Esse artigo faz parte de uma série de estudos onde tentaremos construir um modelo NER (Named Entity Recognition) para identificação de entidades relevantes nas circulares publicadas pelo BNDES para disponibilizar atualizações em suas Políticas Operacionais (PO).

Frequentemente o BNDES realiza ajustes nas suas Políticas Operacionais, cujo objetivo principal é nortear as regras aplicadas aos seus financiamentos. Anualmente as políticas são revisadas para publicação das circulares, já alterações menores geram atualização da circular e podem ocorrer a qualquer instante durante o período de vigência.

A mudança da política operacional impacta equipes de negócio, sistemas e seus respectivos gestores, gerências de testes e equipes de homologação. O esforço empreendido em cada mudança de política operacional é enorme, sendo a maior parte do trabalho realizado de forma manual e sujeita a diversos tipos de falha. A área de auditoria do BNDES faz apontamentos constantes sobre esse processo.

A inteligência artificial não tem como objetivo substituir as tarefas humanas, mas diminuir sua carga de trabalho, automatizar algumas tarefas do processo e apoiar as equipes envolvidas, atendendo a requisitos de *compliance*  e dando para dar mais segurança ao resultado final alcançado.

O projeto PO-NER (Políticas Operacionais Named Entity Reconigtion) tem como objetivo a construção de um ou mais modelos de inteligência artificial aplicados a NLP (Natural Language Processing) para realizar as seguintes tarefas:

* Reconhecimento de entidades nas Políticas Operacionais do BNDES (NER).
* Identificação de diferentes entidades cujo formato é muito similar, por exemplo: Um valor percentual pode estar relacionado ao nível de participação em uma operação de crédito ou a taxa de remuneração aplicada. Os dois valores são percentuais, mas representam diferentes entidades.   
  Esse tipo desambiguação é conhecido como NED (Named Entity Disambiguation) e deverá ser tratado pelo modelo.
* Conectar entidades reconhecidas (Named Entity Linking - NEL).
* Criar uma base de conhecimento (Knowledge Base) para que outros modelos realizem tarefas sobre o conhecimento extraído das circulares.

Caso o(s) modelo(s) construídos alcancem um nível de acurácia satisfatório então será possível utilizá-lo para apoiar gestores e técnicos envolvidos nas mudanças das PO’s do BNDES.

Como estudante do curso de Pós-graduação BI-Master da PUC-RJ e orientado pelo professor Leonardo Mendoza mergulharei na jornada de aprender sobre reconhecimento e conexão de entidades, compartilhando a cada fase do projeto o conhecimento adquirido. Agradeço ao Mestre Leonardo e seus pares na PUC BI-Master pela ajuda e apoio no aprendizado.

Nosso planejamento inicial do projeto, sujeito a mudanças futuras, está relacionado abaixo. Os artigos serão criados ao término de cada etapa.

1. NER Básico – Aplicação de Modelos BERT para NER. Estudo exploratório para compreensão do uso de modelos BERT em tarefas de reconhecimento de entidades.
   1. Utilização de modelos em inglês – maior disponibilidade de datasets e modelos para teste.
   2. Preparação de tokens.
   3. Tratamento dos labels - alinhamento de *wordpiece*, importação de labels por *span* e *token/tag.*
   4. Estudo das métricas para avaliação do modelo.
2. NER Português – Aplicação de Modelos BERT em português para reconhecimento de entidades.
3. Anotações de Politicas Operacionais – Criação de algoritmos para anotar os dados existentes e construir um dataset que servirá para treinar, validar e testar o modelo construído. Utilizaremos algoritmos baseados em heurísticas para criar nossos datasets (Distance Supervision).
4. PO-NER BERT – Construção de um modelo BERT para Extração de Entidades em Politicas Operacionais no BNDES.
5. PO-NEL – Construção de um modelo para Conectar Entidades em Politicas Operacionais no BNDES
6. KB PO – Construção da Base de Conhecimento de Politicas Operacionais.

Não sabemos até onde o projeto chegará, mas isso não importa, o caminho e seu aprendizado são nossos objetivos.

Sem mais delongas, após contextualizar o trabalho, iniciemos a parte técnica do primeiro artigo!

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Está fora do escopo dos artigos a explicação sobre modelos BERT, Word Embedding e conhecimentos de processamento de linguagem natural. No site Medium ou Towards Data Science poderão ser encontrados muitos artigos sobre o tema.

Em nosso estudo utilizaremos modelos BERT pré-treinados, o custo computacional, técnico e financeiro para criação de um modelo fogem ao escopo do trabalho proposto.

O pré-treinamento dos modelo BERT gera um vetor de *embeddings* para o vocabulário por ele selecionado, que geralmente possuirá aproximadamente 30.000 tokens. Um embedding é uma representação vetorial (tensor) de cada token (palavra ou parte de uma palavra) que geralmente possuirá dimensão de 768 (depende do tamanho do modelo: base, large, etc) e definirá o sentido/contexto de cada palavra conhecida e seu relacionamento com outras palavras que são identificadas como próximas ou similares. Importante compreender que o sentido e contexto de cada palavra dependerá do corpus utilizado para treinar o modelo. Um exemplo para demonstrar: Lula pode ser um molusco ou um político, mas também existe um campo de exploração de petróleo com esse nome, ou seja, se documentos da Petrobras estiverem no corpus então lula terá uma representação vetorial diferente do que o treinamento no wikipedia.

Os modelos BERT e outros modelos de inteligência artificial para processamento de linguagem natural não conhecem palavras, mas sim esses embeddings de palavras. Por isso precisamos tratar as palavras antes do treinamento, teste ou inferência do modelo.

