# Construção de um modelo baseado em Transformer para extração de relações de doenças induzidas por substâncias químicas

Cristian E. Muñoz Villalobos, Ricardo Tanscheit
Departamento de Engenharia Elétrica
Pontifícia Universidade Católica do Rio de Janeiro
Rio de Janeiro, RJ-Brazil
crismunoz@ele.puc-rio.br, ricardo@ele.puc-rio.br

Leonardo A. Forero Mendoza
Departamento de Engenharia Elétrica
Universidade do Estado do Rio de Janeiro
Rio de Janeiro, RJ-Brazil
leofome@gmail.com

***Resumo*—A extração da relação Químico-Doença em documentos de texto é uma tarefa desafiadora na área biomédica. Esta tarefa pode compreender interações complexas descritas em mais de uma frase. Trabalhos recentes na área de extração de relações concentram-se na solução deste problema, identificando interações entre entidades presentes em frases diferentes. As abordagens baseadas em Deep Learning modelam estes relacionamentos usando camadas especializadas no aprendizado de representações de forma hierárquica, evitando a dependência de uma sofisticada engenharia de atributos, propensa a erros. Este trabalho propõe um conjunto de mecanismos sobre um módulo *transformer*; o aprendizado de características dentro do *transformer* foi aprimorado usando uma representação de palavras baseada em caracteres, um bloco residual convolutivo e um módulo de tempo de computação adaptativa. Mostra-se que o modelo proposto supera os resultados de outras abordagens por redes neurais.**

***Keywords*—Deep Learning, Biomedicine, Universal Transformer, Name Entity Recognition, Relation Extraction.**

## I. Introdução

Avanços em Processamento de Linguagem Natural (PLN) permitiram um explosivo crescimento na área de extração de informação, especialmente em tarefas como reconhecimento de entidades nomeadas (REN) e extração de relações (ER). A ER, um tipo de extração de informação que consiste em extrair automaticamente relações semânticas entre entidades, tem se tornado um tópico de pesquisa central para uma variedade de aplicações práticas de domínios abertos, como *web*, *newswire*, *blogs*, *chat logs* [1], e também em domínios específicos, como biomedicina [2], documentos da área científica [3], etc.

A diminuição de preço do sequenciamento de genomas[1] tem permitido o diagnóstico e o tratamento de doenças genéticas, como o câncer. Consequentemente, milhares de genomas são analisados e a informação é disseminada em artigos cientificos que são anualmente adicionados em repositórios como o PubMed [2]. Para extrair conhecimento desses repositórios é preciso aprimorar a extração de relação a partir de documentos de texto de forma automática.

[1]https://www.genome.gov/27565109/the-cost-of-sequencing-a-human-genome/

[2]https://www.ncbi.nlm.nih.gov/pubmed/

Em aplicações de PLN para a área biomédica, a extração de relações sobre doenças induzidas por produtos químicos é uma tarefa desafiadora e um processo demorado que tenta lidar com os efeitos colaterais indesejáveis – a toxicidade de produtos químicos. Impulsionar esta tarefa, com base em métodos computacionais, é importante não apenas para aumentar o uso seguro de medicamentos, mas também para evidenciar potenciais relações entre produtos químicos e patologias. Os métodos de ER requerem amostras chamadas de anotações. Cada anotação indica a relação existente entre um par de entidades. As abordagens tradicionais focam-se na classificação de relação de um par de entidades dentro de uma sentença, limitando-se a relações de curtas dependências. Por exemplo:

> Cinco em cada 8 pacientes (63%) melhoraram devido ao tratamento com ácido fusídico: 3 após duas semanas e 2 após quatro semanas. Não houve efeitos colaterais sérios, mas dois pacientes precisaram reduzir o medicamento devido à náusea.

As duas sentenças juntas transmitem o fato de que o ácido fusídico pode produzir náusea. Isso não é expresso em nenhuma das frases separadamente. Visando a conduzir a extração de relações em múltiplas sentenças, alguns trabalhos enfatizaram técnicas como: criação de características adaptativas ([4]), representações por grafos ([5]) e, mais recentemente, abordagens de aprendizado profundo ([6], [7], [8], [9]).

Um dos modelos mais promissores de aprendizado profundo usado para a tarefa de ER foi o modelo BRAN, proposto por [8]. A ideia de fazer uso de um módulo *Transformer* otimiza a recorrência, iterando de forma paralela sobre toda a sequência de palavras, aumentando o grau de paralelismo e reduzindo o tempo de processamento. Neste trabalho, a fim de melhorar a efetividade a partir deste modelo, novos mecanismos de aprendizado profundo são explorados, tais como: *embeddings* de palavras baseados em caracteres, construção de uma função de transição baseada em blocos convolutivos *Depthwise Separable* [10] e a integração de um módulo de tempo de computação adaptativa [11].

