de processamento de linguagem natural, os modelos de última geração tendem a ser famintos

por dados. Grandes conjuntos de dados de treinamento, caros e tediosos para elaborar, são necessários para otimizar os parâmetros desses modelos. Os conjuntos de dados existentes estão se tornando desatualizados e novos domínios com poucos dados precisam ser explorados (YU *et al.*, 2015).

Human in the Loop (ou humano no ciclo) é uma técnica proposta para ampliar o esforço humano por meio de um esquema de rotulagem parcialmente automatizado, onde é feita a junção das classificações do modelo (etapa automatizada) com as correções do humano (etapa manual). Assim é melhorado o aprendizado dos modelos com seres humanos no ciclo (YU *et al.*, 2015). A cada iteração do modelo, o humano faz correções ou acrescenta melhorias, gerando um *feedback*, que posteriormente será utilizado para refinar o modelo (FOLEY *et al.*, 2018).

2.4 Métricas

O próximo passo depois do desenvolvimento de um modelo preditivo, é saber o desempenho do modelo para o futuro (sobre os dados que ele não viu durante o processo de construção). Para problemas de classificação (problema abordado neste trabalho), a principal fonte de medições de desempenho é uma matriz de confusão (OLSON; DELEN, 2008).

O Quadro 1, mostra uma matriz de confusão para um problema de classificação de duas classes.

Quadro 1 – Um exemplo simples de Matriz de Confusão.

	Positivo	**Negativo**
Positivo	*True Positives* (TP)	*False Positives* (FP)
Negativo	*False Negatives* (FN)	*True Negatives* (TN)

Fonte: Adaptado de Olson e Delen (2008).

A matriz de confusão resume o desempenho da classificação de um modelo com relação a alguns dados de teste. É uma matriz bidimensional, indexada em uma dimensão pela classe verdadeira de um objeto e na outra pela classe que o modelo atribuiu. As quatro células da matriz são designadas como TP, FP, FN e TN, conforme indicado na Tabela 1. Algumas medidas de desempenho de classificação são definidas em termos desses quatro resultados (SAMMUT; WEBB, 2011).

2.4.1 Precision

Precision (ou Precisão) é definida como uma razão de TP e o número total de positivos previstos por um modelo (SAMMUT; WEBB, 2011). A precisão é definida em termos de TP e FP da seguinte forma:

$$\frac{TP}{(TP+FP)} \tag{2.1}$$

2.4.2 Recall

Recall ou *Sensitivity*, em português revocação, é a proporção de casos positivos reais que são corretamente previstos como positivos (POWERS, 2007). *Recall* é definido pela seguinte fórmula:

$$\frac{TP}{(TP+FN)} \tag{2.2}$$

2.4.3 F1-Score

F1-Score ou *F-measure* (porque o *recall* e a *precision* são igualmente ponderados) fornece uma maneira de combinar o *recall* e a *precision* para obter uma única medida. O *recall* e a *precision* podem ter pesos relativos no cálculo da *F1-Score*, dando a flexibilidade de ser usada para diferentes aplicações (CHINCHOR, 1992). A fórmula para calcular a *F1-Score* é:

$$\frac{(2 * precision * recall)}{(precision + recall)} \tag{2.3}$$

3 TRABALHOS RELACIONADOS

Este capítulo apresenta alguns trabalhos relacionados, destacando as semelhanças e diferenças com o desenvolvido neste trabalho.

3.1 *Recognition and Extraction of Named Entities in Online Medical Diagnosis Data Based on a Deep Neural Network*

Existem diversos domínios que oferecem grandes quantidades de textos que precisam passar pelo processo de extração de entidades nomeadas. Por exemplo, Liu *et al.* (2019) propõe uma nova rede neural denominada DNN, para reconhecimento de entidades em relatórios médicos. Informações como nomes de doenças, medições médicas e terapias, precisam ser extraídas com eficiência. A DNN é a combinação da *bi-directional long short-term memory (Bi-LSTM)* e da *conditional random field (CRF)* e não depende de regras ou recursos manuais. Liu *et al.* (2019) realizaram experimentos com várias arquiteturas, e obtiveram bons resultados, porém seu modelo não teve melhor desempenho por conta do corpus utilizado possuir uma pequena quantidade de dados para treino.

A grande semelhança entre Liu *et al.* (2019) e este trabalho, está no uso de um modelo de *Deep Learning* para uma tarefa NER. Ambos procuram entidades nomeadas, entretanto, para domínios diferentes. A diferença entre o presente trabalho e Liu *et al.* (2019) está no fato que este trabalho propõe uma ferramenta para tornar eficiente o processo de extração de entidades nomeadas e utiliza redes neurais existentes. Em Liu *et al.* (2019) é proposta uma nova implementação de rede neural e não propõe uma ferramenta.

3.2 *Named Entity Recognition with Extremely Limited Data*

Na abordagem tradicional, o reconhecimento de entidades nomeadas é analisado como uma tarefa de anotação de corpus de pré-indexação. No entanto, existem inúmeras classes de entidades nomeadas, isso torna o trabalho de rotulação bastante custoso. O trabalho de Foley *et al.* (2018) propõe explorar o reconhecimento de entidades nomeadas como uma tarefa de pesquisa, em que a classe de interesse de uma entidade nomeada é uma consulta e as entidades dessa classe são os documentos relevantes. Essa abordagem foi proposta para ser usada sobre uma classe de entidades efêmeras ou contextuais, quando não é vantajoso rotular grandes quantidades de instâncias para treinar um classificador. Em sua abordagem Foley *et al.* (2018) não utiliza

recursos de redes neurais, mas trabalhou com o *feedback* do usuário para melhorar o desempenho das suas consultas.

A interseção entre este trabalho e o de Foley *et al.* (2018) ocorre no uso do *feedback* do usuário como estratégia para corrigir eventuais classificações equivocadas feitas pelo modelo e, em paralelo, construir um significativo conjunto de dados para treinamento. A diferença entre ambos está no uso dos recursos e nos domínios trabalhados. Com o domínio que Foley *et al.* (2018) utilizou, foi possível obter significativos resultados com recursos artesanais. Entretanto neste trabalho, pelas características do domínio e pela forma como as informações que precisam ser extraídas estão dispostas no domínio, o uso de recursos que não utilizavam redes neurais não trouxe resultados satisfatórios. Outra distinção, é a proposta feita por este trabalho de uma ferramenta para deixar a tarefa NER mais acessível para usuários leigos e facilitar o processo de rotulação.

3.3 *Named entity recognition with bidirectional LSTM-CNNs*

Em Chiu e Nichols (2016) é apresentada uma nova arquitetura de rede neural inspirada no trabalho de Collobert *et al.* (2011), que detecta automaticamente recursos a nível de caractere usando uma arquitetura híbrida bidirecional *Long Short Term Memory* (LTSM) e *Convolutional Neural network* (CNN). Além disso, propõe um novo método de codificar correspondências parciais de léxico em redes neurais. Utilizando dados do *CoNLL-2003* e do *OntoNotes*, conseguiu pontuações *F1-Score* de 91.62 e 86.28, respectivamente. Entretanto, seu modelo requer uma grande quantidade de dados para treinamento. Isso é um empecilho para domínios com pequenos volumes de dados para treino.

O trabalho de Chiu e Nichols (2016) assemelha-se com o presente pelo uso das redes neurais LTSM e CNN para executar uma tarefa NER. A diferença está pela nova rede neural que Chiu e Nichols (2016) apresenta e pelo método de entrada dessa nova rede neural. Neste trabalho são utilizadas redes neurais existentes na comunidade e é proposta uma ferramenta para facilitar a execução de algoritmos de *Deep Learning* e diminuir o custo com anotações. Importante lembrar que os domínios trabalhados são totalmente distintos.

