Reconhecimento de entidades nomeadas em tweets sobre política

Arthur Galdino Dangoni[1]

[1]Instituto de Informática – Universidade Federal de Goiás (UFG)
Goiânia – GO – Brazil

arthurdangoni@discente.ufg.br

Abstract. *The named entity recognition (NER) in tweets about policy is an important step in being able to carry out processing applications in natural language training (PLN). This work proposes the use of models of machine learning with the purpose of helping in political decisions and obtaining first place in the competition. Some machine learning models were tested in diferrent configurations. Based on the tests performed, the model that obtained the best result was a Bidirectional Long Short Term Memory with the Conditional Random Fields (BILSTM-CRF).*

Resumo. *O reconhecimento de entidades nomeadas (NER) em tweets sobre política é um passo importante para poder realizar aplicações de processamento de linguagem natural (PLN). Este trabalho propõe a utilização de modelos de machine learning com o propósito de ajudar em decisões políticas e obter o primeiro lugar na competição. Foram testados alguns modelos de machine learning em diferentes configurações. Com base nos testes realizados o modelo que obteve o melhor resultado foi a Long Short Term Memory Bidirecional com o Conditional Random Fields (BILSTM-CRF).*

1. Introdução

O reconhecimento de entidades nomeadas (NER) é a tarefa que identifica entidades nomeadas e classifica essas entidades em categorias a partir de um texto. Essas entidades são normalmente divididas em duas categorias, sendo elas categorias genéricas e de domínio específico. As entidades reconhecidas na parte de categorias genéricas podem ser pessoas, localização e organização. Enquanto as de domínio específico podem ser enzimas, proteínas, genes, nomes de doenças, considerando um domínio médico. Essa tarefa é um passo fundamental para uma variedade de aplicações como atendimento a perguntas (Question answering - QA), recuperação de informação, tradução automática e contrução de uma base de conhecimento [Li et al. 2020]. Este estudo é específico em realizar o reconhecimento de entidades em tweets relacionados a um contexto político. Portanto permite o desenvolvimento de aplicações que podem ajudar na tomada de decisões políticas.

Este artigo propõe a utilização de modelos de machine learning para reconhecer as entidades nomeadas em tweets em um cenário político. Para possibilitar aplicações para ajudar na tomada de deciões políticas além de obter o primeiro lugar na 2ª Competição de processamento de linguagem natural (PLN).

2. Trabalhos Relacionados

Li *et al* (2020) escreveu um survey sobre deep learning na tarefa de NER. Esse estudo mostra diversos datasets, ferramentas e modelos de machine learning para NER. Evidencia as abordagens tradicionais para essa tarefa sendo elas uma abordagem baseada em regras, aprendizado não supervisionado, aprendizado supervisionado e a utilização de deep learning.

A abordagem baseada em regras é baseada em regras criada por pessoas. Essas regras podem ser criadas com base em gazetteers de domínios espefícos ou a partir de padrões léxicos e sintáticos. Portanto essa abordagem funciona bem quando o léxico é exaustivo. Entretanto como é baseada em regras criadas por pessoas não é possivel transferir os sistemas criados nessa abordagem para outros domínios.

O aprendizado não supervisionado são utilizados clusters nos quais permite a extração das entidades nomeadas a partir da similiaridade do contexto. Por fim o aprendizado supervisionado são utilizados modelos de machine learning. A vetorização do texto nessa abordagem é importante, essa vetorização pode ser feita a partir de listas de lookup a partir de wikipedia gazetteer, recursos do dataset como multiplas ocorrências no texto entre outras.

Por fim tem a utilização de deep learning para a tarefa de NER, no qual os modelos de deep learning possuem suas vantagens porque tem uma capacidade de representação e de aprendizado. O deep learning para NER segue um pipeline no qual é mostrado na Figure 1. Em que inicialmente se tem a representação das palavras por meio de word embeddings ou embeddings a nível de caracter entre outros. Em seguida o encoder do contexto no qual vai ter modelos de machines learning e transformers. Por fim o tag decoder que é o passo que vai selecionar a classificação para cada palavra.