## II. Modelo Proposto

O modelo proposto é dividido em três módulos: Camada Encoder, módulo *Transformer* e módulo de tarefas específicas. A Camada Encoder converte a sequência de tokens em uma representação distribuída. O módulo *Transformer* itera sobre esta representação de forma a computar uma nova representação com as características necessárias para resolver uma ou múltiplas tarefas de aprendizado. Por último, os módulos de tarefas específicas projetam as características de saída do *Transformer* para resolver cada tarefa. A Figura 1 apresenta a estrutura do modelo proposto. Os módulos são detalhados a seguir.

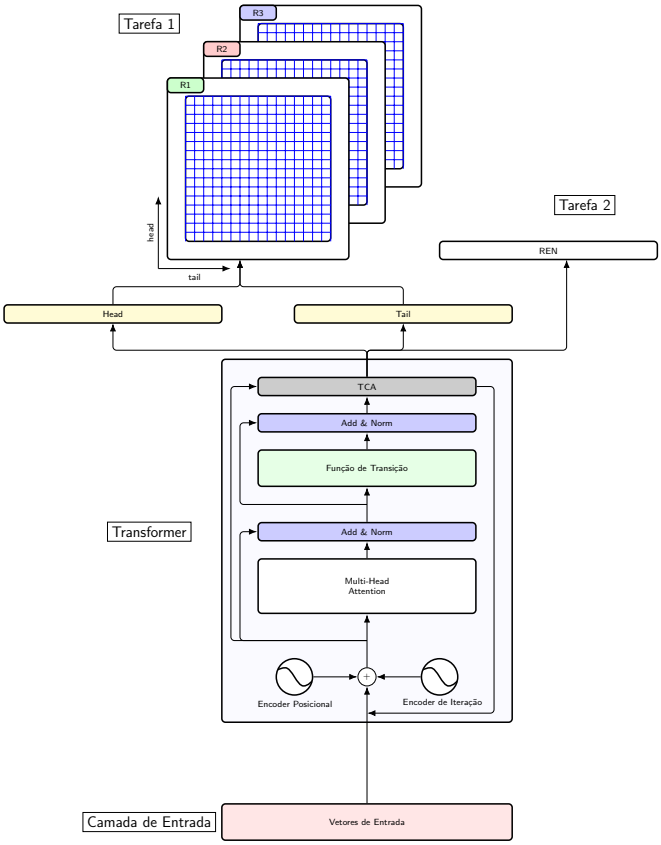


Figura 1. Módulo *Universal Transformer* para extração de relações.

### A. Camada Encoder

A camada converte uma sequência de tokens $\mathbf{x} = \{x_1, .., x_M\}$ em uma sequência de vetores de representação distribuída (*embeddings*). Esses vetores são definidos fazendo-se uso de dois tipos de entrada: *encoders* de palavras $E_w \in \mathbb{R}^{M \times d}$ e *encoders* de palavras baseados em caracteres $E_{w/c} \in \mathbb{R}^{M \times N \times d}$, em que $M$ é o número máximo de palavras na sequência, $N$ é o número máximo de caracteres por palavra e $d$ é o tamanho do *embedding*.

*Representação de palavras em nível de caracteres:* para cada token $x_m$ há uma sequência $E_{c,m} \in \mathbb{R}^{N \times d}$ de $N$ *embeddings* de caracteres de tamanho $d$. A camada *Attention* "individual" (*ATT Individual*) calcula a matriz $M \in \mathbb{R}^{d \times d}$ a partir das representações $h_k$ e $h_q$. Essas representações foram extraídas da sequência $E_{c,m}$, usando as camadas convolutivas $C_3^k$ e $C_3^q$ respectivamente, em que o valor do $kernel = 3$ (obtido experimentalmente). As representações focam-se em capturar padrões de prefixos ou sufixos que formam uma palavra composta. Em seguida, $M$ é processada pelo módulo *Attention-over-Attention* (*ATT-over-ATT*) de forma a obter os pesos de *Attention* $\alpha$ e $\beta$ [12], como apresentado na Figura 2.