3.4 *Improving Named Entity Recognition using Deep Learning with Human in the Loop*

O trabalho Silva *et al.* (2019a) propõe uma estrutura interativa denominada Human NERD, para auxiliar o usuário nas tarefas de classificação NER, desde a criação de um conjunto de dados massivo até a construção/manutenção de um modelo NER de aprendizagem profunda. A arquitetura da ferramenta é baseada em duas visões, cientista de dados e revisor no ciclo. Na visão cientista de dados, o usuário realiza operações (criar, treinar, apagar e adicionar conjunto de treino) sobre o(s) modelo(s). Como revisor, o usuário participa do processo de treino do(s) modelo(s) realizando correções nas predições ou adicionando novos rótulos ao conjunto de treino. A ferramenta incorpora modelos de aprendizado profundo baseados na biblioteca *Keras* e no *framework Spacy*.

Os pontos em comum entre o presente trabalho e Silva *et al.* (2019a) estão na proposta de uma ferramenta para auxiliar o usuário em atividades NER e os modelos de aprendizado profundo que são os mesmos utilizados por ambos os trabalhos. É importante ressaltar que este trabalho é uma extensão do trabalho (SILVA *et al.*, 2019a).

3.5 *Novel approach for Label Disambiguation via Deep Learning*

Em Silva *et al.* (2019b) é proposta uma arquitetura neural livre de bases de conhecimento, recursos específicos de idioma (por exemplo, dicionários geográficos, id de grupos de palavras, *tags* de parte da fala) ou recursos artesanais (por exemplo, padrões de ortografia de palavras e de capitalização). A arquitetura é uma combinação de LTSM bi-direcional e camadas *Classical Receptive Field* (CRF). Silva *et al.* (2019b) obtém resultados satisfatórios utilizando essa arquitetura para resolver um problema denominado Desambiguação de Rótulos. Seu melhor resultado foi obtido utilizando uma rede *Char-BLSTM-CRF*, com um conjunto de dados sobre BOs. O experimento tinha como objetivo o modelo identificar no texto a VÍTIMA, o ASSASSINO e OUTROS.

Silva *et al.* (2019b) assemelha-se com o presente trabalho por trabalharem com o mesmo conjunto de dados (sobre BOs). A rede neural com que obteve o melhor desempenho nos seus experimentos será incorporada pela ferramenta Human NERD, proposta nesse trabalho. O modelo NER construído utilizando a rede neural Bi-LSTM-CRF está sendo utilizado nesse trabalho. A diferença entre ambos os trabalhos, é que este trabalho propõe uma ferramenta e dois modelos NLP, enquanto Silva *et al.* (2019b) foca em propor um modelo NLP.

3.6 Análise Comparativa

O Quadro 2 apresenta um comparativo entre os trabalhos relacionados com o atual, destacando as principais abordagens utilizadas ou desenvolvidas por cada trabalho.

As principais diferenças entre os trabalhos relacionados e o presente, está no fato deste utilizar técnicas já existentes, enquanto os demais criam novas abordagens. Além disso, este trabalho propõe uma ferramenta para facilitar a execução/construção de modelos. Nenhum dos trabalhos relacionados fazem tal proposta.

Quadro 2 – Comparativo entre os trabalhos relacionados e o trabalho proposto.

	Liu *et al.* (2019)	Foley *et al.* (2018)	Chiu e Nichols (2016)	Silva *et al.* (2019a)	Silva *et al.* (2019b)	TRABALHO PROPOSTO
Usa o *feedback* do usuário	-	Seleção das entidades úteis	-	Correções sobre a predição do modelo	-	Correções sobre a predição do modelo
Usa rede neural para tarefa NER	DNN	-	LTSM e CNN	LTSM e CNN	LTSM	LTSM e CNN
Propõe uma nova rede neural	DNN	-	Híbrida (LTSM e CNN)	-	Híbrida (LTSM e CRF)	-
Propõe uma ferramenta	-	-	-	Human NERD	-	Human NERD
Adaptou o NER para um domínio específico	Relatórios Médicos	*CoNLL'03* e *AQUAINT*	*CoNLL-2003* e *OntoNotes*	BOs	BOs	BOs

Fonte: Elaborado pelo autor.

4 PROCEDIMENTOS METODOLÓGICOS

Este capítulo apresenta os passos necessários para a execução deste trabalho. A seguir, cada estágio é melhor detalhado.

4.1 Coleta de Dados

Os BOs são extraídos da base de dados da SSPDS. Existem diversos tipos de BOs (ocorrências). É necessária a realização de um filtro sobre esses documentos para que sejam selecionados apenas os BOs que descrevem uma ocorrência de homicídio. Outros tipos de ocorrências oferecem classes que não são o foco deste trabalho, eliminá-las evita interferência no desempenho do modelo.

O filtro é realizado manualmente e, inicialmente, é realizada uma leitura em busca de informações chaves, como nome da vítima e a palavra “Homicídio”. Encontradas essas informações, o texto é selecionado para o conjunto de treino do modelo NER, caso contrário, ele é ignorado.

4.2 Pré-processamento

Este trabalho utiliza um mapeamento <*token*, número>, construído a partir do conjunto de *tokens* de todos os BOs selecionados. Cada *token* possui um número associado. Esse número é definido pelo intervalo de 1 até o tamanho do conjunto de *tokens*.

Particularmente, tanto o *framewok Spacy* quanto a biblioteca *Keras*, possuem entradas de dados distintas. O conjunto de treino para o *Spacy* é construído a partir de um *Array* para cada texto, esse *Array* contém o texto propriamente dito e outro *Array* contém informações (onde começa e termina e qual é a classe) sobre as classes presentes no texto, da seguinte forma: *Array*(texto, ‘entidades’: [(‘inicio’, ‘fim’, ‘classe’)]).

Para o conjunto de entrada do *Keras* são necessários dois *Arrays*, o *X_train* e o *Y_train*. O *X_train* é um *Array* com números extraídos do dicionario mencionado na Seção 4.1, onde cada numero representa uma palavra do texto original. O *Y_train* também é um *Array* numérico, com N outros *Arrays*, onde N é o número de palavras do texto. Uma sequência binaria é pré determinada para identificar cada classe, cada um dos N *Arrays* possuem essa sequência identificando a palavra daquela posição.

4.3 Definição da Arquitetura Human NERD

O Human NERD é desenvolvido utilizando os recursos da linguagem *Python* (para o *back-end*) e da biblioteca *React* (para o *front-end*). A primeira implementação a ser feita é o suporte para o *framework Spacy*, e posteriormente o suporte a biblioteca *Keras*. Abrangendo diversos algoritmos de *Deep Learning*. A ferramenta possui um fluxo de trabalho dividido em visões. A seguir é melhor definida cada uma delas.

4.3.1 Revisor no Ciclo

O modelo pré-treinado (fornecido pelo *Spacy*) e o humano participam no ciclo do processo de refinamento do modelo NER. O usuário como revisor interage com a estrutura por meio de uma interface de clicar e arrastar. A estrutura apresenta um conjunto de documentos anotados pelo modelo pré-treinado de acordo com o conjunto de classes pré-definidas.

O usuário tem autonomia para concordar com as anotações do modelo, caso estejam corretas. Podendo também rejeitar anotações individuais, dado que estão erradas. O Human NERD estimula o usuário a fazer mais anotações, porém essas anotações vão diminuindo à medida que o modelo evolui seu aprendizado em relação as predições sobre o conjunto de dados utilizado.

4.3.2 Cientista de Dados no Ciclo

Nesta visão a ferramenta mostra para o cientista de dados o detalhamento completo sobre os modelos importados ou já treinados.

A ferramenta reporta o *status* da revisão e do processamento do modelo. É exibido quantos documentos o revisor já validou, e quantas épocas já terminaram durante o treinamento do modelo. Informações sobre o esforço do usuário para anotar documentos, também são mostradas.

