PUCRS

ESCOLA POLITÉCNICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO
MESTRADO EM CIÊNCIA DA COMPUTAÇÃO

JOAQUIM FRANCISCO DOS SANTOS NETO

RECONHECIMENTO DE ENTIDADES NOMEADAS PARA O PORTUGUÊS USANDO REDES NEURAIS

Porto Alegre
2019

PÓS-GRADUAÇÃO - *STRICTO SENSU*

Pontifícia Universidade Católica
do Rio Grande do Sul

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO GRANDE DO SUL
ESCOLA POLITÉCNICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

RECONHECIMENTO DE ENTIDADES NOMEADAS PARA O PORTUGUÊS USANDO REDES NEURAIS

JOAQUIM FRANCISCO DOS SANTOS NETO

Dissertação apresentada como requisito parcial à obtenção do grau de Mestre em Ciência da Computação na Pontifícia Universidade Católica do Rio Grande do Sul.

Orientador: Profa. Renata Vieira

Porto Alegre
2019

Ficha Catalográfica

S237r Santos Neto, Joaquim Francisco dos

Reconhecimento de entidades nomeadas para o português usando redes neurais / Joaquim Francisco dos Santos Neto . – 2019.
93 f.
Dissertação (Mestrado) – Programa de Pós-Graduação em Ciência da Computação, PUCRS.

Orientadora: Profa. Dra. Renata Vieira.

1. Reconhecimento de Entidades Nomeadas. 2. Modelos de Linguagem. 3. Redes Neurais. I. Vieira, Renata. II. Título.

Elaborada pelo Sistema de Geração Automática de Ficha Catalográfica da PUCRS
com os dados fornecidos pelo(a) autor(a).
Bibliotecária responsável: Clarissa Jesinska Selbach CRB-10/2051

Joaquim Francisco dos Santos Neto

Reconhecimento de entidades nomeadas para o português usando redes neurais

Tese/Dissertação apresentada como requisito parcial para obtenção do grau de Doutor/Mestre em Ciência da Computação do Programa de Pós-Graduação em Ciencia da Computação, Escola Politécnica da Pontifícia Universidade Católica do Rio Grande do Sul.

Aprovado em 25 de Novembro de 2019.

BANCA EXAMINADORA:

Prof. Dr. Sandro José Rigo (PPGCA/UNISINOS)

Prof. Dr. Rafael Heitor Bordini (PPGCC/PUCRS)

Profa. Dra. Renata Vieira (PPGCC/PUCRS - Orientadora)

DEDICATÓRIA

Aos meus pais e irmã.

A minhas avós Damiana e Pedrina.

A Jamylle, minha luz, minha amada.

“bela bela
mais que bela
mas como era o nome dela?
Não era Helena nem Vera
nem Nara nem Gabriela
nem Tereza nem Maria
Seu nome seu nome era...
Perdeu-se na carne fria
perdeu-se na confusão de tanta noite e tanto dia
perdeu-se na profusão das coisas acontecidas”
(Ferreira Gullar, Poema Sujo, 1975)

“Cravarei no ar pregos
avantajados e coloridos,
e suspenderei versos
ao meu amor. ”
(Fernandes Nogueira, Coisas e Algos, 2003)

AGRADECIMENTOS

Aos meus pais por todo esforço que fizeram e fazem. Por suas duras vidas de muito trabalho árduo e pouquíssimo acesso a Escola. Ainda por eles, por serem minha vontade de continuar e insistir.

À minha namorada, Jamylle Correia, pela incondicional paciência, amor e companheirismo que tem me dedicado. Por nossa longa história e pela paz que me traz.

Às minhas amadas avós, cujas as vidas foram duras e nunca tiveram oportunidade de estudar.

À minha querida irmã Juliana, pela sua força e amizade atemporal.

A todos os meus tios e tias, que sempre me apoiaram em todas as minhas jornadas e nunca mensuraram esforços para me ajudar.

À Tio Cícero por ser sempre uma inspiração de inteligência, força e dedicação. Por sempre me ajudar e, principalmente, acreditar na minha capacidade.

À Tia Marismênia por ser uma inspiração de inteligência, resistência e coragem. Por sempre cuidar tão bem de mim. Por sempre me apresentar tantos horizontes.

À minha orientadora Profa. Dra. Renata Vieira, a quem tenho maior admiração por sua inteligência, capacidade e simplicidade. Agradeço ainda por ter acreditado em minha capacidade e por ter me dado tantas oportunidades maravilhosas. Agradeço por sempre ter me orientado da melhor forma e por ser sempre tão paciente. Serei eternamente grato por fazer parte de minha vida e história.

Aos meus amigos do Grupo de PLN-PUCRS e SMART, por tantos dias compartilhados, por tantos momentos divertidos, por tantos almoços, por tantas histórias contadas, por tudo que foi vivido. Agradeço em especial a Sandrinha, Dani Schmidt, Henrique, Bernardo, Matheus, Avner e Larissa.

À Juliano, bolsista de IC e amigo que topou inúmeros desafios dessa pesquisa junto comigo.

Aos meus queridos amigos Reginaldo e Angela, pelos tantos finais de senama compartilhados, por tantas conversas sobre Arte e Filosofia, por tantas poesias recitadas, por tantos momentos únicos.

Ao CNPq pelo apoio financeiro.

RECONHECIMENTO DE ENTIDADES NOMEADAS PARA O PORTUGUÊS USANDO REDES NEURAIS

RESUMO

Abordagens modernas para o Reconhecimento de Entidades Nomeadas (REN) utilizam Redes Neurais para automaticamente extrair *features* de textos e as incorporar no processo de classificação. *Word Embeddings*, que é um tipo de Modelo de Linguagem (ML), é um ingrediente chave para melhorar a performance dos sistemas de REN. Mais recentemente, ML Contextualizados, que se adaptam de acordo com o contexto em que a palavra aparece, também se mostraram indispensáveis. Nessa dissertação, mostra-se como diferentes combinações de *Word Embeddings* e ML Contextualizados impactam na tarefa de REN em língua portuguesa. Foi explorado como a diversidade textual e o tamanho do corpus de treino usado nos ML impactam nos resultados dessa tarefa. Também, é apresentado um estudo comparativo de 16 combinações de diferentes ML entre contextualizados e *Word Embeddings*. As avaliações foram realizadas no corpus Mini-HAREM, amplamente adotado neste tema. O melhor resultado alcançado nesta pesquisa, ultrapassa a abordagem estado-da-arte em 5,99%, em um cenário de cinco categorias, e 4,31% quando são consideradas as dez categorias do HAREM. Além das avaliações no HAREM, também foram estudados domínios específicos dessa tarefa. Os resultados nestes casos, foram avaliados nos corpora de contexto Clínico, Policial e Geológico. Em todos, foram obtidos resultados superiores ou competitivos em relação a outras abordagens.

Palavras-Chave: Reconhecimento de Entidades Nomeadas, Modelos de Linguagem, Redes Neurais.

NAMED ENTITY RECOGNITION FOR PORTUGUESE USING NEURAL NETWORKS

ABSTRACT

Modern approaches to Named Entity Recognition (NER) use Neural Networks to automatically extract text *features* and incorporate them into the classification process. *Word Embeddings*, a type of Language Model (LM), are a key ingredient for improving the performance of NER systems. More recently, Contextualized LM, which adapt according to the context in which the word appears, have also proved indispensable. This master's thesis shows how different combinations of *Word Embeddings* and Contextualized LM impact the NER task in Portuguese. The impact of textual diversity and size of the training corpus used in the construction of LMs were explored by the results of this task. Also, a comparative study of 16 combinations of different LMs, contextualized and *Word Embeddings*, is presented. Evaluations were performed in the Mini-HAREM corpus, widely adopted in the Portuguese NER task. The best result achieved in this research surpasses the state-of-the-art approach by 5.99% in a five-category scenario and 4.31% when considering the ten HAREM categories. In addition to the HAREM assessments, specific domains of this task were also studied. The results in these cases were evaluated in Clinical, Police and Geological context corpora. Superior or competitive results were obtained for all corpora in relation to other approaches.

Keywords: Named Entity Recognition, Language Models, Neural Networks.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

BILSTM – *Bidirectional Long Short-Term Memory*

CD – Coleção Dourada

CHARLM – *Neural character-level language modeling*

CNN – Convolutional Neural Network

CONLL – Conference on Computational Natural Language Learning

CRF – *Conditional Random Fields*

DNN – Deep Neural Network

EN – Entidades Nomeadas

IO – Inside Outside

LG – Local Grammar

LSTM – Long Short-Term Memory

ML – Modelo de Linguagem

MTL – Multi-Task Learning

NILC – Núcleo Interinstitucional de Linguística Computacional

PLN – Processamento de Linguagem Natural

REN – Reconhecimento de Entidades Nomeadas

WE – *Word Embeddings*

SUMÁRIO

1. INTRODUÇÃO

Reconhecimento de Entidades Nomeadas (REN) é a tarefa de encontrar nomes próprios em um dado texto e classificá-los entre várias categorias de interesse ou em uma categoria padrão chamada Outros [43]. Nos últimos anos, principalmente, após o trabalho de Collobert et al. [10], que propõe o uso de redes neurais para aprender automaticamente as *features* de uma linguagem, a busca de solução para essa tarefa tem tomado abordagens que não se baseiam mais em regras linguísticas manualmente criadas [18]. Neste sentido, houve um grande impacto na capacidade dos sistemas de REN após o uso de redes neurais e Modelos de Linguagem (ML) pré-treinados como forma de vetorizar palavras na intenção de, automaticamente, a rede neural incorporar valores a este vetor e passá-los a um classificador para rotulagem final [4, 8, 10, 18, 36].

Nesse sentido, o estado-da-arte para o REN está concentrado em abordagens que se utilizam da arquitetura de rede neural BiLSTM-CRF juntamente com modelos de linguagem pré-treinados de alta representatividade, como é o caso do *Flair Embeddings* que tem provido resultados estado-da-arte para o Inglês ($F_1 = 93,09\%$) e Alemão ($F_1 = 88,32\%$) [4]. Na língua portuguesa, o estado-da-arte se concentra no uso dos conjuntos de dados anotados das Coleções Douradas do Concurso Avaliativo de Reconhecimento de Entidades Mencionadas (HAREM) [59], especificamente o Primeiro HAREM (como conjunto de treino) e Mini-HAREM (como conjunto de teste).

Os resultados mais avançados para o português também são provenientes de arquiteturas de redes neurais do tipo BiLSTM-CRF que se utilizam de modelos *Word Embeddings* previamente treinados. Os autores conseguiram uma taxa de acerto de $F_1 = 70,33\%$ (quando se usa todas as categorias do HAREM) e $F_1 = 76,27\%$ (usando um conjunto específico de categorias do HAREM).

1.1 Objetivo

Dessa forma, o objetivo desse trabalho é avaliar de forma experimental o impacto de recentes tipos de representação de linguagem *Flair Embeddings* para o Português, na tarefa de REN.

Para isso a estratégia para melhorar os resultados foi desenvolver um *Flair Embeddings* a partir de um corpus de 4,9 bilhões de tokens e combinar esse modelo com os tradicionais *Word Embeddings*, gerando representações de alta qualidade.

O trabalho tem como propósito avaliar as redes consideradas estado-da-arte (BiLSTM-CRF), porém com este novo modelo, na tarefa de REN para o domínio geral, e também em domínios especializados tais como textos clínicos, policiais e geológicos. o HAREM foi o

corpus de domínio geral utilizado, e contém as categorias padrão dessa tarefa (PESSOA, LOCAL e ORGANIZAÇÃO) e é formado por textos de vários estilos textuais. O corpus de textos clínicos e policiais são aqueles apresentados na avaliação conjunta de REN da conferência de avaliação *Iberian Languages Evaluation Forum (IberLEF)* [11] e contém a categoria PESSOA, exclusivamente. Para a área de Geologia foi realizado avaliações com o GeoCorpus-2, que apresenta um outro conjunto de entidades conforme o domínio [16].

1.2 Resultados

Essa abordagem resultou em um aumento de +4,31% na configuração que usa todas as categorias do HAREM e +5,99% para um conjunto específico de categorias do HAREM. Dessa forma, conseguiu-se superar o atual trabalho estado-da-arte.

Em suma, os experimentos e estudos nesta dissertação indicam que modelos de linguagem de alta representatividade melhoram os resultados da tarefa de REN, quando usados com redes BiLSTM *(Bidirectional Long Short-Term Memory)*. Além disso, os resultados encontrados mostram que o tamanho e a diversidade textual do corpus gerador do modelo de linguagem impactam diretamente nos resultados obtidos em tarefas de Reconhecimento de Entidades Nomeadas.

1.3 Organização do volume

Esta dissertação está dividida em oito capítulos: no capítulo dois, apresentam-se os conceitos necessários para compreensão do trabalho; no capítulo três, apresentam-se os trabalhos relacionados que tratam do problema de REN; no capítulo quatro, apresentam-se os recursos utilizados; no capítulo cinco, mostra-se como foram gerados novos modelos de linguagem para a avaliação; o capítulo seis destina-se a formalizar as redes neurais usadas; no capítulo sete, mostram-se as avaliações e discussões; por fim, no capítulo oito, apresentam-se as considerações finais.

2. CONCEITOS PRELIMINARES

Neste capítulo, serão apresentados os principais conceitos a serem entendidos antes de seguir para as relevantes abordagens usadas em soluções de REN. Primeiro serão apresentados os Modelos de Linguagem *Word Embeddings, BERT Embeddings e Flair Embeddings*. Depois são apresentadas as arquiteturas de rede neural LSTM e CNN. E por fim, é apresentado o classificador probabilístico CRF, amplamente usado em conjunto com as redes LSTM e variações.

2.1 Word Embeddings

Nos últimos anos, uma grande quantidade de trabalhos na área de Processamento de Linguagem Natural (PLN) tem sido diretamente influenciada pelo uso representação vetorial de palavras, tais representações vetoriais são conhecidas como *neural embeddings* ou *Word Embeddings* (WE) [37]. Representar palavras por vetores em um espaço vetorial tem ajudado na resolução de determinadas tarefas da área de PLN [41]. Em especial, um grupo de algoritmos, chamado *Word2Vec*, tem se destacado em termos de uso e resultados. O *Word2Vec* é uma ferramenta de código livre, baseada em redes neurais recorrentes, capaz de aprender representações vetoriais de alta dimensionalidade para palavras com base em um grande corpus [67].

Entretanto, há um considerável número de redes neurais para geração de *Word Embeddings* além do *Word2Vec*. As outras mais conhecidas são *FastText* [26], *Glove* [47] e *Wang2Vec* [39]. O treinamento de um modelo *WE* é do tipo não supervisionado. Isso significa que o copus usado para gerar esse tipo de modelo não necessita de nenhum tipo de anotação sintática, semântica, relacionamentos ou qualquer outro tipo de anotação. Em geral, esse tipo de ML consome muito texto durante o treinamento, sendo normalmente usado um corpus com mais de 1 bilhão de tokens.

Segundo Mikolov et al. [42], modelos *WE* são capazes de aprender estruturas sintáticas e semânticas. A figura 2.1 mostra esse tipo de relação aprendida de forma não supervisionada através do *Word2Vec*.

Recentes trabalhos sobre REN têm apresentado abordagens e resultados potenciais fazendo uso de modelos *Word Embeddings* pré-treinados, responsáveis por transformar as palavras em vetores. Nos trabalhos relacionados, capítulo 3, serão detalhadas as estratégias que usam *WE* como representações de entrada.

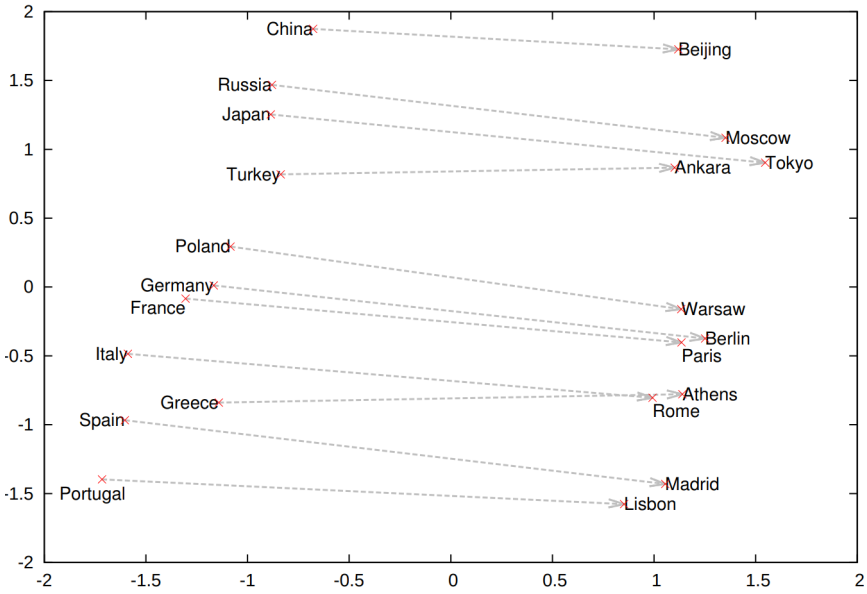


Figura 2.1 – Relações entre país e capital. Estraída de [41]

2.2 BERT Embeddings

Bidirectional Encoder Representations from Transformers (BERT) [15] é um recente modelo de representação de linguagem, que tem alcançado resultados estado-da-arte em várias tarefas de PLN, como para REN, SWAG (tarefa de escolher a continuação de uma frase dentre quatro opções), SQuAD (tarefa que objetiva prever o espaço de resposta para uma dada pergunta e um dado texto que contém a resposta), entre outras tarefas.

O treinamento de um modelo BERT envolve duas tarefas: *Masked LM* e *Next Sentence Prediction*. Na primeira tarefa (*Masked LM*), o objetivo é "mascarar" e prever a palavra mascarada. Assim, BERT mascara até 15% das palavras de uma sentença. Isso significa que 15% dos tokens de uma frase serão substituídos pelo símbolo **[MASK]**. O modelo então tenta predizer o token original, levando em conta o contexto das outras palavras não ocultadas da sequência. Há ainda algumas regras de mascaramento:

- 80% das vezes: a palavra é substituída pelo símbolo [MASK], por exemplo:

  `meu cachorro é cabeludo` $\rightarrow$ `meu cachorro é [MASK].`

- 10% das vezes: a palavra é substituída por uma palavra aleatória, por exemplo:

  `meu cachorro é cabeludo` $\rightarrow$ `meu cachorro é maçã.`

- 10% das vezes: a palavra é mantida, por exemplo:

  `meu cachorro é cabeludo` $\rightarrow$ `meu cachorro é cabeludo.`

Os autores do BERT acreditam que este tipo de treinamento é o que permite maior poder de representação com relação ao treinamento de um modelo *Word Embeddings* tradicional. Por exemplo: em um modelo *Word Embeddings*, na arquitetura *CBOW*, o treinamento ocorre pela predição de uma palavra alvo:

`O gato senta no ___???___`

Já em um modelo BERT, a predição ocorre em todos os termos mascarados, por exemplo:

`Após às artes [MASK] , a escola em ascensão foi a realista [MASK] idéias .`

A segunda tarefa (*Next Sentence Prediction*), no processo de treinamento de um modelo BERT, recebe como *input* pares de sentenças (s_1, s_2), em que o modelo deve predizer se a sentença s_2 é a subsequente a s_1. Durante o treinamento, 50% das entradas são um par em que s_2 é, de fato, a sentença subsequente, enquanto nos outros 50% uma sentença aleatória do corpus é escolhida para ser s_2. A figura 2.2 demonstra visualmente a segmentação de uma frase em sentenças `A` e `B`.

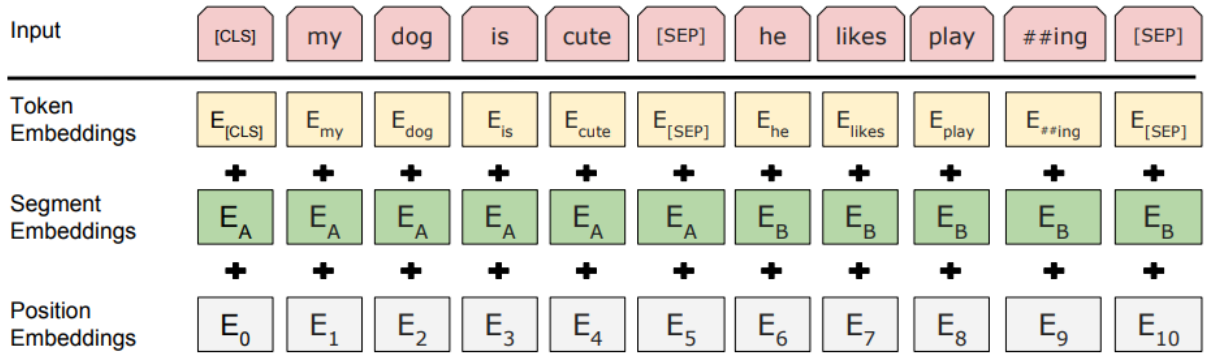


Figura 2.2 – Camadas de incorporação que formam o input do BERT. Extraída de [15]

Os autores de BERT exemplificam esta fase de treinamento com as sentenças:

- Input = `[CLS] the man to [MASK] store [SEP] he bought a gallon [MASK] milk [SEP]`
- Label = `IsNext`
- Input = `[CLS] the man to [MASK] store [SEP] penguin [MASK] are flight ##less birds [SEP]`
- Label = `NotNext`

No exemplo, há rótulos como [CLS] e [SEP], eles servem para ajudar no processo de segmentação das sentenças. O símbolo [CLS] é o primeiro token da sentença, marcando

seu início. O símbolo [SEP] marca a segmentação da sentença, ou seja, após o [CLS] o primeiro [SEP] determina a sentença `A` e um segundo [SEP] marca o final da sequência e também o final da sentença `B`. Esse processo pode ser visto na figura 2.2. A Rede Neural responsável por realizar as tarefas *Masked LM* e *Next Sentence Prediction* é composta por várias camadas do neurônio *Transformer* [63].