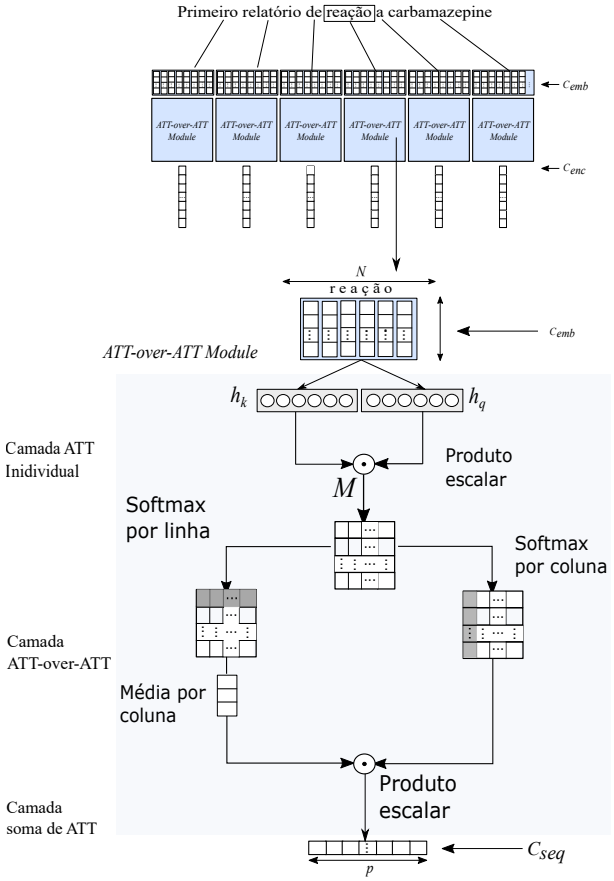


Figura 2. Módulo *Attention over Attention*.

A seguir, são descritas as equações para cada elemento $m$ da sequência de palavras.

$$\begin{gathered} E_{c,m} = E_{w/c}[m, :, :] \\ h_k = \mathrm{C}_3^k(E_{c,m}) \quad h_q = \mathrm{C}_3^q(E_{c,m}) \quad h \in \mathbb{R}^{N \times d} \\ M(i,j) = h_k(i)^\intercal . h_q(j) \quad M \in \mathbb{R}^{d \times d} \end{gathered} \tag{1}$$

Camada de *ATT-over-ATT*: calculam-se os pesos $\alpha$ e $\beta$, em que $i$ e $j$ representam a coluna e linha da matriz $M$, respectivamente.

$$\begin{gathered} \alpha(i) = \text{softmax}(M(1,i), ..., M(d,i)) \\ \alpha = [\alpha(1), \alpha(2), ..., \alpha(d)] \\ \beta(j) = \text{softmax}(M(j,1), ..., M(j,d)) \\ \beta = \frac{1}{d} \sum_{j=1}^{N} \beta(j) \end{gathered} \tag{2}$$

Camada de Soma de ATT:

$$s = \alpha^\intercal \beta \quad , \quad s \in \mathbb{R}^{d \times d}$$
$$E_c[m,:] = \sum_{i=1}^{d} s(i,:) \tag{3}$$

A saída $E$ do módulo *ATT-over-ATT* é o resultado da soma de $E_w$ e $E_c$:

$$E = E_w + E_c \tag{4}$$

### *B. Módulo Universal Transformer*

O módulo *Universal Transformer* (UT) proposto em [11] é uma extensão do modelo *Transformer*, especializado no processamento de sequências. Este modelo dispõe de um módulo de tempo de computação adaptativa (TCA) que controla a quantidade de processamento necessária para cada elemento da sequência.

O módulo age em consecutivas iterações sobre a sequência $H_t \in \mathbb{R}^{M \times d}$ por meio do mecanismo *Multi-Head Self-Attention*, seguido de uma função de transição. A estrutura *Attention* relaciona os elementos de entrada permitindo que os parâmetros do modelo não dependam do tamanho da sequência. O fucionamento desse processo é descrito na Eq (5), em que $H_0 = E$ da saída da camada encoder:

$$\begin{gathered} H_t = \text{LN}(A_t + \text{Transition}(A_t)) \\ \text{onde} \quad \text{A}_t = \text{LN}(H_t^p + \text{MultiHeadSelfAttention}(H_t^p)) \\ \text{e} \quad H_t^p = H_{t-1} + P_t \end{gathered} \tag{5}$$

LN (Layer Normalization) é a camada de normalização proposta em [13] e $P_t$ é o encoder de coordenadas bidimensional (posição, tempo) [11]. O vetor $P_t$ permite a rede considerar características relativas às posições dos elementos da sequência e à iteração em que que ela se encontra. Os componentes do módulo *Transformer* são discutidos na próximas seções.

*1) Multi-Head Self-Attention:* para cada elemento da sequência, o mecanismo *Attention* atribui um grau de importância (pesos) a todos os elementos da sequência de forma a aprender um foco de interação com os elementos da sequência. O mecanismo particiona o módulo de *Self-Attention* em múltiplos focos de *Attention*, permitindo a cada elemento da sequência detectar múltiplas dependências. O tipo de mecanismo *Attention* utilizado é o *Scaled Dot-Product Attention* (Eq. (6):

$$\text{Attention}(Q,K,V) = \text{softmax}(\frac{QK^\intercal}{\sqrt{d}})V \tag{6}$$

em que $d$ é o número de colunas de $Q$, $K$ e $V$. A entrada $H_t$ gera as projeções $Q$, $K$ e $V$. A versão *Multi-Head* considera $k$ partições, ou $heads$, as quais são calculadas na Eq. (7) da seguinte forma:

$$\begin{gathered} \text{MultiHeadSelfAttention}(H) = [head_1, ..., head_k]W^O \\ \text{onde} \quad \text{head}_i = \text{Attention}(HW_i^Q, HW_i^K, HW_i^V) \end{gathered} \tag{7}$$

Os pesos $W^Q, W^K, W^V \in \mathbb{R}^{d \times d/k}$ e $W^O \in \mathbb{R}^{d \times d}$ projetam $H_t^p$ para cada partição de *Attention* $head_i$.

