PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO GRANDE DO SUL
ESCOLA POLITÉCNICA
BACHARELADO EM SISTEMAS DE INFORMAÇÃO

RECONHECIMENTO DE ENTIDADES NOMEADAS NO DOMÍNIO DE BULAS DE DEFENSIVOS AGRÍCOLAS

BRUNO WOLFF ABBAD

Trabalho de Conclusão II apresentado como requisito parcial à obtenção do grau de Bacharel em Sistemas de Informação na Pontifícia Universidade Católica do Rio Grande do Sul.

Orientador: Prof. Dra. Sílvia Maria Wanderley Moraes

Porto Alegre
2023

RECONHECIMENTO DE ENTIDADES NOMEADAS NO DOMÍNIO DE BULAS DE DEFENSIVOS AGRÍCOLAS

RESUMO

A agronomia é um setor de grande impacto na economia brasileira. Em razão disso, as empresas deste setor procuram aumentar o número de sacas por hectare produzidos pelos fazendeiros. Visando este aumento, mais importante se torna a aplicação correta e adequadamente dosada de defensivos agrícolas. Para isso, e pensando sempre do uso das informações mais atuais, é imprescindivel revisitar frequentemente a bula desses defensivos a fim de identificar orientações mais recentes. Frente a esse cenário, foi realizado um estudo para que através de técnicas de processamento de linguagem natural, pudesse ser possível reconhecer as entidades nomeadas no texto de uma bula e sinalizá-las para facilitar a extração de informações relevantes. Com esse fim, foi construído um corpus textual de bulas, realizada uma anotação manual de entidades e implementado um classificador neural para entidades nomeadas. Os resultados foram animadores.

Palavras-Chave: agronomia, defensivos agrícolas, bulas, reconhecimento de entidades nomeadas, processamento de linguagem natural.

NAMED ENTITY RECOGNITION IN THE PESTICIDES PACKAGE INSERTS DOMAIN

ABSTRACT

Agronomy is a sector with a significant impact on the Brazilian economy. As a result, companies in this sector seek to increase the number of bags per hectare produced by farmers. Aiming at this increase, the correct and appropriately dosed application of agricultural pesticides becomes more critical. For this, and always thinking about using the most current information, it is essential to frequently revisit the leaflet of these pesticides to identify more recent guidelines. To recognize the entities named in the text of a pesticide leaflet and signal them to facilitate the extraction of relevant information study was carried out to face this agronomy scenario so that, through natural language processing techniques, it could be possible. To this end, a textual corpus of package inserts was constructed, the manual annotation of entities was performed, and was implemented a neural classifier for named entities. The results were encouraging.

Keywords: agronomy, pesticides, package inserts, named entity recognition, natural language processing.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

PLN – Processamento de Linguagem Natural

POS – Part of Speech

REN – Reconhecimento de Entidade Nomeada

NER – Named Entity Recognition

IOB – inside-outside-beggining

BOW – Bag-of-words

TF – Term Frequency

TF-IDF – Term Frequency-Inverse Document Frequency

CBOW – Continuous Bag of Words

ELMO – Embedding From Language Models

GPT – Generative Pre-Training

BERT – Bidirectional Encoder Representations from Transformers

LSTM – Long Short-Term Memory

CRF – Conditional Random Field

SGD – Stochastic Gradient Descent

SUMÁRIO

1. INTRODUÇÃO

No Brasil, a agricultura é um dos principais pilares da economia do país desde os primórdios de sua colonização até a atualidade [28]. Com diversas culturas de plantio no vasto território brasileiro, existem diferentes tipos de doenças que podem ser transmitidas para as plantações e pela sua importância econômica, os produtores empregam diversas técnicas para evitar que os seus cultivos sejam atingidos, dentre elas a utilização de produtos que combatem as doenças.

Para que um produto possa ser utilizado e comercializado para os produtores, ele precisa ser registrado pelo Ministério da Agricultura atendendo às diversas especificações e regulamentações impostas. Além disso é também necessário que os produtos disponibilizem bulas indicando com transparência as formulações e as doenças que o laboratório indica como eficaz ao combate, quantas aplicações máximas devem ser realizadas, a dosagem por aplicação, efeitos colaterais contra as plantas e contra os humanos que consomem os produtos, entre outros.

A DigiFarmz é uma plataforma[1] que auxilia aos produtores e consultores no manejo fitossanitário, economizar recursos e investimentos com práticas de campo otimizadas, maximizando o rendimento das colheitas enquanto reduz o impacto ambiental através de pesquisas e modelos estatísticos que indicam ao produtor os momentos mais indicados para aplicação e dosagens conforme o produto selecionado. Para isso, são necessárias as informações constantes nas bulas dos produtos. A plataforma é dividida em dois momentos: planejamento e safra. Na parte de safra o produtor coloca as informações depois que ele realizar o plantio, para ter as datas recomendadas de aplicações, mas antes disso ele precisa se preparar e realizar o planejamento de quais produtos irá utilizar a partir das eficácias apresentadas pela performance de cada fungicida para a realidade dele.

São encontrados, porém, dois problemas: os laboratórios que produzem os defensivos fazem pesquisas com seus produtos, podendo adicionar ou retirar doenças em que são eficazes, alterar valores de aplicações máximas ou quantidades das dosagens recomendadas, sendo descobertas tais modificações apenas acessando a página do Ministério da Agricultura e baixando a bula do produto para conferir se houve ou não alterações. Já o outro problema identificado figura a partir de relatos dos membros do time de pesquisa e desenvolvimento, que afirmam que o site em que são encontradas as informações dos produtos, nem sempre acompanham as atualizações em um tempo real, dificultando assim as pesquisas realizadas com aplicação de filtros no site.

Dado ao número de produtos registrados se torna inviável aferir uma validação de forma totalmente manual das alterações nos registros das bulas que podem ser atua-

[1] A plataforma DigiFarmz, de mesmo nome da empresa, da qual o autor participava como cientista de dados.

lizadas inúmeras vezes ao longo do período. São verificadas anualmente todas as bulas de forma manual pelo time de pesquisa e desenvolvimento se tornando um problema pois caso haja uma alteração na bula no dia seguinte ao que foi validado pelo time, apenas após um ano ela será revista e terá alterados os dados que serão atualizados na plataforma. Através deste problema, o presente trabalho tem como objetivo geral: Sugerir um processo de anotação nas tabelas encontradas nas bulas, utilizando técnicas de reconhecimento de entidades nomeadas e aprendizado de máquina, para assim diminuir o tempo necessário de pesquisa pelo time interno e a periodicidade das revisões. Este documento está organizado em 9 capítulos. No capitulo 1, é feita uma introdução a este trabalho. No capitulo 2, é descrito com mais detalhes o problema abordado no trabalho. No capitulo 3, é introduzida a área de processamento de linguagem natural e a subárea do reconhecimento de entidades nomeadas. No capitulo 4, são descritas técnicas envolvendo processamento de linguagem natural e trabalhos usados como referência. No capitulo 5, é descrito como o corpus foi montado e a arquitetura do modelo. No capítulo 6, as ferramentas testadas para anotação e o processo para anotar são descritos. No capítulo 7, é descrito cada um dos *datasets* montados. No capítulo 8, são apresentados os resultados do modelo treinado com diferentes versões do AGROBULA-NER. No capítulo 9, são descritas as formas de continuidade para o trabalho realizado.

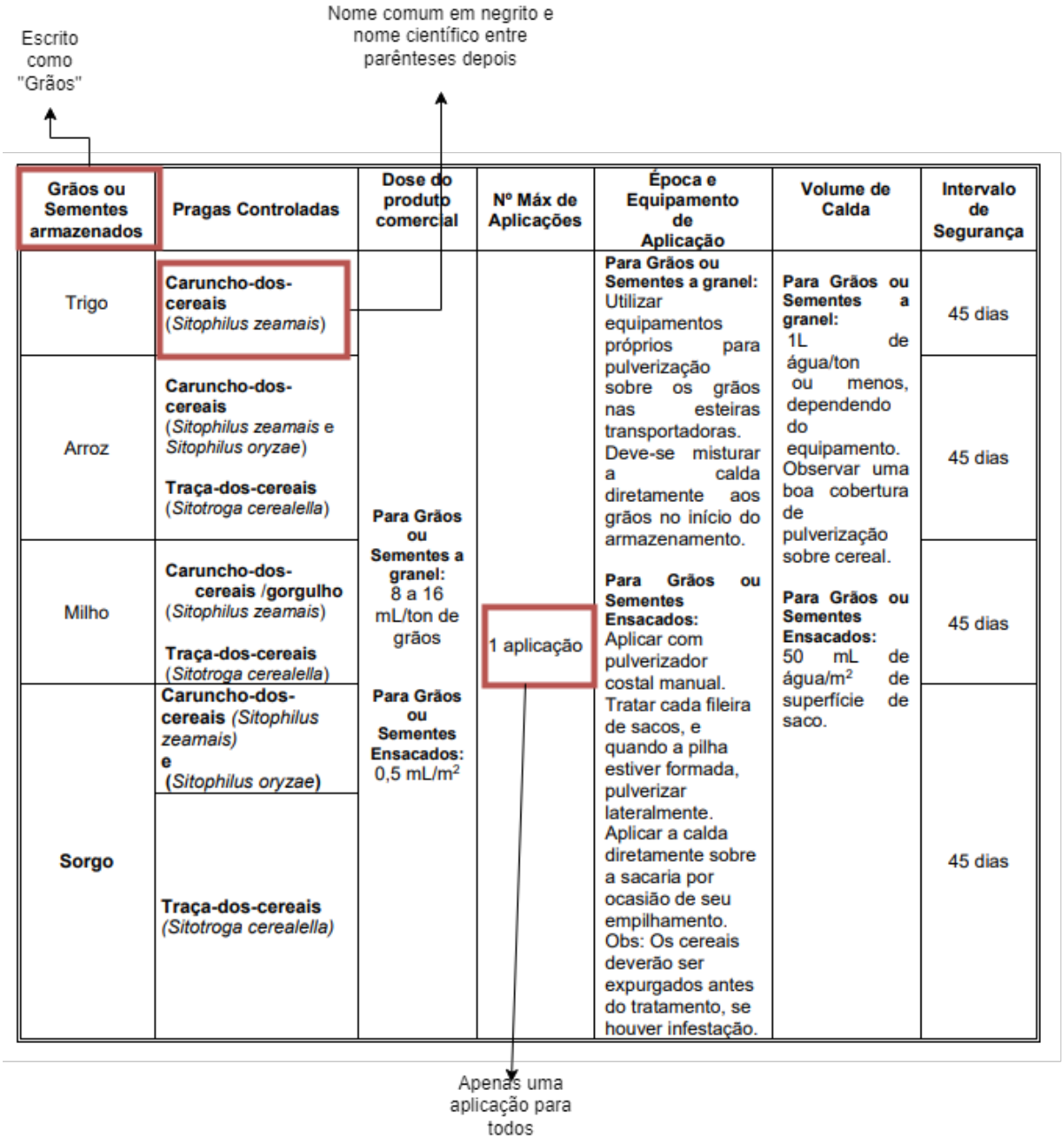
2. PROBLEMA

Como mencionado previamente, na atualidade, para poder comercializar um defensivo agrícola, ele deve ser registrado no Ministério da Agricultura junto de sua bula e diversas outras informações. No entanto, as bulas podem ser alteradas conforme pesquisas dos laboratórios vão sendo elaboradas, podendo ter suas doses, culturas e doenças ajustadas, removidas ou adicionadas. Além disso, quando um produto é modificado, sua página no site do Ministério da Agricultura, não necessariamente acompanha as atualizações, podendo demorar para que esteja devidamente atualizado.

Um dos desafios dos time de pesquisa de empresas no setor do agro, é conseguir conferir manualmente todas as bulas dos produtos comercializados no Brasil em busca de alterações. Justamente por não saber quando foi alterado algum produto, é necessária uma constante validação dessas informações. Dado ao grande número de produtos, é inviável que tal conferência seja feita frequentemente. Há modificações que impactam diretamente a produção. Segundo os próprios fabricantes dos produtos, quando as novas orientações não são seguidas perde-se na eficácia dos produtos, pois os ajustes podem ser de dosagem, alterando os valores anteriormente estabelecidos nas bulas, causando, assim, imprecisões caso não sejam utilizados os dados mais recentes. Outro problema se dá no momento de pesquisa dos produtos no portal do Ministério da Agricultura, em que não é possível pesquisar os produtos por doenças. Essa limitação obriga a realização de uma busca exaustiva entre os diversos produtos que se encontram por cultura.