4.4 Implementação do suporte para o *framework Spacy*

Inicialmente foram realizados alguns testes com o *Spacy*. Para um conhecimento mais a fundo sobre seu funcionamento, modelos foram criados e treinados. Nesse passo o objetivo é encapsular as principais funcionalidades disponibilizadas pelo *Spacy* ao Human

NERD, de forma abstrata para o usuário. A partir de cliques e de uma interface interativa o usuário realiza tarefas NER, como retreinar modelos, rotular novos textos para o conjunto de treino e realizar correções sobre as previsões feitas pelo modelo.

4.5 Implementação do suporte para a biblioteca *Keras*

Treinar uma rede neural disponibilizada pelo *Keras* não é uma tarefa simples para o usuário que ainda é leigo no assunto. A forma como o Human NERD implementa esse suporte ao *Keras*, disponibiliza uma interface interativa, onde o usuário executa redes neurais e cria conjuntos de treinos para alimenta-las, sem a necessidade de conhecimentos abrangentes sobre *Deep Learning*. Vale ressaltar que a facilidade oferecida pelo Human NERD não dispensa o conhecimento do usuário sobre o uso das redes neurais para obter resultados compatíveis com o estado da arte.

4.6 Validação da ferramenta Human NERD

Com a ferramenta em execução, é necessário expor a estrutura em cenários reais para fins de testes. Adicionalmente elabora-se um modelo NER com boa precisão para identificar entidades nomeadas em BOs.

Este trabalho avalia os modelos de NER através das métricas *Precision*, *Recall* e *F1-Score*. A avaliação empírica baseada nesses indicadores desempenha um papel central na estimativa de desempenho dos sistemas de processamento de linguagem natural NLP (GOUTTE; GAUSSIER, 2005).

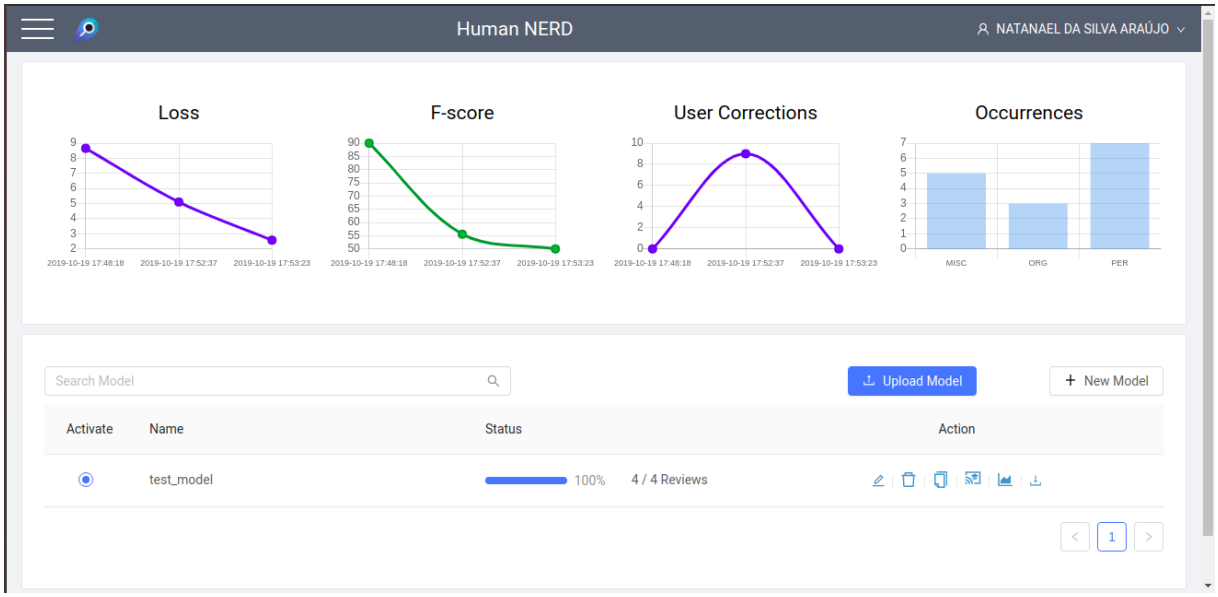
5 RESULTADOS

Este capítulo apresenta os resultados obtidos na execução deste trabalho.

5.1 Ferramenta Human NERD

Na sua versão atual, denominada como “Versão 1.0”. O Human NERD disponibiliza suporte para o *framework Spacy*, possibilitando a criação e manutenção de modelos NER, por meio de uma interface interativa com o humano participando ativamente do ciclo de criação e aperfeiçoamento dos modelos. A Figura 2 apresenta a página inicial do Human NERD com um modelo ativo.

Figura 2 – Página inicial do Human NERD.

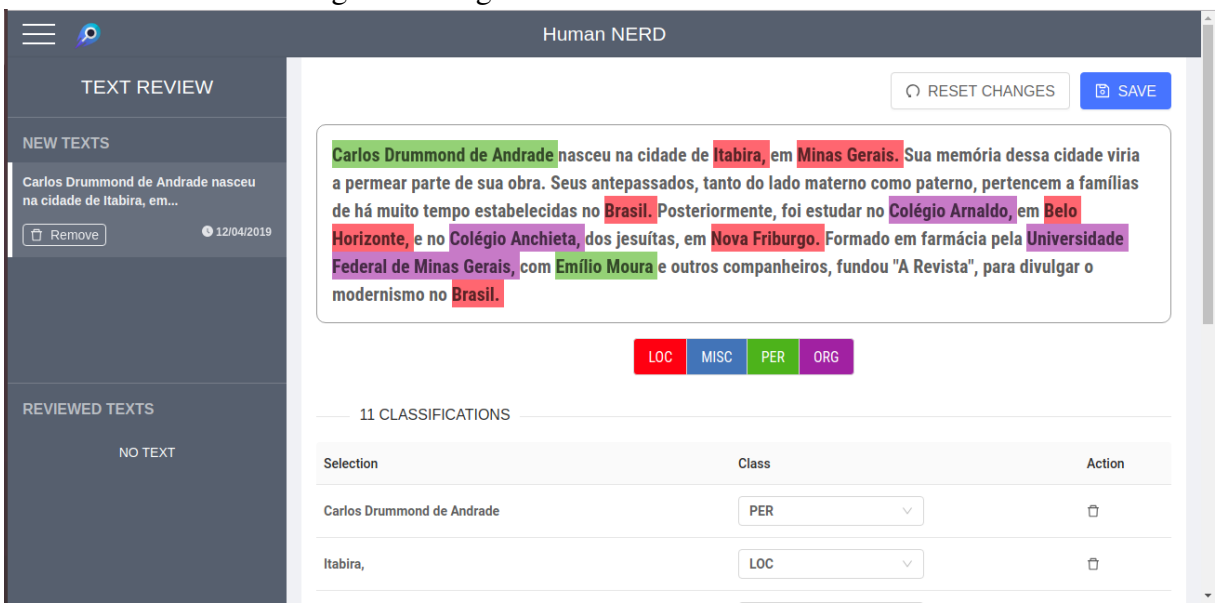


Fonte: Elaborado pelo autor.

Na página inicial, o cientista de dados pode realizar operações sobre modelos, como criar, excluir, atualizar, treinar e importar. Podendo também observar alguns gráficos estatísticos sobre o modelo que está ativo. A partir dessa tela, é possível acessar a página do revisor assim como todas as outras funcionalidades da ferramenta.

A Figura 3 apresenta a página do revisor, onde ele pode adicionar rótulos que não foram rotulados pelo modelo ou aceitar a rotulação feita pelo modelo. É importante ressaltar que nessa fase acontece o refinamento do conjunto de treino e do modelo pelo *feedback* do usuário. O texto apresentado na Figura 3 é ilustrativo, por motivos de confidencialidade não é permitido exibir textos de BOs.

Figura 3 – Página do Revisor no Human NERD



Fonte: Elaborado pelo autor.

A partir da página inicial (visão do cientista de dados) é possível visualizar informações estatísticas sobre os modelos existentes no Human NERD. Na Figura 4 é apresentada a página com as estatísticas de um modelo que foi previamente treinado com dados fictícios.