*2) Função de Transição:* foi utilizada uma variante do bloco convolutivo com conexão residual baseado na MobileNetV2 proposto emr [10]. Uma das vantagens desta estrutura é o emprego de convoluções *Depth Wise*, que utilizam um menor número de parâmetros do que a convolução tradicional, reduzindo assim a probabilidade de ocorrência de *overfitting*. Esta estrutura apresentou melhor desempenho do que o bloco de convolutivas indicado inicialmente por [8]. Este módulo foi modificado de forma a realizar convoluções 1D, como se observa na Figura 3:

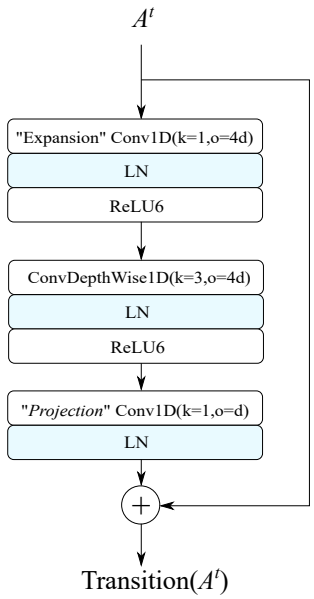


Figura 3. Função de Transição: Bloco Convolutivo com Conexão Residual

O bloco convolutivo é composto por três camadas convolutivas. A camada *Expansion* é uma convolução de $1 \times 1$. Seu propósito é expandir o número de características por um fator 4 antes de entrar na convolução *Depth Wise*. A camada convolutiva *Depth Wise* mantém o número de características constantes e a camada *Projection* reduz o número de características até o seu número inicial. Na Figura **??**, as funções de convolução, $k$ é a dimensão do kernel e $o$ é o número de características na saída. Uma camada de normalização é colocada após cada convolução. As equações do bloco convolutivo são apresentadas na equação (8):

$$\begin{gathered} \text{Transition}(A_t) = \text{LN}(C_1(A_t^{(1)})) + A_t \\ \text{onde} \quad A_t^{(1)} = \text{ReLU6}(\text{LN}(C_3(A_t^{(0)}))) \\ A_t^{(0)} = \text{ReLU6}(\text{LN}(C_1(A_t))) \end{gathered} \tag{8}$$

*3) Encoder Posicional:* a matriz de representação de coordenadas $P_t \in \mathbb{R}^{M \times d}$ é obtida calculando-se o valor senoidal da posição $m$ e tempo $t$ separadamente para cada dimensão $j = \{1, .., d\}$ do vetor de representação. Estas posições são somadas ponto a ponto na Eq. (9):

$$P_t(pos, 2j) = \sin(pos/10000^{2j/d}) + \sin(t/10000^{2j/d})$$
$$P_t(pos, 2j+1) = \cos(pos/10000^{2j/d}) + \cos(t/10000^{2j/d}) \quad (9)$$

Após as iterações, a saída do *Transformer* $H_T \in \mathbb{R}^{M\times N\times d}$ armazena toda a informação necessária para resolver uma ou múltiplas tarefas atribuídas.

*4) Tempo de Computação Adaptativa:* quando uma sequência de elementos é processada, interpretar certas posições desta sequência apresenta maior complexidade computacional do que para outras, o que se reflete em uma maior quantidade de recursos para processamento. O módulo TCA, proposto em [14], permite que a rede neural distribua "quantidades" de computação diferentes para cada posição da sequência. Um diagrama do TCA em conjunto com o modelo *Transformer* é apresentado na Figura 4:

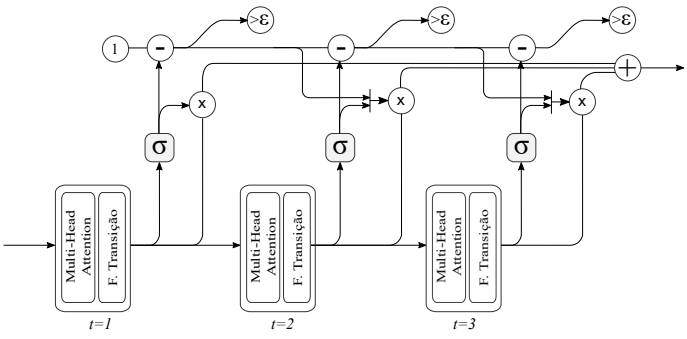


Figura 4. Bloco Transformador com modulo TCA.