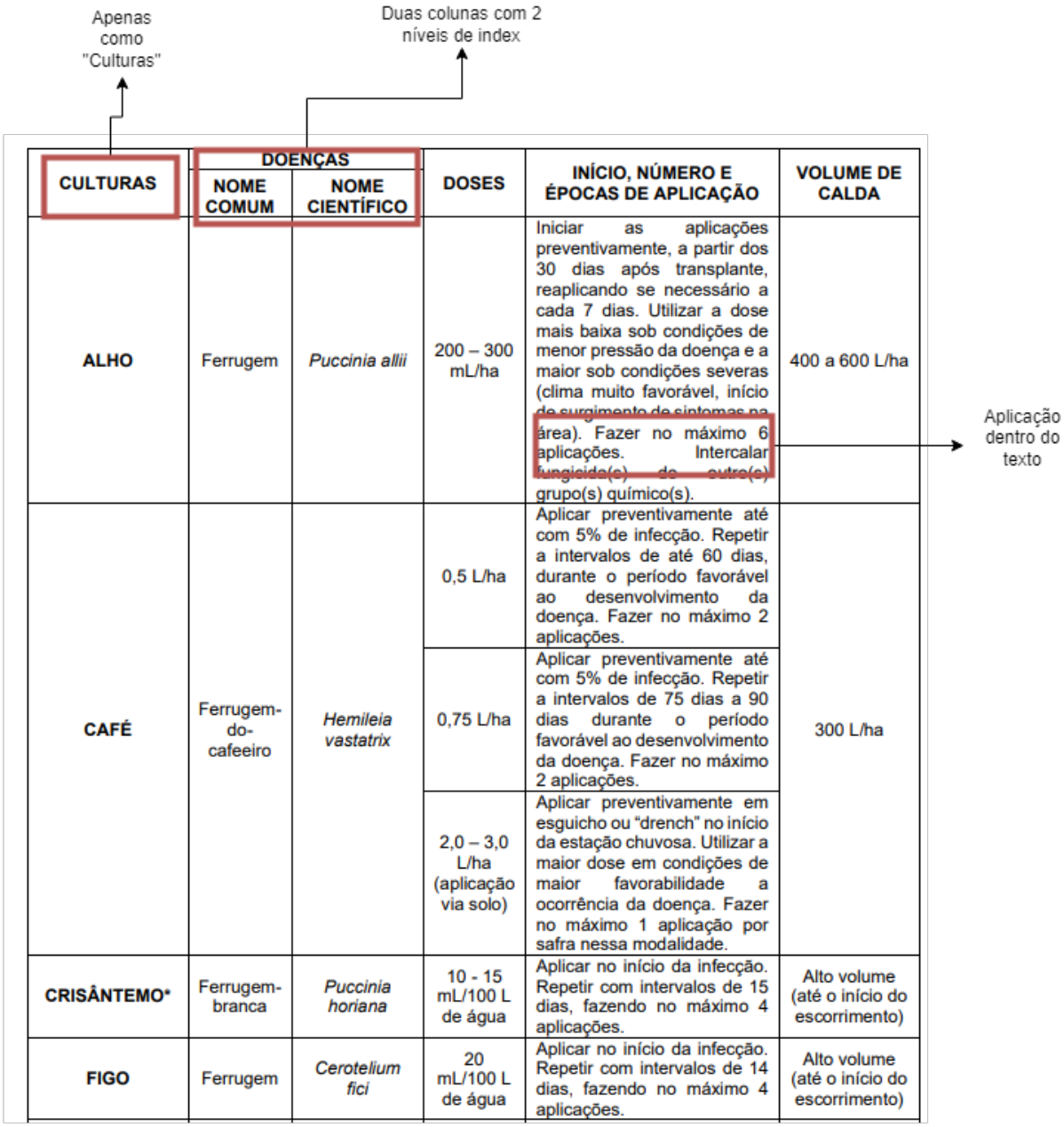
De forma um geral, dados importantes nas bulas são encontrados dentro de tabelas. Cada laboratório define o próprio formato dessas tabelas e das suas bulas, as quais são disponibilizadas em pdf, por padrão. Dito isso, a extração dos dados com bibliotecas de extração comuns, como por exemplo o Tabula ou o Camelot, ambos disponíveis para a linguagem de programação Python, mostraram-se inadequadas. Apesar as bulas de laboratórios diferentes terem layout diferentes, há um certo padrão na estrutura e no conteúdo, elevando o tipo de texto para algo quase semiestruturado. Pensando-se nisso ambas bibliotecas citadas foram testadas, mas os resultados não foram animadores. Não se tornou viável a implementação de um algoritmo genérico capaz de extrair por si só grande parte das informações em tabelas. É importante ressaltar que até mesmo o mesmo laboratório apresentou bulas com layouts diferentes, especialmente para as tabelas, como exemplo na figura 2.1. Na figura, a primeira coluna está utilizando a nomenclatura de "Grãos ou Sementes armazenados", porém normalmente é encontrada nas bulas como "Culturas". Além disso, a segunda coluna está unificando as doenças, que muitas vezes possuem duas colunas, uma com o nome comum da doença e uma segunda com o científico. Outra diferença está no número máximo de aplicações. Foi definido de forma geral para todas as culturas ao invés de separado por cultura. Já na 2.2, a primeira coluna está utilizando o nome usual

da coluna, porém na aba de doenças, utiliza a forma de coluna com dois níveis. Neste caso, as aplicações estão escritas dentro de uma coluna com um texto mais elaborado. Essas são algumas diferenças relatadas em duas bulas do mesmo laboratório mas de diferentes produtos, que apresentam diferentes formatos. Porém essa falta de consistência de formatação única está presente entre todos os produtos de diferentes laboratórios.



Grãos ou Sementes armazenados	Pragas Controladas	Dose do produto comercial	Nº Máx de Aplicações	Época e Equipamento de Aplicação	Volume de Calda	Intervalo de Segurança
Trigo	**Caruncho-dos-cereais** (*Sitophilus zeamais*)	**Para Grãos ou Sementes a granel:** 8 a 16 mL/ton de grãos **Para Grãos ou Sementes Ensacados:** 0,5 mL/m²	1 aplicação	**Para Grãos ou Sementes a granel:** Utilizar equipamentos próprios para pulverização sobre os grãos nas esteiras transportadoras. Deve-se misturar a calda diretamente aos grãos no início do armazenamento. **Para Grãos ou Sementes Ensacados:** Aplicar com pulverizador costal manual. Tratar cada fileira de sacos, e quando a pilha estiver formada, pulverizar lateralmente. Aplicar a calda diretamente sobre a sacaria por ocasião de seu empilhamento. Obs: Os cereais deverão ser expurgados antes do tratamento, se houver infestação.	**Para Grãos ou Sementes a granel:** 1L de água/ton ou menos, dependendo do equipamento. Observar uma boa cobertura de pulverização sobre cereal. **Para Grãos ou Sementes Ensacados:** 50 mL de água/m² de superfície de saco.	45 dias
Arroz	**Caruncho-dos-cereais** (*Sitophilus zeamais* e *Sitophilus oryzae*) **Traça-dos-cereais** (*Sitotroga cerealella*)					45 dias
Milho	**Caruncho-dos-cereais /gorgulho** (*Sitophilus zeamais*) **Traça-dos-cereais** (*Sitotroga cerealella*)					45 dias
Sorgo	**Caruncho-dos-cereais** (*Sitophilus zeamais*) **e** (*Sitophilus oryzae*) **Traça-dos-cereais** (*Sitotroga cerealella*)					45 dias

Figura 2.1 – Tabela do produto Actellic 500 EC [1] do laboratório Syngenta



CULTURAS	DOENÇAS		DOSES	INÍCIO, NÚMERO E ÉPOCAS DE APLICAÇÃO	VOLUME DE CALDA
	NOME COMUM	NOME CIENTÍFICO			
ALHO	Ferrugem	*Puccinia allii*	200 – 300 mL/ha	Iniciar as aplicações preventivamente, a partir dos 30 dias após transplante, reaplicando se necessário a cada 7 dias. Utilizar a dose mais baixa sob condições de menor pressão da doença e a maior sob condições severas (clima muito favorável, início de surgimento de sintomas na área). Fazer no máximo 6 aplicações. Intercalar fungicida(s) de outro(s) grupo(s) químico(s).	400 a 600 L/ha
CAFÉ	Ferrugem-do-cafeeiro	*Hemileia vastatrix*	0,5 L/ha	Aplicar preventivamente até com 5% de infecção. Repetir a intervalos de até 60 dias, durante o período favorável ao desenvolvimento da doença. Fazer no máximo 2 aplicações.	300 L/ha
			0,75 L/ha	Aplicar preventivamente até com 5% de infecção. Repetir a intervalos de 75 dias a 90 dias durante o período favorável ao desenvolvimento da doença. Fazer no máximo 2 aplicações.	
			2,0 – 3,0 L/ha (aplicação via solo)	Aplicar preventivamente em esguicho ou "drench" no início da estação chuvosa. Utilizar a maior dose em condições de maior favorabilidade a ocorrência da doença. Fazer no máximo 1 aplicação por safra nessa modalidade.	
CRISÂNTEMO*	Ferrugem-branca	*Puccinia horiana*	10 - 15 mL/100 L de água	Aplicar no início da infecção. Repetir com intervalos de 15 dias, fazendo no máximo 4 aplicações.	Alto volume (até o início do escorrimento)
FIGO	Ferrugem	*Cerotelium fici*	20 mL/100 L de água	Aplicar no início da infecção. Repetir com intervalos de 14 dias, fazendo no máximo 4 aplicações.	Alto volume (até o início do escorrimento)

Figura 2.2 – Tabela do produto Alto 100 [2] do laboratório Syngenta

3. INTRODUÇÃO AO PROCESSAMENTO DE LINGUAGEM NATURAL

Dentro da ciência da computação, existe uma subárea que é compartilhada entre as áreas da inteligência artificial e da linguística que se chama Processamento de Linguagem Natural (PLN), que faz o estudo da forma de entendimento da linguagem humana natural de forma automatizada, sejam os dados em textos, sons ou imagens.

Segundo o Eisenstein (2019)[6] PLN é focado no desenvolvimento e análise de algoritmos e representações para o processamento da linguagem humana natural, tendo como objetivo gerar novas capacidade computacionais em torno da linguagem humana, dando como exemplos extração de informações de textos, traduções entre linguagens, manter conversações etc.

3.1 Desafios do PLN

Diferentemente de nós humanos, que cometemos erros gramaticais e lógicos em nossa comunicação e mesmo assim conseguimos nos compreender, os computadores precisam de estruturas perfeitas e numéricas, assim, o desafio desafio inicial do PLN será transformar a nossa linguagem em uma estrutura numérica em que o computador seja capaz de compreender.

Segundo o autor Indurkhya et al. (2010, pp.10-11)[7] quando se faz um sistema PLN, muitas das dificuldades podem ser abordadas como parte da triagem de documentos na preparação de um corpus para análise. O tipo de sistemas de escrita da linguagem é o fator mais importante quando se vai determinar a melhor abordagem para o pré-processamento do texto. Os sistemas de escrita podem ser:

- Logográficos: quando um grande número de símbolos individuais representam palavras.
- Silábicos: quando símbolos individuais representam sílabas.
- Alfabéticos: São sistemas em que símbolos individuais representam sons.

3.2 Pré-processamento

Para utilizar a linguagem natural, deve se fazer uma sequência de passos para pré-processar os dados, segundo os autores Naseem et al. (2021)[19] os *datasets* de texto

contém diversas palavras que não queremos, como as *stop-words*, pontuações, escritas incorretas, gírias, etc. Esses ruídos e palavras podem ter efeitos negativos na performance de modelos. Passando por formas de pré-processamento de texto, temos as seguintes etapas citadas pelos autores:

- Tokenização: O processo inicial usual do pré-processamento quando se fala em tarefas de NLP, em que frases são convertidas em *tokens*, em que uma sequência do texto é dividida em palavras, símbolos, frases ou próprios *tokens*, tendo seu objetivo na tokenização, descobrir as palavras em uma sentença.
- Remoção de Ruídos: Durante a extração de dados, algumas vezes, devido a natureza da tarefa, alguns caracteres indesejados acabam vindo junto no texto, e por isso, os mesmos são removidos.
- Substituição de Abreviações e Gírias: é comum vermos em textos informais, gírias ou abreviações, portanto é crucial que essas palavras, sejam substituídas pelos seus significados em palavras comuns, sendo a técnica comum a conversão direta das palavras.
- Correção de Erros Ortográficos: Outra peculiaridade comum quando se fala da linguagem humana, é que encontramos erros de ortografia, de forma mais comum dependendo da formalidade do texto, tendo sua correção afetando o número de vezes que a palavra é escrita em um texto.
- Expansão de Contrações: Para textos em linguagens como a inglesa, as contrações são mais facilmente encontradas, por isso, para padronizar as palavras, são removidas as contrações e expandidas as palavras reduzidas para sua raiz natural.
- Remoção de Pontuação: Em mídias sociais por exemplo, pessoas usam pontuações para expressar sentimentos e *emoticons*, eles dificultam na classificação de textos, então a remoção de pontuações é uma prática comum, mas mantendo alguns como interrogações e exclamações, tendo suas próprias *tags* como citado pela Balahur (2013 pp.120-128)[3]
- Remoção de Números: Outra remoção feita são as dos números que também podem diminuir a precisão dos resultados pelo computador. Sendo como padrão remover os números apenas, mas também deve se atentar já que é possível perder informações úteis como gírias ou abreviações que utilizam números.
- Transformar em Caixa Baixa: Transformar todas as palavras em caixa baixa ajuda a evitar diferentes cópias da mesma palavra, podendo diminuir a performance do computador. Essa técnica, tem suas ressalvas quanto é importante manter por exemplo a sigla de *United States* "US"e o pronome inglês "us", por isso não são em todos os casos que se deve realizar esta técnica.

- Removendo *Stop-words*: Existem diversas palavras que não possuem um significado relevante e estão presentes em grande frequência em um texto. Ou seja, palavras que não ajudam a melhorar a performance das tarefas e não carregam muitas informações, por isso é recomendado normalmente remover as *stop words* antes de selecionar features. Sendo sugerido pelos autores algumas bibliotecas para remover *stop-words* como NLTK, scikit-learn e spaCy.

- *Stemming*: Quando uma palavra tem o mesmo valor semântico mas de diferentes formas, a técnica de stemming remove sufixos e afixos para obter a raiz da palavra ou a base da palavra. A importância do *stemming* foi estudada por Mejova e Srinivasan (2011, pp.546-549) [15].

- Lematização: O propósito da lematização é o mesmo do stemming, que seria cortar palavras para a sua base ou raiz, porém na lematização, flexões de palavras não são cortadas, se usa conhecimento léxico para transformar as palavras em usas formas base.

- Part of Speech (POS): o POS, serve para dar a classificação gramatical das palavras, agrupando as palavras com mesmo valor gramatical juntas.

- Lidando com Negações: Na análise de textos , às vezes a negação da palavra pode não ajudar o computador em sua compreensão, por isso algumas técnicas como adicionar o prefixo ‘NEG_‘ já foram estudadas, ou outra forma similar é a utilização de antônimos.

3.3 O que é Reconhecimento de Entidades Nomeadas (NER)

O PLN possui suas próprias subáreas, sendo o enfoque deste trabalho a subárea do Reconhecimento de Entidade Nomeada (REN) ou também conhecido como Named Entity Recognition (NER). Segundo Jurafsky e Martin(2020)[9], uma entidade nomeada é, de forma resumida, tudo que pode ser referenciado por um nome próprio, dando exemplos como pessoas, localizações, organizações, etc. como mostra na figura 3.1. E a tarefa de reconhecer essas entidades nomeadas tem como finalidade encontrar no texto onde estão localizados esses nomes próprios e marcar os seus tipos de entidade. O livro também explica que essas marcações de texto das entidades nomeadas é um primeiro passo importante em diversas outras tarefas de PLN, como por exemplo, a compreensão do sentimento de um cliente, que para isso primeiramente é preciso entender onde está sendo falado do cliente e sobre o que, ou quem o cliente está falando. São citadas também algumas das dificuldades causadas principalmente por ambiguidades como por exemplo "JFK", sigla para

o nome "John F. Kennedy", que dado o contexto pode ser o antigo presidente dos Estados Unidos, o aeroporto de Nova Iorque, ou alguma das diversas escolas, ruas e pontes espalhadas dentro do país.