2.3 Flair Embeddings

Flair Embeddings é um novo modelo de *embedding* que permite a linguagem ser modelada, levando em conta a distribuição das sequências de caracteres em vez de palavras (como faz as abordagens para treinar modelos como *Skip-Gram, CBOW...*). Ou seja, *Flair Embeddings* é um modelo em nível de palavra, mas cuja geração não só depende do contexto das palavras vizinhas, mas também do nível de caractere das palavras vizinhas. Neste sentido, os autores se referem ao *Flair Embedding* como um modelo de linguagem contextualizado.

Uma característica singular dos ML contextualizados é a capacidade de contornar situações que envolvem palavras polissêmicas. Como os modelos são gerados a partir dos caracteres, é possível gerar novas representações para uma palavra sempre que algum caractere muda, inclusive pela capitalização empregada. Evidentemente, as representações mudam diante do contexto em que a palavra está inserida. A figura 2.3 ilustra esse processo de mudança de representação. Primeiro, tem-se uma única representação tensorial para a palavra “Mangueira” (com a primeira letra maiúscula). Depois, dois sucessivos empregos da palavra “mangueira” aparecem (com a capitalização trocada), todas com uma nova representação para a palavra. Por último, tem-se a aplicação da palavra original (com a primeira letra maiúscula), mas dessa vez aplicada em uma frase, ou seja, dentro de um contexto e, portanto, recebe uma nova representação.

A geração de modelos contextualizados é realizada a partir de uma rede neural profunda chamada *Neural character-level language modeling (CharLM)*, que aprende as representações das palavras. A figura 2.4 ilustra como o tensor do vocábulo “Washington” é extraído levando em consideração o contexto em que a palavra está inserida no nível de palavra e caractere. *CharLM* é formada principalmente por uma rede LSTM que age em um sentido, concatenando os estados ocultos de um determinado intervalo. Na figura 2.4, em vermelho, na direção *forward*, a rede neural concatena informações desde o primeiro caractere da sentença até o último caractere da palavra “Washington”. Em azul, na direção *backward*, a informação é concatenada desde o início da palavra "Washington"até o final da frase.

Uma vez estes modelos gerados, podem ser carregados em uma rede neural para funcionarem como representações das palavras. Ou seja, em vez de se trabalhar especifi-

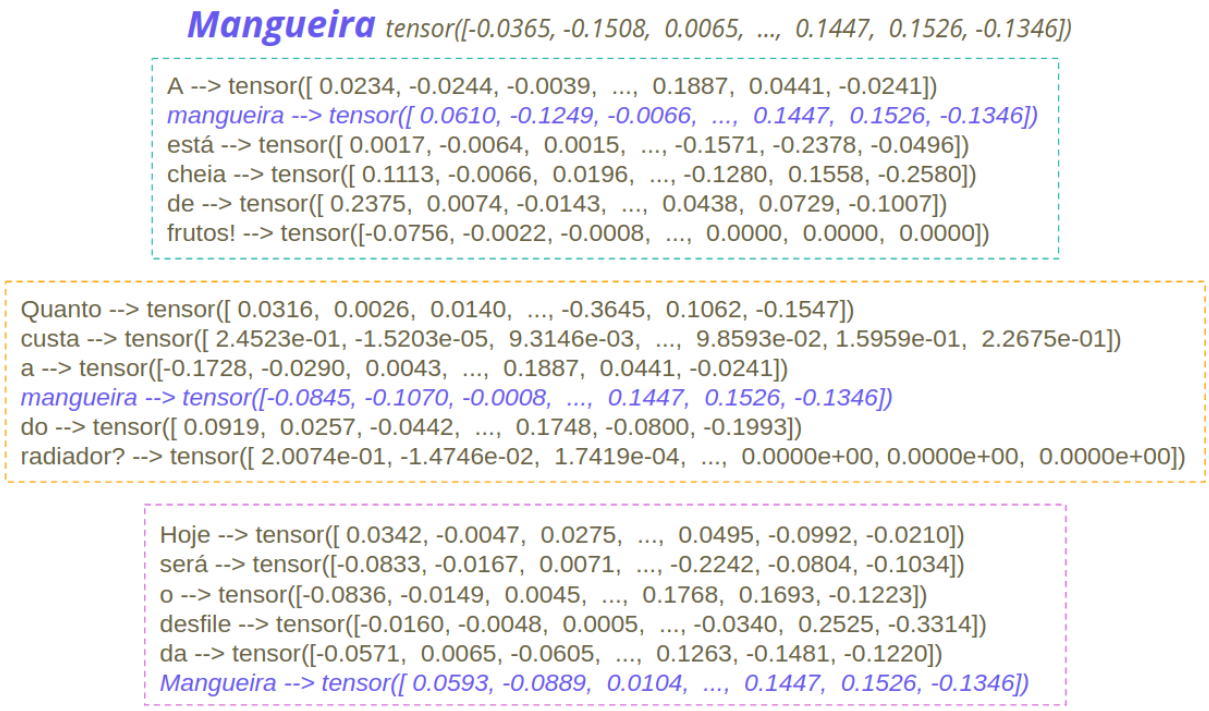


Figura 2.3 – Representações tensoriais usando *Flair Embeddings*

camente com a palavra, trabalha-se com a representação vetorial ou tensorial da mesma. A seção 6.1, trata-se, rigorosamente, da arquitetura da geração destes *embeddings*.

2.4 Redes Long Short-Term Memory (LSTM)

Redes Neurais Recorrentes (do inglês *Recurrent Neural Network - RNN*) é uma ferramenta padrão para tarefas de PLN [45]. Mais especificamente, *Long Short-Term Memory (LSTM)* é um tipo de Rede Neural Recorrente, com uma estrutura computacional complexa, que tem tido sucesso na resolução de tarefas sequenciais [62]. A seguir, uma descrição de como as redes LSTM funcionam, segundo Tai et al. [62] e Hochreiter et al. [29].

Sejam x_t um vetor de entrada, h_t um vetor de estado oculto e t o passo de tempo atual. Define-se h_t como a função do vetor de entrada x_t que a rede recebe em um tempo t. Logo, seu estado oculto anterior é h_{t-1}. Quando não houver dúvidas, trata-se de vetores, que naturalmente estão contidos em um espaço vetorial de dimensão n. Assim, $h_t \in \mathbb{R}^n$.

Define-se uma unidade LSTM, cada passo de tempo t como uma coleção de vetores contidos num espaço $\mathbb{R}^n$. A coleção é formada por um vetor *input gate* i_t, um *forget gate* f_t, um *output gate* o_t, uma *memory cell* c_t e um estado oculto h_t.

Primeiro, decide-se qual informação será armazenada pela unidade, para isso usa-se uma camada sigmoide, que decide quais valores serão atualizados, tal camada retorna um vetor i_t, definido por:

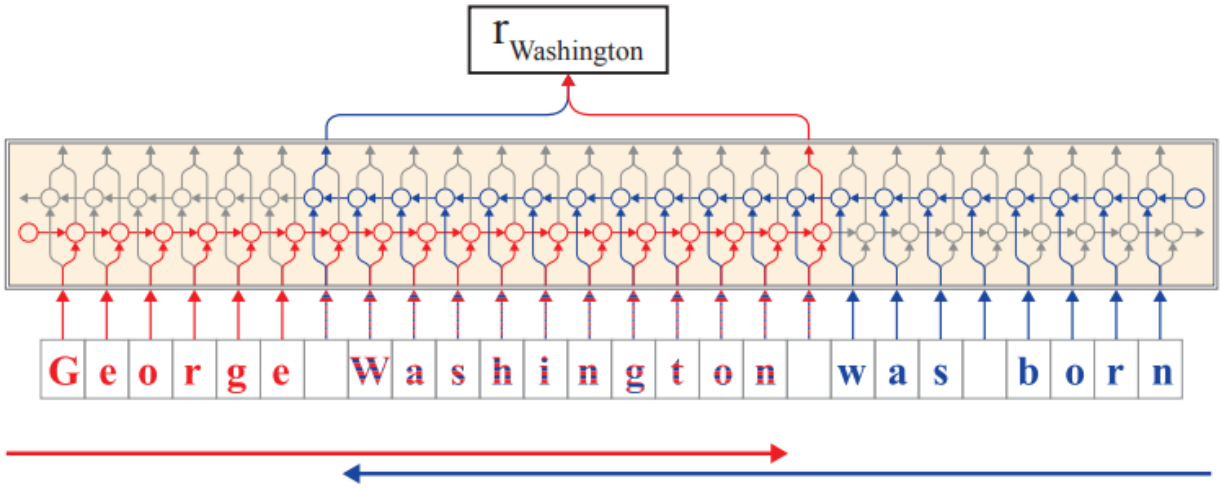


Figura 2.4 – Ilustração da geração do *embedding* “Washington” no nível de palavra e caractere. Extraída de [4]

$$i_t = \sigma(W_i x_t \oplus U_i h_{t-1} \oplus b_i)$$

Agora, será decidida qual informação será descartada da rede, para isso uma camada sigmoide age sobre os vetores de estado oculto h_{t-1} e o vetor de entrada x_t, retornando um vetor f_t.

$$f_t = \sigma(W_f x_t \oplus U_f h_{t-1} \oplus b_f)$$

Ainda neste estágio, uma camada de tangente hiperbólica retorna um vetor candidato a novos valores da unidade, esse vetor é chamado de u_t e definido por:

$$u_t = tgh(W_u x_t \oplus U_u h_{t-1} \oplus b_u)$$

Nesse passo, atualiza-se o estado da unidade com base nas saídas anteriores. Assim, define-se um vetor c_t da seguinte forma:

$$c_t = i_t \odot u_t \oplus f_t \odot c_{t-1}$$

Finalmente, é preciso produzir uma saída baseada no estado anterior c_{t-1} (note que, neste momento, se leva em consideração informações passadas, o que torna essa arquitetura propícia para entendimento de contexto), então aplica-se uma camada sigmoide para tomar a decisão de quais partes da unidade serão atualizadas produzindo um vetor o_t. Uma outra camada de tangente hiperbólica recebe o estado c_t e multiplica pelo vetor o_t resultando no vetor de saída h_t:

$$o_t = \sigma(W_o x_t \oplus U_o h_{t-1} \oplus b_o)$$

$$h_t = o_t \oplus tgh(c_t)$$

Em suma, a estrutura complexa de uma LSTM funciona a partir da ativação de algumas funções, decidindo em alguns momentos se deve manter, alterar ou descartar alguma informação anterior para relacionar com uma informação atual. Esses engenhosos processos, que levam em consideração um estado passado para relacionar com um estado atual, demonstram a direta aplicação dessas redes para tarefas de PLN.

Não menos importante, existe uma variante das redes LSTM chamada *Bidirectional LSTM (BiLSTM)*. As redes BiLSTM consistem em duas LSTM que funcionam em paralelo. Assim, quando dada uma sequência de entrada, uma LSTM percorre tal sequência em uma direção - digamos para frente *(Forward LSTM)* - enquanto a outra LSTM percorre a direção inversa - para trás *(Backward LSTM)* [27]. Uma estrutura bidirecional permite que o vetor de estado oculto capture informações passadas e futuras, dando maior poder de aprendizado para a rede [27].

2.5 Redes Convolucionais para classificação de texto

Redes Neurais Convolucionais (do inglês *Convolutional Neural Networs - CNN*) é uma rede neural originalmente criada para a área de visão computacional, porém elas têm alcançado resultados estado-da-arte na resolução de tarefas de PLN [34]. São exemplos de trabalhos da área de PLN que usam CNN como abordagem principal: Santos et al. [58] alcançaram resultados estado-da-arte para tarefa de *POS Tagging*; também Zhang et al. [68] fizeram diversos experimentos com grandes corpora, mostrando a eficiência desta rede para classificação de texto; as CNN também têm sido usadas para aprender estruturas semânticas para serem usadas em buscas [61].

Zhang et al. [69] apresenta toda a arquitetura e funcionamento de uma Rede Neural Convolucional para classificação de texto. A ilustração 2.5 exibe as várias camadas de processamento que é feito pela rede, começando pelo recebimento do *token* até a saída binária.

Após uma tokenização das sentenças, cada token será representado por um vetor. Pode-se fazer isso usando um modelo *Word Embeddings* apresentado anteriormente. Agora, sendo s a quantidade de *tokens* que contém uma sentença, tem-se uma matriz: a matriz A de dimensão $s \times d$. Essa matriz pode ser vista na figura 2.5, em que a sentença tem comprimento sete e a dimensão dos vetores é cinco, logo uma matriz 7×5.

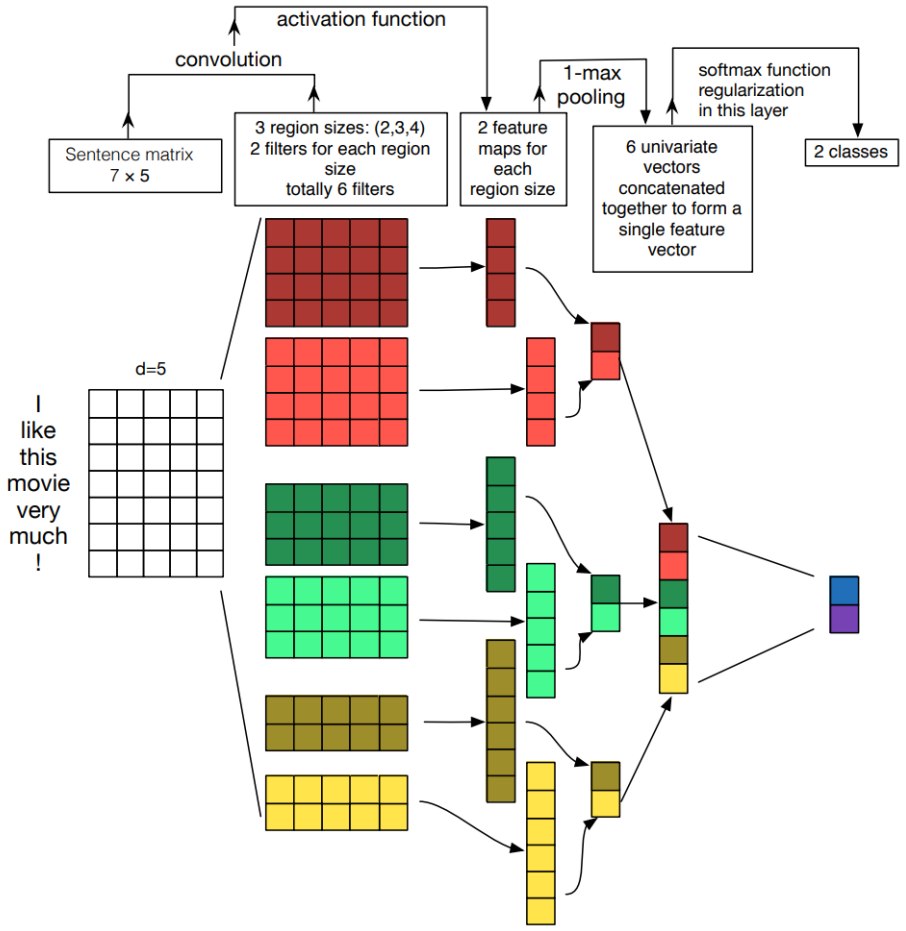


Figura 2.5 – Arquitetura de uma CNN. Extraído de [69]

Após formar a matriz de sentenças A, passa-se para os filtros, estruturas responsáveis por realizar as convoluções em A e gerar os *features maps*. Segundo Zhang et al. [69] é razoável usar um filtro com a mesma dimensão dos vetores *Word Embedding*. Por exemplo, na figura 2.5 há três filtros duplos com tamanhos 2, 3 e 4. Assim, quando dados a matriz de sentenças $A \in \mathbb{R}^{s \times d}$ e uma sequência de operações convolucionais $o \in \mathbb{R}^{s-h+1}$ será obtida pela repetida aplicação dos filtros nas submatrizes de A:

$$\mathbf{o_i} = \mathbf{w} \cdot \mathbf{A}[i : i + h - 1]$$

Assim, w é a matriz de pesos que está parametrizada pelo filtro; $\cdot$ é o produto escalar das submatrizes $\mathbf{A}[i : j]$. Um termo bias $b \in \mathbb{R}$ e uma função de ativação f são adicionados para cada o_i, produzindo o vetor *feature map* $c \in \mathbb{R}^{s-h+1}$:

$$\mathbf{c_i} = f(o_i + b)$$

Os *features maps* gerados têm tamanhos variados e para que tenham tamanhos fixos uma função é aplicada. A função mais conhecida é a *1-max pooling* [6], que extrai um escalar de cada *feature map*. Na figura 2.5, pode-se ver os variáveis tamanhos das saídas

dos filtros e em seguida a camada de *1-max pooling* criando vetores de tamanho padrão. Uma vez criados, os vetores da camada *1-max* passam para a camada de concatenação responsável por unir os vetores anteriores e em seguida para a camada de *softmax*, a fim de retornar a classificação final.

Deve-se notar que a engenhosidade e as várias camadas das CNN possibilitam que a rede aprenda automaticamente informações (*features*) de um determinado corpus. Note que quando são concatenados os vetores da camada *1-max* gera-se um vetor de *features* que já foi extraído automaticamente.

2.6 Conditional Random Fields

Conditional Random Fields (CRF) é uma estrutura matemática usada para construir modelos probabilísticos objetivando segmentar e rotular dados sequenciais, segundo Lafferty et al. [35]. Em outras palavras, o CRF pode ser entendido como um classificador probabilístico. Esse tipo de classificador tem sido amplamente usado em tarefas de REN e *Part-Of-Speech Tagging* [4, 17, 50, 66]. Isso acontece porque o CRF é capaz de receber sequências de elementos e calcular uma classe para cada elemento sem assumir a independência de cada elemento, ou seja, o CRF leva em conta as relações entre os elementos da sequência.

Por conseguinte, pode-se apresentar formalmente a definição de um classificador CRF por: sejam $\bar{x}$ e $\bar{y}$ sequências de palavras e rótulos, respectivamente. E sejam ainda $x \in \bar{x}$ e $y \in \bar{y}$, se quer-se calcular a probabilidade de y acontecer dado x, pode-se usar a probabilidade condicional $P(y|x;w)$ com base em um parâmetro w. Assim, define-se:

$$P(y|x;w) = \frac{1}{Z(x,w)} \cdot e^{\sum_{j=1}^{J} w_j F_j(x,y)} \tag{2.1}$$

Na equação 2.1 $F_j(x,y)$ é a *Feature Function(FF)*, uma medida de compatibilidade entre o exemplo x e o rótulo y. É fácil ver que o termo w_j influencia o valor da FF:

- Quando $w_j > 0$, faz y ser um rótulo mais provável de x;
- Quando $w_j < 0$, faz y ser um rótulo menos provável de x;
- Quando $w_j = 0$, significa que F_j é irrelevante para predizer y;

Ainda na equação 2.1, $Z(x,w)$ é a constante de normalização (também pode ser chamada de fator de normalização), definida como:

$$Z(x, w) = \sum_{y' \in Y} e^{\sum_{j=1}^{J} w_j F_j(x, y')} \tag{2.2}$$

Um classificador CRF está bem definido quando sua *FF* está descrita por:

$$F_j(\bar{x}, \bar{y}) = \sum_{i=1}^{n} f_j(y_{i-1}, y_i, \bar{x}, i) \tag{2.3}$$

Note que n mede exatamente a sequência $\bar{y}$, pois F_j é a soma sobre todos os rótulos de uma sequência $\bar{y}$. Por exemplo:

$$\begin{array}{lllllll} \bar{x} & = & [& \text{Bolero} & \text{de} & \text{Ravel} &] \\ \bar{y} & = & [& \text{O} & \text{O} & \text{B-PER} &] \end{array}$$

Tendo em vista $\bar{x}$ e $\bar{y}$, segue que $F_j = \sum_{i=1}^{3} f_j(y_{i-1}, y_i, \bar{x}, i)$. Agora, pode-se reescrever a equação 2.1 como um classificador CRF:

$$P(\bar{y}|\bar{x}; w_j) = \frac{1}{Z(x, w)} e^{\sum_{j=1}^{J} w_j \sum_{i=1}^{n} f_j(y_{i-1}, y_i, \bar{x}, i)} \tag{2.4}$$

Sem perda, treinar um classificador CRF significa encontrar o parâmetro w que melhor prediz y. Significando, ainda, que estima-se uma sequência de rótulos $\bar{y}$. Para melhor estimar $\bar{y}$ deve-se encontrar valores w_j de modo que eles maximizem $P(\bar{y}|\bar{x}; w)$. Essa estimativa pode ser feita tomando as derivadas parciais de $P(\bar{y}|\bar{x}; w)$. Para um simples exemplo $P(y|x; w)$, aplicam-se as derivadas parciais:

$$\begin{aligned} \frac{\partial}{\partial w_j} \ln P(y|x; w) &= \frac{\partial}{\partial w_j} \ln \frac{1}{Z(x, w)} e^{\sum_j w_j F_j(x, y)} \\ &= \frac{\partial}{\partial w_j} \ln e^{\sum_j w_j F_j(x, y)} - \frac{\partial}{\partial w_j} \ln Z(x, w) \\ &= \underbrace{\frac{\partial}{\partial w_j} \sum_j w_j F_j(x, y)}_{(*_1)} - \underbrace{\frac{\partial}{\partial w_j} \ln Z(x, w)}_{(*_2)} \end{aligned}$$

Derivando $(*_1)$ tem-se:

$$\begin{aligned}\frac{\partial}{\partial w_j}\sum_j w_j F_j(x,y) &= F_j(x,y)\frac{\partial}{\partial w_j}\sum_j w_j \\ &= F_j(x,y)\sum_j \frac{\partial}{\partial w_j} w_j \\ &= F_j(x,y)\end{aligned}$$

Agora, derivando $(*_2)$, vem:

$$\frac{\partial}{\partial w_j}\ln Z(x,w) = \frac{1}{Z(x,w)}\frac{\partial}{\partial w_j}Z(x,w)$$

Portanto,

$$\frac{\partial}{\partial w_j}\ln P(y|x;w) = F_j(x,y) - \frac{1}{Z(x,w)}\underbrace{\frac{\partial}{\partial w_j}Z(x,w)}_{(*_3)}$$

Novamente, calcula-se $(*_3)$ separadamente:

$$\begin{aligned}\frac{\partial}{\partial w_j}Z(x,w) &= \frac{\partial}{\partial w_j}\sum_{y'} e^{\sum_{j'} w_{j'}F_{j'}(x,y')} \\ &= \sum_{y'}\frac{\partial}{\partial w_j} e^{\sum_{j'} w_{j'}F_{j'}(x,y')} \\ &= \sum_{y'} e^{\sum_{j'} w_{j'}F_{j'}(x,y')}\cdot\frac{\partial}{\partial w_j}\sum_{j'} w_{j'}F_{j'}(x,y') \\ &= \sum_{y'} e^{\sum_{j'} w_{j'}F_{j'}(x,y')}\cdot F_{j'}(x,y')\end{aligned}$$

Logo,

$$\begin{aligned}\frac{\partial}{\partial w_j}\ln P(y|x;w) &= F_j(x,y) - \frac{1}{Z(x,w)}\cdot\sum_{y'} e^{\sum_{j'} w_{j'}F_{j'}(x,y')}\cdot F_j(x,y') \\ &= F_j(x,y) - \sum_{y'} F_j(x,y')\cdot\frac{e^{\sum_{j'} w_{j'}F_{j'}(x,y')}}{Z(x,w)}\end{aligned}$$

Ora, mas,

$$\frac{e^{\sum_{j'} w_{j'} F_{j'}(x,y')}}{Z(x,w)} = P(y'|x;w)$$

Finalmente,

$$\frac{\partial}{\partial w_j} \ln P(y|x;w) = F_j(x,y) - \sum_{y'} F_j(x,y') \cdot P(y'|x;w) \tag{2.5}$$

Por fim, a equação 2.5 mostra que os valores de w_j que maximizam a probabilidade $P(y|x;w)$ são o valor da *FF* com respeito a (x,y) menos a soma dos produtos entre as funções *FF* com relação a (x,y') (onde y' é um rótulo qualquer) e a probabilidade desse rótulo qualquer acontecer dado a palavra x com base nos parâmetros w [21].