Ao final de cada iteração do *Transformer*, uma unidade neuronal com ativação sigmoide $h^i_{t-1} \in [0, 1]$ determina quantas atualizações são necessárias por elemento $i$ da sequência $H_{t-1}$.

$$h_t = \sigma(W_h H_{t-1} + b_h) \quad (10)$$

Para cada elemento $m$ da sequência, o valor de $h^m_t = h_t[m]$ determina o peso $p^m_t$ que indica a porcentagem do estado $H_t$ que é agregado para a saída final.

$$\hat{H}_T = \sum_{t=1}^{N_{max}} p_t H_t \quad (11)$$

Quando o valor acumulado de $p^m_t$ é maior do que o limiar $(1-\epsilon)$, o elemento $m$ não realiza mais atualizações; isto ocorre na iteração $N^m$.

$$p^m_t = \begin{cases} h^m_t & \text{Se} \quad t < N^m \\ R^m_t & \text{Se} \quad t = N^m \\ 0 & \text{outro caso} \end{cases} \quad (12)$$

$$N^m = \min\{t' : \sum_{t=0}^{t'} h^m_t >= 1 - \epsilon\} \quad (13)$$

em que $\epsilon$ é uma pequena constante que permite a computação pausar após uma única atualização. Quando o valor ultrapassar o limiar, o valor de $p^m_t$ é ajustado com o fator de recordação $R^m_t$, que permite seguir a definição $\sum_t^{N^m} p_t = 1$.

$$R^m_t = 1 - \sum_{m=1}^{N^m - 1} h^m_t \quad (14)$$

Mais detalhes sobre a implementação podem ser encontrados em [14].

### C. Módulos de tarefas específicas

A saída do módulo *Transformer* entra em um bloco de expansão de forma a gerar $H^T_{exp}$, que expande por um fator 4 o número de características. Assim, esse módulo filtra ou projeta, a partir de $H^T_{exp}$, em $S_{task\in\{\text{NER,RE}\}}$ as características necessárias para resolver cada tarefa. O procedimento é ilustrado na Figura 5:

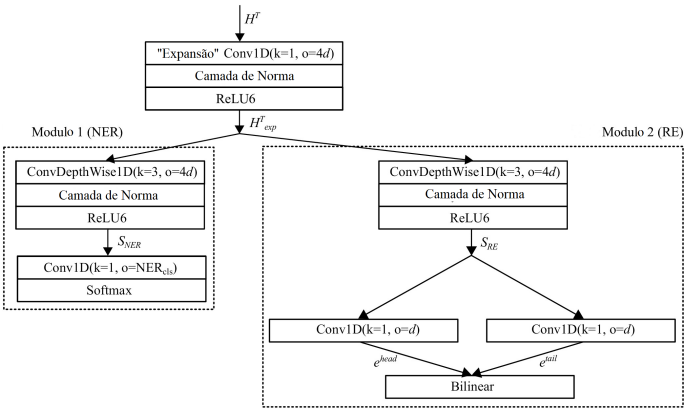


Figura 5. Projeção de Tarefas

$$S_{task\in\{\text{NER,RE}\}} = \text{ReLU6}(\text{LayerNorm}(\text{ConvDW1D}(H^T_{exp})))$$
$$\text{onde} \quad H^T_{exp} = \text{ReLU6}(\text{LayerNorm}(\text{Conv1D}(H^T))) \quad (15)$$

*1) Tarefa 1 – Reconhecimento de Entidade Nomeadas (REN):* para classificar o tipo de entidade de cada posição da sequência, $S_{\text{NER}} \in \mathbb{R}^{M\times d}$ passa por uma camada convolutiva com $REN_{cls}$ neurônios, em que $REN_{cls}$ é o número de classes de entidades. Em seguida, a saída é normalizada por uma função softmax.

$$b = \text{Conv1D}(S_{NER})$$
$$\hat{y}_{NER_i} = \text{softmax}(b(i,1), ..., b(i, NER_{cls})) \quad (16)$$

A função de perda utilizada é a *Negative Log Likelihood* para um mini-batch de $N$ amostras:

$$L = \frac{1}{N}\sum_{j=1}^{N}\sum_{i=1}^{M} y_{NER_{(i,j)}} \log(\hat{y}_{NER_{(i,j)}}) \quad (17)$$

em que $y_{NER_{(i,j)}} \in \mathbb{R}^{M\times NER_{cls}}$ é um mini-batch de sequências one-hot indica a classe correta de entidade.

*2) Tarefa 2 – Extração de Relações:* com base no modelo de [8], a função score na Eq. (19) define a presença de relação entre pares de elementos da sequência $e^{head}, e^{tail} \in \mathbb{R}^{M \times d}$ usando uma função Bilinear:

$$\begin{aligned} e^{head} &= \text{Conv1D}(S_{RE}) \\ e^{tail} &= \text{Conv1D}(S_{RE}) \\ A_{ilj} &= (e_i^{head} L) e_j^{tail} \end{aligned} \tag{18}$$

onde $L \in \mathbb{R}^{d \times L \times d}$ é a matriz de pesos para cada uma das $L$ relações. A matriz $A \in \mathbb{R}^{N \times L \times N}$ é denominada matriz de afinidade e contém o score para a tripleta (head, relation, tail).