Figura 4 – Página de Estatísticas no Human NERD

Fonte: Elaborado pelo autor.

Na página de estatísticas, são apresentados os gráficos com os valores ao longo do tempo das métricas (*losses*, *Fscore* e *kappa*) que são utilizadas para avaliar o modelo durante o treino. Os gráficos também ilustram ações feitas pelo revisor. As quantidades de correções feitas

são mostradas no gráfico *User Corrections*. O gráfico *Occurrences* apresenta a frequência de cada classe no conjunto de treino.

5.2 Modelo NER para classes não ambíguas

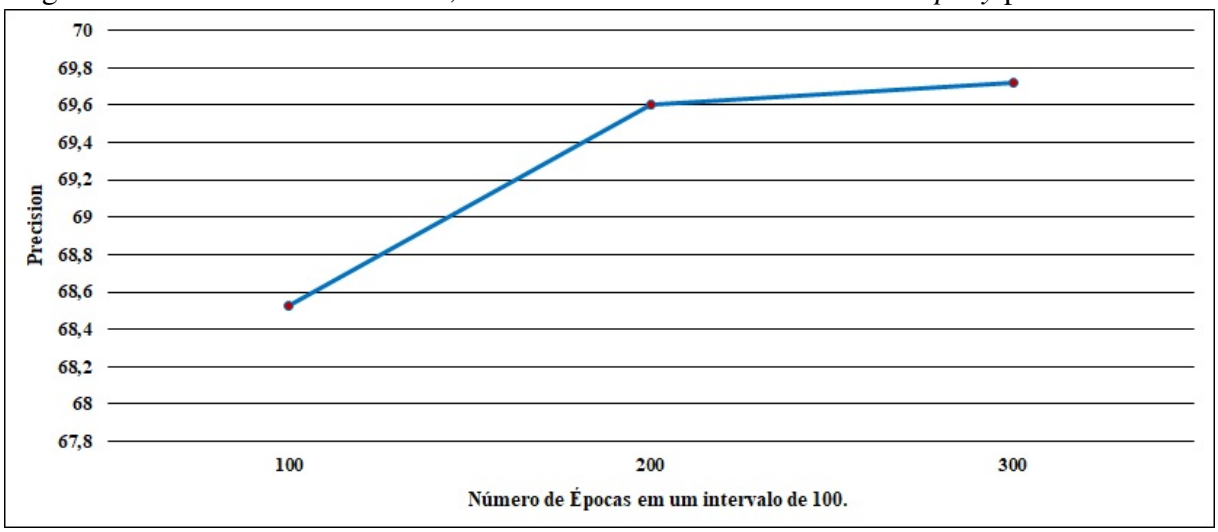
Esta seção apresenta os resultados obtidos com a construção de dois modelos NER para classes que não são ambíguas (Pessoa (PER), Localização (LOC) e Organização (ORG)). Esse tipo de classe permite aos classificadores às rotularem sem levar em consideração o contexto que estão inseridas.

5.2.1 Resultados para classes não ambíguas utilizando o Spacy

Um modelo NER foi construído a partir do *framework Spacy* e treinado com um conjunto de dados sobre BOs. O conjunto de dados é real, com textos dos relatórios da Polícia do estado do Ceará. Este modelo apresenta resultados satisfatórios, identificando informações como tipo da arma, se a ação foi um feminicídio, um achado de cadáver ou um acidente de trânsito. É importante ressaltar que foi utilizado o modelo base (contendo apenas as classes LOC, PER, ORG e Misto (MISC)) do *Spacy* para a língua portuguesa, que posteriormente foi adaptado para esse conjunto de dados.

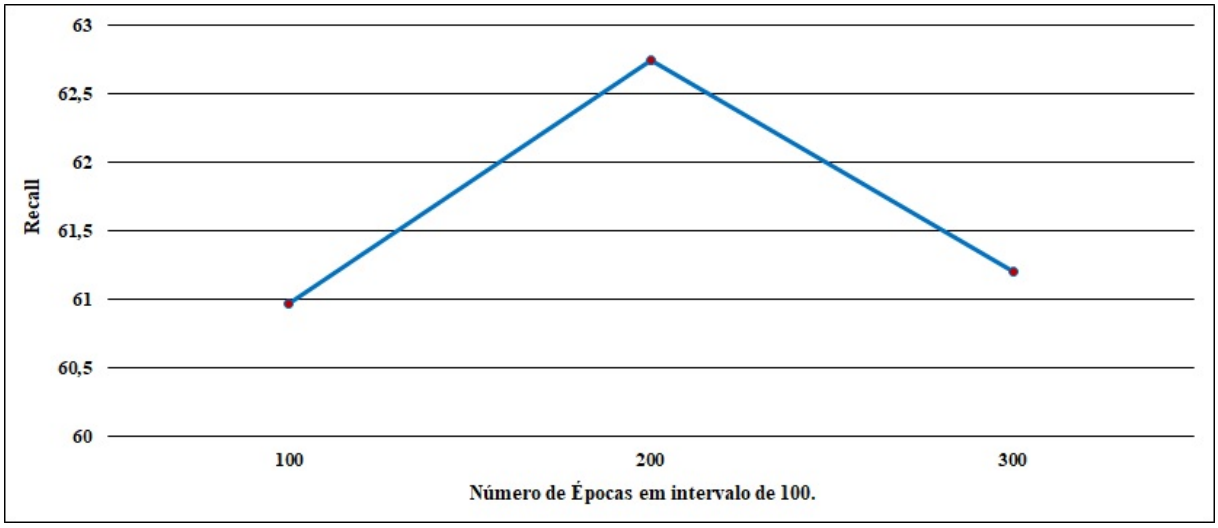
A seguir nas Figuras 5, 6 e 7, são apresentados os valores das métricas *F1-Score*, *Precision* e *Recall* obtidos durante o treino do modelo. Os valores foram capturados em intervalos durante o processo de treino. Nesse experimento foi utilizado um conjunto de dados com 994 textos, sendo 788 (80%) para treino e 198 (20%) para teste e 300 épocas. O modelo foi adaptado com 24 classes, incluindo as classes (PER, LOC, ORG e MISC) que compõem por padrão o modelo em português.

Figura 5 – Resultados de Precisão, obtidos durante o treino do modelo *Spacy* para 24 classes.



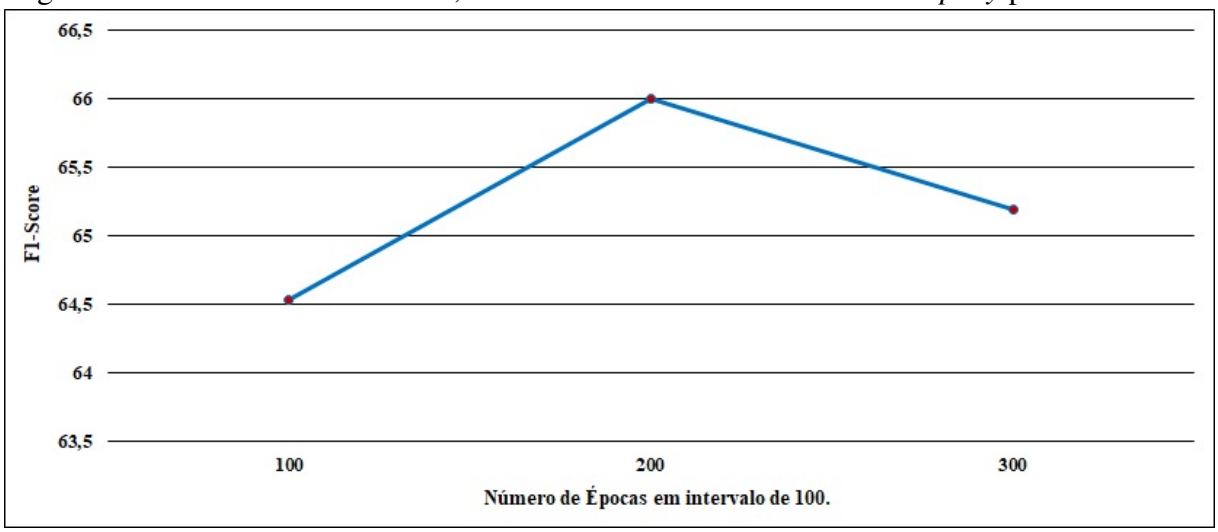
Fonte: Elaborado pelo autor.