3. TRABALHOS RELACIONADOS

O objetivo desse capítulo é apresentar os avanços nas soluções empregadas para o REN. Esse capítulo está organizado em duas seções de estratégias: as que usam abordagens clássicas, com base em regras linguísticas manualmente criadas, e as abordagens mais modernas que usam Redes Neurais. Nesta perspectiva, a seção 3.1 apresenta os trabalhos com abordagens baseadas em regras e a seção 3.2 apresenta as abordagens por meio de Redes Neurais.

A escolha dos trabalhos de REN em Português se deu pelos últimos trabalhos mais influentes para a área e que compartilhavam de recursos similares aos usados na abordagem proposta nesta pesquisa, visando fins de comparação. No caso do inglês, por haver uma grande quantidade de trabalhos com diferentes abordagens, focou-se nas metodologias que empregam redes neurais LSTM com o classificador CRF ou ainda aqueles que usam ML Contextualizados.

Outros trabalhos aparecem no decorrer do texto tendo em vista comparações com os resultados alcançados nesta dissertação. Esses trabalhos não foram incluídos nesta seção por não influenciar diretamente na definição da abordagem seguida, pois foram publicados após a sistematização das metologias seguidas.

3.1 Abordagens baseadas em Regras Linguísticas

Aqui serão apresentados dois trabalhos relacionados ao REN em Português, que fazem uso de abordagens linguísticas: a ferramenta NERP-CRF [17] e a ferramenta CRF+LG [50].

No trabalho de Amaral et al. [17] foi desenvolvida uma ferramenta de REN chamada NERP-CRF, que reconhece entidades para o Português Brasileiro e Europeu. O NERP-CRF gera um modelo capaz de reconhecer as entidades nomeadas em um dado texto. O modelo preditivo é gerado em duas fases: treino e teste. Para a fase treino, é tomado um corpus etiquetado com uma marcação POS *(Part-of-Speech)*; em seguida, todas as entidades são marcadas com a notação BILOU e gerado um vetor de *features* que será dado como entrada para o classificador probabilístico CRF.

Na sequência, etapa de teste, um outro dado conjunto de textos anotados é passado ao NERP-CRF. Entaõ um vetor de POS e um vetor de *features* são criados pela ferramenta e enviados para o CRF que, com base na fase anterior, classifica as novas entidades. Como retorno, o NERP-CRF apresenta as métricas de avaliação: precisão, abrangência e medida-F. Neste trabalho, foram realizados dois testes: o *teste 1*, configurado pelo corpus da Coleção Dourada (CD) do Segundo HAREM, tanto para o treino como para o teste,

usando a técnica de validação cruzada; o *teste 2*, configurado pelo uso do corpus da CD do Primeiro HAREM para fase de treino e o corpus da CD do Segundo HAREM na fase de teste. A tabela 3.1 apresenta os resultados do NERP-CRF.

Tabela 3.1 – Tabela de resultados do NERP-CRF por tipo de teste

Tipo	Treino	Teste	Precisão	Abrangência	F1
Teste 1	I HAREM	II HAREM	83,48%	44,35%	57,92%
Teste 2	II HAREM	II HAREM	80,77%	34,59%	48,43%

Uma abordagem semelhante foi desenvolvida no trabalho de Pirovani et al. [50], em que são apresentados os resultados de um sistema para reconhecimento de EN em português, com base em Gramáticas Locais. Segundo Williams et al. [65] as Gramáticas Locais (do Inglês *Local Grammar - LG*) são formas de agrupar ou capturar expressões de um certo domínio, de tal modo que esse domínio possui características comuns, sejam elas sintáticas ou semânticas. Segundo Pirovani et al. [50] o sistema CRF+LG obtém a rotulação das entidades por um CRF Linear e a classificação dessa entidade pelas LG.

O CRF+LG é capaz de gerar um modelo de CRF que é usado para as avaliações da ferramenta. Para gerar esse modelo, primeiro são passados os arquivos de texto a serem aprendidos, que por sua vez são segmentados em sentenças. Na sequência, todas as marcações do corpus são substituídas pela notação IO. Análogo ao NERP-CRF, o CRF+LG faz uso de um conjunto de dezoito *features* que são adicionadas para cada token do corpus juntamente com a classe da EN atribuída pela LG. Essas informações são passadas para o classificador CRF, que naturalmente retorna uma saída, ou seja, um rótulo para aquela entidade e por fim é gerado um modelo a partir do corpus aprendido pelo mecanismo.

Para a fase de testes do sistema, os procedimentos são praticamente os mesmos, exceto pelo corpus de entrada que não contém marcação de EN. Assim, a predição das EN ocorre com base no modelo treinado. No sentido de avaliar o sistema desenvolvido, foram realizados dois tipos de testes para extração das métricas de avaliação (precisão, abrangência e F1). O *teste 1:* usa a CD do Primeiro HAREM para treinamento e geração do modelo CRF e a CD do Segundo HAREM para realizar os testes. O *teste 2* usa o Primeiro HAREM para treino e o Mini-HAREM para teste, entretanto somente cinco categorias são consideradas: PESSOA, LOCAL, ORGANIZAÇÃO, TEMPO E VALOR. A tabela 3.2 exibe os resultados dos respectivos testes.

Tabela 3.2 – Tabela de resultados do CRF+LG por tipo de teste

Teste	Treino	Teste	Precisão	Abrangência	F1
Teste 1	I HAREM	II HAREM	65,46%	51,75%	57,80%
Teste 2	I HAREM	MiniHAREM	67,09%	54,85%	60,36%

3.2 Abordagens baseadas em Redes Neurais

No trabalho de dos Santos et al. [18] foi criada uma Rede Neural Profunda chamada CharWNN, usando Redes Neurais Convolucionais para resolver REN nos idiomas Português e Espanhol. CharWNN é uma arquitetura de Rede Neural Profunda (do inglês *Deep Neural Network - DNN*) que usa nível de palavra e caractere para classificação sequencial. A classificação é feita a partir de informações capturadas por uma camada convolucional, capaz de extrair padrões das palavras no nível de caractere. A primeira camada da CharWNN cria os vetores de palavras V^{wrd}, agindo como uma camada de *embeddings*; os autores consideram que as palavras são compostas de caracteres que, por sua vez, estão contidos em um vetor fixo V^{chr}; esses vetores desempenham papel fundamental no decorrer da aprendizagem da rede, pois eles capturam as informações morfológicas, sintáticas, e semânticas das palavras.

Assim, quando dada uma sentença S consistindo de N palavras $\{w_1, w_2, ..., w_n\}$, todas elas são convertidas em um vetor $u_n = [r^{wrd}, r^{wch}]$. O vetor u_n é composto por dois subvetores: r^{wrd} e r^{wch} que são responsáveis por:

$$\begin{cases} r^{wrd} \in \mathbb{R}^{d^{wrd}} & \textit{captura as informações sintáticas e semânticas em nível de palavra} \\ r^{wch} \in \mathbb{R}^{cl_u} & \textit{captura as informações morfológicas em nível de caractere} \end{cases}$$

Logo, uma série de procedimentos matemáticos (internos à rede) são realizados para gerar pontuações e pesos para cada palavra aprendida e, no caso da predição, o algoritmo de Viterbi [64] é usado para inferir a classe do *token*.

Para o treinamento e teste da rede foram usados, para o caso do Português, o corpus do I HAREM, na fase de treino, e o MiniHAREM, na fase de teste da rede. No idioma Espanhol, foi utilizado o corpus do SPA CoNLL-2002 para treino e teste. A tabela 3.3 apresenta os resultados obtidos pela CharWNN.

Tabela 3.3 – Tabela de resultados da CharWNN por idioma

Idioma	**Treino**	**Teste**	**Cenário Total**			**Cenário Seletivo**		
			Pre.	**Abr.**	**F1**	**Pre.**	**Abr.**	**F1**
PT/BR	I HAREM	MiniHAREM	74,54%	68,53%	71,41%	78,38%	77,49%	77,93%
SPA	CoNLL-02	CoNLL-02	-	-	-	82,21%	82,21%	82,21%

Na tabela 3.3 os resultados estão categorizados por dois cenários: total e seletivo. No cenário total, o treinamento e teste são com todas as dez categorias da CD do Primeiro HAREM; por outro lado, no cenário seletivo, tem-se as categorias PESSOA, LOCAL,

ORGANIZAÇÃO, TEMPO e VALOR. No espanhol, são consideradas apenas as categorias PESSOA, LOCAL, ORGANIZAÇÃO e OUTROS.

Um outro trabalho para o Português é o de Castro et al. [13], que também usa aprendizado em nível de palavra e caractere. Primeiro, uma camada de *embeddings* pré-treinados transforma as palavras em vetores. Os vetores são levados para as camadas de BiLSTM para extração de informações adicionais. Assim como Chiu et al. [8], os autores usam a arquitetura BiLSTM para melhor entendimento do contexto em que o token se encontra. Cada token de uma sentença é passado para o nível de caractere, em que novamente uma BiLSTM extrai vetores de informações. Por fim, uma camada final de CRF recebe os vetores extraídos e faz a predição do rótulo para o token. Atualmente este trabalho é o estado-da-arte para o Reconhecimento de Entidades Nomeadas em Português. A tabela 3.4 apresenta os resultados obtidos.

Tabela 3.4 – Tabela de resultados da rede LSTM-CRF

Treino	**Teste**	**Cenário Total**			**Cenário Seletivo**		
		Pre.	**Abr.**	**F1**	**Pre.**	**Abr.**	**F1**
I HAREM	Mini-HAREM	72,78%	68,03%	70,33%	78,26%	74,39%	76,27%

Chiu et al. [8] também usaram redes convolucionais para a tarefa de REN em inglês, porém os autores adicionaram mais uma arquitetura de rede neural: as BiLSTM. Para os autores, tarefas de marcação sequencial, como é o caso do REN, uma arquitetura BiLSTM, pode levar em conta uma quantidade significativa de contexto em ambas as direções de um token, o que dá maior poder de desambiguação e classificação. Uma das principais contribuições deste trabalho é a proposta de uma arquitetura de rede que combina as LSTM e CNN, tornando o modelo capaz de extrair automaticamente padrões linguísticos em nível de palavra e de caractere.

Quando a rede recebe as sentenças tokenizadas na camada de entrada, elas passam por uma camada de *embeddings* que faz a representação vetorial das palavras, então uma camada adicional em nível de palavra adiciona informações como capitalização e léxico. A primeira regra examina se o token está em caixa alta, caixa alta somente na primeira letra, caixa baixa ou misturado; já a segunda regra age como um recurso externo à rede, como uma lista de EN que foram retiradas da DBpedia, tais entidades são das categorias PESSOA, LOCAL, ORGANIZAÇÃO.

Em seguida, uma camada convolucional extrai automaticamente, como na CharWNN, outras informações em nível de caractere, gerando um vetor em nível de palavra e caractere, tal vetor é passado para as camadas de BiLSTM. Nesse sentido, uma camada *Forward LSTM* compreende uma parte do contexto à direita do token, enquanto a camada *Backward LSTM* compreende o contexto do lado esquerdo ao token. Finalmente, a camada de saída da rede é composta por um decodificador, responsável por passar os valores das duas

direções (*Forward LSTM* e *Backward LSTM*) para a função *softmax* para cálculo das probabilidades do rótulo. A tabela 3.5 apresenta os resultados organizados por modelo de rede e corpus de avaliação.

Tabela 3.5 – Resultados da rede LSTM-CNN por corpus e modelo de teste

Modelo	**CoNLL-2003**			**OntoNotes 5.0**		
	Pre.	**Abr.**	**F1**	**Pre.**	**Abr.**	**F1**
BLSTM	80,14%	72,81%	76,29%	79,68%	75,97%	77,77%
BLSTM-CNN	83,48%	83,28%	83,38%	82,58%	82,49%	82,53%
BLSTM-CNN + emb	90,75%	91,08%	90,91%	85,99%	86,36%	86,17%
BLSTM-CNN + emb + lex	**91,39%**	**91,85%**	**91,62%**	**86,04%**	**86,53%**	**86,28%**

Como apresentado, os trabalhos que envolvem redes neurais se utilizam de informações em nível de palavra e caractere para o REN. Em geral, os vetores com padrões de informações dos caracteres são gerados durante a execução do treinamento. Já no caso do nível de palavra, há dois momentos de incorporação: o primeiro acontece nas camadas de *embeddings*, que antes do treinamento iniciar, foram carregadas com modelos *Word Embeddings* pré-treinados; o segundo momento acontece durante a execução da rede, que é o próprio funcionamento dela sobre as entradas.

Trabalhos mais recentes para o inglês têm alavancando o estado-da-arte com novas arquiteturas de rede e sofisticados modelos de linguagem contextualizados. Uma arquitetura de destaque é a BiLSTM-CRF proposta por Lample et al. [36] que recebe um modelo *Word Embeddings* pré-treinado como representação da linguagem. Recentemente, os resultados de Lample et al. foram superados por outros dois trabalhos que usam novas formas de representar a linguagem: ELMo [48], BERT [15] e *Flair Embeddings* [4]. A tabela 3.6 compara a evolução dos resultados desses últimos quatro trabalhos. Os trabalhos que apresentam os modelos BERT e *Flair Embedding* são de suma importância para esta pesquisa e portanto dedica-se duas seções 2.2 e 6.1 para os modelos BERT e *Flair Embedding*, respectivamente. Nessas seções o leitor poderá encontrar minuciosamente como são gerados os modelos e o formalismo matemático do processo.

Tabela 3.6 – Evolução do estado-da-arte para o Inglês

Ano	**Arquitetura**	**Treino/Teste**	**F1**
2016	BiLSTM-CRF [36]	CoNLL-2003	90,94%
2018	BiLSTM-CRF+ELMo [48]		92,22%
2018	BERT [15]		92,80%
2018	BiLSTM-CRF+Flair [4]		**93,09**%

Com base em [36], foi apresentado por [13] uma rede neural que usa uma camada de CRF para classificação dos rótulos das EN em português. Os autores combinaram na

estrutura de sua rede as arquiteturas Bi-LSTM e o classficador probabilístico CRF, comum nas abordagens clássicas, tal como as técnicas implementadas no NERP-CRF e CRF+LG [17, 50]. Segundo [13], a atribuição de *tags* para *tokens* em um texto é baseada em informação, isto é, depende de um contexto em que estas estão inseridas. Assim, para determinar se um *token* é uma EN, deve-se analisar a forma da palavra e as tendências de localização da palavra em um dado contexto.

Sem perda, pode-se perceber que houve evolução com relação às técnicas usadas para resolver a tarefa de REN. Partindo de estratégias com engenharia de *features* manuais até redes neurais com extração automática de padrões. Assim, os trabalhos estado-da-arte têm convergido para uma arquitetura de rede neural profunda capaz de capturar informações no nível de sentença e caractere usando, sempre, modelos de linguagem *Word Embeddings* ou os sofisticados modelos contextualizados, como é o caso do ELMo, BERT e *Flair Embeddings*. No capítulo 7 serão apresentados os resultados da abordagem proposta nesta pesquisa, valendo-se de uma rede neural com funcionamento semelhante às discutidas nesta seção, mas que difere das outras, para o português, por produzir ganho considerável em medida F1 ao usar os *Flair Embeddings*.

4. RECURSOS

Neste Capítulo, serão apresentados os recursos usados para avaliação da rede neural responsável pelo REN; os ML para o Português disponíveis para uso; e os corpora para geração de novos ML.

Neste sentido, a Seção 4.1 detalha os dois corpora usados para comparação com abordagens anteriores para o REN em Português, além de detalhar o corpus GeoCorpus de domínio geológico do qual também avaliou-se a rede neural. Já na Seção 4.2 apresenta-se os corpora usados nas avaliações do IberLEF 2019 [11]. A Seção 4.4 apresenta os recursos de ML já existentes para o Português, que estão disponíveis para uso. Por fim, a Seção 4.3 apresenta três grandes corpora em Português para geração de novos ML.

4.1 Corpora para REN

4.1.1 Primeiro HAREM e Mini-HAREM

Segundo Santos e Cardoso (2007) [59], o HAREM (Avaliação de Reconhecimento de Entidades Mencionadas) é um concurso avaliativo para o REN no Português. Dois corpora foram utilizados na primeira edição dessa competição: o primeiro corpus de avaliação (chamado de Primeiro HAREM), com mais de cinco mil Entidades Nomeadas (EN); e o corpus Mini-HAREM, com mais de três mil EN. Ambos foram anotados para o REN e compostos de uma variedade de gêneros textuais: jornalístico, literário, político, textos da web e transcrições de entrevistas. Um e outro são considerados coleção ouro devido ao fato de que suas EN foram manualmente anotadas e estão entre os corpora mais citados na literatura do REN para Português. Os dois anotados com dez categorias. Usamos esses corpora em nossos experimentos para possibilitar a comparação com trabalhos anteriores sobre REN em Português. Há também o Mini-HAREM, que é um corpus menor, composto de textos do mesmo domínio que o primeiro HAREM e conta com 129 textos também anotados manualmente. A tabela 4.1 descreve as categorias e quantidades de EN.

Tabela 4.1 – Tabela com categorias e quantidades do I HAREM e Mini HAREM

Categoria	Quantidade	
	Primeiro HAREM	**Mini-HAREM**
Abstração	406	204
Acontecimento	128	57
Coisa	135	170
Local	1.236	877
Obra	200	190
Organização	924	599
Pessoa	1.031	832
Tempo	436	361
Valor	463	326
Outro	70	28
Total	**5.029**	**3.644**

4.2 Corpora para REN em textos de domínio

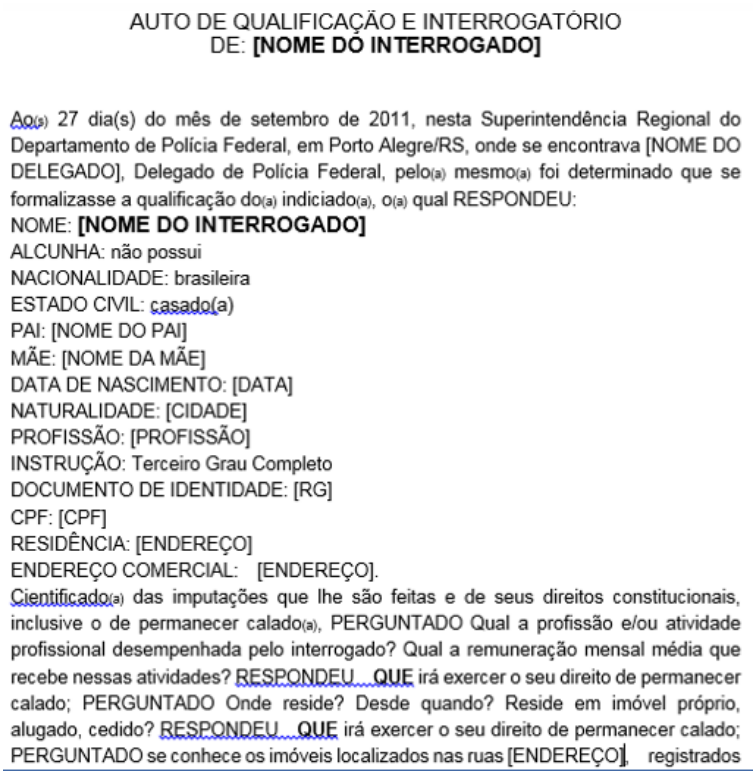
4.2.1 Corpus para domínio Policial (*Police Dataset*)

Este corpus é formado por dados textuais da Polícia Federal do Brasil e foi anotado manualmente para a categoria PESSOA pelos organizadores da tarefa de REN no IberLEF 2019. Os dados estão divididos em dez textos de Depoimentos, dez textos de Declarações e dez textos de Interrogatório.