A função de perda utlizada é LogSumExp, que soma o escore das relações em $A$ para todos os pares de menções $p^{head}$ e $p^{tail}$:

$$score(p^{head}, p^{tail}) = \log \sum_{\substack{i \in P^{head} \\ j \in P^{tail}}} \exp(A_{ij}) \tag{19}$$

onde $P^{head}$ e $P^{tail}$ representam o conjunto de índices de menções $p^{head}$ e $p^{tail}$, respectivamente.

## III. Experimentos

### A. Pré-processamento dos Dados

Os dados foram pré-processados utilizando-se a biblioteca SpaCy[3], com uma tokenização padrão "en" para o idioma inglês. Acompanhando o trabalho de [8], o problema de hiperonímia foi solucionado empregando-se o controlador hierárquico de vocabulário de MESH [4].

A indexação foi realizada por meio do vocabulário completo de palavras. Uma forma de aperfeiçoar a invariância do sistema foi adicionar ruído à entrada. Assim, as palavras do texto foram alteradas aleatoriamente pela tecla UNK. De forma a gerar os exemplos "negativo" no treinamento, os pares de entidades que não possuíam relacionamentos anotados foram catalogados como *NULL*.

### B. Extração de relações de doenças induzidas por substâncias químicias - Biocriative V

*1) Conjunto de dados:* foi criado a partir do Dataset Comparativo Toxicogenômico (DCT), que lida com interações entre genes, substâncias químicas e doenças. Foram processados 1500 resumos de texto com anotações de tipos de entidades e relações sobre doenças induzidas quimicamente[5]. Nestas, os efeitos colaterais foram relacionados a medicamentos e reações adversas. Os tipos de entidades categóricas foram denominados doenças e substâncias químicas.

A relação entre o par de entidades doença-químico foi categorizada como "doença induzida por substância químicas". Os relacionamentos anotados entre pares de entidades no texto foram considerados "exemplos positivos". Os relacionamentos não anotados no texto foram considerados "exemplos negativos". De forma a realizar uma comparação com o modelo base BRAN [8], foi utilizado também o conjunto de dados ruidosos (gerados automaticamente utilizando a técnica de supervisionamento a distância) proposto pelo autor. Este dados foram criados a partir do conjunto de relações DCT e artigos de PunMed. A distribuição de dados para treinamento e validação são apresentados na Tabela I:

Tabela I
DIVISÃO DE DADOS PARA TREINO / VALIDAÇÃO / TESTE E DADOS ADICIONAIS UTILIZADOS DO BANCO DE DADOS DCT.

| Tipos | Documents | Positivos | Negativos |
|---|---|---|---|
| Train | 500 | 1038 | 4280 |
| Valid | 500 | 1012 | 4136 |
| Teste | 500 | 1066 | 4270 |
| DCT | 15.448 | 26.657 | 146.057 |

*2) Resultados:* a Tabela II compara o modelo proposto com os melhores modelos registrados na literatura para o *benchmark* Biocreative V. Particularmente, dada a disponibilidade de dados ruidosos adicionais apresentados pelo autor do modelo BRAN, o modelo proposto foi analisado com o acréscimo também destes dados (+ Data). Os *embeddings* foram treinados apenas sobre a base alvo.

Tabela II
RECALL/PRECISÃO/F1-SCORE DO DATASET BIOCREATIVE V DE EXTRAÇÃO DE RELAÇÃO DE DOENÇAS INDUZIDAS POR SUBTÂNCIAS QUÍMICAS

| Modelo | P | R | F1 |
|---|---|---|---|
| Gu et al. (2016)[15] | 62,0 | 55,1 | 58,3 |
| Zhou (2016) [16] | 55,6 | 68,4 | 61,3 |
| Gu et al.[17] | 55,7 | 68,1 | 61,3 |
| Verga (2018) [8] | 55,6 | 70,8 | 62,1 |
| +Data | 64,0 | 69,2 | 66,2 |
| Verga (2018) [8] (ensemble) | 63,3 | 67,1 | 65,1 |
| +Data | 65,4 | 71,8 | 68,4 |
| Nguyen (2018) [6] | 71,1 | 72,6 | **71,8** |
| Mandya et al. (2018) [9] | 69,0 | 70,0 | 69,0 |
| Modelo Proposto | 70,1 | 75,4 | **72,6** |
| +Data | 74,35 | 78,74 | **76,3** |

As métricas precisão (P), *recall* (R) e *F1-score* (F1) são listadas para todos os modelos comparados. O modelo [17] codifica os pares de menção usando uma CNN, enquanto [16] usa uma LSTM. Ambos utilizam *cross-validation* para definir o classificadores. O modelo [6] propõe uma estrutura que combina LSTM e CNN e [9] utiliza vetores de representação de palavra em nível de caracteres na entrada de uma CNN. Por sua vez, [8] propõe uma abordagem utilizando o módulo *Transformer* com CNNs. Os resultados obtidos apresentam um F1-score de 72.6, superior ao do modelo base BRAN [8] e ao do modelo [6] – o melhor resultado encontrado na literatura. A utilização de dados adicionais permitiram um considerável incremento do desempenho, atingindo-se um F1-score de 76.3. Os melhores resultados estão em negrito.