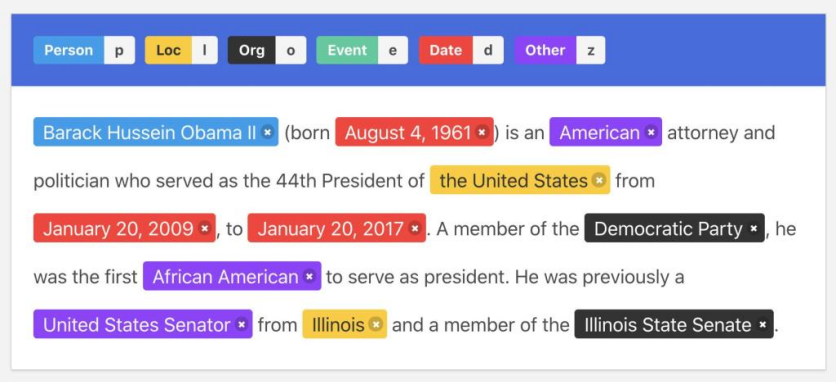


Figura 3.1 – Tabela retirada do site npcloud.io [23]

Recentemente, notou-se que as entidades não se aplicam mais apenas a nomes próprios mas a uma variedade de elementos, cuja identificação das instâncias são importantes para a aplicação. O conceito de entidade foi ampliado e passou a atender a especificidades do domínio. Identificação de números é um desses casos. É comum a anotação de número de identificação como CPF, RG, cartão de crédito, conta corrente ou mesmo valores em moeda.

3.3.1 Exemplos de NER

Para compreensão maior de onde é utilizado o NER, seguem alguns exemplos em que são utilizadas técnicas de NER: Há sistemas de tradução automática nos quais é possível pesquisar frases citando cidades e as mesmas se mantém sem ser traduzidas, evitando diversos transtornos, como trazer uma frase contendo a cidade brasileira "Rio de Janeiro"mantendo seu nome ao invés de em uma tradução para o inglês ser transformada em "River from January", ou temos também exemplos como em *chatbots* para suporte ao cliente em que precisa compreender termos muitas vezes com entidades categorizadas de forma customizada para necessidade do negócio e ter capacidade de atender ao cliente de forma acurada, a sumarização de textos através da extração de informações facilitadas pelo NER, a própria extração de informações por si só que também nos leva a compreensão de textos, entre outras diversas outras aplicações que utilizam ou podem se aproveitar da utilização do NER.

3.3.2 Formato de Anotação NER

Para fazer as anotações nos textos, segundo os autores [9], são usualmente utilizadas três formatos:

- inside-outside-beggining: a forma padrão é utilizando a técnica de formato IOB ou BIO, que foi proposta por Ramshaw and Marcus (1999 pp.157-176) [22], onde é extraído um *chunk* que nada mais é que a extração de frases para compreensão de contexto, a técnica de IOB em que palavras marcadas com o I estão dentro de um *chunk*, palavras marcadas com O estão fora e palavras marcadas com B são o começo de um *chunk*, sendo pontuações marcadas como se fossem palavras, essa técnica de IOB, nos permite compreender o tipo de entidade nomeada e suas ligações.

  Devido a ser a forma padrão, foi escolhido o formato IOB para ser utilizado ao longo do trabalho.

- START/END: Citada no por Vijay Krishnan and Vignesh Ganapathy (2005)[11] tambpém conhecida como IOBES/BIOES, em que o E representa palavras no fim de um *chunk* e a letra S representa palavras de um *chunk* único.

- IO: Abordagem também citada no por Vijay Krishnan and Vignesh Ganapathy (2005)[11] em que existem duas categorias apenas, portanto, não seria capaz de diferenciar entre *chunks* adjacentes da mesma entidade nomeada.

Na figura 3.2 a mesma frase é demonstrada com as três técnicas.

Words	IO Label	BIO Label	BIOES Label
Jane	I-PER	B-PER	B-PER
Villanueva	I-PER	I-PER	E-PER
of	O	O	O
United	I-ORG	B-ORG	B-ORG
Airlines	I-ORG	I-ORG	I-ORG
Holding	I-ORG	I-ORG	E-ORG
discussed	O	O	O
the	O	O	O
Chicago	I-LOC	B-LOC	S-LOC
route	O	O	O
.	O	O	O

Figura 3.2 – Tabela retirada do livro Speech and language processing : an introduction to natural language processing, computational linguistics, and speech recognition [9]

4. TÉCNICAS

Para a extração de *features* e classificadores, são utilizados diversos modelos de linguagens que vêm sendo pesquisados e desenvolvidos por diversos pesquisadores para entender relações sintáticas e semânticas entre palavras segundo Naseem et al. (2021, pp.1-35)[19] que explicam os modelos clássicos e outros modelos de aprendizado de representação famosos.

4.1 Modelos Clássicos

Nesta seção, alguns dos modelos utilizados mais frequentemente no passado para classificação de textos, sendo a frequência das palavras a base dessas formas de representação de palavras. Fazendo de suas formas, o texto é representado em vetores contendo os números de vezes que as palavras aparecem em um documento. Tendo os métodos de *categorical word representation* e a de *weightedweighted word representation*.

4.1.1 Categorical Word Representation

Conhecido por ser a forma mais simples de representação de texto, nesse método as palavras são representadas por símbolos de “1“ ou “0“. Sendo os modelos de *One-hot-encoding* e *Bag-of-words* (BoW) sendo os dois modelos mais utilizados.

- One-hot-encoding: A forma mais direta de representação textual, tendo o número de dimensões igual ao número de palavras encontradas no texto, sendo representado por “1“ o índice correspondente da palavra e “0“ as outras colunas, tendo todas as palavras únicas, uma unica dimensao que será representada por 1.
- BoW: Conhecido por ser uma extensão do *One-hot-encoding*, adicionando a representação do *One-hot-encoding* em uma frase. A matriz gerada usando o BoW, ignora o relacionamento semântico entre palavras, assim como a ordem das palavras.

Podemos encontrar no livro a figura 4.1 que demonstra como a frase ‘Hello World‘ ficaria utilizando respectivamente o One-hot-encoding e o BoW.

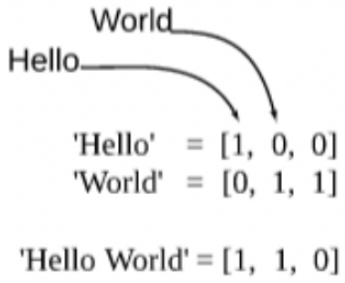


Figura 4.1 – Representacao retirada do artigo A Comprehensive Survey on Word Representation Models: From Classical to State-Of-The-Art Word Representation Language Models [19]

4.1.2 Weighted Word Representation

Falando da representação de *Weighted word*, normalmente são dois métodos que se pensam, o Term Frequency (TF) e o Term Frequency-Inverse Document Frequency (TF-IDF). Ambos são associados aos métodos de representação categóricos, porém são mais sofisticados que apenas contadores, são baseados na frequência de palavras, sendo ambos relatados pelos autores [19] da seguinte forma:

- Term Frequency: Método mais direto de extração de *features*; Esse método calcula o número de vezes que uma palavra ocorre em um documento dividido pelo tamanho do documento.
- Term Frequency-Inverse Document Frequency: O TF-IDF foi apresentado por Jones (1988 pp.132-142)[26] para representação textual, cortando o impacto causado de palavras comuns dentro do corpus. Diferentemente do método TF, o TF-IDF denota o inverso da frequência no documento, técnica para reduzir o efeito de palavras comuns , dando mais pesos a palavras com maior ou menor frequência.

4.2 Representações por Aprendizado

Devido a falha em capturar o significado sintático e semânticos das palavras nos modelos de representação categóricos, e esses modelos sofrem da maldição da dimensionalidade, fez com que pesquisadores aprendessem os modelos de representação distribuída para diminuir o espaço dimensional, segundo Bolukbasi et al. (2016) [4] alguns dos modelos citados por Naseem et al. (2021)[19] são os contínuos que são sem contexto e os modelos contextuais:

4.2.1 Modelos Contínuos

Técnica em que o texto no corpus é mapeado em vetores, permitindo que palavras com o mesmo significado, tenham a mesma representação. Tendo como maior benefício na técnica, também conhecida como *Word Embedding*, maior eficiência e expressividade na representação por manter a similaridade das palavras por contexto e por sua dimensionalidade dos vetores baixa.

Dentro da área dos modelos contínuos, temos os dois modelos mais famosos: O Word2Vec publicado por Mikolov et al. (2013)[18] e o GloVe publicado por Manning et al. (2014, pp.55-60)[14].

- Word2Vec: O modelo Word2Vec usa duas camadas escondidas que são usadas em uma rede neural para criar um vetor de cada palavra. Os vetores capturados pelo Continuous Bag of Words (CBOW) e por modelos Skip-gram do Word2Vec, que são detalhados pelos autores Naseem et al. [19] conseguem ter a informação semântica e sintática das palavras. E para ter melhor representação das palavras, se é recomendado treinar com um corpus grande. O Word2Vec se provou util em diversas atividades de NLP. Esse modelo foi criado para criar *embeddings* de treinados de forma mais significante, e desde então vem sido usado o padrão para desenvolvimento de representações pré treinadas de palavras. O Word2Vec prevê baseado no contexto utilizando um dos dois modelos de rede neural, tendo o tamanho da janela predefinido movido junto com o corpus em ambos os modelos, e o treinamento é feito com as palavras dentro dessa janela em cada passo. Esse algoritmo é uma ferramenta robusta para descobrir relacionamentos no corpus e as similaridades entre *tokens*.

  - Continuous Bag of Words: O CBOW prevê as palavras atuais baseado no contexto, comunicando com as palavras vizinhas dentro da janela de texto. Três camadas são usadas no processo do CBOW. O contexto é considerado como a primeira camada, onde combina com a estimativa de todas as palavras do *input* para a matriz com os pesos, que depois é estimada para relacionar com o *output* e a própria tarefa para melhorar a representação baseada no método de *backpropagation* do gradiente de erro.
  - Skip-gram: O reverso do modelo CBOW seria o Skip-gram, pois diferentemente do outro modelo, a predição se dá na palavra central após o treinamento do contexto. As camadas de *input* se relaciona com a palavra alvo, e a camada de *output* se relaciona com o contexto. O modelo visa estimar o contexto dada a palavra. A última fase do modelo é a correlação entre o *output* e todas as palavras no contexto para ajustar a representação baseada no método de *backpropagation*. O Skip-gram performa melhor quando se tem menos dados de treinamento

e poucas palavras frequentes são apresentadas, porém comparando com o outro modelo, o CBOW performa mais rápido e melhor com palavras repetidas.

- GloVe: GloVe foi apresentado por Maning et al. (2014)[14], sendo um algoritmo usado amplamente para tarefas de classificação de texto. O GloVe é a expansão do Word2Vec para aprendizado de vetores de palavras de forma eficiente, onde as predições de palavras são feitas baseadas nas palavras ao redor. Diferentemente do Word2Vec em que as informações estatísticas do todo, não são usadas bem, o GloVe consegue abranger as estatísticas para compreender melhor o contexto.

Métodos de representação de palavras como o Word2Vec e o GloVe são simples e acurados, e em *data sets* grandes, eles podem aprender as representações semânticas das palavras, porém eles não aprendem as correlações de palavras de fora do vocabulário, sendo elas definidas de duas formas: Palavras que não são inclusas no vocabulário atual e palavras que não aparecem no corpus de treino atual.

4.2.2 Modelos Contextuais

Embora os modelos contínuos possam reter as informações semânticas e sintáticas de um documento, ainda existem os problemas relacionados a manter a representação completa de um contexto específico de um documento. O entendimento do contexto atual é crucial para diversas tarefas em NLP, por isso alguns trabalhos recentes foram feitos para tentar incorporar as relações de palavras aos modelos de linguagens para resolver o problema de significado. Alguns dos modelos mais comuns baseados em contexto foram relatados pelos autores [19] da forma abaixo:

- Generic Context word representation (Context2Vec): Modelo proposto por Melamud et al. [16] (2016, pp. 51-61) para gerar representações de palavras dependentes de contexto. O modelo é baseado no CBOW do Word2Vec, mas substitui a representação da palavra média fixada em uma janela por uma melhor e mais poderosa rede neural bidirecional LSTM. Foi usado um corpus de texto grande para ensinar a rede neural as relações de contexto de uma frase e as palavras alvo com a mesma dimensão, que depois foi otimizada para refletir as interdependências entre o alvo e o contexto da sentença como um todo.

- Embedding From Language Models (ELMo): O modelo Embedding from Language Models (ELMo) foi proposto por Peters et al. (2018)[20] é capaz de dar representações contextuais das palavras de forma profunda. Os pesquisadores concordam que dois problemas devem ser levados em consideração em um modelo de sucesso: A natureza dinâmica da palavra usada na semântica e gramática, e que conforme

a o ambiente de linguagem evolui, esses usos devem ser alterados. Os vetores de palavras finais são aprendidos por um modelo de linguagem bi direcional. O ELMo usa uma concatenação linear das representações aprendidas pelo modelo de linguagem bidirecional ao invés de apenas a camada final de representação como outros modelos de representação contextual. Em diferentes frases, ELMo consegue prover diferentes representações de palavras.

- Generative Pre-Training (OpenAI Transformer): O Generative Pre-Training (GPT) proposto por Radford et al. (2019)[21], é o primeiro modelo de linguagem pré-treinado baseado em transformação que consegue manipular a semântica das palavras de forma efetiva no termo de contexto. O aprendizado foi feito em um corpus enorme de textos livres, o GPT estende o modelo de linguagem não supervisionado em uma escala muito maior. Diferentemente do ELMo, o GPT usa o decodificador do transformador no modelo de linguagem de forma auto regressiva, fazendo com que o modelo preveja a próxima palavra de acordo com o contexto. O GPT vem se mostrando tendo uma boa performance em muitas tarefas, porém possui o contra de ser unidirecional. A representação geral do modelo é vista na figura 4.2

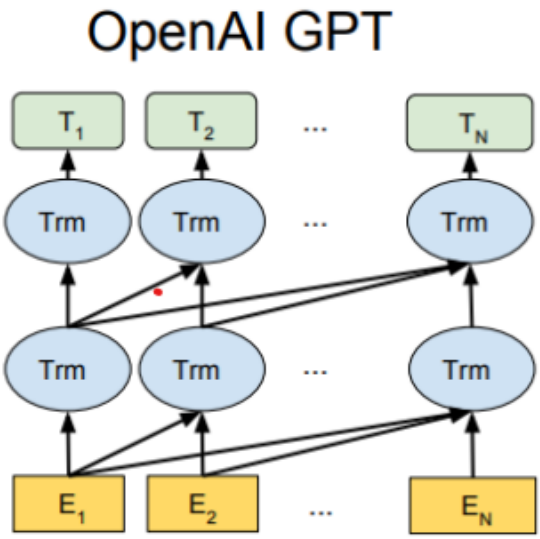


Figura 4.2 – Representação do modelo GPT retirada do [21]

- Bidirectional Encoder Representations from Transformers: Como descendente do GPT, temos o Bidirectional Encoder Representations from Transformers (BERT) proposto por Devlin et al. (2018)[5], é outro modelo contextual que segue a mesma premissa do GPT, treinar um modelo de linguagem enorme em um texto aberto e depois calibrar em tarefas individuais com arquiteturas de redes neurais customizadas. A rede neural do BERT utiliza camadas de atenção paralelas ao invés de recorrências sequenciais. O BERT é treinado com duas tarefas para encorajar a predição bidirecional e a compreensão do nível de frases. O BERT é treinado com duas tarefas não su-

pervisionadas: (1) O modelo de linguagem mascarado onde 15% dos *tokens* são arbitrariamente trocados por "MASK", e o modelo é treinado para predizer quais são os *tokens* mascarados. (2) A tarefa de prever a próxima sentença, então um par de sentenças é dado ao modelo e treinado para identificar quando a segunda segue a primeira, sendo essa tarefa feita para coletar informações adicionais que são de longo prazo ou pragmáticas. O BERT é treinado no *dataset* de corpus de livros [29] e nas passagens de texto da Wikipédia em inglês. O modelo pré treinado do BERT está disponível publicamente nas versões BERT-Base e BERT-Large, podendo ser utilizado em dados sem anotações ou calibrados em dados para tarefas específicas no modelo pré treinado. Na figura 4.3 é apresentado o modelo do BERT.