Akbik *et al.* (2018) propõe a criação de um novo embedding chamado de contextual string embeddings. Esse embedding considera cada caractere para fazer o embedding da palavra como um todo. Portanto cada caractere considera as palavras vizinhas, ou seja considera o contexto, o que possibilita diferentes embeddings para uma palavra em que se tem diversos usos dependendo do contexto. Além de poder realizar um pré-treino em um dataset que não foi anotado. Por fim como considera os caracteres da palavra é útil para explorar informações como o prefixo e o sufixo.

Esse embedding foi comparado com outros do estado da arte em três tarefas diferentes sendo elas o Part-of-Speech (POS) , NER e chunking. Os testes utilizaram uma Bidirectional Long Short Term Memory (BILSTM) com Conditional Random Fields (CRF) e foram testados diferentes formas de embeddings sendo eles inicialmente o método proposto e os outros utilizaram esse método com a adição de outros embeddings sendo eles word embedding, representação a nível de caracteres, word embedding concatenado com representação a nível de caracteres e todos os embedding propostos anteriormente. Por fim os resultados mostraram que para a tarefa de NER o método foi melhor em comparação com os outros, sendo a melhor configuração para essa tarefa o método proposto com word embedding. A partir desse estudo é proposto verificar a utilização desse embedding para o problema atual.

Lample *et al.* (2016) propôe dois novos modelos sendo o primeiro uma BILSTM com o CRF e o segundo a Stacked LSTM(S-LSTM). Foram utilizados word em-

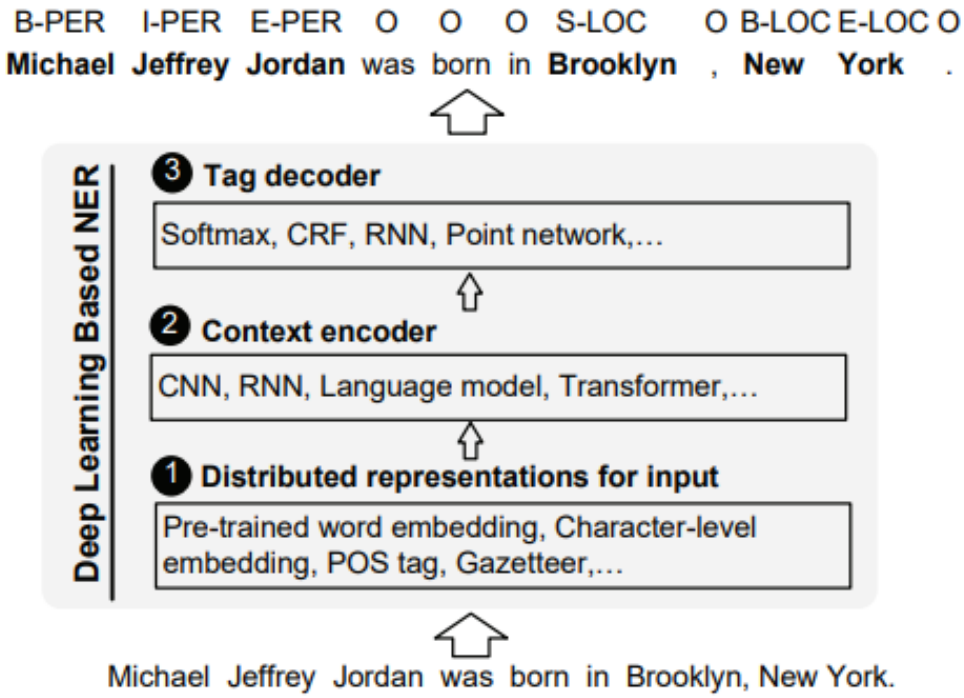

Figure 1. Pipeline geral para a tarefa de NER

beddings pré-treinados sendo eles o skip-n-gram (Ling *et al.* 2015) e uma variação do word to vec (Mikolov *et al.* 2013). Foram realizados testes nos datasets do CoNLL-2002 e CoNLL-2003 para tarefas de NER. Diferentes configurações foram testadas sendo as possibilidades com uma representação a nível de caracteres, word embedding e a adição ou remoção de uma camada de dropout. O otimizador utilizado em ambas arquiteturas foi o stochastic gradient descent (SGD). Os resultados mostraram que a BILSTM-CRF obteve resultados melhores do que o do estado da arte até o momento. Além de que a S-LSTM obteve resultados melhores ou parecidos com o do estado da arte. Esse estudo é interessante, pois propõe novas arquiteturas que produziram bons resultados para a tarefa de NER.