Figura 6 – Resultados de *Recall*, obtidos durante o treino do modelo *Spacy* para 24 classes.



Fonte: Elaborado pelo autor.

Figura 7 – Resultados de *F1-Score*, obtidos durante o treino do modelo *Spacy* para 24 classes.



Fonte: Elaborado pelo autor.

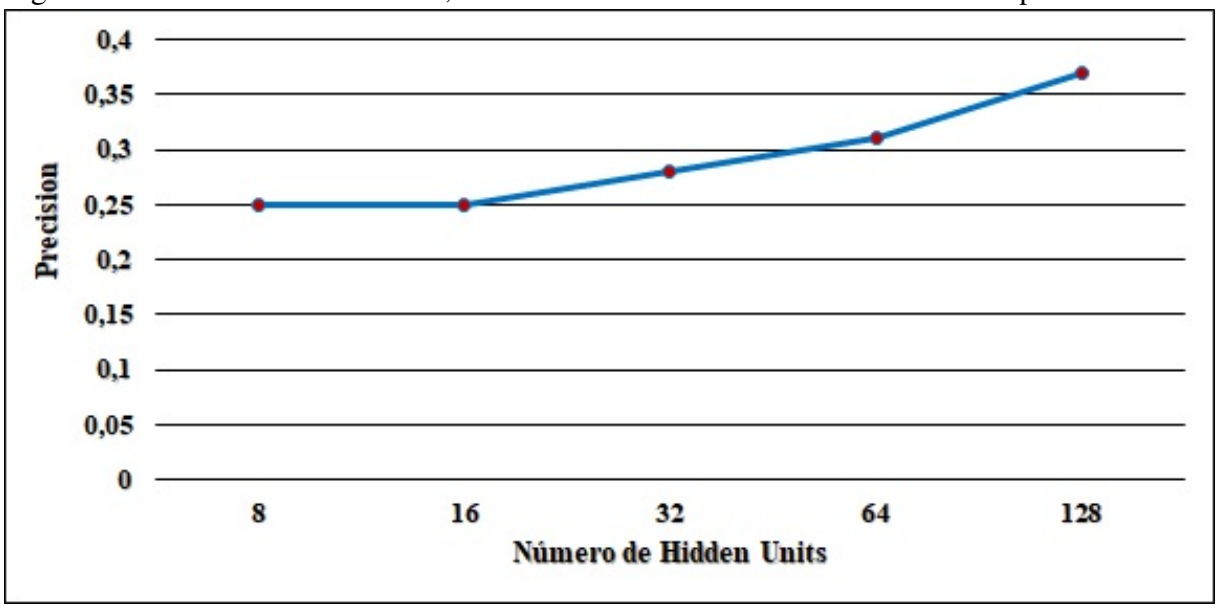
Como observado nas Figuras 5, 6 e 7, o modelo atingiu um valor de *precision* (69%) considerado satisfatório. Com essa precisão (*precision*) o modelo consegue extrair varias informações (tipo da arma, se foi latrocínio e etc) dos BOs de forma eficiente. Entretanto para distinguir o nome da vitima do nome do acusado o modelo confunde-se com alta frequência.

5.2.2 Resultados para classes não ambíguas utilizando o Keras

Nesta seção, serão apresentados os resultados de *Precision*, *Recall* e *F1-Score* de um modelo construído a partir dos mesmos dados (conjunto de treino, teste e as 25 classes, esse modelo leva em consideração a classe OTHER) utilizados para a construção do modelo apresentado na seção 5.2.1. O modelo aqui apresentado foi construído com a biblioteca *Keras*, utilizando uma rede neural *Bi-LSTM* com CRF. Para cada execução foi definido o número máximo de 300 épocas e uma variação no número de *hidden units* (8, 16, 32, 64, 128), onde cada variação representa um modelo.

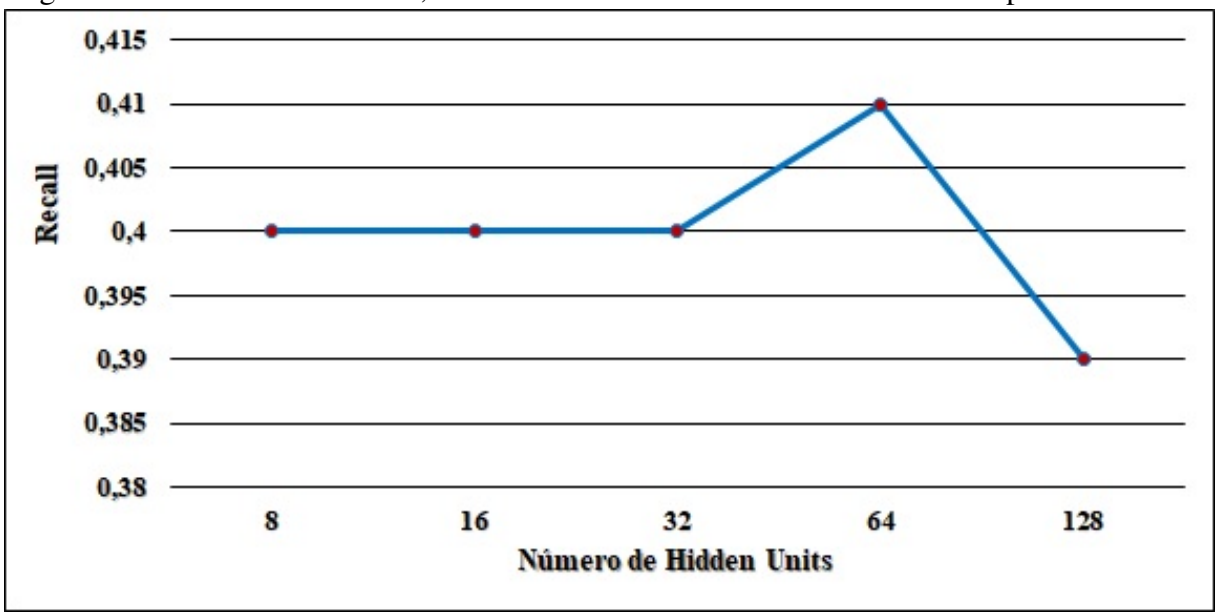
Nas Figuras 8, 9 e 10, são apresentados os valores de *precision*, *recall* e *f1-score*. Esses valores representam uma média ponderada dos valores de *y_true* (valores verdadeiros) e *y_pred* (valores estimados pelo modelo). Os gráficos ilustram os valores obtidos em cada execução do modelo com a variação de *hidden units*.

Figura 8 – Resultados de *Precision*, obtidos durante o treino do modelo *BiLSTM* para 25 classes.