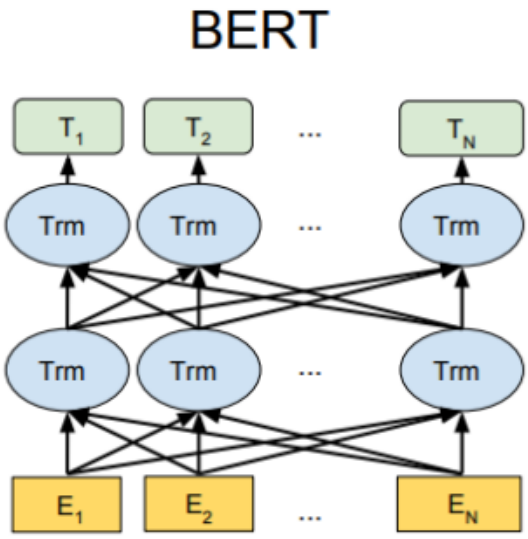


Figura 4.3 – Representação do modelo BERT retirada do [5]

As conclusões dos autores Naseem et al. [19] são:

Os modelos de clássicos de representação, são de fácil implementação, porém muito diretos e ingênuos, já que não conseguem capturar as informações sintáticas e semânticas, já que não consideram as ordens das palavras e nem o relacionamento entre elas. Hoje em dia já são legado, devido sua baixa performance, causada por exemplo de que o tamanho do vetor de *input* é proporcional ao tamanho do vocabulário do texto, fazendo ser muito pesado computacionalmente, embora tenham sido muito utilizados no passado para diferentes tarefas como classificação de documentos, NLP, extração de informações e até visão computacional.

Métodos de representações por aprendizado, ajudaram a comunidade a construir modelos poderosos, porém, tem o contra de que as *features* precisam ser selecionadas manualmente, então para resolver isso, foi preciso apresentar métodos capazes de descobrir e aprender essas representações de forma automática para realizar as tarefas. Porém mesmo tendo grande sucesso esses modelos em tarefas como classificação e superar os modelos categóricos, ainda existem limitações que atingem tais modelos, como a polissemia, já

que palavras polissêmica são endereçadas ao mesmo vetor, ignorando seu contexto, como um exemplo clássico da palavra "Banco"que pode ter dois significados completamente diferentes entre o banco de sentar e o a instituição financeira. Outro problema encontrado nesses modelos são com palavras fora do vocabulário que são atribuídas a vetores aleatórios durante o treinamento. Todos esses contras diminuem a performance dos modelos, e performam mal em textos de baixa qualidade.

Embora os modelos contextuais tenham a solução para problemas contextuais que os modelos contínuos não são capazes de capturar, é preciso salientar que os modelos contextuais exigem mais capacidade computacional, sendo considerados caros computacionalmente falando, além disso, exigem a utilização de redes neurais para realizar suas tarefas, o que pode ser uma tarefa mais complicada de implementação.

4.3 Trabalhos Relacionados

Esta seção será para abordar os trabalhos e estudos relacionados ao estudo proposto no trabalho. Sendo selecionados artigos e pesquisas de assuntos semelhantes ao proposto.

- No artigo de Araujo et al. (2018, pp.313-323)[13] foi percebido pelos autores, que a área judicial havia potencial para extração de informações com NER, permitindo que aplicações como prover *links* para as leis citadas e casos legais, ou agrupar documentos similares. Devido aos documentos legais de texto terem idiossincrasias a respeito de maiusculização, pontuação e estrutura, não possuem estrutura frasal de sujeito e predicado, como eles demonstraram no exemplo da figura 4.4, por isso, mesmo já existindo corpus manualmente anotados em português, como o HAREM [25] e o Paramopama [8], foi concluído que a distribuição dos documentos são diferentes dos corpus existentes de forma que modelos treinados neles iriam performar mal quando processando documentos legais. Também como não existem anotações específicas para entidades judiciais, os modelos iriam falhar para extrair o conteúdo específico legal. Os autores apresentam um *dataset* da língua português para reconhecimento de entidades nomeadas composto inteiramente por documentos legais manualmente anotados, criando duas novas categorias de entidades foram adicionadas para extrair o conhecimento legislativo, "LEGISLACAO", para entidades relacionadas a leis; e "JURISPRUDENCIA", para entidades que se referem a casos legais.

  Para compor o *dataset*, 66 documentos foram utilizados de diferentes cortes brasileiras e quatro documentos foram utilizados de documentos legislativos, totalizando um total de 70 documentos.

EMENTA: APELAÇÃO CÍVEL - AÇÃO DE INDENIZAÇÃO POR DANOS MORAIS - PRELIMINAR - ARGUIDA PELO MINISTÉRIO PÚBLICO EM GRAU RECURSAL - NULIDADE - AUSÊNCIA DE INTERVENÇÃO DO PARQUET NA INSTÂNCIA A QUO - PRESENÇA DE INCAPAZ - PREJUÍZO EXISTENTE - PRELIMINAR ACOLHIDA - NULIDADE RECONHECIDA.

HABEAS CORPUS 110.260 SÃO PAULO RELATOR : MIN. LUIZ FUX PACTE.(S) :LAERCIO BRAZ PEREIRA SALES IMPTE.(S) :DEFENSORIA PÚBLICA DA UNIÃO PROC.(A/S)(ES) :DEFENSOR PÚBLICO-GERAL FEDERAL COATOR(A/S)(ES) :SUPERIOR TRIBUNAL DE JUSTIÇA

Figura 4.4 – Estrutura do documento legal demonstrada no artigo [13]

Para cada documento, foi utilizada a biblioteca NLTK para separar o texto em uma lista de frases e tokeniza-las. Sendo o *output* final de cada documento um arquivo com apenas uma palavra por linha e linhas vazias delimitando o fim da sentença.

Após pré-processar os documentos, foi utilizada a ferramenta WebAnno para manualmente anotar cada um dos documentos com as *tags* de:

- "ORGANIZACAO": Para organizações
- "PESSOA": Para pessoas
- "TEMPO": Para entidades temporais
- "LOCAL": Para localizações
- "LEGISLACAO": Para leis
- "JURISPRUDENCIA": Para decisões relacionadas a casos legais

  Sendo "LEGISLACAO"e "JURISPRUDENCIA"correspondentes às classes "Ato de Lei"e "Decisão", respectivamente.

Foi utilizado o esquema de anotação IOB e as entidades nomeadas foram assumidas para que não se sobrepusessem e não se estendessem para mais de uma frase.

Na criação do *dataset*, 50 documentos foram selecionados aleatoriamente para o treinamento e 10 documentos para cada um os conjuntos de desenvolvimento e testes. Foi comparado que o total de *tokens* no LeNER-Br é comparável a outro corpus de NER como o Paramopama e o CONLL-2003 English [24], estando os três entre 300.000 e 320.000 *tokens*.

Para estabelecerem uma linha de base no *dataset*, foi escolhido o modelo LSTM-CRF, proposto por Lample et al. (2016) [12], pois esse modelo é capaz de atingir a performance de estado da arte de 90,94% no F1-Score quando utilizado no conjunto de testes do English CoNLL-2003.

A arquitetura do modelo consiste em uma camada memória de curto prazo longa bidirecional, também conhecida por Long Short-Term Memory (LSTM), seguida por uma camada de campo aleatório condicional, ou também chamada por Conditional Random Field (CRF). O *input* do modelo é uma sequência de representações vetoriais de cada palavra individual contruída da concatenação de ambas as relações de palavras e de nível de caracteres.

Para a tabela de pesquisa, foram utilizadas 300 dimensões do GloVe pré treinadas em corpus de gêneros múltiplos formados por textos em português brasileiro e português europeu, e a tabela de pesquisa de caracteres foi obtida por valores aleatórios com relações para cada caractere no *dataset*. As relações são alimentadas para uma camada LSTM separada bidirecional. O resultado é concatenado com a outra relação de palavras pré treinada, resultando no vetor final de representação. A figura 4.5 representa o panorama geral do processo.

Para a redução do *overfitting* e melhora das capacidades de generalização do modelo, foi utilizada técnica proposta por Srivastava et al. (2014, pp.1929-1958)[27], utilizando uma máscara nos *outputs* das duas camadas bidirecionais LSTM. Sendo a figura 4.6 a demonstração da arquitetura principal do modelo.

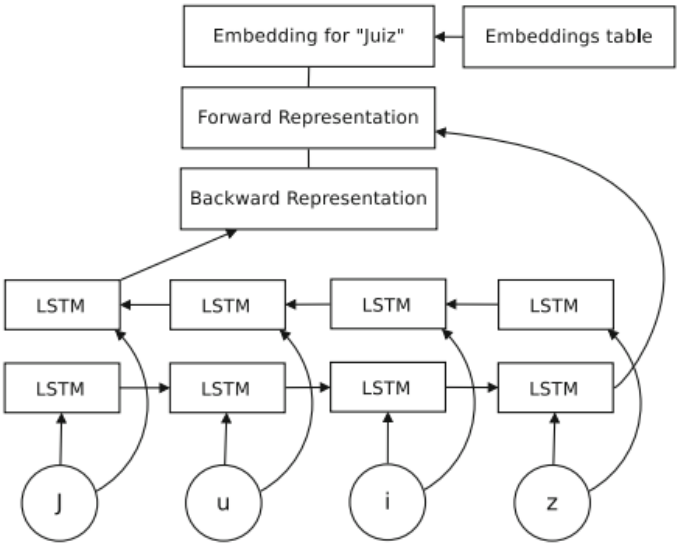


Figura 4.5 – Panorama geral do processo de concatenação demonstrada no artigo [13]

Para o *tuning* de hiperparâmetros, foram utilizados o Adam, proposto por Kingma e Ba [10] e o Stochastic Gradient Descent (SGD) como otimizadores, tendo como preferência o SGD pois mesmo tendo convergência menor que o Adam, atingiu melhores resultados. Utilizaram também a técnica de *Gradient Clipping* para evitar a explosão de gradientes.

Após experimentarem diversos hiperparâmetros, os que performaram melhores foram os mesmos utilizados por Lample et al. (2016)[12], apresentados como na figura 4.7.

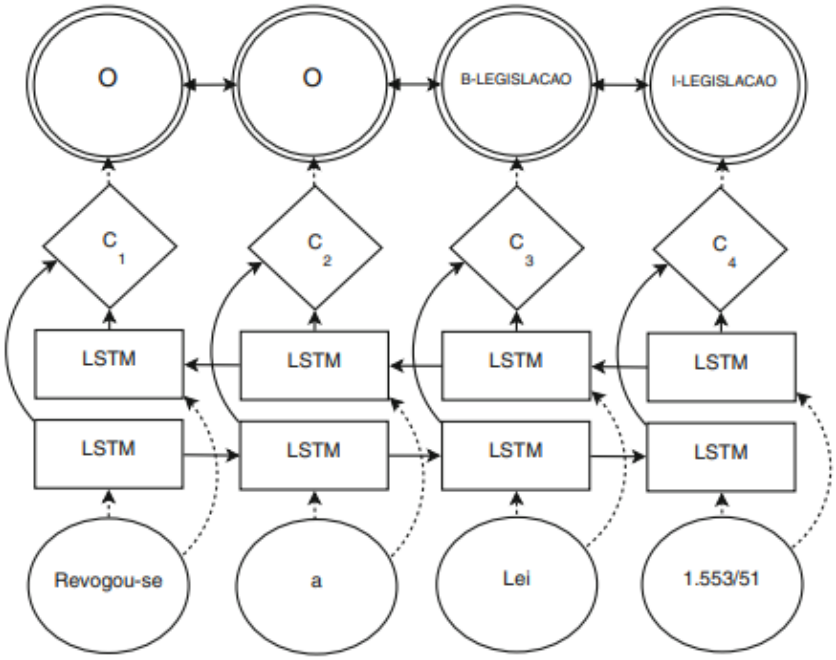


Figura 4.6 – Arquitetura principal do modelo demonstrada no artigo [13]

Foi utilizada a técnica de *Learning Rate Decay* em todas as épocas, salvando os parâmetros da rede apenas quando atingia melhor performance no conjunto de validação de épocas passadas. O modelo foi primeiro treinado com corpus do Paramopama para avaliar se poderia atingir a performance do estado da arte em um *dataset* português, e após confirmar que o modelo performava melhor que o modelo do estado da arte proposto por Mendonça et al. (2016)[17] a rede LSTM-CRF foi treinada com o *dataset* proposto.

Os passos de pré-processamento aplicados foram transformar as palavras em caixa baixa e substituir todos os dígitos por zero, sendo ambos os passos necessários para combinar o pré-processamento das relações de palavras pré treinadas. Como o nível de representação de caracteres preserva a maiusculização, essa informação não é perdida quando as palavras estão em caixa baixa.

Os resultados obtidos do F1-Score no LeNER-BR foram de 97,04% e 88,82% para as entidades de legislação e jurisprudência respectivamente, atingindo uma média de 92,53% como média geral entre o reconhecimento das entidades no texto.

Hyper-parameter	Value
Word embedding dimension	300
Character embedding dimension	50
Number of epochs	55
Dropout rate	0.5
Batch size	10
Optimizer	SGD
Learning rate	0.015
Learning rate decay	0.95
Gradient clipping threshold	5
First LSTM layer hidden units	25
Second LSTM layer hidden units	100

Figura 4.7 – Hiperparâmetros utilizados no modelo demonstrados no artigo [13]

5. RECONHECIMENTO DE ENTIDADES NOMEADAS EM BULAS

O protótipo desenvolvido ao longo deste trabalho de conclusão de curso seguiu a metodologia descrita por Araujo et al. [13] com algumas adaptações.