A anotação desse corpus foi realizada por quatro anotadores: cada um fez sua anotação individual e, no final do processo, houve um alinhamento de todas as entidades anotadas. A ferramenta de anotação usada foi o WebAnno[1], por ter a possibilidade de produzir saídas no formato CoNLL-2002 [57]. Analogamente, Moreira et al. [44] anotou um conjunto de 15 peças policiais (depoimentos, declarações e interrogatórios) e avaliou o desempenho de duas ferramentas de REN no corpus criado. As figuras 4.1 e 4.2 exemplificam o texto de interrogatório e declaração, respectivamente. Os textos de depoimentos são idênticos aos de declarações, exibidos na figura 4.2. Os dados pessoais nas imagens foram ocultados para preservar a identidade das pessoas citadas e envolvidas.

O *Police Dataset* contém textos bem estruturados, bem como gramaticalmente corretos, pois são todos documentos oficiais. A técnica de pré-processamento usada no texto antes da anotação foi a tokenização. No total, de 30 textos, tivemos 1.388 sentenças, 37.706 tokens e um total de 916 entidades nomeadas da categoria PESSOA. Esses dados também são de natureza sensível e por isso não são de acesso público.

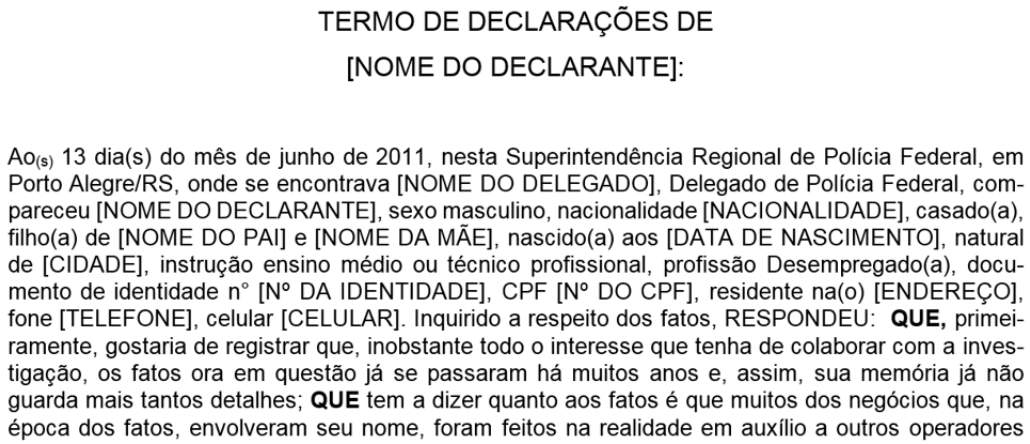
[1] https://webanno.github.io/webanno/



AUTO DE QUALIFICAÇÃO E INTERROGATÓRIO
DE: **[NOME DO INTERROGADO]**

Ao(s) 27 dia(s) do mês de setembro de 2011, nesta Superintendência Regional do Departamento de Polícia Federal, em Porto Alegre/RS, onde se encontrava [NOME DO DELEGADO], Delegado de Polícia Federal, pelo(a) mesmo(a) foi determinado que se formalizasse a qualificação do(a) indiciado(a), o(a) qual RESPONDEU:
NOME: **[NOME DO INTERROGADO]**
ALCUNHA: não possui
NACIONALIDADE: brasileira
ESTADO CIVIL: casado(a)
PAI: [NOME DO PAI]
MÃE: [NOME DA MÃE]
DATA DE NASCIMENTO: [DATA]
NATURALIDADE: [CIDADE]
PROFISSÃO: [PROFISSÃO]
INSTRUÇÃO: Terceiro Grau Completo
DOCUMENTO DE IDENTIDADE: [RG]
CPF: [CPF]
RESIDÊNCIA: [ENDEREÇO]
ENDEREÇO COMERCIAL: [ENDEREÇO].
Cientificado(a) das imputações que lhe são feitas e de seus direitos constitucionais, inclusive o de permanecer calado(a), PERGUNTADO Qual a profissão e/ou atividade profissional desempenhada pelo interrogado? Qual a remuneração mensal média que recebe nessas atividades? RESPONDEU **QUE** irá exercer o seu direito de permanecer calado; PERGUNTADO Onde reside? Desde quando? Reside em imóvel próprio, alugado, cedido? RESPONDEU **QUE** irá exercer o seu direito de permanecer calado; PERGUNTADO se conhece os imóveis localizados nas ruas [ENDEREÇO], registrados

Figura 4.1 – Exemplo de Auto de Qualificação e Interrogatório. Extraído de [44].



TERMO DE DECLARAÇÕES DE
[NOME DO DECLARANTE]:

Ao(s) 13 dia(s) do mês de junho de 2011, nesta Superintendência Regional de Polícia Federal, em Porto Alegre/RS, onde se encontrava [NOME DO DELEGADO], Delegado de Polícia Federal, compareceu [NOME DO DECLARANTE], sexo masculino, nacionalidade [NACIONALIDADE], casado(a), filho(a) de [NOME DO PAI] e [NOME DA MÃE], nascido(a) aos [DATA DE NASCIMENTO], natural de [CIDADE], instrução ensino médio ou técnico profissional, profissão Desempregado(a), documento de identidade n° [Nº DA IDENTIDADE], CPF [Nº DO CPF], residente na(o) [ENDEREÇO], fone [TELEFONE], celular [CELULAR]. Inquirido a respeito dos fatos, RESPONDEU: **QUE,** primeiramente, gostaria de registrar que, inobstante todo o interesse que tenha de colaborar com a investigação, os fatos ora em questão já se passaram há muitos anos e, assim, sua memória já não guarda mais tantos detalhes; **QUE** tem a dizer quanto aos fatos é que muitos dos negócios que, na época dos fatos, envolveram seu nome, foram feitos na realidade em auxílio a outros operadores

Figura 4.2 – Exemplo de Termo de Declaração. Extraído de [44].

4.2.2 Corpus para domínio Clínico (*Clinical Dataset*)

Evoluções médicas são dados textuais relatados por funcionários de um hospital (técnicos de enfermagem, enfermeiros, médicos...) sobre cada paciente [19, 51]. Esse tipo de texto contém nomes de pacientes, médicos e residentes, resultados de exames médicos e outras informações médicas variadas. Um conjunto de evoluções médicas foi selecionado e anotado com a categoria PESSOA por quatro anotadores da organização da tarefa de REN no IberLEF 2019. Assim como no *Police Dataset* o *Clinical Dataset* também foi anotado manualmente por quatro pessoas e passou por um processo de alinhamento

das anotações ao término. O WebAnno foi escolhido como ferramenta de anotação. A figura 4.3 extraída do trabalho de Quaini et al. [51] apresenta um exemplo textual de uma evolução médica.

O *Clinical Dataset* apresenta desafios particulares quando se trata de sua estrutura textual: palavras que devem ser separadas por um espaço não são (por exemplo, “AnaR1") e várias abreviações médicas. Nestes casos, foi entendido que “AnaR1"é uma Pessoa, assim como “####Paulo" também é uma PESSOA e, portanto, pertencem à categoria Pessoa. No total, 77 entidades nomeadas da categoria PESSOA foram anotadas em um conjunto de 50 evoluções com 9.523 tokens. Por se tratar de corpus de natureza sensível, não é de domínio público.

R104 - (Dor abdominal e pelvica) Outras dores abdominais e as nao especificadas Evolução HNSC:
*****enfermagem****paciente MUITO AGITADA PELAMANHÃ.
RECEBEU TODAS MEDICAÇÕES PRESCRITAS SEM EFEITO CONSIDERÁVEL SIC NA PASSAGEM DE PLANTÃO.
MÃE ENTENDEU NÃO SER POSSÍVEL LEVÁ-LA A CONSULTA NA SANTA CASA PELA AGITAÇÃO ATUAL.
LIGO PARA GASTRO E DEIXO RECADO COM RESIDENTE NoInfo.
Evoluído por: XXXXX em XX/XX/XX às XX:XX

Figura 4.3 – Exemplo de evolução médica. Extraído de [51]

4.2.3 GeoCorpus

Segundo Amaral (2017) [16], o GeoCorpus é um corpus para REN no domínio da Geologia, especificamente sobre bacias sedimentares brasileiras. O GeoCorpus pode ser considerado como um corpus de domínio por não possuir categorias tradicionais como PESSOA, LOCAL e ORGANIZAÇÃO. Os textos presentes neste corpus foram obtidos de teses, dissertações, artigos e boletins de Geociências da Petrobras, todos em Português brasileiro.

Para o uso do GeoCorpus fez-se uma revisão em todo o conjunto, resultando no GeoCorpus-2. Primeiro, houve a conversão de formatos, de modo que a versão original do GeoCorpus é um XML, que não é compatível com as necessidades das estratégias usadas nessa pesquisa. Nesse sentido, fez-se um trabalho de passar o GeoCorpus para o padrão CoNLL-2002 [57]. Após, notou-se que 230 sentenças estavam repetidas e estas foram retiradas. Com isso, o corpus foi dividido em três conjuntos: Treino, Teste e Validação. Essa divisão é essencial para treinamento de redes neurais.A tabela 4.2 apresenta as categorias e quantidades das entidades geológicas do GeoCorpus-2. As colunas “Original” e “Revisão” significam, respectivamente, os números de entidades geológicas no corpus Original e após a revisão.

Tabela 4.2 – Quantidade de EN no GeoCorpus-2

Categoria	Original	Revisão	Treino	Teste	Validação
Eon	288	286	206	60	20
Era	326	324	235	69	20
Período	637	628	464	125	39
Época	650	647	478	134	35
Idade	796	756	566	157	33
Rocha Sedimentar Siliciclástica	743	738	543	150	45
Rocha Sedimentar Carbonática	240	240	173	50	17
Rocha Sedimentar Química	5	5	3	1	1
Rocha Sedimentar Orgânica	22	22	15	5	2
Bacia Sedimentar Brasileira	243	240	168	58	14
Contexto Geológico de Bacia	262	260	188	56	16
Unidade Litoestratigráfica	581	574	425	107	42
Outro	739	736	543	156	37
Total	**5.532**	**5.456**	**4.007**	**1.128**	**321**

4.3 Corpora para Geração de ML em Português

Nesta seção, serão apresentados quatro corpora que foram usados para treinar novos modelos de linguagem usados em nossos experimentos: BlogSet-BR, brWaC, ptwiki-20190301 e GeoBoletins. Como prelúdio dos motivos que levaram a treinar novos modelos de linguagem, é que um grande corpus vasto em diversidade textual pode trazer ganhos em avaliações extrínsecas, como é o caso do Reconhecimento de Entidades Nomeadas.

BlogSet-BR é um grande corpus formado por textos em Português Brasileiro advindos de blogs online da internet. Este corpus contém mais de 86 milhões de sentenças, com mais de 2,7 bilhões de tokens. Dos 7,4 milhões de posts coletados, as principais *tags* foram: 'notícias', 'dicas', 'amor', 'música', 'moda' e 'filmes' [20]. A Tabela 4.3 detalha o tamanho do BlogSet-BR.

brWaC é outro grande corpus com textos em Português Brasileiro construído a partir de uma metodologia chamada WaCky [5], que consiste em quatro passos:

- Identificação de sementes para URLs;
- Limpeza;
- Remoção de conteúdo duplicado;
- Marcação *Part-of-speech tagging*;

O brWaC contém mais de 145 milhões de sentenças, com um total de 3 bilhões de tokens [23]. A Tabela 4.3 detalha o tamanho do brWaC.

ptwiki-20190301[2] é um dos armazenamentos de dados mensais da *Wikimedia Foundations*. É composto de uma cópia completa de todas as páginas da Wikimedia para o Português Brasileiro até o dia 1 de Março de 2019, e está disponível tanto no formato de texto quanto de metadado em XML. A tabela 4.3 detalha o tamanho do corpus.

Tabela 4.3 – Dimensão dos corpora de treino

Corpus	Tokens	Palavras	Sentenças
BlogSet-BR	2.750.700.677	2.146.206.009	86.803.291
brWaC	3.207.918.165	2.764.098.344	145.370.673
ptwiki-20190301	162.210.780	163.962.460	7.053.963

GeoBoletins corpus formado por boletins[3] de geociências da Petrobras. O corpus consiste de 2.276.554 bilhões de tokens e 95.454 sentenças. Vale ressaltar que esse corpus foi usado especificamente para um processo de afinamento, que será descrito na seção 5.2.2.

4.4 Modelos de Linguagens para o Português

4.4.1 Word Embeddings

Como foi apresentado no capítulo 3, as abordagens mais sofisticadas para reconhecimento de entidades nomeadas se utilizam de modelos *Word Embeddings*, previamente treinados, com um grande corpus de variados gêneros textuais. Neste sentido, nossos experimentos consideraram vários modelos *Word Embeddings* (WE) públicos.

Uma parcela dos WE utilizados veio de um repositório online[4] de modelos pré-treinados para o Português. Esses recursos são de livre uso e mantidos pelo Núcleo Interinstitucional de Linguística Computacional (NILC) da Universidade de São Paulo (USP). Tais modelos foram treinados com um corpus composto de 1.395.926.282 tokens em Português Brasileiro e Europeu. A figura 4.4 detalha a composição desse corpus, que apresenta uma vasta diversidade textual. Os modelos foram treinados utilizando os seguintes algoritmos: *Word2Vec, FastText, Wang2Vec* e *Glove*.

Como pode ser visto, os *WE* do NILC podem ser considerados de domínio geral, pois não se restrigem a um contexto especifico. Neste sentido, um modelo *WE* de domínio da Geologia também foi adicionado às avaliações, e será feita referência a esse *WE* por

[2] https://dumps.wikimedia.org/ptwiki/20190301/
[3] http://publicacoes.petrobras.com.br/portal/revista-digital/pt_br/pagina-inicial.htm
[4] http://nilc.icmc.usp.br/embeddings

Corpus	Tokens	Types	Genre	Description
LX-Corpus [Rodrigues et al. 2016]	714,286,638	2,605,393	Mixed genres	A huge collection of texts from 19 sources. Most of them are written in European Portuguese.
Wikipedia	219,293,003	1,758,191	Encyclopedic	Wikipedia dump of 10/20/16
GoogleNews	160,396,456	664,320	Informative	News crawled from GoogleNews service
SubIMDB-PT	129,975,149	500,302	Spoken language	Subtitles crawled from IMDb website
G1	105,341,070	392,635	Informative	News crawled from G1 news portal between 2014 and 2015.
PLN-Br [Bruckschen et al. 2008]	31,196,395	259,762	Informative	Large corpus of the PLN-BR Project with texts sampled from 1994 to 2005. It was also used by [Hartmann 2016] to train word embeddings models
Literacy works of public domain	23,750,521	381,697	Prose	A collection of 138,268 literary works from the Domínio Público website
Lacio-web [Aluísio et al. 2003]	8,962,718	196,077	Mixed genres	Texts from various genres, e.g., literary and its subdivisions (prose, poetry and drama), informative, scientific, law, didactic technical
Portuguese e-books	1,299,008	66,706	Prose	Collection of classical fiction books written in Brazilian Portuguese crawled from Literatura Brasileira website
Mundo Estranho	1,047,108	55,000	Informative	Texts crawled from Mundo Estranho magazine
CHC	941,032	36,522	Informative	Texts crawled from Ciência Hoje das Crianças (CHC) website
FAPESP	499,008	31,746	Science Communication	Brazilian science divulgation texts from Pesquisa FAPESP magazine
Textbooks	96,209	11,597	Didactic	Texts for children between 3rd and 7th-grade years of elementary school
Folhinha	73,575	9,207	Informative	News written for children, crawled in 2015 from Folhinha issue of Folha de São Paulo newspaper
NILC subcorpus	32,868	4,064	Informative	Texts written for children of 3rd and 4th-years of elementary school
Para Seu Filho Ler	21,224	3,942	Informative	News written for children, from Zero Hora newspaper
SARESP	13,308	3,293	Didactic	Text questions of Mathematics, Human Sciences, Nature Sciences and essay writing to evaluate students
Total	1,395,926,282	3,827,725		

Figura 4.4 – Composição do corpus do NILC Embeddings. Extaída de [28].

GeoWE. O *GeoWE* foi desenvolvido por Gomes et al. [25], treinado sobre um corpus de 10.109.732 milhões de tokens. O algortimo escolhido pelos autores foi o *Word2Vec* na arquitetura *Skip-Gram*.

É importante ressaltar que além dos *embeddings* do NILC, também foram gerados novos *Word Embeddings* como parte dos cenários de modelos de linguagem usados na tarefa de REN. O detalhamento desses novos recursos são descritos nas subseções 4.3 e 5.1.

4.4.2 BERT Embeddings

Devlin et al. [15] tornou públicos, no GitHub[5], vários modelos BERT. Apenas os modelos Inglês e Chinês foram treinados com um corpus monolíngue referente ao idioma. Entretanto, há um modelo multilíngue, que contempla 104 idiomas, incluindo o Português. Os modelos BERT para o Inglês se dividem em *Large* e *Base* e subdivididos por *Cased* (Capitalização preservada) e *Uncased* (Todo o texto deve estar em minúsculo). No caso

[5]https://github.com/google-research/bert

do modelo Chinês e Multilíngue há apenas versões *Base* com as variações *Large* e *Base*. A tabela 4.4 apresenta os modelos disponíveis e seus respectivos corpora de treino. Nos casos em que apenas a Wikipédia foi usada como corpus, não foi informada pelos autores do BERT a quantidade de palavras.

Como mencionado no capítulo 2, seção 2.2, o BERT foi desenvolvido pelo *Google AI Language*[6], significando que a versão original dele foi implementado usando a biblioteca de *Deep Learning TensorFlow* [1]. Posteriormente, uma versão *PyTorch*[7] (outra biblioteca de *Deep Learning*) foi desenvolvida e disponibilizada[8].

Tabela 4.4 – Modelos BERT e seus respectivos corpora de treino

Tipo	Idioma	Corpus	Qtd. Palavras
BERT-Base BERT-Large	Inglês	BooksCorpus [70] Wikipedia-EN	3,3 Bilhões
BERT-Base	Chinês	Wikipedia-CH	-
BERT-Base	Multilíngue	Wikipedia Top 100 idiomas	-

4.4.3 *Flair Embeddings*

Há um potencial recurso de Modelos *Flair Embeddings* para o Português, desenvolvido por Lief (2019) [38]. O modelo encontra-se disponível no GitHub[9]. Esse modelo foi treinado sobre um corpus de 0.9 bilhões de tokens em português, extraídos de diversas web páginas escritas em português, não fazendo distinção entre português europeu e brasileiro [7]. Quando não houver dúvidas, o Flair Embeddings desenvolvido por Lief (2019) será mencionado como *FlairEL*.

4.5 A biblioteca *Flair*

Por fim, há um recurso fundamental para o desenvolvimento dos resultados e experimentos apresentados nesta pesquisa: a biblioteca *Flair* (para Python[10]) desenvolvida por Akbik et al. [3], que acolhe vários componentes já mencionados. Como forma de organizar os conceitos, a figura 4.5 apresenta os três principais recursos usados da biblioteca:

[6] https://ai.google/research/teams/language/
[7] https://pytorch.org/
[8] https://github.com/huggingface/transformers
[9] https://github.com/zalandoresearch/flair
[10] https://www.python.org/

Modelos de Linguagens já pré-treinados; a rede neural *CharLM* para geração de novos modelos de linguagem *Flair Embeddings*; e a rede neural BiLSTM-CRF para identificação e classificação das entidades nomeadas, que diante da temática dessa pesquisa, foi usada para Reconhecimento de Entidades Nomeadas.

Ainda na figura 4.5, as caixas com preenchimento branco significam que o recurso não foi usado. Quando azul, significam que foi usado. Sendo assim, recursos como o modelo de linguagem *FlairEL* e os modelos BERT apresentados nas seções 4.4.3 e 4.4.2 já estão embutidos na biblioteca *Flair* e foram usados via a própria biblioteca.