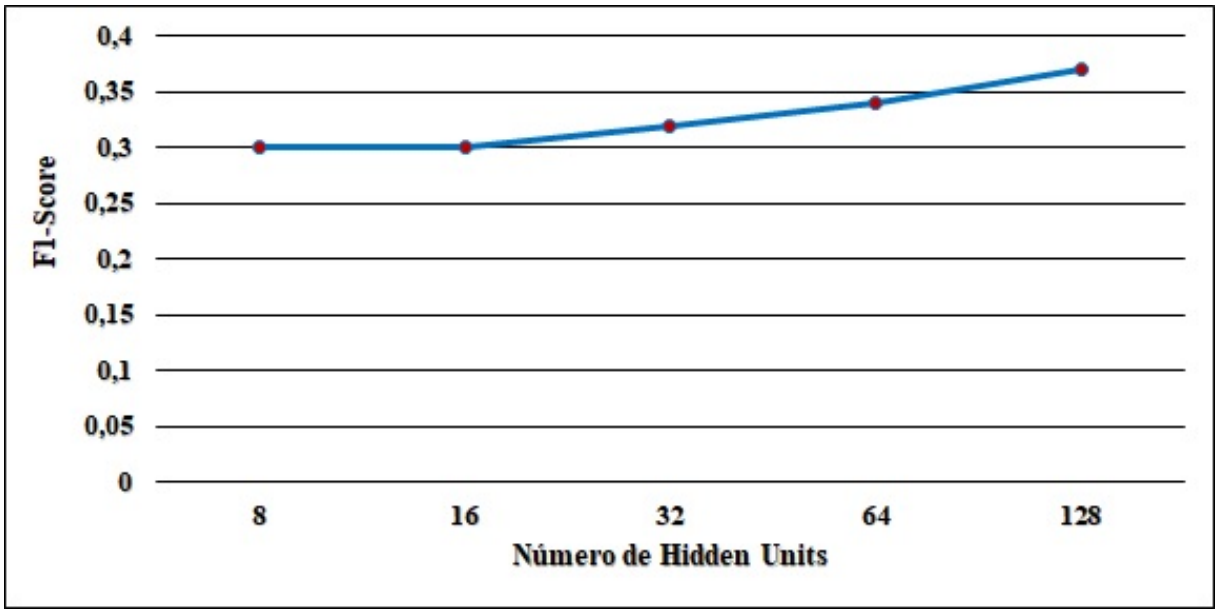
Fonte: Elaborado pelo autor.

Figura 9 – Resultados de *Recall*, obtidos durante o treino do modelo *BiLSTM* para 25 classes.



Fonte: Elaborado pelo autor.

Figura 10 – Resultados de *F1-Score*, obtidos durante o treino do modelo *BiLSTM* para 25 classes.



Fonte: Elaborado pelo autor.

Os gráficos (Figura 8, 9 e 10) expõem resultados insatisfatórios. Esse baixo desempenho justifica-se pela pequena quantidade de dados para treino de algumas classes. Algumas categorias acontecem com baixa frequência, por exemplo, as classes Triplo Homicídio (TRIPLO_HOM) e Legítima Defesa de Terceiros (LEG_DEF_TER), o fato da não ocorrência desses eventos, impacta em uma baixa ocorrência no conjunto de treino e compromete o desempenho de modelos como esse que são redes neurais que necessitam de grande quantidade de dados para treino. A partir desses resultados, evidencia-se que o modelo construído utilizando o *framework Spacy*, possui melhor desempenho na identificação dessas classes nesse conjunto de dados.

5.3 Modelo NER para classes ambíguas

Esta seção apresenta os resultados obtidos com a construção de dois modelos NER para classes ambíguas (Vítima e Assassino). Rotular esse tipo de classe é um problema desafiador. Os modelos tradicionais se confundem e rotulam Vítima e Assassino como pessoas, entretanto, é necessária uma rotulagem menos genérica.

5.3.1 Resultados para classes ambíguas utilizando o Keras

Esta seção apresenta os detalhes sobre o treinamento do modelo NER construído com a biblioteca *Keras*. Visto na Seção 5.2.1, que existe um problema para distinguir o nome da

vítima e o nome do assassino. O experimento aqui apresentado, tem como objetivo solucionar esse problema utilizando o mesmo conjunto de dados utilizado no experimento da Seção 5.2.1, entretanto, trabalhando apenas sobre duas classes (VÍTIMA e ASSASSINO). Nesse conjunto de dados, contém 1.399 textos para treino e 350 textos para teste, classificados manualmente.

O conjunto de dados foi rotulado usando três tipos diferentes de classes: VÍTIMA, ASSASSINO E OUTROS. Note que a VÍTIMA e o ASSASSINO podem ser confundidos pelos modelos tradicionais de classificação NER, uma vez que ambas as palavras preservam o mesmo recurso ortográfico. O conjunto de treino contém 3.507 menções para VÍTIMA, 2.902 para ASSASSINO e 499.628 para OUTROS. O conjunto de testes é composto por 1.274 menções rotuladas como VÍTIMA, 686 como ASSASSINO e 125.090 como OUTROS.

Os experimentos foram realizados com diferentes tipos de arquiteturas de redes, como BLSTM-CRF, BRNN-CRF e Char-BLSTM-CRF. É importante ressaltar que a qualidade dos resultados é afetada ao variar o número de unidades ocultas para uma camada. Os resultados apresentados nas Tabelas 2 e 2, foram obtidos usando os mesmos hiper-parâmetros para todos os modelos.

Quadro 2 – Resultados de *Precision* para as classes VÍTIMA, ASSASSINO e OUTROS.

Rede Neural	*Hidden Units*	VÍTIMA	ASSASSINO	OUTROS
BRNN-CRF	8	0,57	**0,47**	0,99
	16	0,60	0	0,99
	32	0,59	0	0,99
	64	0,60	0	0,99
	128	**0,62**	0,19	0,99
BLSTM-CRF	8	0,71	0,57	0,99
	16	0,71	0,57	0,99
	32	**0,79**	0,60	0,99
	64	**0,79**	0,60	0,99
	128	0,77	**0,61**	0,99
Char-BLSTM-CRF	8	0,80	0,56	0,99
	16	0,67	0,63	0,99
	32	0,67	0,47	0,99
	64	**0,85**	**0,65**	0,99
	128	0,76	0,49	0,99

Fonte: Elaborado pelo autor.

Para os três modelos avaliados, as redes foram treinadas usando o algoritmo de propagação reversa. Vários métodos têm sido propostos para melhorar o desempenho do

gradiente estocástico descendente (SGD), como Adadelta, Rmsprop (DAUPHIN *et al.*, 2015) e Adam. Apesar de que os melhores resultados foram obtidos usando o Rmsprop, o que é esperado, já que o otimizador do Rmsprop é geralmente uma escolha boa o suficiente, seja qual for o seu problema (CHOLLET, 2017).

Quadro 2 – Resultados de *Recall* para as classes VÍTIMA, ASSASSINO e OUTROS.

Rede Neural	*Hidden Units*	VÍTIMA	ASSASSINO	OUTROS
BRNN-CRF	8	**0,86**	0,01	0,99
	16	0,80	0	0,99
	32	0,81	0	0,99
	64	0,80	0	0,99
	128	0,70	**0,02**	0,99
BLSTM-CRF	8	**0,82**	0,43	0,99
	16	0,81	0,33	0,99
	32	0,76	0,47	0,99
	64	0,79	**0,61**	0,99
	128	0,77	0,47	0,99
Char-BLSTM-CRF	8	0,80	0,58	0,99
	16	**0,83**	0,30	0,99
	32	0,78	0,37	0,99
	64	0,73	**0,64**	0,99
	128	0,82	0,54	0,99

Fonte: Elaborado pelo autor.