5.1 Corpus

O primeiro passo foi montar um corpus do entitulado AGROBULA-NER para o treinamento do modelo. Assim como no caso do LeNER-BR, os corpora já existentes em portugês não possuem dados de domínio específico relacionados área agrícola, e, por isso, os modelos que tivessem sido gerados a partir desses corpora possivelmente iriam falhar na extração desse conteúdo de conhecimento específico, Além disso, as bulas dos defensivos agrícolas também não são formatadas de forma que tenha um ordenamento de frases com sujeitos e predicados. Sendo optou-se por reunir um conjunto de bulas. Foram coletadas 2678 bulas do site do ministério da agricultura[1], mas apenas 92 foram usadas devido ao escopo e tempo para produção do projeto ser em um tempo enxuto. Por isso, foram selecionadas as 92 bulas inicais de forma sequencial.

Para a bula estar pronta para ser anotada e depois utilizada no modelo, ela teve de passar por diversos passos que serão descritos abaixo.

Primeiramente, as bulas são diretamente salvas no formato .pdf, portanto, o primeiro passo para poder começar manipular a bula foi transformar de .pdf para .html, porém isso não era o suficiente, dado que na conversão de pdf para html, eram mantidas diversas *tags* de configuração da página, como demonstrado na figura 5.1. Foi utilizada a biblioteca Aspose [2] assim sendo capaz de encontrar no texto onde estavam os elementos dentro das tabelas das bulas.

```
<html><head><meta http-equiv="Content-Type" content="text/html; charset=utf-8" /><meta http-equiv="Content-Style-Type" content="text/css" /><meta
name="generator" content="Aspose.Words for Python via .NET 22.8.0" /><title></title></head><body style="font-family:'Times New Roman';
font-size:12pt"><div><div style="-aw-headerfooter-type:header-primary; clear:both"><p style="margin-top:0pt; margin-bottom:0pt"><span
style="height:0pt; display:block; position:absolute; z-index:-65537"><img src="bula-5119-2021-09-01.002.png" width="717" height="390" alt=""
style="margin-top:244.54pt; -aw-left-pos:0pt; -aw-rel-hpos:margin; -aw-rel-vpos:margin; -aw-top-pos:0pt; -aw-wrap-type:none; position:absolute"
/></span><img src="bula-5119-2021-09-01.001.png" width="717" height="390" alt="" style="margin-left:0pt; -aw-left-pos:0pt; -aw-rel-hpos:margin;
-aw-rel-vpos:margin; -aw-top-pos:0pt; -aw-wrap-type:none; position:absolute; z-index:0" /><span
```

Figura 5.1 – Trecho do arquivo .html gerado do .pdf

Após gerar o .html da bula, o arquivo precisava ser limpo, para isso, foi utilizada a biblioteca para manipulação de arquivos html: lxml [3], gerando um .html limpo, visto na figura 5.2, com apenas elementos textuais e sem configurações de aparência.

[1] agrofit.agricultura.gov.br
[2] products.aspose.com/words/python-net/
[3] lxml.de/api/lxml.html.clean.Cleaner-class.html

<body> <img src="bula-5119-2021-09-01.002.png"><img src="bula-5119-2021-09-01.001.png"><0xa0> Evaluation Only. Created with Aspose.Words. Copyright 2003-2022 Aspose Pty Ltd. Intrepid® 240 SC <logomarca do produto> Registrado no Ministério da Agricultura, Pecuária e Abastecimento - MAPA sob nº 00699 COMPOSIÇÃO: N-tert-butyl-N’-(3-methoxy-o-toluoyl)-3,5-xylohydrazide (METOXIFENOZIDA).....240,00 g/L (24,0% m/v) Outros Ingredientes.....860,00 g/L (86,0% m/v) <0xa0> <table><tr><td> GRUPO </td><td> 18 </td><td> INSETICIDA </td></tr></table> CONTEÚDO: VIDE RÓTULO CLASSE: Inseticida não sistêmico acelerador de ecdise. GRUPO QUÍMICO: METOXIFENOZIDA: Diacilhidrazina TIPO DE FORMULAÇÃO: Suspensão Concentrada (SC) TITULAR DO REGISTRO(*): Dow AgroSciences Industrial Ltda. Alameda Itapecuru, 506 - 2º andar, Bloco B, Parte-1 - Alphaville Centro Industrial e Empresarial / Alphaville CEP: 06454-080 - Barueri/SP - CNPJ: 47.180.625/0001-46 Fone: 0800 772 2492 - Registro no Estado nº 650 - CDA/SP<0xa0> (*) IMPORTADOR DO PRODUTO FORMULADO FABRICANTE DO PRODUTO TÉCNICO INTREPID TÉCNICO Registrado no

Figura 5.2 – Trecho do arquivo .html após ter sindo limpo

Para finalizar o *pipeline*, as bulas foram transformadas em arquivos texto, utilizando a biblioteca html2text [4], assim tendo arquivos texto sem as divisões e *tags* html como visto na figura 5.3 , prontas para serem passadas para o WebAnno e serem anotadas.

```
![](bula-5119-2021-09-01.002.png)![](bula-5119-2021-09-01.001.png)

Evaluation Only. Created with Aspose.Words. Copyright 2003-2022 Aspose Pty
Ltd.

Intrepid® 240 SC

<logomarca do produto>

Registrado no Ministério da Agricultura, Pecuária e Abastecimento - MAPA sob
nº 00699

COMPOSIÇÃO:

N-tert-butyl-N’-(3-methoxy-o-toluoyl)-3,5-xylohydrazide
(METOXIFENOZIDA).....240,00
g/L (24,0% m/v)
```

Figura 5.3 – Trecho do arquivo .txt após ter sindo transformado do .html

Foi utilizado o esquema de anotação IOB de Ramshaw e Marcus [22], visto que é o padrão citado por Jurafsky e Martin [9] e também por ter sido utilizado desta forma no LeNER-Br.

5.2 Arquitetura

O modelo LSTM-CRF foi o escolhido para ser possível estabelecer a linha de base visto que existem já valores de performance no estado da arte em tanto o LeNER-Br como no English CoNLL-2003, assim facilitando a avaliação do modelo no *dataset* de bulas.

Foi utilizada para a tabela de pesquisa, também a de 300 dimensões do GloVe pré treinada em corpus de gêneros múltiplos formados por textos em português brasileiro e português europeu, e a tabela de pesquisa de caracteres obtida por valores aleatórios com relações para cada caractere no *dataset* para resultar no vetor final de representação com os valores concatenados.

[4] https://github.com/Alir3z4/html2text

A técnica de utilização de máscaras por Srivastava et al. [27] para melhorar as capacidades de generalização do modelo e reduzir o *overfitting* também foi utilizada. E o *tuning* de hiperparâmetros também foram testados tanto o otimizador Adam, de Kingma e Ba (2014)[10] quanto o SGD, assim como utilizado no desenvolvimento do LeNER-Br devido aos seus resultados elevados.

Os hiperparâmetros utilizados, foram os mesmos utilizados por Lample et al [12] e por Araújo et al [13]. Utilizando técnica de *Learning Rate Decay* em todas as épocas. Após esses ajustes, a rede foi treinada com o textitdataset construído. Como visto na figura 5.4 onde é possível visualizar as etapas realizadas no desenvolvimento do projeto.

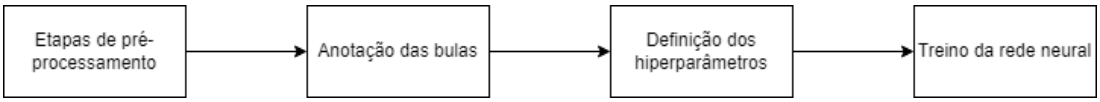


Figura 5.4 – Etapas de desenvolvimento do projeto.

Uma ressalva importante, diferentemente de como foi feito no LeNER-Br, as palavras não tiveram a caixa normalizada, como exigiria mais uma etapa de pré-processamento, não foi feito para que a conclusão do projeto se adequasse melhor dentro do prazo definido. No caso do LeNER-BR, todas as palavras foram transformadas em caixa baixa e dígitos substituídos por zero. Essas transformações não foram feitas no *dataset* na etapa de pré-processamento.

Conforme visto na arquitetura, a escolha do modelo e dos hiperparâmetros, foram idênticos aos utilizados por [13] et al. se diferenciando nas etapas de pré-processamento para tornar os arquivos necessários viáveis para anotação.

6. ANOTAÇÕES

Neste capítulo são abordados os tópicos: ferramentas utilizadas para anotação e o processo de anotação.

6.1 Ferramentas

Para as anotações manuais, foram testadas diversas ferramentas que serão descritas abaixo juntamente com a análise comparativa realizada.

- Light Tag: Ferramenta *web based*, sem necessidade de instalação. Aceita *inputs* em que o texto deve ser colado diretamente no site, ou arquivos .json e .csv. Tem como funcionalidade capacidade de criar relações entre as entidades e exporta o arquivo no formato .json. Como mostra na imagem 6.1, pode se ver a interface da ferramenta.

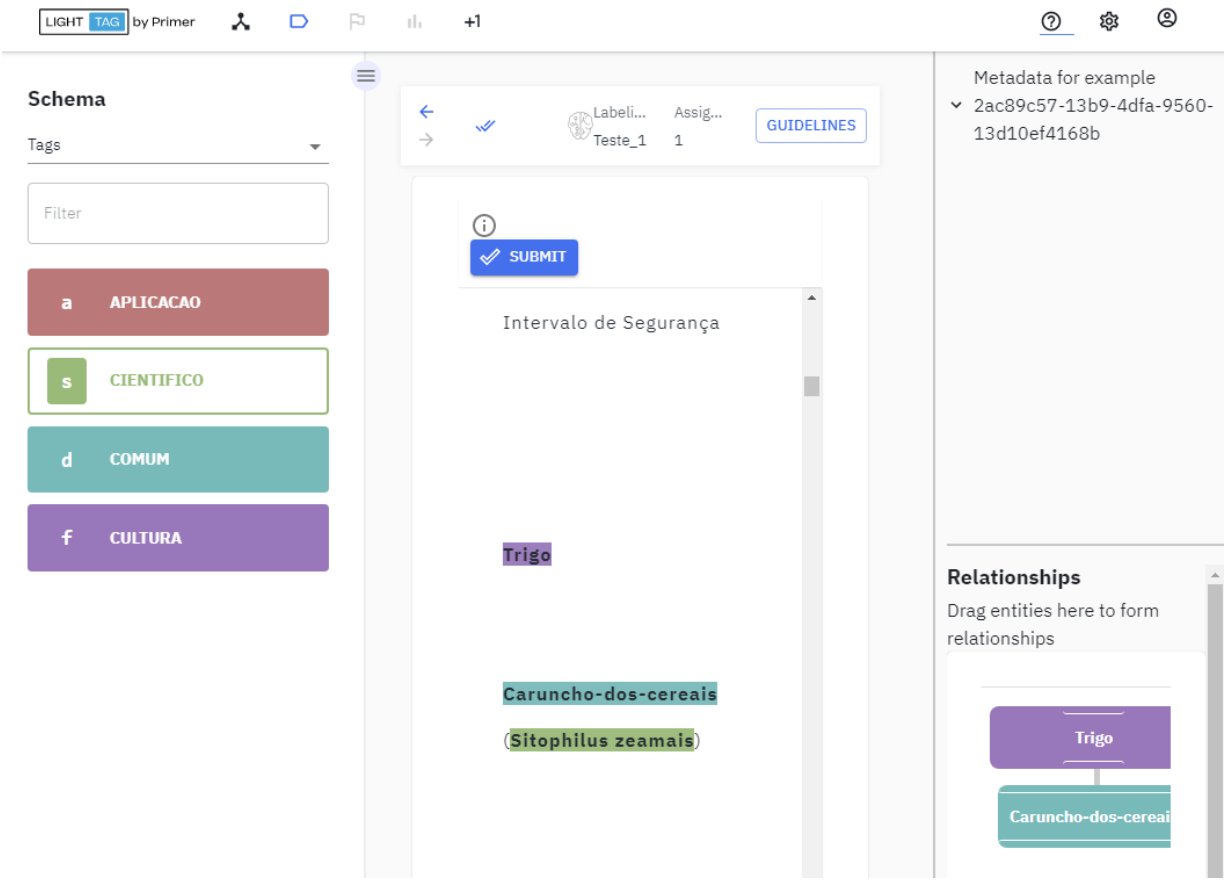


Figura 6.1 – Interface da ferramenta Light Tag

- General Architecture for Text Engineering (GATE): O GATE aparenta ser uma ferramenta em que cria *pipelines* com diversas tarefas envolvendo NLP, porém se mostrou

necessário que o texto esteja na língua inglesa. Capaz de receber como *inputs* arquivos em formtao .html e exporta em .xml. Como demonstrado na imagem 6.2, é visto a interface do GATE.

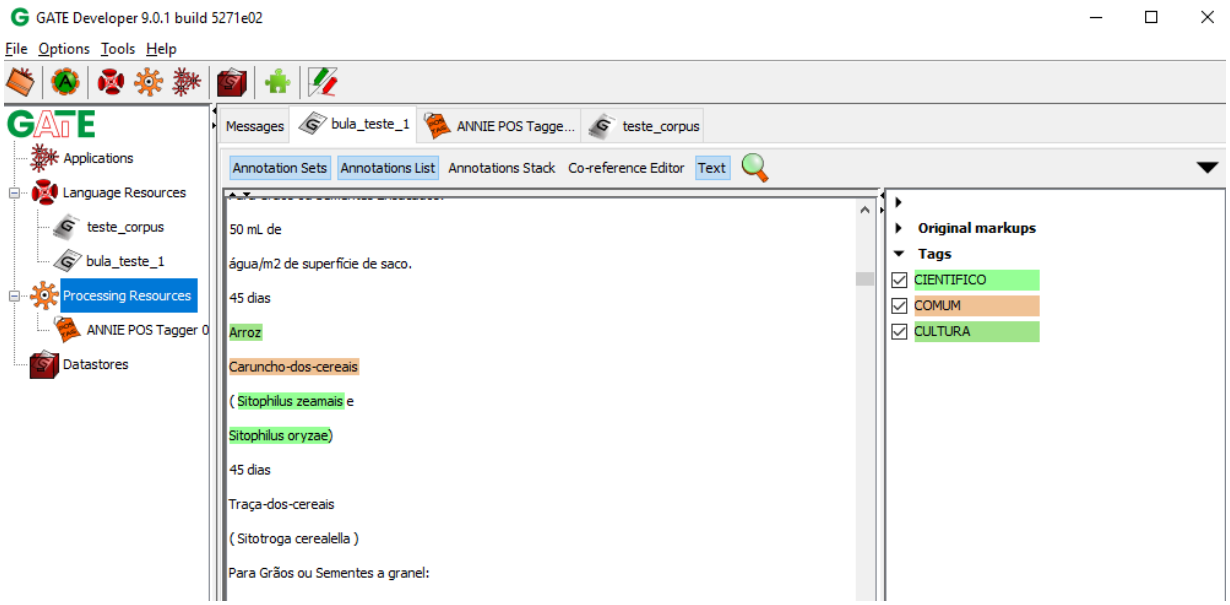


Figura 6.2 – Interface da ferramenta GATE

- Tag Editor: Aparentou ser uma ferramenta com diversas funcionalidades, podendo anotar POS manual por exemplo, recebendo *inputs* em formato .txt e exporta no formato .json, .txt ou arquivos no formato do SpaCy, tendo o lado negativo como limitado a 30 frases na versão gratuita, sendo necessário adquirir a versão que custa entre 25 a 35 dólares para ter acesso completo. Pode se ver a interface da ferramenta na imagem 6.3.