3. Metodologia

O conjunto de dados foi disponibilizado pela professora Nádia Félix Felipe da Silva. Esse conjunto de dados é composto por textos do twitter relacionados a política. Além das palavras o conjunto de dados também possui as classificações de acordo com o formato inside-outside-beginning (IOB). Esse formato considera que uma sentença ou chunk começe com ”B” que é o begin, não podendo começar com a tag ”I” que representa que a tag está dentro da sentença e o ”O” representa o fim de uma sentença ou palavras que não fazem parte de uma sentença.Esse conjunto de dados é dividido em conjunto de treino que possui 10392 amostras e conjunto de teste que possui 11239 amostras, sendo ao todo 21631 amostras.

Os experimentos foram realizados no Google Colaboratory que é um ambiente de notebooks. Nesse ambiente foram realizados a codificação das etapas para resolver o problema. Todos os modelos seguiram essas etapas que foram a leitura dos dados a partir

do dataset de treino, embedding, encoder e decoder como mostra a Figura 1.

Por fim as predições de cada modelo foram salvas em arquivos '.csv' para a submissão no kaggle. O kaggle é um site utilizado para competições de machine learning onde foi realizado a 2ª Competição de PLN da Universidade Federal de Goiás (UFG) para a disciplina de PLN.

4. Experimentos

Nesta seção apresentamos as etapas realizadas para fazer os experimentos. Foi utilizado o flairNLP que é um framework para NLP. Primeiro foi realizado o embedding do dataset em que foram testados diferentes métodos na seção 4.1. A seguir a seleção do modelo que vai estar na parte de encoder na seção 4.2. Por fim é realizado o treinamento selecionando alguns hyperparâmetros como número de épocas, e o otimizador na seção 4.3. Após isso se tem os resultados gerados a partir dos modelos implementados na seção 4.4. Por fim nas conclusões são mostradas as contribuições deste estudo na seção 4.5.

4.1. Embedding

Foi utilizado o framework FlairNLP no qual já é possivel fazer a utilização de alguns embeddings sendo eles word embedding, fast text embedding, embeddings a nível de caracteres, Elmo embedding, flair embedding, embeddings utilizando transformers entre outros. Foram testados principalmente o word embedding, flair embedding e o elmo embedding. No qual eles foram usados utilizando stacked embeddings no qual se concatena esses embeddings em um único. Por fim foi realizado um experimento alterando o valor do dropout na BILSTM com CRF.

O flair embeddings implementa o contextual string embeddings. Nesse embedding pode ter a representação das palavras em uma direção ou nas duas direções da sentença, este estudo somente usou a forma bidirecional desse embedding. Enquanto os word embeddings utilizados foram baseados no fastText treinado no idioma português. O fastText é uma biblioteca que tem modelos pré-treinados baseados no continuous bag of words(CBOW) em datasets como a wikipedia. O CBOW é uma forma de word embedding em que a partir de uma sentença para cada palavra é considerado as palavras vizinhas, ou seja considera o contexto para se ter a representação das palavras. Por fim o elmo embedding utiliza o modelo elmo para fazer o embedding das palavras, o modelo utilizado foi o treinado para o idioma português.

4.2. Modelos Testados

Em alguns modelos foi utilizado o CRF como decoder, porque essa técnica leva em consideração o formato da classificação sende ele o IOB. Na hora de se realizar a classificação é considerado o contexto das outras palavras além de encorajar o modelo a gerar classificações que sejam condizentes com o formato do IOB. Os modelos testados foram a BILSTM com CRF e sem, a S-LSTM com crf e sem, além de que esses modelos foram treinados com diferentes épocas e embeddings. Por fim todas as LSTMs tinham 256 neurônios na feedforward dentro da célula LSTM.

4.3. Seleção de Hiperparâmetros no Treinamento

O treinamento desses modelos chegava ao fim caso chegasse ao número máximo de épocas ou o learning rate fosse menor do que 0.0001, sendo o learning rate inicial igual a

0.1. O learning rate sofre um decrescimo de acordo com a paciência, a paciência utilizada foi 3, ou seja a cada 3 épocas em que não se teve uma melhora do modelo o learning rate era diminuido.