Na Tabela III, são comparadas as variantes do modelo proposto. Os resultados são calculados com base em 10

[3]https://spacy.io/

[4]https://www.nlm.nih.gov/databases/download/mesh.html

[5]https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4860626/

repetições, considerando os exemplos positivos, negativos e dados adicionais. Outras variantes são avaliadas: Bloco com Conexão Residual (BCR), Encoder de palavras baseado em Caracteres (EC) e Tempo de Computação Adaptativo (TCA).

Tabela III
RESULTADOS DO BIOCREATIVE V PARA OS VÁRIOS TIPOS DE RELAÇÃO PROPOSTOS NO MODELO.

| BCR | EC | TCA | +Data | P | R | F1 |
|---|---|---|---|---|---|---|
| x | x | x | x | 74,35 | 78,74 | $76,36 \pm 0.58$ |
| | x | x | x | 67,61 | 76,79 | $71,77 \pm 2,68$ |
| x | | x | x | 75,46 | 76,00 | $75,64 \pm 0,71$ |
| x | x | | x | 75,10 | 77,82 | $76,32 \pm 1,29$ |
| x | x | x | | 74,65 | 67,92 | $71,03 \pm 1,85$ |
| | x | x | | 68,12 | 76,02 | $71,75 \pm 2,64$ |
| x | | x | | 62,17 | 72,10 | $66,70 \pm 3,5$ |
| x | x | | | 69,94 | 75,82 | $72,66 \pm 1,00$ |

Dos experimentos realizados, pode-se observar que cada componente acrescenta uma melhora no desempenho quando comparado à configuração sem tal componente. O TCA proporcionou um melhor desempenho apenas quando utilizado em conjunto com dados adicionais.

De forma a avaliar a significância estatística dos resultados, o teste de Wilcoxon [18] foi empregado na comparação entre o modelo base e as duas melhores configurações do modelo proposto (considerando as que não utilizaram dados adicionais). A hipótese considera que as distribuições de onde os resultados (F1-Score) dos dois sistemas foram amostrados são idênticas. Por convenção, a hipótese nula é rejeitado para valores de $p$ menores ou iguais a 0,05. Os resultados do teste são apresentados na Tabela IV.

Tabela IV
TESTE DE SIGNIFICÂNCIA ESTATÍSTICA (TESTE DE WILCOXON)

| Hipótese | $p$ |
|---|---|
| BRAN - Modelo (BCR+EC+TCA) | 0,00038 |
| BRAN - Modelo (BCR+EC) | 0,00152 |
| Modelo (BCR+EC+TCA) - Modelo (BCR+EC) | 0,01867 |

Observa-se que as duas configurações propostas, quando comparadas com o modelo base, são estatisticamente independentes ($p < 0,05$). O mesmo pode ser afirmado a respeito da comparação entre as duas melhores configurações propostas.

### *C. Conjunto de Dados baseada na Toxicogenômica Comparativa e PubMed (DTC)*

*1) Conjunto de dados:* o conjunto, proposto em [8], foi construído utilizando o método de supervisão à distância contendo anotações de relações definidas em nível de documento. O conjunto de dados do Dataset baseado na Toxicogenômica Comparativa (DTC) organiza as interações entre genes, substâncias químicas e doenças. Cada relação no DTC está associada especificamente a um par de entidades e um artigo PubMed onde a relação é observada. O conjunto de dados pode ser encontrado no Github [5].

[5]https://github.com/patverga/bran

Os documentos utilizados para fins desta pesquisa contém sequências com menos de 500 tokens, enquanto o conjunto de dados é formado por 68.400 documentos. No total, são 166.474 exemplos positivos, organizados em 13 tipos de relações. Estas relações são dadas entre pares de entidades químico-doença, gene-químico e gene-doença. A Tabela V contém a distribuição de pares de entidades para conjuntos de dados.

Tabela V
DESTRUIBUIÇÃO DE EXEMPLOS POSITIVOS/NEGATIVOS ORGANIZADOS POR TIPO DE RELAÇÃO

| Tipos | Documentos | Positivos | Negativos |
|---|---|---|---|
| Químico/Doença | 64.139 | 93.940 | 571.932 |
| Químico/Gene | 34.883 | 63.463 | 360.100 |
| Gene/Doença | 32.286 | 9.071 | 266.461 |

O conjunto de dados DTC contém mais de 100 tipos de relações organizadas de forma hierárquica. O mapeamento resultou em 13 tipos de relações. A distribuição de relações para treino, validação e teste é mostrada na Tabela VI.