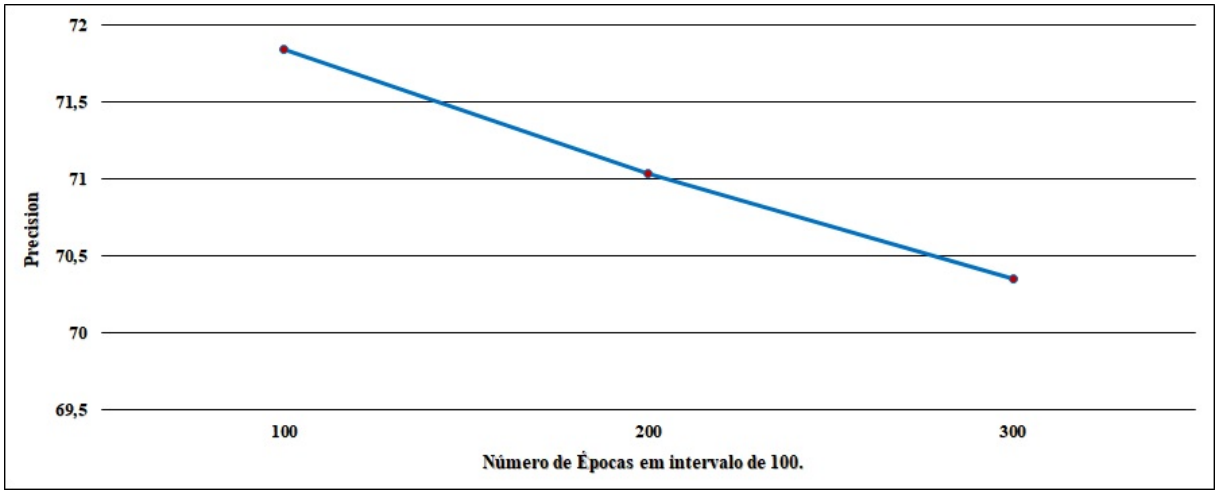
De acordo com os resultados mostrados nas Tabelas 2 e 2, Char-BLSTM-CRF e BLSTM-CRF superam o BRNN-CRF em métricas de precisão (*precision*) e recuperação (*recall*) para as classes VÍTIMA e ASSASSINO. Detectar a classe OUTROS não é o foco desse trabalho. Note que foi adicionada uma camada de CRF às redes neurais consideradas, uma vez que foi demonstrado que sequências de etiquetas de decodificação conjuntas podem beneficiar significativamente o desempenho final de modelos de redes neurais (HUANG *et al.*, 2015).

5.3.2 Resultados para classes ambíguas utilizando o Spacy

A seguir, são apresentados os resultados obtidos a partir da construção de um modelo NER utilizando o *Spacy*. Para esse modelo, foi utilizado o mesmo conjunto de dados e classes apresentados na Seção 5.3.1. É importante ressaltar que o modelo foi adaptado apenas para duas classes (VITIMA e ASSASSINO) a partir do modelo base em português fornecido pelo *Spacy* que possui quatro classes (LOC, PER, ORG e MISC). O modelo foi treinado por 300 épocas.

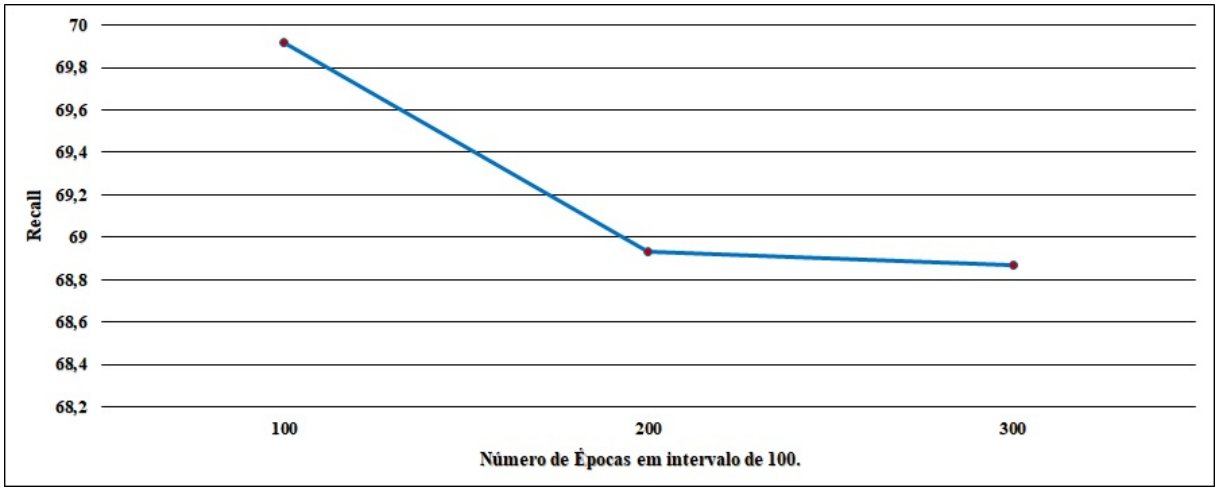
Nas Figuras 11, 12 e 13, são exibidos os resultados de *Precision*, *Recall* e *F1-Score*, capturados durante o treino. É importante ressaltar que esse modelo foi treinado com os mesmos hiper-parâmetros utilizados para a construção do modelo apresentado na Seção 5.2.1.

Figura 11 – Resultados de *Precision*, capturados durante o treino do modelo *Spacy* para 2 classes.



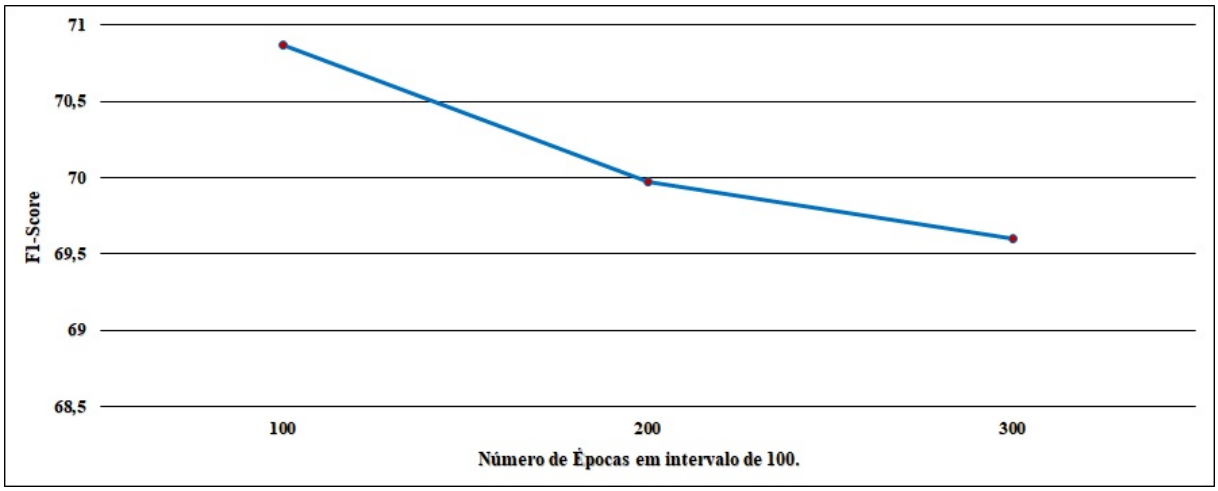
Fonte: Elaborado pelo autor.

Figura 12 – Resultados de *Recall*, capturados durante o treino do modelo *Spacy* para 2 classes.



Fonte: Elaborado pelo autor.

Figura 13 – Resultados de *F1-Score*, capturados durante o treino do modelo *Spacy* para 2 classes.



Fonte: Elaborado pelo autor.

Nos gráficos (Figuras 11, 12 e 13), é possível observar um desempenho que pode ser considerado mediano chegando ao final da execução com 71% de *precision*, 68% de *recall* e 69,2% de *F1-Score*. Entretanto, como solução para o problema exposto (identificar o nome da vítima e do assassino) esse modelo não é suficiente. Os modelos construídos com o *Spacy* possuem capacidade de rotulação mais genérica, nesse caso o modelo confunde-se entre o nome da vítima e do assassino, e acaba rotulando como se fossem uma única pessoa.

A Seção 5.2 apresenta dois modelos para classes não ambíguas, esses modelos possuem arquiteturas diferentes. Os resultados obtidos mostram um melhor desempenho do modelo construído com o *framework Spacy*. O modelo com o *Keras* não se sai tão bem para esse problema e conjunto de dados. É importante ressaltar que os modelos foram treinados com a mesma quantidade de épocas e mesmo conjunto de dados, mas alguns hiper-parâmetros de diferem pela peculiaridade da arquitetura de cada modelo.