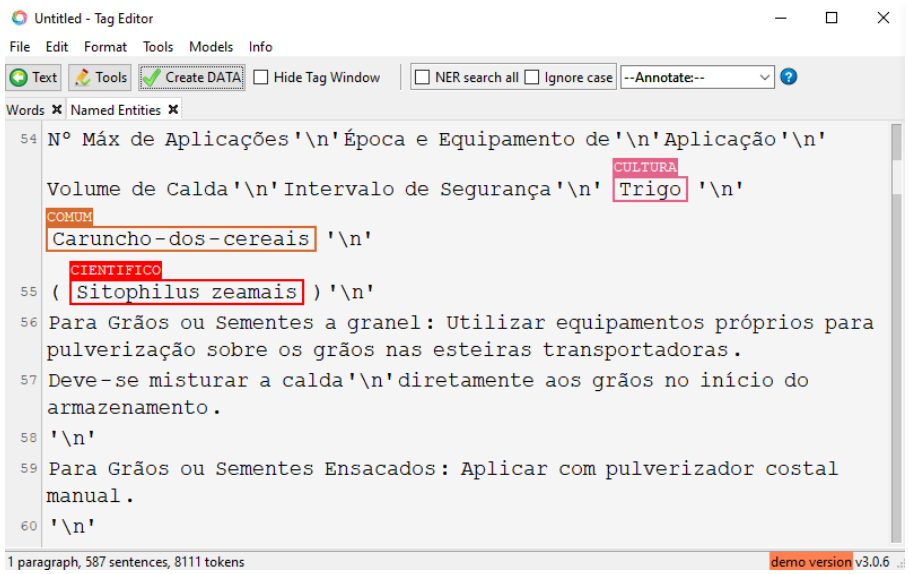


Figura 6.3 – Interface da ferramenta Tag Editor

- Docanno: A ferramenta Docanno é facil de instalar e usar, além disso aceita os formatos de .txt, .json, CoNLL, etc. e salva no formato JSONL, tendo o contra de que não se pode criar relações entre as entidades. Na figura 6.4 pode se ver como a ferramenta se apresenta.

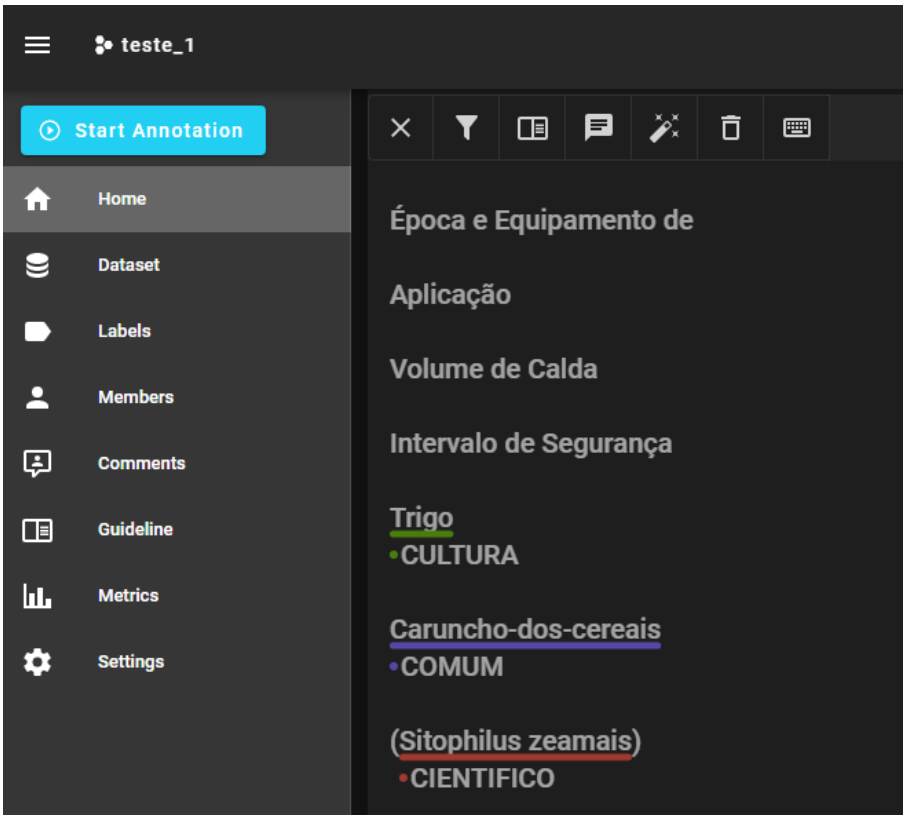


Figura 6.4 – Interface da ferramenta Docanno

- Browser-Based Rapid Annotation Tool (brat): O brat possui fácil instalação, diversas formas de configurações, podendo criar as *labels* e relações, recebendo como *input* arquivos de texto e exporta no formato de texto ou .ann. Sendo na figura 6.5 demonstrado a interface do brat.

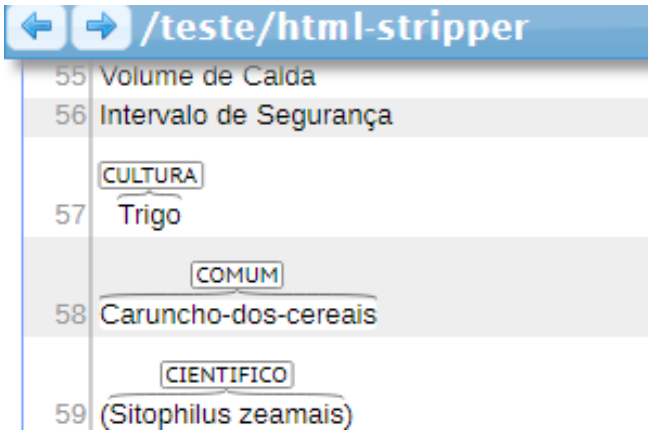


Figura 6.5 – Interface da ferramenta brat

- WebAnno: Foi utilizada no trabalho do [13] et al. por sua capacidade de receber os *inputs* no formato de *text* e salvar os *outputs* no formato CoNLL 2003, demonstrado na figura 6.6 que é o aceito pelo modelo do trabalho. Além disso a ferramenta tem interface simples e fácil de usar como demonstrada na figura 6.7.

```
alvos O
biológicos O
: O
Bonagota B-CIENTIFICO
salubricola I-CIENTIFICO
( O
Lagarta B-COMUM
enroladeira I-COMUM
) O
, O
Cryptoblabes B-CIENTIFICO
gnidiella I-CIENTIFICO
```

Figura 6.6 – Formato do arquivo CoNLL 2003

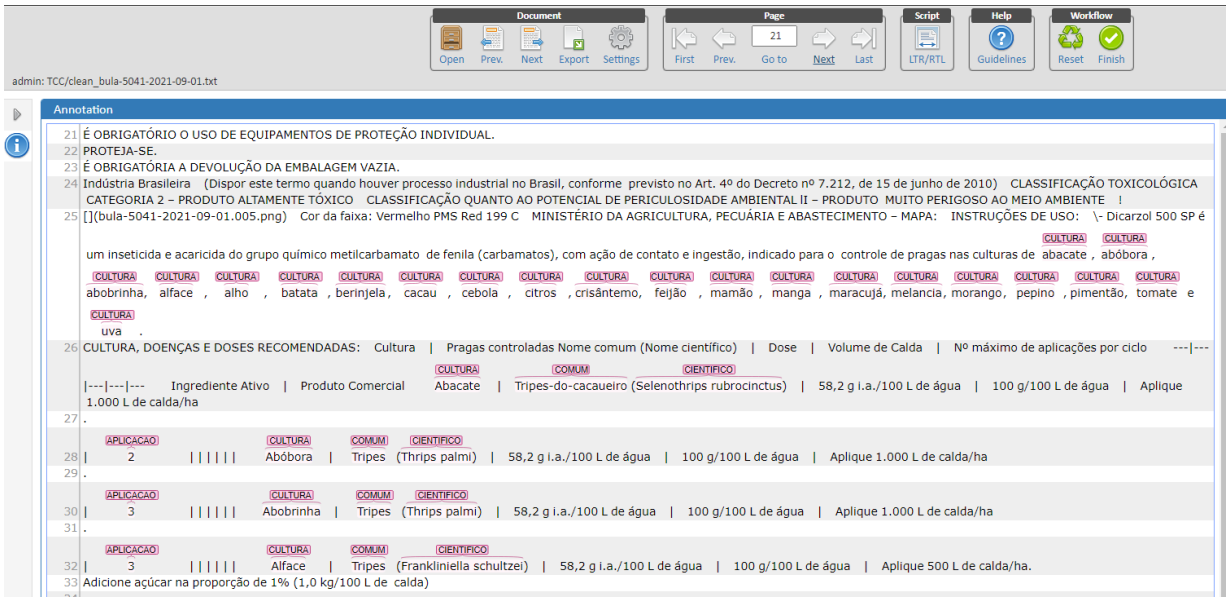


Figura 6.7 – Interface da ferramenta WebAnno

Conforme mostra na figura 6.8 é possível ver os comparativos entre as ferramentas. Principalmente pela questão de aceitar os *inputs* como arquivos texto e gerar *outputs* no formato CoNLL, foi escolhido o WebAnno.

Ferramenta	Aceita Input .txt?	Salva .conll?	Fácil Instalação?	Interface Amigável?	É gratuito?
Light Tag	X	X	N/A	✓	✓
GATE	X	X	✓	X	✓
Tag Editor	✓	X	✓	✓	X
Docanno	✓	X	✓	✓	✓
BRAT	✓	X	✓	✓	✓
WebAnno	✓	✓	✓	✓	✓

Figura 6.8 – Tabela comparativa das ferramentas

6.2 Processo de Anotação

Nesta seção, serão descritos alguns pontos, sendo eles a forma que foi anotada, as quantas versões foram geradas e suas devidas quantidades de entidades e o que foi procurado.

Foi anotado de forma manual em arquivos textos as entidades de: "CULTURA", para as entidades relacionadas ao tipo de semente; "COMUM"e "CIENTIFICO"para entidades relacionadas as doenças das plantas; e "APLICACAO"para entidades que serão relacionadas as aplicações dos defensivos agrícolas nas plantas.

Após o modelo ter sido treinado com a primeira versão do AGROBULA-NER, nos testes, foi observado que, em bulas não vistas antes (bulas do conjunto de teste), o modelo estava se perdendo em símbolos que formavam a tabela, como por exemplo em hifens utilizados para criar espaçamento na tabela. Isso motivou uma nova anotação. A anotação nova procurou estender a entidade COMUM quando haviam dois nomes juntos por exemplo, como demonstrado na figura 6.9. Já, no caso de aplicações, foi necessário alguns ajustes de anotação. Os resultados não estavam bons. Por isso, houve um foco maior em anotar aplicações, aumentando o número de bulas no corpus, tentando abranger um maior contexto como demonstrado na figura 6.10.

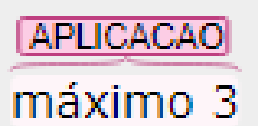


Figura 6.9 – Anotação de APLICACAO com contexto maior

O modelo com a segunda versão do AGROBULA-NER foi treinado e apresentou melhora, porém não foi capaz de resolver as anotações em símbolos separadores da ta-

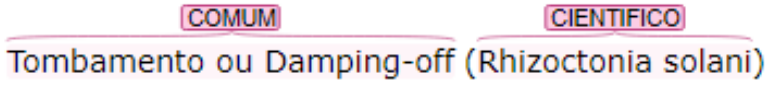


Figura 6.10 – Demonstração da anotação de COMUM para mais nomes

bela, além disso houve piora nas métricas de aplicações, foi levantada a hipótese de ser devido a primeira versão terem sido anotadas as aplicações máximas de forma com menor contexto, diferentemente do segunda versão, por isso, foi feito uma terceira versão do AGROBULA-NER revisando as bulas da primeira versão do AGROBULA-NER, onde foram também excluídas bulas que estavam sem entidades de aplicação.

7. *DATASETS*

Durante o desenvolvimento deste trabalho, foram gerados quatro conjuntos de dados diferentes mencionados no capítulo de anotações. Ao longo deste capítulo, serão descritas as métricas por entidade e também dividido nos dados de treino, o conjunto de dados de teste Dev, que foi utilizado durante o treinamento como referência em cada época treinada para validação e o conjunto de dados de teste que é o conjunto isolado. Contextualizando a formação dos três conjuntos treino, test e dev, foram feitos da seguinte forma: 15% aproximadamente do número total de bulas foi separado para o conjunto dev, 15% aproximadamente para o test e o restante para o treino, mantendo o mais próximo possível a proporção de entidades por entidade em cada conjunto. Nos *datasets* seguintes em que se mantiveram todas as bulas, foram mantidas as mesmas bulas para test e dev a fim de manter a mesma referência para validação de melhora de performance.

- Na primeira versão do AGROBULA-NER foi anotado o total de 2758 entidades, tendo como métricas de anotações cada entidade na tabela 7.1

	COMUM	CIENTIFICO	CULTURA	APLICACAO	Total
Train	492	633	294	162	1581
Dev	195	284	118	62	659
Test	146	219	82	71	518

Tabela 7.1 – Métricas AGROBULA-NER Versão 1

- Na segunda versão, foram anotadas 8409 entidades, e as métricas de anotação referenciadas na tabela 7.2

	COMUM	CIENTIFICO	CULTURA	APLICACAO	Total
Train	1948	2886	1329	1069	7232
Dev	195	284	118	62	659
Test	146	219	82	71	518

Tabela 7.2 – Métricas AGROBULA-NER Versão 2

- Na terceira versão, houveram 7011 entidades anotadas, com as métricas encontradas na tabela 7.3

	COMUM	CIENTIFICO	CULTURA	APLICACAO	Total
Train	1426	2127	1018	861	5432
Dev	260	411	146	111	928
Test	163	256	135	97	651

Tabela 7.3 – Métricas AGROBULA-NER Versão 3

- Foi feito uma quarta versão do AGROBULA-NER, com um total de 8409 entidades. Essa quarta versão é uma mescla da segunda versão, com a terceira. Foram utilizadas todas as bulas já antes anotadas, porém as bulas que estavam na primeira versão foram revistas para abranger maior contexto em aplicações. As métricas são descritas na tabela 7.4

	COMUM	CIENTIFICO	CULTURA	APLICACAO	Total
Train	1948	2886	1329	1069	7232
Dev	195	284	118	62	659
Test	146	219	82	71	518

Tabela 7.4 – Métricas *dataset* 4

Até onde se sabe, este é o AGROBULA-NER é o primeiro *dataset* que consiste inteiramente de textos de bulas de defensivos agrícolas manualmente anotados com as entidades de aplicação, nome comum, nome científico e cultura na língua portuguesa.