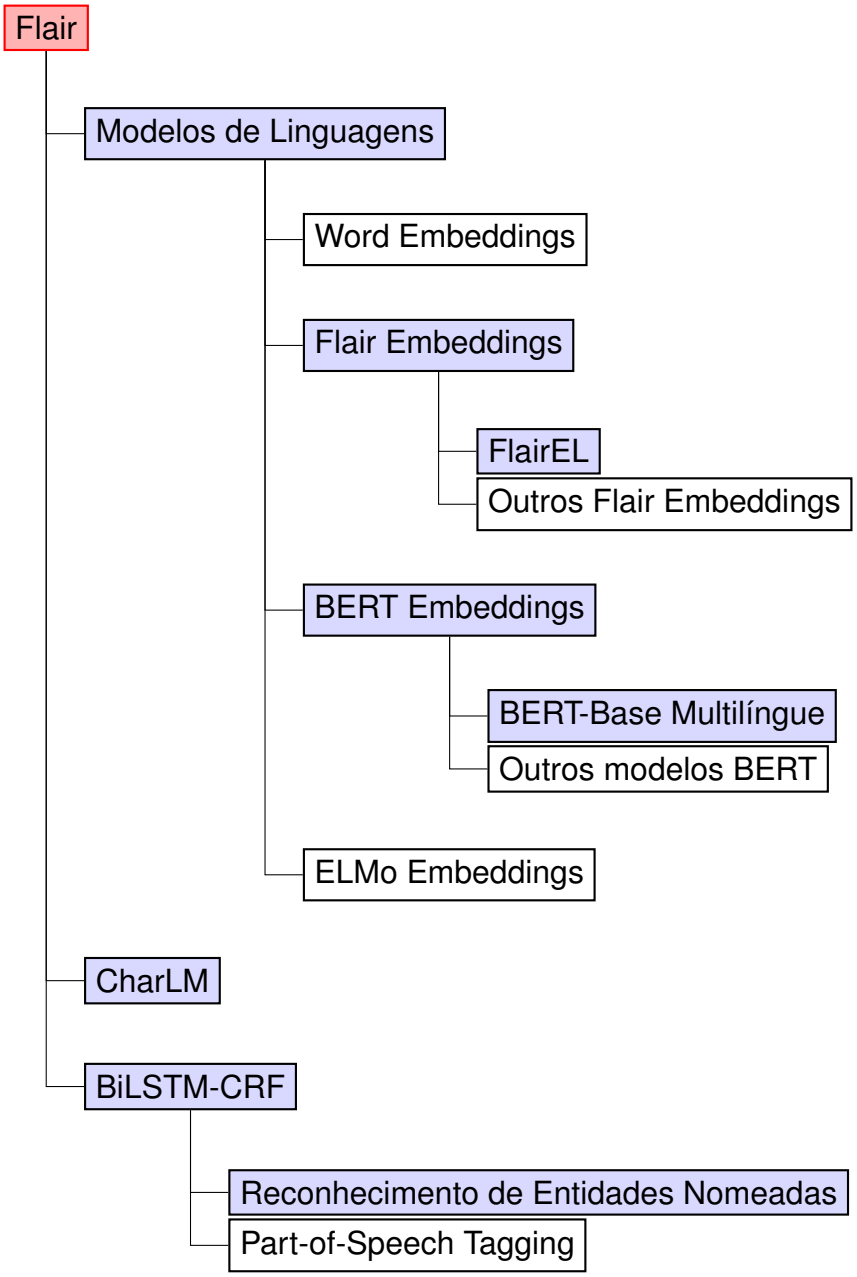


Figura 4.5 – Diagrama de componentes da biblioteca *Flair*

5. GERAÇÃO DE MODELOS DE LINGUAGEM

5.1 Gensim para *Word Embeddings*

No capítulo anterior, seção 4.3 foi apresentado os corpora usados para geração de novos modelos de linguagem, com a intenção de aprofundar as avaliações de REN propostas nesta pesquisa. A seguir explica-se as fases e técnicas usadas para pré-processar os corpora, bem como a geração dos modelos *Word Embeddings (WE)* a partir do corpus selecionado.

O pré-processamento desses três corpora (brWaC, BlogSet-BR e ptwiki-20190301) usados para treinar os modelos de linguagem foi realizado em três estágios:

i Limpeza ampla;

ii Limpeza fina;

iii Divisão de corpus;

O estágio *(i)* envolve a tokenização pela biblioteca NLTK[1]; a remoção de todo código HTML; a remoção de todas as anotações de *Part-of-speech tagging*; e a remoção de todos os caracteres não-latinos.

O estágio *(ii)* envolve seguir as especificações de Hartmann et al. [28], utilizando o *script* disponibilizado pelo NILC[2]. As especificações adotadas são:

- Todos emails são mapeados para um token padrão “EMAIL”;
- Todos os números são mapeados para o token “0”;
- Todas as urls são mapeadas para o token “URL”;
- Diferentes aspas são normalizados
- Diferentes hífens são normalizados;
- Tags de HTML são removidas;
- Todo texto entre colchetes é removido;
- Sentenças com menos de 5 tokens são removidas;

[1] https://github.com/nltk/nltk
[2] https://github.com/nathanshartmann/portuguese_word_embeddings

Por fim, o estágio *(iii)* envolve a divisão de todo o corpus em arquivos com não mais que 10 milhões de tokens. Foi adotada a divisão do corpus para facilitar o processo de treino dos modelos de linguagem, uma vez que o corpus final somou 25GB. A Tabela 5.1 apresenta algumas estatísticas dos corpora selecionados após o pré-processamento.

Tabela 5.1 – Detalhes do Corpus após pré-processamento

Corpus	**Sentenças**	**Tokens**
brWaC	127.272.109	2.930.573.938
BlogSet-BR	58.494.090	1.807.669.068
ptwiki-20190301	7.053.954	162.109.057
Corpus final	192.820.153	4.900.352.063

Após o pré-processamento do corpora, foi realizado o treinamento dos *WE* utilizando a biblioteca Gensim [52] no Python, que oferece duas redes neurais para produzir *WE*: *Word2Vec* [40] e *FastText* [26]. Ambos algoritmos são divididos em duas arquiteturas (ou estratégias) de treinamento: *CBOW* e *Skip-Gram*. O que resultou em quatro *WE* produzidos. Cada modelo levou em média 2 dias e 10 horas para finalizar o treinamento.

Todos os modelos desenvolvidos têm 300 dimensões e foram treinados em 5 épocas. Durante o treinamento, sempre que um token aparecesse mais de sete vezes em um lote de treino, o token faria parte do vocabulário do *WE*. Um dos parâmetros fundamentais na produção dos *WE* é o tamanho da janela de contexto. Para essa pesquisa, foi considerada uma janela de contexto de tamanho 5, ou seja, para cada token de uma sequência (de tokens), as probabilidades serão calculadas levando em conta 5 tokens à esquerda e 5 tokens à direita.

Word2Vec: é um grupo de algoritmos para geração de vetores de palavras que se dividem em duas arquiteturas de rede neural: *Continuous Bag Of Words (CBOW)* e *Skip-gram*. A figura 5.1, do trabalho original de Mikolov et al. [40] ilustra essas arquiteturas. Assim, considere uma sequência de tokens s tal que:

$$s = (t_1, t_2, ..., t_c, ..., t_{n-1}, t_n)$$

Onde t_c é o token central de s e os tokens a sua esquerda e direita, ou seja, $s - \{t_c\}$, são o contexto em que t_c está inserido. Neste sentido, na arquitetura *CBOW*, uma camada de entrada recebe a sequência $s - \{t_c\}$ com o objetivo de prever t_c. Enquanto que no *Skip-Gram* o processo é invertido, de modo que a camada de entrada recebe $\{t_c\}$ com o objetivo de prever o contexto relacionado, ou seja, $s - \{t_c\}$ [40, 33].

FastText: assim como o *Word2Vec*, o *FastText* é outro algoritmo para *WE*, que também é dividido nas arquiteturas *CBOW* e *Skip-gram*. Esse tipo de ML tem sido usado com sucesso para várias tarefas de PLN, tal como classificação de textos e reconhecimento

de entidades nomeadas. Uma das principais diferenças entre o *Word2Vec* e o *FastText* é que o *FastText* pode estimar vetores para palavras que não fazem parte do modelo pré-treinado. Isso acontece porque o treinamento do modelo usa n-gramas em vez de palavras completas. Por exemplo, dado o token “renoir” e $n = 3$ tem-se a palavra representada por uma coleção de trigramas $<re, ren, eno, noi, oir, ir>$ [26, 33].

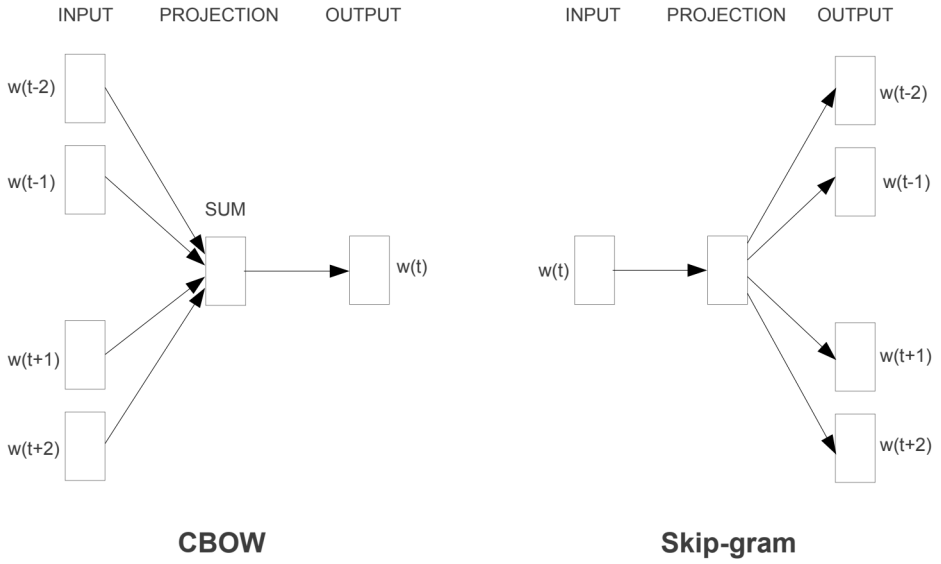


Figura 5.1 – Estrutura das arquiteturas dos modelos *CBOW* e *Skip-Gram*

5.2 CharLM para *Flair Embeddings*

No Capítulo 3, foi mostrado que os modelos de linguagem *Flair Embeddings* são um dos componentes mais importantes para as abordagens que usam redes neurais profundas, pois eles têm um alto poder de representatividade da linguagem. Neste sentido, nesta seção, serão apresentados os detalhes envolvidos no desenvolvimento de um novo modelo *Flair Embeddings* para o Português.

O treinamento de um modelo de linguagem *Flair Embeddings* consiste em receber sequências de caracteres por meio de sentenças de uma linguagem natural, de modo que serão atribuídos significados contextuais às sequências de caracteres. A extração e atribuição dos significados das sequências são feitas por uma rede neural chamada *Neural Character-level Language Modeling (CharLM)* [4]. No capítulo seguinte, será detalhado o funcionamento da *CharLM*.

5.2.1 *FlairBBP*

Segundo Akbik et al. [4], não é necessário um meticuloso pré- processamento dos textos usados para gerar os *Flair Embeddings*. Até mesmo uma simples tokenização pode ser desnecessária. Entretanto, alguns passos de limpeza do corpus foram realizados. Assim, foram realizados os três estágios descritos na seção 5.1, porém não houve tokenização e foram revertidas as desmembrações do POS nos casos em que foram separados preposição seguida de artigo e/ou preposição seguida de pronome.

Esses procedimentos de limpeza foram feitos nos corpora: BlogSet-Br, brWac e ptwiki-20190301. Esses três corpora foram escolhidos para gerar um modelo *Flair Embeddings* de cunho amplo e geral. Esse modelo *Flair Embeddings* será referido como *FlairBBP*, onde “BBP” vem das iniciais dos corpora usados no treino.

Novamente, todo o corpus foi dividido em arquivos com não mais de 10 milhões de tokens (como não houve tokenização, qualquer sequência de caracteres separados dos outros por um espaço foi considerado um token). Com o pré-processamento completo, partiu-se para o treino dos modelos com rede a *CharLM* utilizando uma GPU Tesla K40. O treinamento durou aproximadamente 7 dias ininterruptos. A Tabela 5.2 apresenta os hiperparâmetros usados no treinamento.

É importante destacar que um modelo *Flair Embeddings* consiste de duas partes: um modelo *Forward* e um *Backward*. Isso significa que ao treinar um modelo *Flair Embeddings* deve-se fazer dois treinamentos separados: o *Flair Embeddings Forward* e o *Flair Embeddings Backward*. Sendo assim, o modelo *FlairBBP* é composto por dois outros: *FlairBBP Forward* e *FlairBBP Backward*. Por simplicidade será abstraído para *FlairBBP*.

Tabela 5.2 – Hiperparâmetros de Treino

Parâmetro	**Valor**
Camadas de LSTM	1
Camadas ocultas	2048
Comprimento de sentenças	250
Tamanho do lote	100

5.2.2 FlairBBP-GeoFT

Como já foi dito, também foi feito avaliações para o REN no domínio da Geologia. Como se trata de um domínio muito especifico, decidiu-se fazer um processo de afinamento

do modelo geral *FlairBBP* para um mais rico em informações do domínio trabalhado. Esse processo normalmente é conhecido por *Fine Tuning*.

Para fazer o *Fine Tuning* foi usada novamente a rede *CharLM*. Entretanto, agora com parâmetros apropriados para carregar o modelo já treinado (que se pretende ajustar), ou seja, o *FlairBBP*, e o novo corpus de treino. No caso, foi usado o corpus GeoBoletins (apresentado na seção 4.3) para enriquecer/afinar o *FlairBBP*. Os outros parâmetros de treino se resumem a tabela 5.2.

No final do processo, gerou-se um novo modelo adaptado para o contexto geológico. Mas, assim como o *FlairBBP*, o $FlairBBP_{GeoFT}$ também é composto por dois outros modelos: *Forward* e *Backward*. Mas por simplicidade será referido como $FlairBBP_{GeoFT}$, fazendo alusão ao *FlairBBP*, o corpus GeoBoletins e o procedimento de *Fine Tuning*.

6. REDES NEURAIS PARA ML E REN

Nas seções 2.3 e 3.2 foi discutido o recente modelo de linguagem chamado *Flair Embeddings* apresentado por Akbik et al. [4], que junto com modelos *Word Embeddings* e a rede neural BiLSTM-CRF formam a abordagem estado-da-arte para REN em Inglês e Alemão. Neste capítulo, será detalhado o funcionamento das redes neurais usadas nessa pesquisa. Na seção 6.1, será mostrado como a rede neural *CharLM* aprende um ML a partir de um grande corpus, resultando em um *Flair Embeddings*. Na sequência, a seção 6.2 detalha a rede neural BiLSTM-CRF que recebe os modelos de linguagem previamente treinados e com base nessas representações, mais novas informações aprendidas, faz a classificação dos tokens.

6.1 Rede Neural para ML (CharLM)

Nesta seção, tratar-se-á formalmente como são gerados os modelos *Flair Embeddings*.

Segundo Rosenfeld [54], é possível modelar uma linguagem natural por meio do aprendizado das distribuições de caracteres dessa linguagem. Nesse sentido, seja $X_{0:T}$ uma sequência de caracteres, que produz uma linguagem natural, tal que $X_{0:T}$ é definida por:

$$X_{0:T} := (x_0, x_1, ..., x_t)$$

Nessa perspectiva, um corpus é uma coleção de sequências de caracteres $\bigcup_{i=1}^{S} X^i_{0:T}$, em que S é a quantidade de sentenças do corpus. Assim, os procedimentos de treino dos modelos *Flair Embeddings* se resumem ao processo de aprender a predizer qual o próximo caractere de uma dada sequência $X_{0:T}$. Por exemplo, considere a sequência:

$$\begin{aligned} X_{0:7} &= (c, o, u, b, e, r, t) \\ &= (x_0, x_1, ..., x_7) \end{aligned}$$

Nota-se que $X_{0:7} = X_{0:6} \cup \{x_7\}$, o que se induz a questionar se dado apenas $X_{0:6}$, será possível predizer $\{x_7\}$ de modo que $X_{0:6} \cup \{x_7\} = X_{0:7}$. Como já mencionado, segundo Rosenfeld (2000) [54] é possível fazer predições desse tipo. Em outros termos, o que está sendo feito é o calculo da probabilidade condicional:

$$p(x_t|x_0, x_1, ..., x_{t-1}) = p(x_{0:T})$$

Então, tem-se:

$$\begin{aligned} p(x_{0:T}) =& p(x_t|x_0, x_1, ..., x_{t-1}) \\ =& p(x_0|X_{0:-1}) \cdot p(x_1|X_{0:0}) \cdot p(x_2|X_{0:1}) \cdot p(x_3|X_{0:2}) \cdot ... \cdot p(x_t|X_{0:t-1}) \\ =& \prod_{t=0}^{T} p(x_t|x_{0:t-1}) \end{aligned}$$

No capítulo 2, detalhou-se a arquitetura das redes LSTM e viu-se que a saída da rede é produzida por uma função h_t, que segundo Akbik et al. [4] a $P(x_t|x_{0:t-1})$ é aproximadamente h_t. Logo,

$$\begin{aligned} p(x_{0:T}) &= \prod_{t=0}^{T} p(x_t|x_{0:t-1}) \\ &\approx \prod_{t=0}^{T} p(x_t|h_t;\theta) \end{aligned} \tag{6.1}$$

Na equação 6.1 h_t, representa-se a dada sequência $x_{0:t-1}$, e θ significa os parâmetros do modelo. A equação 6.1 é computada recursivamente em uma LSTM com ajuda da célula de memória c_t, repensável por atualizar os estados internos da rede. Assim, para cada sequência $X_{0:t-1}$, h_t e c_t serão calculadas por:

$$h_t(X_{0:t-1}) = f_h(x_{t-1}, h_{t-1}, c_{t-1};\theta) \tag{6.2}$$

$$c_t(X_{0:t-1}) = f_c(x_{t-1}, h_{t-1}, c_{t-1};\theta) \tag{6.3}$$

Nota-se que para $t = 0$ em $h_t(X_{0:t-1})$ e $c_t(X_{0:t-1})$, vem:

$$h_t(X_{0:-1}) := h_{-1} \tag{6.4}$$

$$c_t(X_{0:-1}) := c_{-1} \tag{6.5}$$

É de se causar estranheza em 6.4 e 6.5, porque naturalmente não há sequências $X_{0:-1}$. Neste caso, o problema pode ser solucionado por duas alternativas: instanciando

$h_{-1} = c_{-1} = 0$ ou tratando $h_{-1} = c_{-1}$ como parte dos parâmetros de θ. Por fim, a probabilidade final de cada caractere x_t é dada por:

$$p(x_t|h_t; V) = \frac{e^{Vh_t+b}}{||e^{Vh_t+b}||_1}$$

Onde V e b são os pesos e o bias, respectivamente. Para criar os *embeddings* contextualizados, os autores utilizaram os estados ocultos de uma LSTM, ou seja, a equação 6.1 representa o modelo que concatena os estados ocultos da esquerda para direita. Esse modelo recebe o nome de *forward*. Analogamente, um modelo que concatena no sentido inverso (direita-esquerda), chama-se *backward*. O modelo *backward* pode ser descrito da seguinte forma: tome $X_{0:T}$ uma sequência de caracteres. Pode-se subdividir $X_{0:T}$ de tal forma que toma-se intervalos sempre à direita de um $x_i \in X_{0:T}$, de modo a predizer x_i.

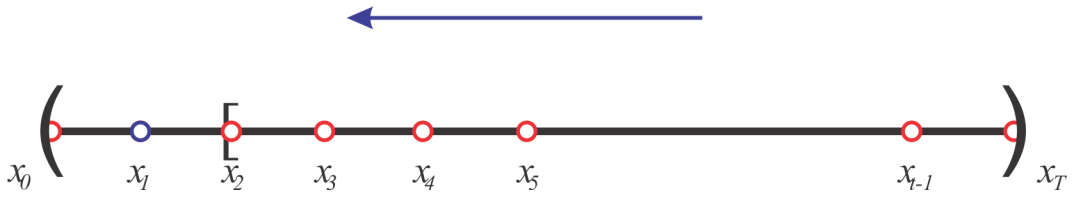


Figura 6.1 – Ilustração de predição do caractere x_1 na direção *backward*

Ou seja, na figura 6.1, objetiva-se predizer x_1 (em azul), dado $x_2, x_3, ..., x_T$. Em probabilidade condicional, temos:

$$p^b(x_t|X_{t+1:T}) \approx \prod_{t=0}^{T} p^b(x_t|h_t^b; \theta)$$

Onde,

$$h_t^b = f_h^b(x_{t-1}, h_{t-1}^b, c_{t-1}^b; \theta)$$

Que é novamente calculada recursivamente com ajuda de c_t^b:

$$c_t^b = f_c^b(x_{t-1}, h_{t-1}^b, c_{t-1}^b; \theta)$$

Para fins de padronização, pode-se definir: $h_t^f := h_t$ da equação 6.2 e $c_t^f := c_t$ da equação 6.3.

Formalmente, os processos de *forward* e *backward* produzem duas saídas para cada palavra: h_t^f e h_t^b. Uma indexação das palavras é necessária, assim cada palavra é

indexada por um t_i, $i \in \mathbb{N}$. Dito isto, para cada *embedding* contextualizado será definido por w:

$$w_i^{CharLM} := \begin{bmatrix} h^f_{t_{i+1}-1} \\ h^b_{t_i-1} \end{bmatrix} \tag{6.6}$$

Tomando novamente o exemplo da figura 2.4, $h^f_{t_{i+1}-1}$ representa a concatenação das saídas (estados ocultos) de cada caractere das palavras: *was* e *born*. Por outro lado, $h^b_{t_i-1}$ é a concatenação com relação a palavra *George*.

6.2 Rede Neural para REN

Essa seção descreve formalmente a arquitetura da rede neural BiLSTM-CRF, responsável pela classificação sequencial de tokens, caso do REN. Nesse sentido, os resultados apresentados no capítulo seguinte competem ao uso da BiLSTM-CRF.

Sejam $BiLSTM = w_0, w_1, ..., w_n$ as entradas da rede e Δ um corpus. Assim,

$$\forall w_i \in \Delta, \exists r_i | r_i := \begin{bmatrix} r^f_i \\ r^b_i \end{bmatrix} \tag{6.7}$$

Onde r^f_i e h^b_i são os estados de saída da *BiLSTM*. Nota-se que r_i é um tensor contendo informações capturadas automaticamente pela rede BiLSTM no momento da passagem das entradas. Uma vez as informações contidas no tensor r_i, elas são passadas para um classificador probabilístico sobre uma sequência de rótulos y, dando a classificação final do token. Isso equivale dizer, que a probabilidade de uma sequência de rótulos $y_{0:n}$ acontecer dada uma sequência de *tokens* $r_{0:n}$ pode ser estimada por um classificador CRF:

$$\hat{P}(y_{0:n}|r_{0:n}) \propto \prod_{i=1}^{n} \psi_i(y_{i-1}, y_i, r_i) \tag{6.8}$$

Onde,

$$\psi_i(y', y, r) = e^{(W_{y',y} r + b_{y',y})}$$

Experimentalmente, os autores dos *Flair Embeddings* também fizeram uma projeção linear dos estados ocultos da rede:

$$r_i = W_r w_i + b_r$$

Cuja intenção é passar r_i diretamente para uma função de probabilidade *softmax*, resultando na classificação do token:

$$P(y_i = j|r_i) = softmax(r_i)[j]$$

Uma representação visual da arquitetura desta rede neural pode ser vista na figura 6.2, em que o primeiro módulo da rede, *Character Language Model*, recupera os tokens de entrada em *embeddings*. Ou seja, o *Character Language Model* guarda os modelos pré-treinados e ao receber um token, produz um tensor r_{token} que é a representação da palavra.