Na Seção 5.3 são apresentados dois modelos NER para o problema de classes ambíguas. Com base nos resultados, o modelo construído com o *Keras* obteve um desempenho satisfatório. O modelo do *Spacy* teve um desempenho razoável, levando em consideração os valores de métricas obtidos. O *Spacy* trabalha com uma arquitetura diferente do *Keras*, essa arquitetura não consegue extrair todas as informações do contexto que a classe está, isso se traduz em um desempenho ruim do modelo para esse problema.

6 CONCLUSÕES E TRABALHOS FUTUROS

Este trabalho tem como propósito desenvolver uma ferramenta para facilitar o manuseio de modelos NER e integrar o usuário no ciclo de construção dos modelos. Também como objetivo, tem a criação de dois modelos NER para a extração de entidades nomeadas em BOs. Um modelo foi criado utilizando o *framework Spacy* e o outro utilizando a biblioteca *Keras*, cada um extrai informações diferentes, mas os dois resultados são unificados em apenas um resultado final.

Para o desenvolvimento da ferramenta Human NERD, foram necessárias leituras sobre outras ferramentas com os mesmos objetivos ou semelhantes, além de reuniões com técnicos da SSPDS para que fosse possível esclarecer o problema e propor uma solução viável. Em paralelo a isto acontecia a construção dos modelos NER, no qual foi necessário realizar observações sobre outros trabalhos que utilizaram técnicas semelhantes às utilizadas no presente trabalho.

Com base nos resultados apresentados no capítulo de resultados (Capítulo 5), a ferramenta Human NERD incorpora o *framework Spacy*, possibilitando a criação de modelos com o usuário no ciclo, realizando revisões no conjunto de treino, com o objetivo de refinar o modelo. Operações sobre os modelos como criar, remover, editar, atualizar, adicionar textos, fazer *upload* de um modelo e realizar a exportação de uma planilha com os resultados da previsão do modelo podem ser realizadas de forma intuitiva dentro do Human NERD.

Ainda nos resultados, é possível observar resultados satisfatórios no modelo NER utilizando o *Spacy*, atingindo 70% de *precision*. Esse modelo foi elaborado com 25 classes (PER, Homicídio (HOM), Latrocínio (LATROC), etc), treinado sobre um conjunto de textos sobre BOs. Apresentou bons resultados, levando em consideração que suas classes são genéricas, por exemplo a classe PER, é uma informação importante mas não é suficiente para identificar os envolvidos nos BOs. Como visto, o modelo não apresentou bons resultados para informações específicas como, nome da vitima e do assassino.

Também é possível observar o modelo construído utilizando o *Keras*, criado apenas com duas classes, vítima e assassino, apresenta seu melhor resultado com a rede neural *Char-BLSTM-CRF* chegando a 85% de *precision* para vítima e 65% para assassino. As redes *BRNN-CRF* e *BLSTM-CRF* obtiveram resultados inferiores.

Os resultados apresentados esclarecem a necessidade do uso de dois modelos com arquiteturas diferentes, para resolução do problema proposto sobre os dados de BOs.

Como trabalhos futuros pretende-se implementar no Human NERD o suporte para a biblioteca *Keras*. Outro trabalho seria melhorar o desempenho de ambos os modelos NER, adicionando variância de hiper-parâmetros e refinando o conjunto de treino.

REFERÊNCIAS

BEZERRA, E. Introdução à aprendizagem profunda. In: **SIMPÓSIO BRASILEIRO DE BANCO DE DADOS–SBBD**. Salvador: [s.n.], 2016. p. 1–30.

CHINCHOR, N. Muc-4 evaluation metrics. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. **Proceedings of the 4th conference on Message understanding**. [S.l.], 1992. p. 22–29.

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.

CHOLLET, F. **Deep learning with python.** [S.l.]: Manning Publications Co., 2017.

CHOLLET, F. **Deep Learning mit Python und Keras: Das Praxis-Handbuch vom Entwickler der Keras-Bibliothek**. [S.l.]: MITP-Verlags GmbH & Co. KG, 2018.

COLLOBERT, R.; WESTON, J.; BOTTOU, L.; KARLEN, M.; KAVUKCUOGLU, K.; KUKSA, P. Natural language processing (almost) from scratch. **Journal of machine learning research**, v. 12, n. Aug, p. 2493–2537, 2011.

DAUPHIN, Y.; VRIES, H. D.; BENGIO, Y. Equilibrated adaptive learning rates for non-convex optimization. In: **Advances in neural information processing systems**. [S.l.: s.n.], 2015. p. 1504–1512.

DENG, L.; YU, D. *et al.* Deep learning: methods and applications. **Foundations and Trends® in Signal Processing**, Now Publishers, Inc., v. 7, n. 3–4, p. 197–387, 2014.

FOLEY, J.; SARWAR, S. M.; ALLAN, J. Named entity recognition with extremely limited data. **arXiv preprint arXiv:1806.04411**, 2018.

GOUTTE, C.; GAUSSIER, E. A probabilistic interpretation of precision, recall and f-score, with implication for evaluation. In: **EUROPEAN CONFERENCE ON INFORMATION RETRIEVAL**. [S.l.: s.n.], 2005. p. 345–359.

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

KONKOL, M.; BRYCHCÍN, T.; KONOPÍK, M. Latent semantics in named entity recognition. **Expert Systems with Applications**, Elsevier, v. 42, n. 7, p. 3470–3479, 2015.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. nature 521. 2015.

LIU, X.; ZHOU, Y.; WANG, Z. Recognition and extraction of named entities in online medical diagnosis data based on a deep neural network. **Journal of Visual Communication and Image Representation**, Elsevier, 2019.

MA, X.; XIA, F. Unsupervised dependency parsing with transferring distribution via parallel guidance and entropy regularization. In: **PROCEEDINGS OF THE 52ND ANNUAL MEETING OF THE ASSOCIATION FOR COMPUTATIONAL LINGUISTICS (Volume 1: Long Papers)**. [S.l.: s.n.], 2014. v. 1, p. 1337–1348.

MOTA, C.; SANTOS, D.; RANCHHOD, E. Avaliação de reconhecimento de entidades mencionadas: princípio de arem. **Avaliação conjunta: um novo paradigma no processamento computacional da língua portuguesa**, p. 161–175, 2007.

NADEAU, D. **Semi-supervised named entity recognition**: learning to recognize 100 entity types with little supervision. Tese (Doutorado) — University of Ottawa, 2007.

OLSON, D. L.; DELEN, D. **Advanced data mining techniques**. [S.l.]: Springer Science & Business Media, 2008.

POWERS, D. M. **Evaluation**: From precision, recall and f-factor to roc, informedness, markedness & correlation, school of informatics and engineering, flinders university, Adelaide, Australia. 2007.

SAMMUT, C.; WEBB, G. I. **Encyclopedia of machine learning**. [S.l.]: Springer Science & Business Media, 2011.

SILVA, T. L. C. da; ARAUJO, N. da S.; MACEDO, J. A. F. de; MAGALHAES, R. P.; ABREU, D. A.; MELO, V. T. de; PINHEIRO, P. O.; REGO, P. A. L.; NETO, A. V. L. Improving named entity recognition using deep learning with human in the loop. **EDBT/ICDT 2019 JOINT CONFERENCE**, 2019.

SILVA, T. L. C. da; ARAUJO, N. da S.; MACEDO, J. A. F. de; ARAUJO, D.; SOARES, F. M.; REGO, P. A. L.; NETO, A. V. L. Novel approach for label disambiguation via deep learning. **INTERNATIONAL CONFERENCE ON MACHINE LEARNING AND DATA MINING**, 2019.

SPECK, R.; NGOMO, A.-C. N. Ensemble learning for named entity recognition. In: **INTERNATIONAL SEMANTIC WEB CONFERENCE**. [S.l.: s.n.], 2014. p. 519–534.

YU, F.; SEFF, A.; ZHANG, Y.; SONG, S.; FUNKHOUSER, T.; XIAO, J. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. **arXiv preprint arXiv:1506.03365**, 2015.