8. RESULTADOS

Neste capítulo serão retratados os resultados obtidos ao longo dos treinamentos dos modelos.

- O primeiro modelo treinado, já foi capaz de atingir um resultado satisfatório, com suas métricas descritas nas tabelas 8.1, 8.2 e 8.3

Entidade	Precision	Recall	F1-Score	Support
COMUM	96.81%	98.58%	97.68%	492
CIENTIFICO	98.28%	99.21%	98.74%	633
CULTURA	90.65%	98.98%	94.63%	294
APLICACAO	94.12%	98.77%	96.39%	162
Média	95.98%	98.92%	97.41%	1581

Tabela 8.1 – Resultados conjunto de treino do AGROBULA-NER Versão 1

Entidade	Precision	Recall	F1-Score	Support
COMUM	95.52%	65.64%	77.81%	195
CIENTIFICO	92.08%	85.92%	88.89%	284
CULTURA	81.89%	88.14%	84.90%	118
APLICACAO	89.47%	82.26%	85.71%	62
Média	91.03%	79.97%	84.60%	659

Tabela 8.2 – Resultados conjunto de teste 1 (Dev) do AGROBULA-NER Versão 1

Entidade	Precision	Recall	F1-Score	Support
COMUM	80.00%	93.15%	86.08%	146
CIENTIFICO	94.95%	94.52%	94.74%	219
CULTURA	91.57%	92.68%	92.12%	82
APLICACAO	94.23%	69.01%	79.67%	71
Média	90.10%	90.35%	89.82%	518

Tabela 8.3 – Resultados conjunto de teste 2 (Test) do AGROBULA-NER Versão 1

Contudo, foi levantada a hipótese de que aumentando o tamanho da amostra, seria possível obter resultados melhores, por isso, foram anotadas mais bulas para realizar um novo treinamento.

- Após anotar mais 75 bulas, foi gerado uma segunda versão que após o treino apresentou os resultados descritos nas tabelas 8.4, 8.5 e 8.6

  Foi visto após o treino do modelo usando a segunda versão que em um contexto geral, houve melhora na performance do modelo, principalmente na métrica do F1-Score nos conjuntos de teste, mas foi notado que a performance na aplicação, houve piora.

Entidade	Precision	Recall	F1-Score	Support
COMUM	98.46%	98.72%	98.59%	1948
CIENTIFICO	97.49%	99.55%	98.51%	2886
CULTURA	95.16%	97.67%	96.40%	1329
APLICACAO	85.34%	88.77%	87.02%	1069
Média	95.53%	97.39%	96.45%	7232

Tabela 8.4 – Resultados conjunto de treino do AGROBULA-NER Versão 2

Entidade	Precision	Recall	F1-Score	Support
COMUM	99.48%	98.46%	98.97%	195
CIENTIFICO	97.58%	99.30%	98.43%	284
CULTURA	89.31%	99.15%	93.98%	118
APLICACAO	58.43%	83.87%	68.87%	62
Média	92.98%	97.57%	95.01%	659

Tabela 8.5 – Resultados conjunto de teste 1 (Dev) do AGROBULA-NER Versão 2

Entidade	Precision	Recall	F1-Score	Support
COMUM	97.99%	100%	98.98%	146
CIENTIFICO	98.64%	99.54%	99.09%	219
CULTURA	95.24%	97.56%	96.39%	82
APLICACAO	73.68%	98.59%	84.34%	71
Média	94.50%	99.23%	96.61%	518

Tabela 8.6 – Resultados conjunto de teste 2 (Test) do AGROBULA-NER Versão 2

- Foi levantada a hipótese que devido a diferença na abordagem de como foi anotado o contexto de aplicação da primeira e o segunda versão do AGROBULA-NER, seria interessante fazer uma terceira versão, revisando as bulas iniciais para dar maior contexto de aplicações, além disso, como o foco se dava em aumentar a performance de aplicações, foram removidas bulas que não haviam ao menos uma aplicação anotada do conjunto de dados e redistribuídas entre os conjuntos de teste para que se mantivesse proporção próxima. Após a criação da terceira versão e o modelo treinado, foram obtidos os resultados descritos nas tabelas 8.7, 8.8 e 8.9

Entidade	Precision	Recall	F1-Score	Support
COMUM	98.39%	98.74%	98.56%	1426
CIENTIFICO	98.51%	99.76%	99.14%	2127
CULTURA	97.26%	97.74%	97.50%	1018
APLICACAO	94.71%	95.59%	95.14%	861
Média	97.64%	98.45%	98.05%	5432

Tabela 8.7 – Resultados conjunto de treino do AGROBULA-NER Versão 3

Após a análise dos resultados, foi visto que embora o modelo estivesse com uma média de precisão e F1-Score elevados, ainda sim a aplicação houve mais uma piora, também é possível observar que no conjunto de testes 2 (Test) em quase todas as

Entidade	Precision	Recall	F1-Score	Support
COMUM	97.95%	91.92%	94.84%	260
CIENTIFICO	91.67%	99.03%	95.20%	411
CULTURA	81.14%	97.26%	88.47%	146
APLICACAO	50.69%	65.77%	57.25%	111
Média	86.87%	92.78%	89.50%	928

Tabela 8.8 – Resultados conjunto de teste 1 (Dev) do AGROBULA-NER Versão 3

Entidade	Precision	Recall	F1-Score	Support
COMUM	88.82%	92.64%	90.69%	163
CIENTIFICO	90.61%	98.05%	94.18%	256
CULTURA	94.44%	88.15%	91.19%	135
APLICACAO	48.39%	77.32%	59.52%	97
Média	84.67%	91.55%	87.52%	651

Tabela 8.9 – Resultados conjunto de teste 2 (Test) do AGROBULA-NER Versão 3

entidades e na média geral, foi onde apresentaram os piores resultados, porém, é esperado que no modelo haja imperfeições até mesmo pelo tamanho da amostra, que dando como exemplo, no trabalho do LeNER-BR do Araujo et al. [13] foram anotados cerca de 300.000 *tokens*, uma diferença de 293.000 *tokens* aproximadamente dado o escopo e tamanho do projeto.

- Uma quarta versão foi criada para que fosse testada também o quanto as bulas sem aplicação estariam afetando a performance do modelo, por isso, foram utilizadas as bulas anotadas na versão 3 com a revisão de contexto das bulas d versão 1, somadas com as bulas da versão 2. Também foram mantidas as bulas do conjunto de teste utilizadas no AGROBULA-NER 1 e 2, porém com o contexto maior aplicado. Gerando o resultado demonstrado nas tabelas 8.10, 8.11 e 8.12

Entidade	Precision	Recall	F1-Score	Support
COMUM	99.13%	99.44%	99.28%	1948
CIENTIFICO	98.87%	99.83%	99.34%	2886
CULTURA	96.98%	99.17%	98.07%	1329
APLICACAO	80.11%	98.32%	88.28%	1069
Média	95.82%	99.38%	97.46%	7232

Tabela 8.10 – Resultados conjunto de treino do AGROBULA-NER Versão 4

Foi visto que a versão 4 do AGROBULA-NER houve em grande maioria os melhores resultados em termos de métricas, com exceções da aplicação, onde no treino e dev, obteve o pior resultado dos conjuntos de dados. O que chamou a atenção, foram resultados de 100% em diversas métricas, sendo um ponto que necessita de maior estudo, pois pode indicar um grau de *overfit*.

Entidade	Precision	Recall	F1-Score	Support
COMUM	100%	99.49%	99.74%	195
CIENTIFICO	100%	100%	100%	284
CULTURA	90.08%	100%	94.78%	118
APLICACAO	47.66%	98.39%	64.21%	62
Média	93.30%	99.70%	95.62%	659

Tabela 8.11 – Resultados conjunto de teste 1 (Dev) do AGROBULA-NER Versão 4

Entidade	Precision	Recall	F1-Score	Support
COMUM	100%	100%	100%	146
CIENTIFICO	99.55%	100%	99.77%	219
CULTURA	97.59%	98.78%	98.18%	82
APLICACAO	73.96%	100%	85.03%	71
Média	95.86%	98.81%	97.56%	518

Tabela 8.12 – Resultados conjunto de teste 2 (Test) do AGROBULA-NER Versão 4

Analisando os quatro modelos treinados, o modelo treinado usando a versão 3 é o que mais indica apresentar balanço em suas métricas, por isso é possível ponderar que seja o modelo mais realístico, apresentando um grau de abstração maior. Porém é notável que o modelo treinado com a versão 4, apresentou as métricas mais elevadas, seria necessário evoluir na pesquisa para validar se de fato o modelo é o mais adequado ou acabou gerando um *overfit*.

Na figura 8.3 podemos ver um trecho de uma bula em que o modelo 4 foi utilizado para fazer a anotação de uma bula nova, em amarelo a entidade de cultura, roxo o nome científico, verde o nome comum e vermelho a aplicação. Podemos ver que existem alguns erros, como por exemplo na cultura do feijão em que o modelo classificou "(***)"como uma cultura, mas as outras entidades estavam corretas.

Foram feitos dois gráficos expondo visualmente os resultados obtidos, estão referenciados nas figuras 8.1 e 8.2.

8.1 Discussão

Após o desenvolvimento do trabalho e análise dos resultados, nesta seção serão retratados os pontos de discussão.

Conforme observado, nenhum dos modelos foi capaz de resolver completamente a questão de indicar entidades principalmente de nome comum e científico em caracteres gráficos com propósito de modelar a tabela como visto na figura 8.4 em que vemos em verde o que o modelo compreende como nome comum e roxo nome científico. Como descrito antes, houveram cerca de 8000 *tokens* anotados ao longo de todos os processos, um número relativamente pequeno, ainda mais comparado ao total de 2676 bulas, onde foram

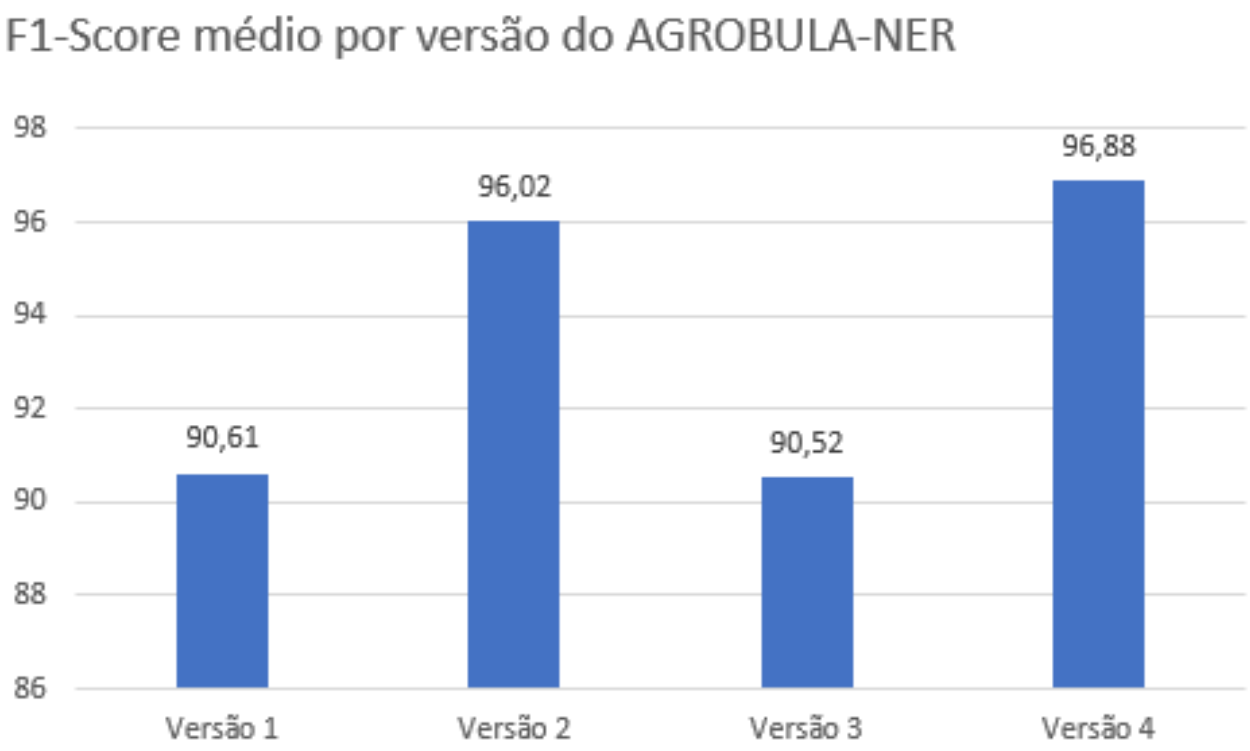


Figura 8.1 – Gráfico do F1-Score médio por versão do *dataset*

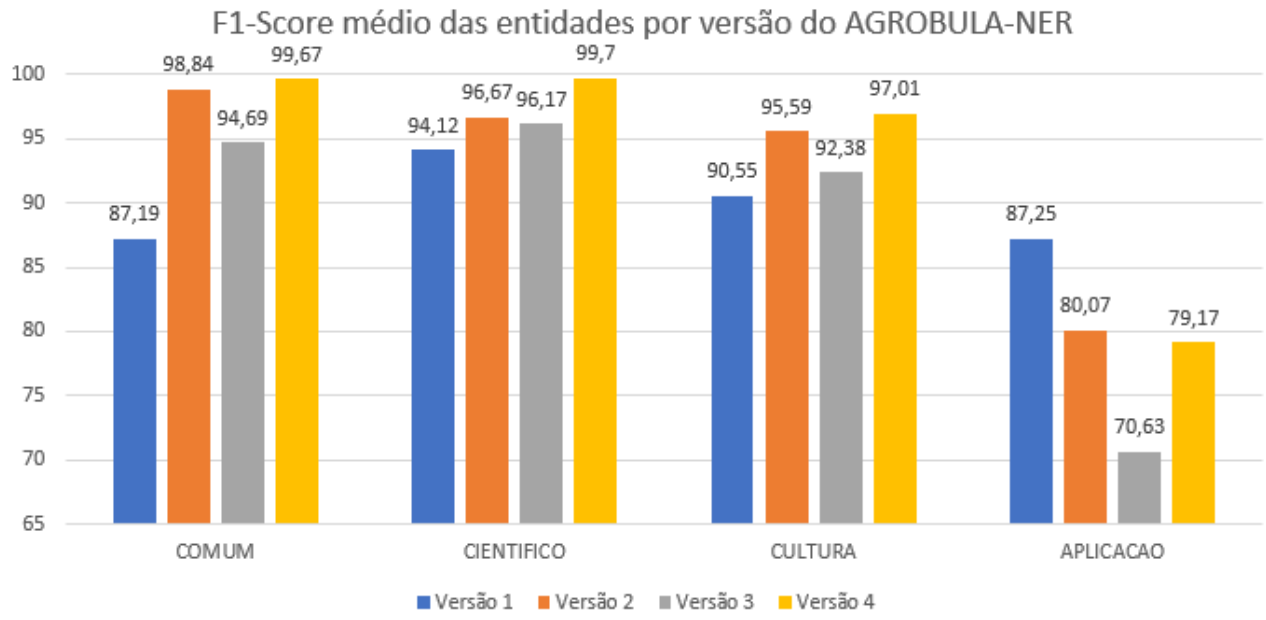


Figura 8.2 – Gráfico do F1-Score médio por entidade de cada versão do *dataset*

anotadas cerca de 130 bulas devido a ser realizado por um único individuo e pelo tamanho do escopo do projeto. Portanto é seguro afirmar que há espaço para melhora no modelo e na qualidade do AGROBULA-NER conforme aumento de anotações.