Em seguida, o módulo *Sequence Labeling Model* é de fato a rede BiLSTM-CRF, como é possível ver na figura 6.2: o módulo recebe os tensores r_{token} e a eles novas informações são incorporadas para que, finalmente, o tensor final seja passado para o classificador CRF, retornando a classificação final do token.

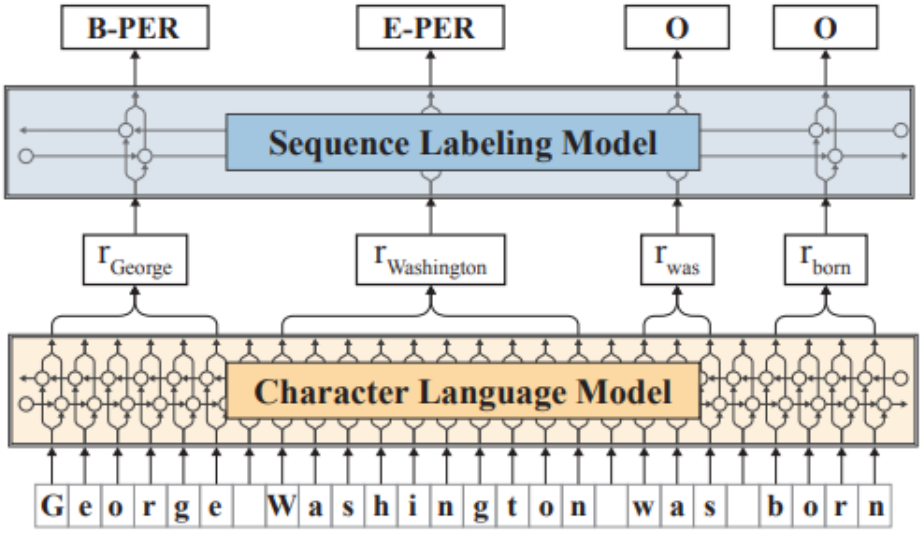


Figura 6.2 – Rede neural para REN. Extraída de [4]

7. EXPERIMENTOS

Neste capítulo, serão apresentados os experimentos e os detalhes envolvidos na abordagem empregada. No capítulo 3, em que fala-se sobre os trabalhos relacionados, são apresentados ao leitor os trabalhos estado-da-arte para o inglês e português na tarefa de Reconhecimento de Entidades Nomeadas. Neste sentido, os trabalhos, de maiores impactos para a tarefa de REN, advêm de abordagens baseadas em Redes Neurais Profundas que usam modelos de linguagem pré-treinados como forma de representação das palavras. Com base em toda teoria já dissertada até aqui, serão apresentados os resultados tomando base nas técnicas estado-da-arte.

Este capítulo está dividido em sete seções, nas quais define-se as métricas de avaliação, seção 7.1; resultados usando os modelos *Flair Embeddings*, seção 7.2.1; resultados usando o modelo multilíngue *BERT*, seção 7.2.2; comparação dos resultados alcançados com outros trabalhos 7.2.3; apresentação e discussão dos resultados alcançados no IberLEF 2019 7.3.1; resultados obtidos no domínio da geologia (GeoCorpus), seção 7.3.2; e, por fim, a análise de erro na seção 7.2.4.

7.1 Métricas de Avaliação

7.1.1 Métricas para REN

As avaliações apresentadas neste capítulo seguem os padrões de avaliação do CoNLL-2002 [57]. Também foi usado o script original do CoNLL-2002 em Perl[1] para extrair as métricas. Há um formato padrão de saída para rodar o script de avaliação: cada linha do arquivo de saída (.txt) deve conter um único token, seguido do rótulo predito pelo sistema e o rótulo correto. Um exemplo do formato pode ser visto na tabela 7.1.

Recentes trabalhos para Reconhecimento de Entidades Nomeadas em Português também se utilizaram do CoNLL-2002 para avaliação da performance de seus sistemas, o que torna viável a comparação das abordagens. Segundo o CoNLL-2002 os sistemas de REN devem ser avaliados pela métrica $\mathbf{F}_{\beta=1}$, definada por:

$$\mathbf{F}_{\beta=1} = \frac{(\beta^2 + 1) \cdot \mathit{Precision} \cdot \mathit{Recall}}{\beta^2 \cdot \mathit{Precision} + \mathit{Recall}} \tag{7.1}$$

Como $\beta = 1$, vem:

[1] https://www.perl.org/

$$\mathbf{F}_1 = \frac{2 \cdot \mathit{Precision} \cdot \mathit{Recall}}{\mathit{Precision} + \mathit{Recall}} \tag{7.2}$$

Onde, *Precision* é a porcentagem de EN encontrada corretamente pelo sistema (Equação 7.3). *Recall* é a porcentagem de EN, presente no corpus, que foi encontrada pelo sistema (Equação 7.4). Uma EN está correta, somente quando, há a exata correspondência entre o rótulo anotado e o rótulo de saída do sistema.

$$\mathit{Precision} = \frac{\mathit{Número\ de\ EN\ encontradas\ corretamente\ pela\ Rede}}{\mathit{Número\ de\ EN\ encontradas\ pela\ Rede}} \tag{7.3}$$

$$\mathit{Recall} = \frac{\mathit{Número\ de\ EN\ encontradas\ corretamente\ pela\ Rede}}{\mathit{Número\ de\ EN\ no\ corpus}} \tag{7.4}$$

Tabela 7.1 – Exemplo de formatação do CoNLL-2002

Vincent	B-PER	B-PER
Willem	I-PER	I-PER
van	I-PER	I-PER
Gogh	I-PER	I-PER
nasceu	O	O
no	O	O
dia	O	O
30	B-TMP	B-TMP
de	I-TMP	I-TMP
março	I-TMP	I-TMP
de	I-TMP	I-TMP
1853	I-TMP	I-TMP
em	O	O
Zundert	B-LOC	B-LOC
.	O	O

7.1.2 Métricas para ML

Com relação aos modelos Flair Embeddings, usou-se a métrica de Perplexidade (PPL) para avaliar a qualidade do modelo durante o treinamento. Quanto mais baixo a PPL melhor é o modelo preditivo [31, 56]. A equação 7.5 mostra o cálculo da PPL sobre um corpus de tamanho N.

$$PPL = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i|w_{i-1})}} \tag{7.5}$$

7.2 Avaliação em Domínio Geral

7.2.1 Resultados usando *Flair Embeddings*

Antes de seguir para os resultados, é importante introduzir a notação de empilhamento de *embeddings*. Os experimentos dessa pesquisa se utilizam dessa estratégia de empilhamento que faz a concatenação de diferentes modelos de linguagem. Os experimentos desta seção seguem o padrão de empilhamento de dois modelos de linguagem: *Word Embeddings* + *Flair Embeddings*. Sendo assim, cada token w_i do corpus será representado por um tensor a partir de um empilhamento de *embedding*, ou seja:

$$w_i = \begin{bmatrix} w^{Word\ Embeddings} \\ w^{Flair\ Embeddings} \end{bmatrix} \tag{7.6}$$

As suposições iniciais são que o tamanho, a diversidade dos estilos textuais e a abrangência dos assuntos dos corpora usados para modelar a linguagem pudessem causar variações na efetividade do modelo resultante. Para constatar as suposições, foram avaliadas várias combinações de modelos de linguagem (empilhamentos de *embeddings*). Para esses experimentos iniciais, usou-se o corpus do Primeiro HAREM no Cenário Seletivo para treino da rede neural e avaliou-se a performance da rede no corpus Mini-HAREM. Todos os experimentos de REN foram treinados em uma GPU Tesla K80 na plataforma Google Cloud [2].

Os experimentos foram divididos em dois grandes grupos de avaliação, de acordo com o corpus usado para o treino dos modelos *word embeddings*: os modelos do *Grupo 1* foram treinados usando o brWaC, BlogSet-BR e ptwiki-20190301; e os modelos do *Grupo 2* foram obtidos através do repositório de *embeddings* do NILC (apresentado na seção 4.4.1).

Os grupos 1 e 2 foram subdivididos pelo modelo *Flair Embeddings*. O primeiro conjunto usou o modelo *FlairBBP* (treinado usando o brWaC, BlogSet-BR e ptwiki-20190301); e o segundo conjunto utilizou o *FlairEL* (treinado por Lief (2019) [38]).

Finalmente, existe uma última subdivisão que segue a arquitetura das redes neurais usadas para criar os *Word Embeddings (WE)*. Os *Word Embeddings* usados foram:

[2] https://cloud.google.com/

Word2Vec Skip-gram; Word2Vec CBOW; FastText Skip-gram; e *FastText CBOW*. A Tabela 7.2 detalha as divisões bem como as métricas obtidas.

Tabela 7.2 – Avaliação de diferentes combinações de ML no Cenário Seletivo do HAREM

Grupo	Modelo Flair	WE	Prec	Rec	F1
Grupo 1	FlairBBP	W2V-SKPG	81,40%	79,55%	80,47%
		W2V-CBOW	81,90%	79,76%	80,82%
		FT-SKPG	81,92%	79,38%	80,63%
		FT-CBOW	82,02%	80,10%	81,05%
	FlairEL	W2V-SKPG	81,96%	78,97%	80,43%
		W2V-CBOW	82,27%	79,72%	80,98%
		FT-SKPG	80,91%	78,80%	79,84%
		FT-CBOW	82,12%	79,69%	80,89%
Grupo 2	FlairBBP	W2V-SKPG	**83,38%**	**81,17%**	**82,26%**
		W2V-CBOW	82,06%	80,48%	81,27%
		FT-SKPG	82,03%	79,83%	80,91%
		FT-CBOW	77,64%	75,90%	76,76%
	FlairEL	W2V-SKPG	81,65%	79,97%	80,80%
		W2V-CBOW	82,65%	80,34%	81,48%
		FT-SKPG	83,07%	80,72%	81,88%
		FT-CBOW	76,59%	75,22%	75,89%

De acordo com a tabela 7.2, é possível perceber que a melhor combinação de modelos de linguagem, ou ainda, o melhor empilhamento de *embeddings* é a composição *FlairBBP+Word2Vec Skip-Gram* do NILC ($F_1 = 82,26\%$). Já o pior caso foi usando a combinação *FlairEL+FastText CBOW* ($F_1 = 75,89\%$). Essas informações podem ser observadas graficamente nas figuras 7.1 e 7.2. Também, uma análise mais rica pode ser feita observando a variância e o desvio padrão das medidas F1 de cada grupo.

Observa-se que, com exceção da combinação *FlairEL+W2V-CBOW*, os empilhamentos do *FlairBBP* foram superiores aos do *FlairEL*. Os empilhamentos que usaram o *FlairBBP* também foram mais consistentes: a variância da F1 é de $s^2 = 0,0624$ e o seu desvio padrão foi de $s = 0,2497$; enquanto a variância de F1 do *FlairEL* foi de $s = 0,2727$ e desvio padrão de $s = 0,5222$.

Para o grupo 2, observou-se que há uma variação considerável nos resultados. A F1 do *FlairBBP* teve uma variância de $s^2 = 5,8954$, com um desvio padrão de $s = 2,4280$; enquanto o do *FlairEL* teve uma variância de $s^2 = 7,7520$ com um desvio padrão de $s = 2,7842$. A tabela 7.3 reapresenta os dados de variância e desvio padrão.

Pode-se perceber que o *FlairBBP* apresenta menor variância amostral e desvio padrão das medida F1 quando comparado ao *FlairEL*. Dois fatos podem ser mencionados a partir dessa menor variação: o corpus de treino do *FlairBBP* é cinco vezes maior do que o tamanho do corpus de treino do *FlairEL*; e a perplexidade do *FlairBBP* é ligeiramente

Tabela 7.3 – Variância e desvio padrão das medidas por grupo e modelo *Flair Embeddings*

Grupo	**Modelo**	s^2	s
Grupo 1	FlairBBP	0,0624	0,2497
	FlairEL	0,2727	0,5222
Grupo 2	FlairBBP	5,8954	2,4280
	FlairEL	7,7520	2,7842

menor do que a do *FlairEL*. A Tabela 7.4 apresenta uma comparação do valor de perplexidade (PPL) entre ambos os modelos. Além do mais,conjectura-se que a baixa variação amostral do *FlairBBP* para o Grupo 1 se deve ao fato de que o *Flair Embeddings* e os *Word Embeddings* foram treinados usando o mesmo corpora.

Tabela 7.4 – Perplexidade entre os modelos FlairEL e FlairBBP

Modelo Flair Embeddings		**PPL**
FlairEL	Forward	2,78
	Backward	2,81
FlairBBP	Forward	**2,76**
	Backward	**2,80**

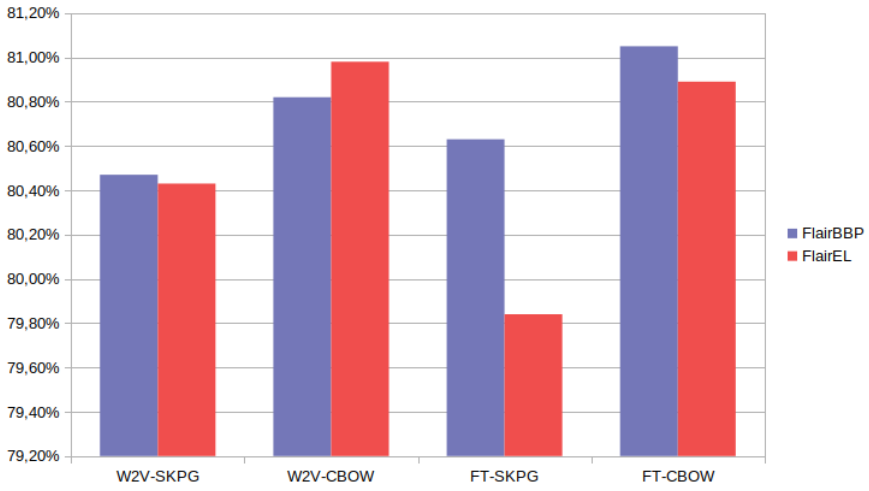


Figura 7.1 – Gráfico de barras das medidas F1 do Grupo 1

Como dito, a melhor combinação foi composta pelos ML *FlairBBP* e *Word2Vec Skip-Gram* do NILC, apesar dos *Word Embeddings (WE)* desenvolvidos nesta pesquisa (seção 5.1) terem sidos treinados com 4,9 bilhões de tokens. Os *WE* do NILC foram treinados com pouco mais de 1 bilhão de tokens. Entende-se que os modelos do NILC produzem melhores resultados devido à vasta diversidade textual presente no corpus de treino. Também deve-se destacar o tamanho do vocabulário do modelo que pode ser outro fator determinante.

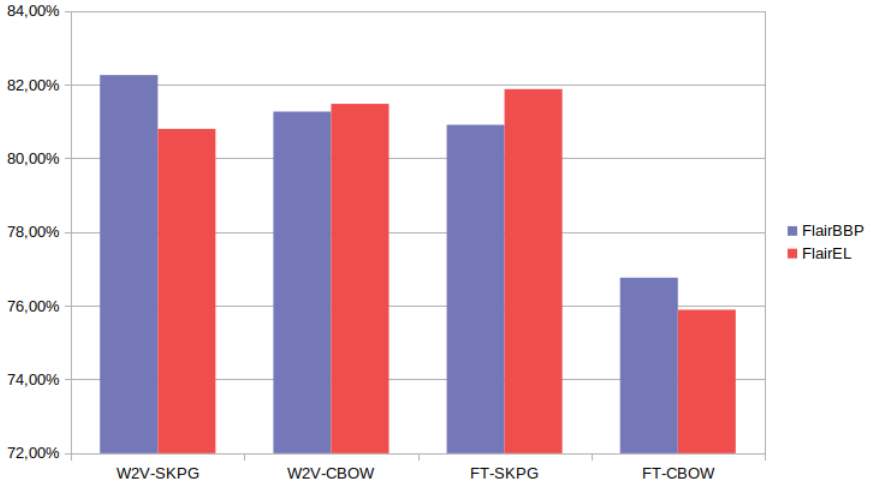


Figura 7.2 – Gráfico de barras das medidas F1 do Grupo 2

Outro ponto é o parâmetro de treino que regula quantas vezes uma palavra deve se repetir para fazer parte do vocabulário do *WE*. Durante o treinamento do *WE* do NILC, um token deve se repetir ao menos cinco vezes no corpus para ser adicionado ao vocabulário. Já os modelos desenvolvidos para esse trabalho requerem, no mínimo, sete vezes. Isso, junto à diversidade do corpus de treino, resultou em um vocabulário maior para o modelo do NILC, uma diferença de 381.329 tokens em relação aos gerados.

Assim como dos Santos et al. [18] e Castro et al. [13], foram realizados testes em dois cenários do HAREM: total e seletivo. A tabela 7.5 apresenta os resultados para o cenário total, e a tabela 7.2.1 apresenta os resultados para o cenário seletivo. Em ambos os casos, usou-se a melhor combinação de MLs, conforme resultado apresentado na tabela 7.2.

Tabela 7.5 – Cenário Total do HAREM

Categoria	**Precision**	**Recall**	**F1**
Geral	74,91%	74,37%	74,64%
ABSTRAÇÃO	53,90%	42,13%	47,29%
ACONTECIMENTO	22,37%	34,00%	26,98%
COISA	54,72%	35,80%	43,28%
LOCAL	81,40%	85,75%	83,52%
OBRA	46,82%	43,09%	44,88%
ORGANIZAÇÃO	67,23%	77,07%	71,82%
OUTRO	10,00%	7,14%	8,33%
PESSOA	82,44%	77,08%	79,67%
TEMPO	91,43%	90,40%	90,91%
VALOR	82,92%	81,90%	82,41%

Tabela 7.6 – Cenário Seletivo do HAREM

Categoria	Prec	Rec	F1
Geral	83,38%	81,17%	82,26%
LOCAL	84,78%	84,68%	84,73%
ORGANIZAÇÃO	72,61%	76,19%	74,35%
PESSOA	85,07%	76,84%	80,75%
TEMPO	93,81%	89,83%	91,77%
VALOR	84,81%	82,21%	83,49%

7.2.2 Resultados usando BERT Embeddings

As tabelas 7.8 e 7.9 apresentam os resultados com um diferente empilhamento de *embeddings*:

$$w_i = \begin{bmatrix} w^{Word2Vec\ Skip\text{-}Gram} \\ w^{BERT\ Embeddings} \end{bmatrix} \tag{7.7}$$

O objetivo com os experimentos que envolvem os modelos BERT (Multilíngue) e o *Word2Vec Skip-Gram* do NILC é comparar recentes e diferentes modelos de linguagem contextualizados. Apenas uma combinação de ML foi tomada para teste e por isso escolheu-se o *Word2Vec Skip-Gram*, que foi o *WE* que melhor se desempenhou nos testes com o *FlairBBP*. A tabela 7.7 exibe um comparativo entre as medidas alcançadas usando o BERT e o *FlairBBP*. Manteve-se os mesmos corpora de treino e teste a fim de comparar resultados.

Pode-se observar que a medida F1, no Cenário Total, não teve uma grande diferença quando comparada ao Cenário Seletivo, em que a diferença foi de 5, 15%. É possível conjecturar que o desempenho do BERT não foi melhor, pelo fato de que o BERT (Multilíngue) usado não foi treinado com um corpus monolíngue, caso do *FlairBBP*. Por exemplo, Rönnqvist et al. [53] mostraram que modelos para os idiomas inglês e alemão têm bom desempenho para a tarefa de geração de linguagem, enquanto o modelo multilíngue tem menor desempenho para a mesma tarefa.

Tabela 7.7 – Comparação entre o FlairBBP e o BERT

ML	Cenário Total F1	Cenário Seletivo F1
FlairBBP	**74,64%**	**82,26%**
BERT (Multilíngue)	72,22%	77,11%

Tabela 7.8 – Resultados usando BERT para Cenário Seletivo

Categoria	Precision	Recall	F1
Geral	77,63%	76,59%	77,11%
LOCAL	78,87%	79,81%	79,34%
ORGANIZAÇÃO	67,05%	71,43%	69,17%
PESSOA	81,35%	75,37%	78,24%
TEMPO	90,51%	80,79%	85,37%
VALOR	73,08%	75,77%	74,40%

Tabela 7.9 – Resultados usando BERT para Cenário Total

Categoria	Precision	Recall	F1
Geral	74,36%	70,19%	72,22%
ABSTRAÇÃO	44,51%	37,06%	40,44%
ACONTECIMENTO	24,64%	34,00%	28,57%
COISA	50,00%	29,01%	36,72%
LOCAL	81,33%	81,24%	81,28%
OBRA	53,38%	37,77%	44,24%
ORGANIZAÇÃO	67,99%	72,31%	70,09%
OUTRO	50,00%	7,14%	12,50%
PESSOA	83,00%	75,98%	79,33%
TEMPO	89,52%	84,46%	86,92%
VALOR	74,10%	75,46%	74,77%

7.2.3 Comparação com o Estado-da-Arte

Nas tabelas 7.11 e 7.10 compara-se os resultados dessa pesquisa em relação aos recentes sistemas para REN para o Português. Como já abordado, todos os trabalhos usaram o corpus do Primeiro HAREM e Mini-HAREM para a treino e teste, respectivamente. A abordagem do uso dos modelos de linguagem contextualizados, proposto nesta pesquisa, mostra um melhoramento nas medidas F1: +4,31% para o Cenário Total e +5,99% para o Cenário Seletivo.