A seleção de hiperparâmetros foi feita de forma empírica no qual foi variado o número máximo de épocas e no flairNLP tem dois possíveis otimizadores sendo o padrão o SGD e o outro é o adaptive moment estimatiom (ADAM).

4.4. Resultados

Foi criado uma tabela composta pelos seus respectivos números de épocas, embeddings, otimizadores e modelo utilizado com ou sem CRF. É mostrado as acurácias obtidas no resultado público do kaggle e no privado como é mostrado na tabela 1.

Épocas + Embedding + Otimizador + Modelo	Privado	Público
5 + W + SGD + BILSTM-CRF	0.89249	0.89027
5 + WF + SGD + BILSTM-CRF	0.93637	0.94080
10 + WF + SGD + BILSTM-CRF	0.95111	0.94894
30 + WE + SGD + BILSTM-CRF	0.85776	0.85845
110 + WF + SGD + BILSTM-CRF	**0.97747**	**0.97311**
43 + WF + SGD + S-LSTM-CRF	0.97341	0.96778
30 + WF + ADAM + BILSTM-CRF	0.64382	0.65502
110 + WF + SGD + BILSTM-CRF + Dropout	0.97337	0.96337

Table 1. W: Word Embedding fastText do português, F: Flair Embedding, E: Elmo Embedding do português, Dropout: valor alterado para 0.5 sendo o padrão do flairNLP igual a 0.

A partir da tabela podemos mostrar que o modelo teve uma boa capacidade de generalização porque os resultados público e privado do kaggle foram parecidos. Os melhores resultados em questão do otimizador foi utilizando o SGD, enquanto a melhor configuração de embeddings foi o word embedding do fastTest mais o flair embedding. Por fim em relação ao modelo utilizado a BILSTM teve resultado melhores do que a S-LSTM e a utilização do CRF foi eficaz, pois teve uma melhora nos resultados ao utilizar ele. Inicialmente foi feito alguns experimentos com poucas épocas para verificar o comportamento do modelo e não foi utilizado a concatenação de 3 embeddings porque quanto mais embeddings foram utilizados mais pesado se torna o modelo.

Por fim levando em consideração que o resultado privado foi o resultado definitivo da competição, o melhor modelo obtido foi a BILSTM-CRF com otimizador SGD utilizando o flair embedding bidirecional mais o word embedding fastText com 110 épocas.

4.5. Conclusões

Este trabalho apresentou diversos modelos e diferentes embeddings que poderiam ser usados para solucionar o problema de reconhecer entidades nomeadas em tweets sobre política com o objetivo de possibilitar aplicações para ajudar na tomada de decisões políticas. Além de obter o primeiro lugar na 2ª Competição de PLN da UFG para a disciplina de PLN. Neste trabalho a partir dos testes realizados foi obtido o 4ª lugar na competição com o modelo BILSTM-CRF que obteve a acurácia de 97.7%. Por fim foram

aplicados todos os conhecimentos obtidos na disciplina até o momento e pode servir como um guia inicial para pessoas que estão estudando a tarefa de NER.

Para trabalhos futuros é proposto utilizar a biblioteca do AllenNLP possibilitando alterações maiores na arquitetura do modelo e implementar arquiteturas transformer para verificar uma possível melhoria nos resultados.

5. Referências

[Li et al. 2020] [Akbik et al. 2018] [Lample et al. 2016] [Ling et al. 2015] [Mikolov et al. 2013]

References

Akbik, A., Blythe, D., and Vollgraf, R. (2018). Contextual string embeddings for sequence labeling. In *Proceedings of the 27th international conference on computational linguistics*, pages 1638–1649.

Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., and Dyer, C. (2016). Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360*.

Li, J., Sun, A., Han, J., and Li, C. (2020). A survey on deep learning for named entity recognition. *IEEE Transactions on Knowledge and Data Engineering*.

Ling, W., Tsvetkov, Y., Amir, S., Fermandez, R., Dyer, C., Black, A. W., Trancoso, I., and Lin, C.-C. (2015). Not all contexts are created equal: Better word representations with variable attention. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1367–1372.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.