Também é visto que a entidade de aplicação, precisa ser melhor estudada em formas de tentar melhorar sua performance dado que suas métricas apresentam ainda es-

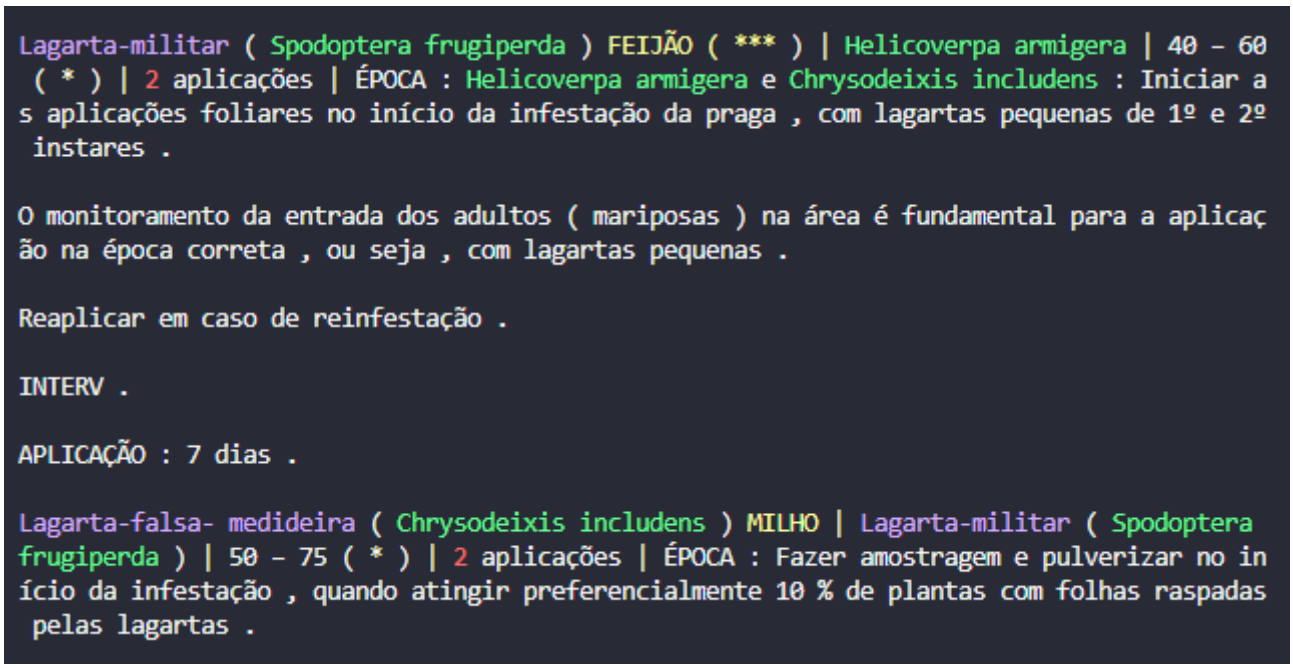


Figura 8.3 – Demonstração da anotação do modelo em trecho de bula

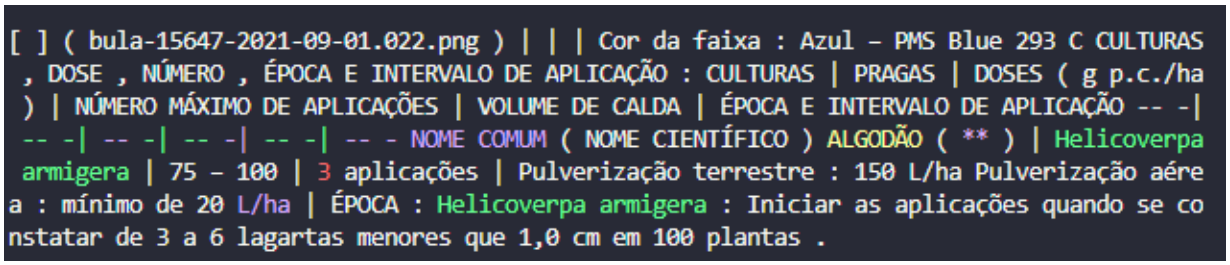


Figura 8.4 – Trecho de bula anotada pelo modelo

tarem abaixo. Uma possibilidade é utilizar da técnica de pré-processamento de transformar os números todos em 0, porém pode também estar atrelado ao tamanho da amostra, que com mais textos rotulados, o modelo pode se tornar mais capaz de aprender os casos específicos de aplicação máxima, porém semelhante ao caso do trabalho do Araujo et al. [13] em que as entidades de localização tiveram performance abaixo comparadas as outras devido diversas razões, mas principalmente por ter menor quantidade de entidades nos textos, o mesmo acontece na entidade de aplicação, havendo diversas bulas sem citar as aplicações máximas, ou nem todas as culturas descritas as aplicações máximas, por isso, acredito que é possível atingir uma performance melhor.

Houve um pequeno teste para compreender se com a versão do AGROBULA-NER atual, seria capaz de nomear as entidades dentro de textos agronômicos em formatos diferentes dos vistos nas bulas, e o resultado foi que talvez seja possível, dado que no teste, foi capaz de fazer algumas anotações corretas no texto como descrito nas figuras 8.5 e 8.6 demonstrando que pode ser uma outra frente para abordar.

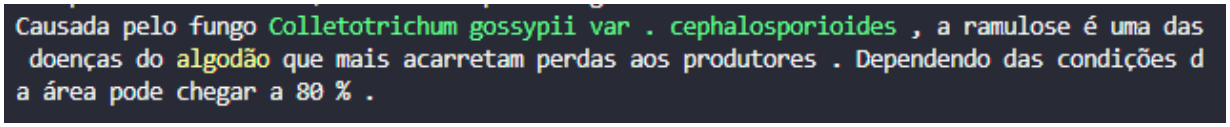


Figura 8.5 – Trecho de texto anotado pelo modelo

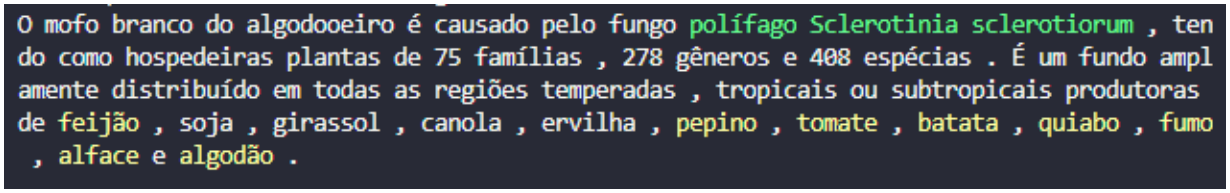


Figura 8.6 – Trecho de texto anotado pelo modelo

9. TRABALHOS FUTUROS

Ao longo do desenvolvimento, foi possível notar que as oportunidades dentro do escopo de NER para bulas agronômicas é extremamente vasto, por isso neste capítulo será retratado algumas pontos focais.

Dentro da bula além das quatro entidades abordadas no trabalho, dentro das bulas se pode identificar outras entidades que para trabalhos futuros, é possível anotar outras entidades dentro do texto, como por exemplo a quantidade da dose que deve ser feita, o intervalo entre as aplicações, a época, entre outras que estão contidas nas bulas, assim conseguindo um modelo mais completo.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] Capturado em: https://www.syngenta.com.br/sites/g/files/zhg256/f/actellic_500_ec.pdf?token=1655328002, Mar 2022.

[2] Capturado em: https://www.syngenta.com.br/sites/g/files/zhg256/f/alto_100_2.pdf?token=1649892839, Mar 2022.

[3] Balahur, A. “Sentiment analysis in social media texts”. In: Proceedings of the 4th workshop on computational approaches to subjectivity, sentiment and social media analysis, 2013, pp. 120–128.

[4] Bolukbasi, T.; Chang, K.-W.; Zou, J. Y.; Saligrama, V.; Kalai, A. T. “Man is to computer programmer as woman is to homemaker? debiasing word embeddings”, *Advances in neural information processing systems*, vol. 29, 2016.

[5] Devlin, J.; Chang, M.-W.; Lee, K.; Toutanova, K. “Bert: Pre-training of deep bidirectional transformers for language understanding”, *arXiv preprint arXiv:1810.04805*, 2018.

[6] Eisenstein, J. “Introduction to natural language processing”. MIT press, 2019.

[7] Indurkhya, N.; Damerau, F. J. “Handbook of natural language processing”. Chapman and Hall/CRC, 2010.

[8] Júnior, C. M.; Macedo, H.; Bispo, T.; Santos, F.; Silva, N.; Barbosa, L. “Paramopama: a brazilian-portuguese corpus for named entity recognition”, *Encontro Nac. de Int. Artificial e Computacional*, 2015.

[9] Jurafsky, D.; Martin, J. H. “Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition”, 2020.

[10] Kingma, D. P.; Ba, J. “Adam: A method for stochastic optimization”, *arXiv preprint arXiv:1412.6980*, 2014.

[11] Krishnan, V.; Ganapathy, V. “Named entity recognition”, *Stanford Lecture CS229*, 2005.

[12] Lample, G.; Ballesteros, M.; Subramanian, S.; Kawakami, K.; Dyer, C. “Neural architectures for named entity recognition”, *arXiv preprint arXiv:1603.01360*, 2016.

[13] Luz de Araujo, P. H.; Campos, T. E. d.; de Oliveira, R. R.; Stauffer, M.; Couto, S.; Bermejo, P. “Lener-br: a dataset for named entity recognition in brazilian legal text”. In: International Conference on Computational Processing of the Portuguese Language, 2018, pp. 313–323.

[14] Manning, C. D.; Surdeanu, M.; Bauer, J.; Finkel, J. R.; Bethard, S.; McClosky, D. “The stanford corenlp natural language processing toolkit”. In: Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations, 2014, pp. 55–60.

[15] Mejova, Y.; Srinivasan, P. “Exploring feature definition and selection for sentiment classifiers”. In: Proceedings of the International AAAI Conference on Web and Social Media, 2011, pp. 546–549.

[16] Melamud, O.; Goldberger, J.; Dagan, I. “context2vec: Learning generic context embedding with bidirectional lstm”. In: Proceedings of the 20th SIGNLL conference on computational natural language learning, 2016, pp. 51–61.

[17] Mendonça Jr, C.; Barbosa, L. A.; Macedo, H. T.; São Cristóvão, S. “Uma arquitetura híbrida lstm-cnn para reconhecimento de entidades nomeadas em textos naturais em língua portuguesa”, *XIII Encontro Nacional de Inteligência Artificial e Computacional (ENIAC). SBC*, 2016.

[18] Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; Dean, J. “Distributed representations of words and phrases and their compositionality”, *Advances in neural information processing systems*, vol. 26, 2013.

[19] Naseem, U.; Razzak, I.; Khan, S. K.; Prasad, M. “A comprehensive survey on word representation models: From classical to state-of-the-art word representation language models”, *Transactions on Asian and Low-Resource Language Information Processing*, vol. 20–5, 2021, pp. 1–35.

[20] Peters, M.; Neumann, M.; Iyyer, M.; Gardner, M.; Clark, C.; Lee, K. “Luke” zettlemoyer. 2018. deep contextualized word representations”. In: Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies.

[21] Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I.; et al.. “Language models are unsupervised multitask learners”, *OpenAI blog*, vol. 1–8, 2019, pp. 9.

[22] Ramshaw, L. A.; Marcus, M. P. “Text chunking using transformation-based learning”. In: *Natural language processing using very large corpora*, Springer, 1999, pp. 157–176.

[23] Salinas, J. “Few-shot ner: Entity extraction without annotation and training based on gpt”. Capturado em: https://nlpcloud.io/few-shot-ner-entity-extraction-without-annotation-training-based-on-gpt.html, Mar 2022.

[24] Sang, E. F.; De Meulder, F. “Introduction to the conll-2003 shared task: Language-independent named entity recognition”, *arXiv preprint cs/0306050*, 2003.

[25] Santos, D.; Seco, N.; Cardoso, N.; Vilela, R. “Harem: An advanced ner evaluation contest for portuguese”. In: quot; In Nicoletta Calzolari; Khalid Choukri; Aldo Gangemi; Bente Maegaard; Joseph Mariani; Jan Odjik; Daniel Tapias (ed) Proceedings of the 5 th International Conference on Language Resources and Evaluation (LREC’2006)(Genoa Italy 22-28 May 2006), 2006.

[26] Sparck Jones, K. “A Statistical Interpretation of Term Specificity and Its Application in Retrieval”. GBR: Taylor Graham Publishing, 1988, pp. 132–142.

[27] Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. “Dropout: a simple way to prevent neural networks from overfitting”, *The journal of machine learning research*, vol. 15–1, 2014, pp. 1929–1958.

[28] Wikipédia. “Agricultura no brasil — wikipédia, a enciclopédia livre”. [Online; accessed 1-maio-2022], Capturado em: https://pt.wikipedia.org/w/index.php?title=Agricultura_no_Brasil&oldid=63505391, 2022.

[29] Zhu, Y.; Kiros, R.; Zemel, R.; Salakhutdinov, R.; Urtasun, R.; Torralba, A.; Fidler, S. “Aligning books and movies: Towards story-like visual explanations by watching movies and reading books”. In: Proceedings of the IEEE international conference on computer vision, 2015, pp. 19–27.