Ainda sobre as tabelas 7.11 e 7.10, duas perspectivas interessantes podem ser discutidas:

1. O tamanho dos conjuntos de treino dos modelos de linguagem;
2. A efetividade dos ML Contextualizados;

Sobre o primeiro ponto, de fato as suposições iniciais (tamanho do corpus e diversidade textual) sobre o treinamento dos ML causam impacto na tarefa de REN. A principal diferença dentre as abordagens comparadas é o tamanho e diversidade textual do corpus

usado para treinar os ML. A CharWNN [18] usa um modelo *WE*, treinado com 401 milhões de tokens, e foi ultrapassado por Castro et al. [13], que utilizou os *Word Embeddings* do NILC, treinados com 1 bilhão de palavras. Os empilhamentos de ML propostos, por sua vez, ultrapassam o sistema de Castro et al. [13], combinando os modelos do NILC com o *FlairBBP*.

No segundo ponto, também fica demonstrado o poder de representação que o *Flair Embeddings* tem. Também é importante ressaltar que o tamanho do corpus usado para treinar o *FlairBBP* teve efetividade frente ao *FlairEL*, treinado em um corpus menor quase 5 vezes.

Tabela 7.10 – Comparação com o estado-da-arte para o Cenário Seletivo

Abordagem	**Cenário Seletivo**			
	Precision	**Recall**	**F1**	Δ
BiLSTM-CRF+FlairBBP	**83,38%**	**81,17%**	**82,26%**	+5,99%
BiLSTM-CRF[13]	78,26%	74,39%	76,27%	+5,04%
CharWNN[18]	73,98%	68,68%	71,23%	

Tabela 7.11 – Comparação com o estado-da-arte para o Cenário Total

Abordagem	**Cenário Total**			
	Precision	**Recall**	**F1**	Δ
BiLSTM-CRF+FlairBBP	**74,91%**	**74,37%**	**74,64%**	+4,31%
BiLSTM-CRF[13]	72,28%	68,03%	70,33%	+4,92%
CharWNN[18]	67,16%	63,74%	65,41%	

7.2.4 Análise de Erro

Nesta seção, será apresentada uma análise de erro sobre as saídas da rede neural. Essa foi feita usando a saída da melhor combinação de ML encontrada na seção anterior para o corpus Mini-HAREM. As análises foram feitas com base na interpretação das matrizes de confusão dos Cenários Seletivo (tabela 7.12) e Total (tabela 7.13). Para tanto, as notações BIO foram removidas, ou seja, a verificação de correspondência (entre o rótulo correto e o predito) é feita token a token.

Primeiro, no Cenário Seletivo, pode-se ver que os casos mais críticos foram nas categorias LOCAL e ORGANIZAÇÃO. Pela tabela 7.12, a rede classificou 71 vezes um token como LOCALIZAÇÃO e, na verdade, era ORGANIZAÇÃO. Analogamente, a rede classificou 66 vezes um token como ORGANIZAÇÃO e, na verdade, era LOCALIZAÇÃO.

No Cenário Total, pode-se perceber que o caso mais crítico foi entre as categorias OBRA e ABSTRAÇÃO, em que a rede classificou 247 tokens como OBRA e, na verdade, a categoria era ABSTRAÇÃO. Outro caso crítico é a categoria OUTRO, em que a rede teve vários casos de *falso positivo*. Em geral, a rede teve muitos *falsos positivos* entre as classes OBRA, ABSTRAÇÃO, ACONTECIMENTO, COISA e OUTRO.

Tabela 7.12 – Matriz de Confusão para o Cenário Seletivo do Mini-HAREM

	TMP	**PES**	**ORG**	**LOC**	**VAL**
TMP	536	0	0	3	2
PES	0	1269	34	25	3
ORG	0	19	939	71	0
LOC	0	9	66	1045	0
VAL	3	0	3	1	523

Tabela 7.13 – Matriz de Confusão para o Cenário Total do Mini-HAREM

	TMP	**PES**	**ORG**	**LOC**	**OBR**	**VAL**	**ABS**	**ACO**	**COI**	**OTR**
TMP	539	0	0	4	4	2	0	13	0	0
PES	0	1298	38	36	6	2	23	10	1	0
ORG	0	20	981	86	5	0	13	8	2	0
LOC	5	7	76	1080	14	0	8	4	6	1
OBR	3	27	31	17	240	10	20	77	8	4
VAL	5	0	2	1	0	535	0	0	0	2
ABS	3	19	34	14	247	0	157	11	3	0
ACO	0	1	31	3	37	3	10	83	0	0
COI	0	1	29	16	15	4	9	4	70	0
OTR	1	1	5	1	3	2	3	0	0	5

Outros casos específicos foram selecionados do mesmo conjunto de dados analisado e apresentado na tabela 7.14. Os pontos de 1 a 11 discutem cada exemplo apresentado na tabela.

1. A entidade “Grossgöpfritz” foi classificada corretamente pela rede, apesar de não ser uma palavra de comum na língua portuguesa;

2. A entidade “Schüssel”, que também não é comum no português e não aparece no corpus de treino foi classificada corretamente;

3. Entidade composta por cinco tokens e todos foram classificados corretamente. Essa entidade não aparece no corpus de treino;

4. Aqui, há três EN, entre elas, as duas primeiras foram classificadas corretamente, entretanto, a rede classificou o token “CE” como ORGANIZAÇÃO, mas a classificação correta é LOCAL. Em outras sentenças, a rede conseguiu classificar corretamente a sigla “SP”;

5. A EN, nessa sentença, tem dez tokens, mas a rede conseguiu apenas identificar e classificar, corretamente, os seis primeiros. Um dos possíveis motivos é que um dos token da EN é uma virgula e normalmente esse token recebe anotação OUTSIDE;

6. A rede conseguiu classificar corretamente as siglas IBET e USP. A terceira EN foi identificada e classificada incorretamente. A rede classificou a menção “Oliveira Neves” como PESSOA, que na Coleção Dourada (CD) está classificada como ORGANIZAÇÃO. O final da entidade, os tokens “e Associados” não foram identificados e portanto não classificados;

7. Neste exemplo, a rede conseguiu identificar corretamente, porém classificou a EN como LOCALIZAÇÃO, enquanto a classe correta é ORGANIZAÇÃO;

8. Neste exemplo, a rede classificou o token “4ª” como categoria VALOR, entretanto, neste caso, não havia anotação para o token. No corpus de treino e teste, é possível identificar várias outras menções como essa (do exemplo) que foram classificadas como VALOR;

9. Este é um exemplo uma classificação correta da categoria TEMPO;

10. Aqui a rede não identificou a EN “Quarta- feira” (dois tokens), que é muito semelhante ao exemplo anterior, porém não foi identificada e, consequentemente, não classificada. Na sequência, a EN “Cruzeiro”, que segundo os guias do HAREM é uma EN da classe PESSOA, não foi identificada. Novamente, a rede não conseguiu identificar a EN “Quarta-feira” da classe TEMPO;

11. Neste exemplo, a entidade “Segunda Guerra Mundial” não foi identificada pela rede. Mas há também uma inconsistência de classificação por parte do HAREM. No conjunto de treino (Primeiro HAREM), a menção “Segunda Guerra Mundial” é classificada como ACONTECIMENTO, já no Mini-HAREM é classificada como TEMPO.

7.3 Avaliação em domínios específicos

7.3.1 Resultados no domínio clínico e policial (IberLEF 2019)

Além das avaliações no corpus do HAREM que tem um caráter mais geral, também aplicou-se a abordagem de empilhamento de ML em avaliações de REN, em domínios específicos. Neste sentido, participou-se da tarefa compartilhada *“Portuguese Named Entity Recognation and Relation Extraction Tasks (NerRelIberLEF2019)”* [11] no *Iberian Languages Evaluation Forum (IberLEF 2019)*. A tarefa *NerRelIberLEF2019* é dividida em três

Tabela 7.14 – Exemplos de sentenças do Mini-HAREM classificadas pela rede

Exemplo	Sentença
1	e um anão de jardim de **Grossgöpfritz** [B-LOC], salvo quanto
2	E **Schüssel** [B-PES], chamado de o sonho à realidade?
3	À **Academia** [B-ORG] **de** [I-ORG] **Belas-Artes** [I-ORG] **de** [I-ORG] **Viena** [I-ORG]
4	**Reginaldo** [B-PES] **Duarte** [B-PES] pertence a uma família de políticos de **Juazeiro** [B-LOC] **do** [I-LOC] **Norte** [I-LOC] (**CE** [B-ORG]), mas nunca havia disputado eleição.
5	É que, apesar de todas as tentativas feitas por os agentes da **Direção** [B-ORG] **Central** [I-ORG] **de** [I-ORG] **Investigação** [I-ORG] **de** [I-ORG] **Corrupção** [I-ORG], **Fraudes e Infracções Económico-Financeiras**, nunca fez qualquer revelação que pudesse incriminar outras pessoas.
6	pós-graduando em direito tributário - **IBET** [B-ORG] / **USP** [B-ORG] e sócio da **Oliveira** [B-PES] **Neves** [I-PES] **e Associados**
7	Encontrará as respostas na página do programa do **Teatro** [B-LOC] **Municipal** [I-LOC] **Maria** [I-LOC] **Matos** [I-LOC] .
8	Naquela altura, ter o exame da 4ª [B-VAL] classe era bastante
9	em **Quarta** [B-TMP] **,** **Julho** [B-TMP] **32** [I-TMP] **2002** [I-TMP] @ **11:21** [B-TMP]
10	**Quarta- feira** tem **Cruzeiro** em **BH** [B-LOC] O time deverá permanecer na capital mineira até a próxima **Quarta-feira**
11	Isto depois da **Segunda Guerra Mundial**

sub-tarefas: uma tarefa sobre REN e outras duas sobre Extração de Relações. Por esse motivo, será feita referência à sub-tarefa de REN como *Tarefa 1*.

Nesta seção, serão descritos os resultados obtidos como participante da *Tarefa 1* em relação às demais equipes que integraram a tarefa compartilhada. O sistema submetido é o melhor modelo gerado pelo treinamento da BiLSTM-CRF, apresentado na seção 7.2.1, no cenário seletivo do HAREM. Ou seja, o sistema é o modelo preditivo gerado pelo treinamento da BiLSTM-CRF com o *FlairBBP* e o *WE Word2Vec Skip-Gram* do NILC.

A tabela 7.15 apresenta os resultados alcançados, segundo o script CoNLL-2002 [57]. Os gráficos nas figuras 7.3 e 7.4 exibem as medidas F1 para cada corpus de avaliação. Os números no eixo das abscissas significam os sistemas participantes da *Tarefa 1*:

1. BiLSTM-CRF-ELMo [14];
2. CRF-LG [49];

3. NLPyPort [22];

4. CVT [24];

5. BiLSTM-CRF-FlairBBP [60] (Abordagem de empilhamento);

6. Linguakit [24];

A tabela 7.16 apresenta os corpora usados para treinar cada sistema participante. O sistema Linguakit não aparece na tabela porque tem funcionamento baseado em eurísticas.

O sistema proposto apresentou resultados competitivos em relação ao sistema que atingiu a maior medida F1 para o *Police Dataset*, diferenciando apenas 2,8%. O bom desempenho nesse conjunto de dados pode ser atribuído aos empilhamentos de *embeddings* e ao fato de que os textos usados para construir o *Police Dataset* são muito bem estruturados, como aqueles usados para construir a coleção HAREM [11].

No caso do *Clinical Dataset*, a diferença entre a medida F1 do sistema proposto e o sistema com a maior medida F1 foi de 12, 98%. Acredita-se que isso se deve ao fato das particularidades estruturais e da linguagem do *Clinical Dataset*. Esse corpus é composto de evoluções médicas que contém abreviaturas, termos médicos e várias outras particularidades encontradas em textos de ambientes hospitalares. Textos como esses diferem muito do estilo tradicional para a tarefa de REN em português, e essas diferenças não foram levadas em consideração durante o treinamento da rede.

Tabela 7.15 – Resultados nos corpora da Tarefa 1

Corpus	**Categoria**	**Prec**	**Rec**	**F1**
Police Dataset	PESSOA	94,21%	82,82%	88,15%
Clinical Dataset	PESSOA	22,08%	41,46%	28,81%

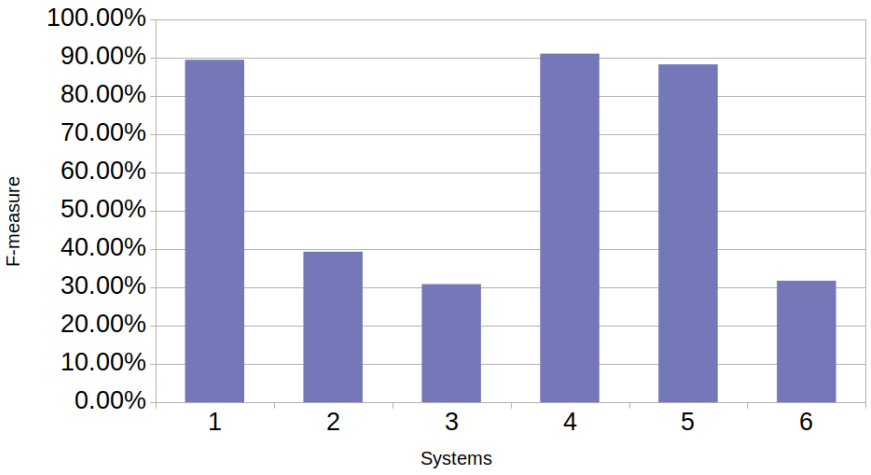


Figura 7.3 – Avaliação no corpus *Police Dataset* - Classe PESSOA

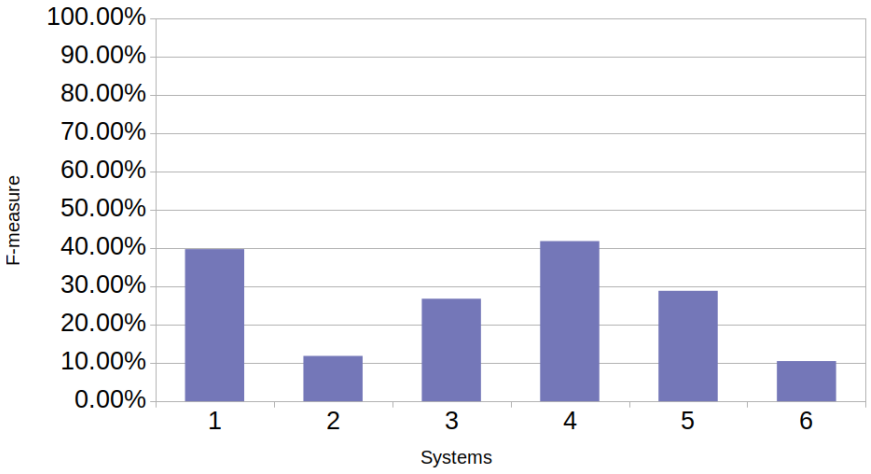


Figura 7.4 – Avaliação no corpus *Clinical Dataset* - Classe PESSOA

Tabela 7.16 – Corpora de Treino por Sistema

Sistema	BiLSTM CRF ELMo	CRF-LG	NLPyPort	CVT (Embeddings)	BiLSTM CRF FlairBBP
Corpora de Treino	WikiNER I HAREM MiniHAREM LeNER-Br Paramopama Datalawyer	I HAREM II HAREM MiniHAREM	II HAREM	LeNER-Br II HAREM FreeLing Corpus	I HAREM

7.3.2 Resultados no domínio geológico (GeoCorpus)

Nesta seção, serão apresentados e discutidos os resultados obtidos no GeoCorpus-2. Continuou-se usando a rede BiLSTM-CRF, bem como os modelos de linguagem *Word2Vec Skip-Gram (W2V-SKPG)* do NILC e o *FlairBBP*. Além desses recursos, também foram incluídos o modelo *Word Embeddings GeoWE* e o modelo $Flair_{GeoFT}$.

Assim, os experimentos foram organizados em oito testes, cada um envolvendo um único ML ou empilhamento. Primeiramente, o desempenho do modelo *WE* de domínio geral (W2V-SKPG) e do modelo *WE* de domínio geológico (GeoWE) foram avaliados utilizando-os separadamente.

O mesmo foi feito para os modelos *Flair Embeddings* de domínio geral (*FlairBBP*) e específico para Geologia ($Flair_{GeoFT}$).

Finalmente, o último experimento envolveu o empilhamento de cada modelo *Flair Embeddings* com cada modelo *WE*, o que resultou em quatro empilhamentos de ML. A tabela 7.17 apresenta os resultados para cada uma dessas experiências.

Como pode ser visto, a menor medida F1 foi a alcançada quando usado somente o *GeoWE*. Por outro lado, o melhor caso ocorre quando é novamente aplicada a estratégia de empilhar ML. Porém, há de se notar, que ao empilhar o *FlairBBP* com os *WE*, o empilhamento resultante apresenta piores medidas.

Esse resultado significa que o *GeoWE* não é adequado para REN em seu estado atual. Acredita-se que seja devido ao tamanho e pré-processamento realizado no corpus de treino do *WE*. Como já construída essa hipótese nessa pesquisa, a qualidade do corpus tem grande peso no resultado final de um modelo de linguagem, principalmente para os *WE*. Neste sentido, um vasto e rico corpus em português para a Geologia ainda é um problema em aberto.

No teste em que foi usado apenas modelos *Flair Embeddings*, nota-se que o melhor desempenho foi para o $FlairBBP_{GeoFT}$, mostrando que o processo de afinamento do modelo geral *FlairBBP* foi bem-sucedido.

Finalmente, apesar de os resultados para as combinações de empilhamentos do *GeoWE* terem sido mais baixos, algumas observações interessantes podem ser feitas. Os resultados alcançados com a composição *GeoWE* + *FlairBBP* e *GeoWE* + $FlairBBP_{GeoFT}$ foram significativamente diferentes, o que é um contraste quando se olha para os empilhamentos que usaram o *W2V-SKPG*, que foram essencialmente semelhantes. Empilhar o *GeoWE* com o $FlairBBP_{GeoFT}$ resultou em um crescimento $4,9\%$ em medida F1, com relação ao *GeoWE* + *FlairBBP*. Já quanto ao *W2V-SKPG* + $FlairBBP_{GeoFT}$ foi alcançada uma medida F1 de apenas $0,59\%$ superior ao *W2V-SKPG* + *FlairBBP*.

Em geral, os resultados encontrados superam os apresentados no trabalho de Amaral (2017) [16], em que foi alcançada uma medida F1 de 54, 33% com um classificador CRF. Porém, o sistema foi avaliado por meio de validação cruzada e não usa o script CoNNL-2002, como usado nestas avaliações, ou seja, não é possível uma comparação estrita.

Tabela 7.17 – Tabela com resultados no GeoCorpus

Modelo de Linguagem		**Precision**	**Recall**	**F1**
Word Embeddings	GeoWE	73,31%	42,38%	53,71%
	W2V-SKPG	80,27%	64,18%	71,33%
Flair Embeddings	FlairBBP	85,97%	80,41%	83,10%
	FlairBBP$_{GeoFT}$	86,03%	82,45%	84,20%
Empilhamento de Embeddings	GeoWE+FlairBBP	86,87%	72,16%	78,84%
	W2V-SKPG+FlairBBP	86,78%	81,47%	84,04%
	GeoWE+FlairBBP$_{GeoFT}$	86,35%	81,29%	83,74%
	W2V-SKPG+FlairBBP$_{GeoFT}$	**86,63%**	**82,71%**	**84,63%**

8. CONSIDERAÇÕES FINAIS

Neste capítulo, serão apresentado as conclusões com base nos resultados obtidos, bem como a listagem das principais contribuições advindas com os esforços empenhados no desenvolvimento desta pesquisa. Por fim, são apresentados tópicos em aberto que servem como perspectivas para futuros trabalhos.

8.1 Conclusões

Nesta dissertação, foram apresentados os resultados alcançados com o desenvolvimento desta pesquisa. Entre eles, apresentou-se uma análise de como diferentes combinações de modelos *Word Embeddings* e *Flair Embeddings* impactam os resultados da tarefa de REN. A avaliação extrínseca foi feita com base em 16 possíveis combinações de ML. Desses, o que obteve melhor resultado foi a combinação que usa o *FlairBBP*. Os resultados experimentais demonstram que o tamanho e a diversidade do corpora, assim como o tipo de treinamento usado para criar o modelo de linguagem, são fatores potencialmente influentes para tarefa de REN.

Observa-se também que a topologia de rede neural BiLSTM-CRF tem sido bem sucedida para tarefas de classificação sequencial (ou classificação de tokens). Pode-se ainda inferir que o fator determinante para o bom desempenho da BiLSTM-CRF é a qualidade das representações de palavras fornecidas nas camadas de *embeddings*. Esse fato fica visível ao olhar a abordagem de Castro et al. [13], que usa a BiLSTM-CRF apenas com representações *WE*. O mesmo acontece na evolução das abordagens de solução para REN em língua inglesa: Lample et al. também usou a BiLSTM-CRF [36] com *WE*, que foi superado por Peters et al. [48] usando *WE* e ML contextualizados (ELMo), sendo sequencialmente superado por Akbik et al. [4] fazendo uso de *WE* e *Flair Embeddings*.

As avaliações extrínsecas também contemplaram as representações de linguagem fornecidas pelo modelo BERT (Multilíngue), o qual conseguiu apenas resultados competitivos com o estado-da-arte, mas não se conseguiu ultrapassar. Sendo assim, é ainda mais vantajoso usar modelos *Flair Embeddings* para a tarefa de REN, pois demandam menos poder computacional e são mais simples de programar.

Seguindo as análises realizadas, nota-se que, em geral, os *Flair Embeddings* se mostram muito eficazes quando aplicados para domínios específicos como é o caso do *Police Dataset* e *GeopCorpus-2*.