A conversão de palavras em embeddings traz o primeiro problema a ser resolvido. Se nosso vocabulário possui 30.000 tokens então está claro que não teremos todas as palavras de um idioma, cada modelo BERT utilizará uma heurística para construir o seu vocabulário, de acordo com os documentos utilizados (corpus) para o seu pré-treinamento. Para tratar o problema palavras fora do vocabulário (OOV – Out Of Vocabulary) existem várias soluções, em nossos artigos utilizaremos a técnica conhecida como *Wordpiece*, mas existem outras como *Byte Pair Encoding e Byte Level Pair Encoding.*

O responsável por transformar uma palavra em token/*embedding* é o *Tokenizer*, sendo que cada modelo BERT possui o seu respectivo *Tokenizer*. Para clarificar o caminho palavra --> token/embedding vamos exemplificar:

Palavra – Escola

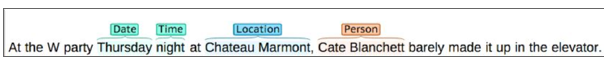
Token - #1234

Embedding – [ 1.2 56 45 78.9 ....... 56]

Token é o índice de um embedding, sendo o embedding a representação vetorial do token.

O tokenizador de um modelo é construído junto com o seu treinamento, por isso devemos utilizá-los em par.

Um problema que será tratado nesse artigo é o alinhamento entre os *labels* e tokens, mas primeiro temos que explicar o que representa um label ... 😊. Label é o rótulo de classificação de uma palavra, informado se a palavra é uma entidade, se inicia uma entidade ou se faz parte de uma entidade iniciada ou continuada na palavra anterior. Exemplo:



night é uma entidade nova após a entidade Thursday, já Marmont é a continuação da entidade Chateau e finalizando temos Cate que inicia uma entidade após “,” que não faz parte de nenhuma entidade.

Os labels de classificação podem ser carregados de várias formas, no artigo trataremos dois formatos:

* Span - Labels com inicio e fim de entidade. Exemplo:

 { label: PERSON, inicio:0, fim:7}   
 { label: LOCATION, inicio:15, fim:23}   
 { label: LOCATION, inicio:43, fim:57}

* Tag - Palavra e classificação. Informa as sentenças separadas por palavras e respectiva classificação. Exemplo:

Vinicius B-PERSON  
foi O  
a O  
Nova B-LOCATION  
York I-LOCATION  
e O  
visitou O  
o O  
museu O  
Madame B-LOCATION  
Tussauds I-LOCATION

Como vimos anteriormente, o modelo BERT não contém todas as palavras de um idioma, é um subconjunto que geralmente contém 30.000 tokens. Para tratar as palavras fora do vocabulário (OOV) o tokenizador gera *wordpieces* quando não encontra um token, concatenando as maiores parte de palavras que possui. Vamos dar um exemplo para explicarÇ

"Let us start pretraining the model."

Ao utilizar o tokenizador obtemos o seguinte resultado:

tokens = [let, us, start, pre, ##train, ##ing, the, model]

O tokenizador vai procurar pelo maior token que atenda a subparte não conhecida, em último caso ele utilizará letras, já que todas estão no vocabulário.

Repare que a palavra pretraining será particionada em 3 tokens, no entanto as anotações por *span* ou *tag* estão relacionados a palavras e não a partes dela. O alinhamento de *labels* e tokens tem o objetivo de resolver as lacunas geradas a partir do particionamento de palavras em um ou mais tokens.

<<<<<< Fonte e Exemplo >>>>>>>>>>>>

**Classificação de Tokens**

Existem diversos formatos para classificação dos tokens no reconhecimento de entidades, em nosso estudo utilizaremos o formato IOB.

**IOB - Inside Outside Begin**

Uma entidade tem seu início com primeiro token (B) e opcionalmente um ou mais tokens internos (I). *Outside* é utilizada quando o token não representa entidade. Importante lembrar que os prefixos B- e I- serão aplicados a todas as entidades que fazem parte do contexto, duplicando o número de classificações.  
 Exemplo:

Para o caso de termos as entidades PERSON e LOCALIZATION, nossos *labels* seriam os seguintes:

O, B-PERSON, I-PERSON, B-LOCALIZATION, I-LOCALIZATION

“George Harris was in New York city.”

[[“George”, “B-PERSON”], [“Harrris”, ”I-PERSON”] , [“was”, “O”], [“in”, “O”], [“New”, “B-LOCALIZATION”], [“York”, “I-LOCALIZATION”], [“city”, “O”]]

Para o modelo BERT não faz diferença esse formato, já que cada item representa uma possível classificação. A preparação dos labels faz parte das tarefas anteriores ao treinamento ou utilização do modelo.

Embora a classificação dos tokens seja melhor entendida por humanos no formato textual, o modelo entende somente números. A classe LableNER foi criada para carregar os labels, aplicando permutação quando necessário (carga de spans) e conversão entre a identificação numérica ou textual das entidades. Essa classe também persistirá o label com seu ID em um arquivo durante o treinamento para que não ocorra mudança do ID, garantindo que os identificadores de cada label serão sempre os mesmos.

Uma decisão importante para o projeto é como tratar classificação das *wordpieces.* Dado que uma palavra foi separada em 2 ou mais tokens, qual será a classificação desses tokens complementares a uma palavra?

Pesquisando na internet descobri duas técnicas: A primeira ignora os *wordpieces* atribuindo o identificar -100 durante o treinamento, informando ao modelo que aquele token deve ser ignorado. A segunda opção, utilizada pelo nosso estudo, considera a wordpiece parte da entidade, ou seja, ela sempre terá o prefixo “I-“.