Contudo, apresentou-se uma abordagem de ponta capaz de produzir um novo estado-da-arte para o Reconhecimento de Entidades Nomeadas em Língua Portuguesa. O resultado foi atingido a partir do uso de um modelo pré-treinado *Word Embeddings* e

um recente tipo de modelo de linguagem contextualizado ao qual gerou-se e nomeou-se como *FlairBBP*. A abordagem valeu-se ainda de uma rede neural profunda BiLSTM-CRF, receptora dos ML de alta representatividade, usando essas incorporações para produzir a classificação final das palavras.

8.2 Contribuições

1. Organização e pré-processamento de um grande corpus de textos em Português (BR e PT) para treinar Modelos de Linguagem;
2. Revisão e transformação do GeoCorpus para o formato CoNLL-2002;
3. Criação de dois corpora para o REN nos domínios policial (*Police Dataset*) e saúde (*Clinical Dataset*);
4. Geração e disponibilização de novos Modelos de Linguagem para o Português:
    - Word2Vec: Skip-Gram e CBOW;
    - FastText: Skip-Gram e CBOW;
    - FlairBBP *Forward*;
    - FlairBBP *Backward*;
    - *FlairBBP*$_{GeoFT}$ *Backward*;
    - *FlairBBP*$_{GeoFT}$ *Forward*;
5. Disponibilização de scripts de treino e avaliação para reprodução de experimentos;
6. Disponibilização de um modelo preditivo no GitHub[1]. O modelo preditivo é produto do processo de treino da Rede Neural. Disponibilizou-se o melhor modelo que combina os ML *FlairBBP* + *Word2Vec Skip-Gram*;
7. Avaliação extrínseca e análise do impacto dos ML Contextualizados para a tarefa de REN em Português;
8. Avaliação extrínseca em corpora de domínio (*Clinical Dataset, Police Dataset* e o GeoCorpus-2);
9. Avaliação nos corpora da Primeira Avaliação do HAREM e, consequentemente, alcance de um novo estado-da-arte para a tarefa de REN em Português;
10. Artigos publicados:

[1] https://github.com/jneto04/ner-pt

- Santos, J., Consoli, B., dos Santos, C., Terra, J., Collonini, S., Vieira, R.: *Assessing the impact of contextual embeddings for portuguese named entity recognition*. In:8th Brazilian Conference on Intelligent Systems, BRACIS 2019, Salvador, Brasil, Outubro 15-18, 2019. pp. 437–442.
- Santos, J.; Terra, J.; Consoli, B. S.; Vieira, R. *Multidomain contextual embeddings for named entity recognition*. In: Proceedings of the Iberian Languages Evaluation Forum co-located with 35th Conference of the Spanish Society for Natural Language Processing, IberLEF@SEPLN 2019, Bilbao, Espanha, Setembro 24, 2019, pp. 434–441.
- Collovini, S.; Santos, J. F. S.; Consoli, B. S.; Terra, J.; Vieira, R.; Quaresma, P.; Souza, M.;Claro, D. B.; Glauber, R. *Iberlef 2019 portuguese named entity recognition and relation extraction tasks*. In: Proceedings of the Iberian Languages Evaluation Forum co-located with 35th Conference of the Spanish Society for Natural Language Processing, IberLEF@SEPLN 2019, Bilbao, Spain, Setembro 24, 2019., 2019, pp. 390–410.
- Consoli, B. S.; Santos, J. F. S.; de Abreu, S. C.; Vieira, R. *Análise da capacidade identificação de paráfrase em ferramentas de resolução de correferência*, Linguamática, vol. 10–2, 2018, pp. 45–51.

8.3 Trabalhos Futuros

Apesar dos resultados alcançados, ainda há potenciais trabalhos a serem desenvolvidos. Os tópicos de 1 a 7 listam os principais trabalhos a serem realizadas:

1. Existe outras variedades de corpora para REN em Português que merecem ser investigados e estudados, por exemplo:
   - WikiNER: é um corpus para REN automaticamente anotado e foi gerado a partir de páginas da Wikipedia [46];
   - Paramopama: formado por textos da wikipédia e anotado automaticamente com revisão [32];
   - LeNER-Br: formado por textos jurídicos de tribunais superiores e estaduais [12];
2. Os modelos *Word Embeddings*, contemplados nesta pesquisa têm 300 dimensões. Mas exitem modelos *WE* com 600 e 1000 dimensões. Nesse sentido, há experimentos a se fazer usando maior dimensionalidade.
3. Os modelos *Word Embeddings*, gerados para compor os cenários de avaliação desta pesquisa, foram avaliados apenas extrinsecamente nos experimentos já apresenta-

dos. Nesse sentido, ainda há um trabalho de avaliação intrínseca dos modelos *WE*, que devem seguir os padrões usados por Hartmann et al. [28].

4. Como mostra a tabela 7.16, usou-se apenas um corpus de treino na abordagem proposta, entretanto há outros corpora que podem ser usados com intenção de melhorar os resultados na *Tarefa 1* (IberLEF).

5. Mostrou-se que os modelos *Flair Embeddings* causam grande impacto na tarefa de REN, mas ainda é uma incógnita o que realmente os modelos *Flair Embeddings* aprendem. Akbik et al. [4] afirma que esses modelos captam informações sintáticas e semânticas, mas não esclarece quais. No caso do BERT, há um recente trabalho em que apresentou-se que o BERT captura informações linguísticas de uma maneira composicional, imitando estruturas clássicas de árvores estruturadas [30].

6. Apesar dos modelos preditivos desenvolvidos nesta pesquisa estarem disponíveis para uso no GitHub, é fundamental existir um serviço online com uma interface gráfica para uso do modelo de reconhecimento das entidades.

7. Alguns estudos recentes mostram que usar estratégias de *Multi-Task Learning (MTL)* tem melhorado os resultados para tarefas de REN em domínios específicos [2, 9, 55].

REFERÊNCIAS BIBLIOGRÁFICAS

[1] Abadi, M.; Barham, P.; Chen, J.; Chen, Z.; Davis, A.; Dean, J.; Devin, M.; Ghemawat, S.; Irving, G.; Isard, M.; Kudlur, M.; Levenberg, J.; Monga, R.; Moore, S.; Murray, D. G.; Steiner, B.; Tucker, P.; Vasudevan, V.; Warden, P.; Wicke, M.; Yu, Y.; Zheng, X. "Tensorflow: a system for large-scale machine learning". In: Proceedings of the 12th Symposium on operating systems design and implementation, 2016, pp. 265–283.

[2] Aguilar, G.; Maharjan, S.; Monroy, A. P. L.; Solorio, T. "A multi-task approach for named entity recognition in social media data". In: Proceedings of the 3rd Workshop on noisy user-generated text, 2017, pp. 148–153.

[3] Akbik, A.; Bergmann, T.; Blythe, D.; Rasul, K.; Schweter, S.; Vollgraf, R. "FLAIR: an easy-to-use framework for state-of-the-art NLP". In: Proceedings of the Conference of the north american chapter of the association for computational linguistics, 2019, pp. 54–59.

[4] Akbik, A.; Blythe, D.; Vollgraf, R. "Contextual string embeddings for sequence labeling". In: Proceedings of the 27th International conference on computational linguistics, 2018, pp. 1638–1649.

[5] Bernardini, S.; Baroni, M.; Evert, S. "Wacky! Working papers on the Web as Corpus". Gedit Edizioni, 2006, 224p.

[6] Boureau, Y.-L.; Ponce, J.; LeCun, Y. "A theoretical analysis of feature pooling in visual recognition". In: Proceedings of the 27th international conference on machine learning, 2010, pp. 111–118.

[7] Buck, C.; Heafield, K.; van Ooyen, B. "N-gram counts and language models from the common crawl". In: Proceedings of the 9th International conference on language resources and evaluation, 2014, pp. 3579–3584.

[8] Chiu, J. P. C.; Nichols, E. "Named entity recognition with bidirectional lstm-cnns", *Transactions of the association for computational linguistics*, vol. 4, Jul 2016, pp. 357–370.

[9] Collobert, R.; Weston, J. "A unified architecture for natural language processing: deep neural networks with multitask learning". In: Proceedings of the 25th International conference on machine learning, 2008, pp. 160–167.

[10] Collobert, R.; Weston, J.; Bottou, L.; Karlen, M.; Kavukcuoglu, K.; Kuksa, P. "Natural language processing (almost) from scratch", *Journal of machine learning research*, vol. 12, Ago 2011, pp. 2493–2537.

[11] Collovini, S.; Santos, J.; Consoli, B.; Terra, J.; Vieira, R.; Quaresma, P.; Souza, M.; Claro, D. B.; Glauber, R. “Iberlef 2019 portuguese named entity recognition and relation extraction tasks”. In: Proceedings of the 35th conference of the spanish society for natural language processing, 2019, pp. 390–410.

[12] de Araujo, P. H. L.; de Campos, T. E.; de Oliveira, R. R.; Stauffer, M.; Couto, S.; Bermejo, P. “Lener-br: a dataset for named entity recognition in brazilian legal text”. In: Proceedings of the 13th International conference on the computational processing of the portuguese language, 2018, pp. 313–323.

[13] de Castro, P. V. Q.; da Silva, N. F. F.; da Silva Soares, A. “Portuguese named entity recognition using LSTM-CRF”. In: Proceeding of the 13th International conference on the computational processing of portuguese, 2018, pp. 83–92.

[14] de Castro, P. V. Q.; da Silva, N. F. F.; da Silva Soares, A. “Contextual representations and semi-supervised named entity recognition for portuguese language”. In: Proceedings of the 35th Conference of the spanish society for natural language processing, 2019, pp. 411–420.

[15] Devlin, J.; Chang, M.; Lee, K.; Toutanova, K. “BERT: pre-training of deep bidirectional transformers for language understanding”, *Computing research repository - arXiv*, vol. abs/1810.04805, Mai 2018, pp. 16.

[16] do Amaral, D. O. F. “Reconhecimento de entidades nomeadas na área da geologia: bacias sedimentares brasileiras”, Tese de doutorado, Programa de Pós-Graduação em Ciência da Computação, PUCRS, 2017, 109p.

[17] do Amaral, D. O. F.; Vieira, R. “NERP-CRF: uma ferramenta para o reconhecimento de entidades nomeadas por meio de conditional random fields”, *Linguamática*, vol. 6–1, 2014, pp. 41–49.

[18] dos Santos, C. N.; Guimarães, V. “Boosting named entity recognition with neural character embeddings”. In: Proceedings of the 50th Named entity workshop, 2015, pp. 25–33.

[19] dos Santos, H. D. P.; Ulbrich, A. H. D. P. S.; Woloszyn, V.; Vieira, R. “An initial investigation of the charlson comorbidity index regression based on clinical notes”. In: Proceeding of the 31st IEEE international symposium on computer-based medical systems, 2018, pp. 6–11.

[20] dos Santos, H. D. P.; Woloszyn, V.; Vieira, R. “Blogset-br: a brazilian portuguese blog corpus”. In: Proceedings of the 11th International conference on language resources and evaluation, 2018, pp. 661–664.

[21] Elkan, C. “Log-linear models and conditional random fields”, Tutorial notes, Department of computer science and engineering, UCSD, 2008, 30p.

[22] Ferreira, J.; Oliveira, H. G.; Rodrigues, R. “Nlpyport: named entity recognition with CRF and rule-based relation extraction”. In: Proceedings of the 35th Conference of the spanish society for natural language processing, 2019, pp. 468–477.

[23] Filho, J. A. W.; Wilkens, R.; Idiart, M.; Villavicencio, A. “The brwac corpus: a new open resource for brazilian portuguese”. In: Proceedings of the 11th International conference on language resources and evaluation, 2018, pp. 4339–4344.

[24] Gamallo, P.; García, M.; Martín-Rodilla, P. “NER and open information extraction for portuguese: notebook for iberlef 2019 portuguese named entity recognition and relation extraction tasks”. In: Proceedings of the 35th Conference of the spanish society for natural language processing, 2019, pp. 457–467.

[25] Gomes, D.; Cordeiro, F.; Evsukoff, A. “Word embeddings em português para o domínio específico de óleo e gás”. In: Proceedings of the 19th Rio oil & gas expo and conference, 2018, pp. 10.

[26] Grave, E.; Mikolov, T.; Joulin, A.; Bojanowski, P. “Bag of tricks for efficient text classification”. In: Proceedings of the 15th Conference of the european chapter of the association for computational linguistics, 2017, pp. 427–431.

[27] Graves, A.; Jaitly, N.; Mohamed, A.-r. “Hybrid speech recognition with deep bidirectional lstm”. In: Proceedings of the Automatic speech recognition and understanding, 2013, pp. 273–278.

[28] Hartmann, N.; Fonseca, E. R.; Shulby, C.; Treviso, M. V.; Silva, J.; Aluísio, S. M. “Portuguese word embeddings: evaluating on word analogies and natural language tasks”. In: Proceedings of the 11th Brazilian symposium in information and human language technology, 2017, pp. 122–131.

[29] Hochreiter, S.; Schmidhuber, J. “Long short-term memory”, *Neural computation*, vol. 9–8, Nov 1997, pp. 1735–1780.

[30] Jawahar, G.; Sagot, B.; Seddah, D. “What does BERT learn about the structure of language?” In: Proceedings of the 57th Annual meeting of the association for computational linguistics, 2019, pp. 3651–3657.

[31] Jelinek, F.; Mercer, R. L.; Bahl, L. R.; Baker, J. K. “Perplexity—a measure of the difficulty of speech recognition tasks”, *The journal of the acoustical society of america*, vol. 62–S1, Dez 1977, pp. S63–S63.

[32] Júnior, C. M.; Macedo, H.; Bispo, T.; Santos, F.; Silva, N.; Barbosa, L. “Paramopama: a brazilian-portuguese corpus for named entity recognition”. In: Anais do XII Encontro nacional de inteligência artificial e computacional, 2015, pp. 218–223.

[33] Jurafsky, D.; Martin, J. H. “Speech & language processing”. Prentice Hall, 2019, 988p.

[34] Kim, Y. “Convolutional neural networks for sentence classification”. In: Proceedings of the Conference on empirical methods in natural language processing, 2014, pp. 1746–1751.

[35] Lafferty, J. D.; McCallum, A.; Pereira, F. C. N. “Conditional random fields: probabilistic models for segmenting and labeling sequence data”. In: Proceedings of the 18th International conference on machine learning, 2001, pp. 282–289.

[36] Lample, G.; Ballesteros, M.; Subramanian, S.; Kawakami, K.; Dyer, C. “Neural architectures for named entity recognition”. In: Proceedings of the Conference of the north american chapter of the association for computational linguistics: human language technologies, 2016, pp. 260–270.

[37] Levy, O.; Goldberg, Y. “Dependency-based word embeddings”. In: Proceedings of the 52nd Annual meeting of the association for computational linguistics, 2014, pp. 302–308.

[38] Lief, E. “Deep contextualized word embeddings from character language models for neural sequence labeling”, Tese de doutorado, Computer Science, CU, 2019, 102p.

[39] Ling, W.; Dyer, C.; Black, A. W.; Trancoso, I. “Two/too simple adaptations of word2vec for syntax problems”. In: Proceedings of the Conference of the north american chapter of the association for computational linguistics: human language technologies, 2015, pp. 1299–1304.

[40] Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. “Efficient estimation of word representations in vector space”. In: Proceedings of the 1st International conference on learning representations, 2013, pp. 12.

[41] Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; Dean, J. “Distributed representations of words and phrases and their compositionality”. In: Proceedings of the Advances in neural information processing systems, 2013, pp. 3111–3119.

[42] Mikolov, T.; Yih, W.-t.; Zweig, G. “Linguistic regularities in continuous space word representations”. In: Proceedings of the Conference of the north american chapter of the association for computational linguistics: human language technologies, 2013, pp. 746–751.

[43] Milidiú, R. L.; Duarte, J. C.; Cavalcante, R. “Machine learning algorithms for portuguese named entity recognition”, *Inteligencia Artificial, Revista Iberoamericana de Inteligencia Artificial*, vol. 11–36, Dez 2007, pp. 67–75.

[44] Moreira, F.; Vieira, R. “Aplicação de reconhecimento de entidades nomeadas em investigação de crimes financeiros”. In: Proceedings of the 2nd Symposium in information and human language technology, 2019, pp. 134–143.

[45] Murdoch, W. J.; Szlam, A. “Automatic rule extraction from long short term memory networks”. In: Proceedings of the 5th International conference on learning representations, 2017, pp. 12.

[46] Nothman, J.; Ringland, N.; Radford, W.; Murphy, T.; Curran, J. R. “Learning multilingual named entity recognition from wikipedia”, *Artificial intelligence*, vol. 194, Jan 2013, pp. 151–175.

[47] Pennington, J.; Socher, R.; Manning, C. D. “Glove: global vectors for word representation”. In: Proceedings of the Conference on empirical methods in natural language processing, 2014, pp. 1532–1543.

[48] Peters, M. E.; Neumann, M.; Iyyer, M.; Gardner, M.; Clark, C.; Lee, K.; Zettlemoyer, L. “Deep contextualized word representations”. In: Proceedings of the Conference of the north american chapter of the association for computational linguistics: human language technologies, 2018, pp. 2227–2237.

[49] Pirovani, J. P. C.; Alves, J.; Spalenza, M.; Silva, W.; da Silveira Colombo, C.; Oliveira, E. “Adapting NER (CRF+LG) for many textual genres”. In: Proceedings of the 35th Conference of the spanish society for natural language processing, 2019, pp. 421–433.

[50] Pirovani, J. P. C.; de Oliveira, E. “Portuguese named entity recognition using conditional random fields and local grammars”. In: Proceedings of the 11th International conference on language resources and evaluation, 2018, pp. 4452–4456.

[51] Quaini, T. E.; dos Santos, H. D. P.; de Abreu, S. C.; Consoli, B. S.; Vieira, R. “Um estudo sobre desidentificação de evoluções clínicas”. In: Anais do IV Workshop de iniciação científica em tecnologia da informação e da linguagem humana, 2019, pp. 386–390.

[52] Rehurek, R.; Sojka, P. “Software framework for topic modelling with large corpora”. In: Proceedings of the 7th Language Resources and Evaluation Conference, 2010, pp. 46–50.

[53] Rönnqvist, S.; Kanerva, J.; Salakoski, T.; Ginter, F. “Is multilingual BERT fluent in language generation?” In: Proceedings of the 1st Workshop on deep learning for natural language processing, 2019, pp. 29–36.

[54] Rosenfeld, R. “Two decades of statistical language modeling: where do we go from here?”, *Proceedings of the IEEE*, vol. 88–8, Ago 2000, pp. 1270–1278.

[55] Ruder, S. “An overview of multi-task learning in deep neural networks”, *Computing research repository - arXiv*, vol. abs/1706.05098, Jun 2017, pp. 14.

[56] Sampson, G. “Handbook of standards and resources for spoken language systems”. Mouton De Gruyter, 1997, 886p.

[57] Sang, T. K.; Erik, F. “Introduction to the conll-2002 shared task: language-independent named entity recognition”. In: Proceedings of the 6th Conference on natural language learning, 2002, pp. 155–158.

[58] Santos, C. D.; Zadrozny, B. “Learning character-level representations for part-of-speech tagging”. In: Proceedings of the 31st International conference on machine learning, 2014, pp. 1818–1826.

[59] Santos, D.; Cardoso, N. “A golden resource for named entity recognition in portuguese”. In: Proceeding of the 7th International conference on the computational processing of portuguese, 2007, pp. 69–79.

[60] Santos, J.; Terra, J.; Consoli, B. S.; Vieira, R. “Multidomain contextual embeddings for named entity recognition”. In: Proceedings of the 35th Conference of the spanish society for natural language processing, 2019, pp. 434–441.

[61] Shen, Y.; He, X.; Gao, J.; Deng, L.; Mesnil, G. “Learning semantic representations using convolutional neural networks for web search”. In: Proceedings of the 23rd International conference on world wide web, 2014, pp. 373–374.

[62] Tai, K. S.; Socher, R.; Manning, C. D. “Improved semantic representations from tree-structured long short-term memory networks”. In: Proceedings of the 53rd Annual meeting of the association for computational linguistics, 2015, pp. 1556–1566.

[63] Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; Polosukhin, I. “Attention is all you need”. In: Proceedings of the Advances in neural information processing systems, 2017, pp. 5998–6008.

[64] Viterbi, A. J. “Automating the design of socs using cores”, *IEEE Transactions Information Theory*, vol. 13–2, Abr 1967, pp. 260 – 269.

[65] Williams, L.; Bannister, C.; Arribas-Ayllon, M.; Preece, A.; Spasić, I. “The role of idioms in sentiment analysis”, *Expert Systems with Applications*, vol. 42–21, Set-Out 2015, pp. 7375–7385.

[66] Wu, M.; Liu, F.; Cohn, T. “Evaluating the utility of hand-crafted features in sequence labelling”. In: Proceedings of the Conference on empirical methods in natural language processing, 2018, pp. 2850–2856.

[67] Zhang, D.; Xu, H.; Su, Z.; Xu, Y. “Chinese comments sentiment classification based on word2vec and svmperf”, *Expert Systems with Applications*, vol. 42–4, Mar 2015, pp. 1857–1863.

[68] Zhang, X.; Zhao, J.; LeCun, Y. “Character-level convolutional networks for text classification”. In: Proceedings of the Advances in neural information processing systems, 2015, pp. 649–657.

[69] Zhang, Y.; Wallace, B. C. “A sensitivity analysis of (and practitioners’ guide to) convolutional neural networks for sentence classification”. In: Proceedings of the 8th International joint conference on natural language processing, 2017, pp. 253–263.

[70] Zhu, Y.; Kiros, R.; Zemel, R. S.; 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 International conference on computer vision, 2015, pp. 19–27.

Pontifícia Universidade Católica do Rio Grande do Sul
Pró-Reitoria de Graduação
Av. Ipiranga, 6681 - Prédio 1 - 3º. andar
Porto Alegre - RS - Brasil
Fone: (51) 3320-3500 - Fax: (51) 3339-1564
E-mail: prograd@pucrs.br
Site: www.pucrs.br