Tabela VI
DISTRIBUIÇÃO DE DADOS PARA TREINO/VALIDAÇÃO/TESTE.OS VALORES SÃO INDICADOS EM PORCETAGEM.

| | Treino | Validação | Teste |
|---|---|---|---|
| **Químico/Doença** | | | |
| marcado/mecanismo | 41.562 | 5.126 | 5167 |
| terapêutico | 24.151 | 2.929 | 3.059 |
| **Gene/Doença** | | | |
| marcador/mecanismo | 5.930 | 825 | 819 |
| terapêutico | 560 | 77 | 75 |
| **Químico/Gene** | | | |
| aumento_expressão | 15.581 | 1.958 | 2.137 |
| aumento_metabólico_proc | 5.986 | 740 | 638 |
| diminuição_expressão | 5.870 | 698 | 783 |
| aumento_atividade | 4.154 | 467 | 497 |
| efeito_resposta | 3.834 | 475 | 508 |
| diminuição_atividade | 3.124 | 396 | 434 |
| efeito_transportação | 3.009 | 333 | 361 |
| aumento_reação | 2.881 | 367 | 353 |
| diminuição_reação | 2.221 | 247 | 269 |
| diminuição_metabólico_proc | 798 | 100 | 120 |

*2) Resultados:* conforme mostrado na Tabela VII, o modelo proposto estende e supera a metodologia de [8] em termos de desempenho para problemas de classificaccão multiclasse. Durante o treinamento, devido a datasets desequilibrados, cada classe tem um peso inversamente proporcional ao número de exemplos dentro na função de custo. O modelo proposto prevê com melhor desempenho as relações com maior quantidade de exemplos. Cada experimento leva de 12 h a 14 h para o treinamento de 60000 iterações em uma máquina Intel (R) Xeon (R) CPU E5-2660 2.00 Ghz. Os *embeddings* foram treinados apenas sobre a base alvo.

Tabela VII
RESULTADOS DA AVALIAÇÃO DE EXTRAÇÃO DE RELAÇÕES NO CONJUNTO DE DADOS CTD (DADOS DE TESTE). OS VALORES SÃO INDICADOS EM PORCENTAGEM.

| | Modelo Proposto | | | BRAN (2018) | | |
|---|---|---|---|---|---|---|
| | P | R | F1 | P | R | F1 |
| **Químico/Doença** | | | | | | |
| marcado/mecanismo | 89,6 | 83,7 | **86,5** | 46,2 | 57,9 | 51,3 |
| terapêutico | 83,7 | 79,4 | **81,5** | 55,7 | 67,1 | 60,8 |
| **Gene/Doença** | | | | | | |
| marcador/mecanismo | 89,1 | 74,7 | **81,3** | 42,2 | 44,4 | 43,0 |
| terapêutico | 63,3 | 26,4 | **37,3** | 52,6 | 10,1 | 15,8 |
| **Químico/Gene** | | | | | | |
| aumento_expressão | 57,0 | 69,7 | **62,7** | 39,7 | 48,0 | 43,3 |
| aumento_metabólico_proc | 50,5 | 53,8 | **52,1** | 26,3 | 35,5 | 29,9 |
| diminuição_expressão | 49,1 | 40,5 | **44,4** | 34,4 | 32,9 | 33,4 |
| aumento_atividade | 50,1 | 34,8 | **41,** | 24,5 | 24,7 | 24,4 |
| efeito_resposta | 74,0 | 41,8 | **53,4** | 40,9 | 35,5 | 37,4 |
| diminuição_atividade | 59,1 | 42,0 | **49,0** | 30,8 | 19,4 | 23,5 |
| efeito_transportação | 50,8 | 36,6 | **42,5** | 28,7 | 23,8 | 25,8 |
| aumento_reação | 30,3 | 5,6 | **9,5** | 12,8 | 5,6 | 7,4 |
| diminuição_reação | 25,8 | 8,7 | **13,0** | 12,3 | 5,7 | 7,4 |
| diminuição_metabólico_proc | 40,0 | 6,7 | **11,5** | 28,9 | 7,0 | 11,0 |

## IV. CONCLUSÃO

Este trabalho explorou a importância de diferentes configurações a partir da arquitetura *Transformer* para extração de relações do modelo BRAN. Vetores de palavras baseadas em caracteres foram utilizados com a finalidade de extrair características dos elementos que compõe cada palavra. O bloco convolutivo com conexão residual como função de transição do *transformer* e um módulo de Tempo de Computação Adaptativa foram utilizados com intenção de melhorar a desempenho do sistema. Resultados experimentais para os conjuntos de dados Biocreative e DTC mostraram que o modelo supera outras abordagens baseadas em rede neurais. Estes resultados foram obtidos a partir de treinamentos sem a utilização de *embeddings* pré-treinados; neste caso apenas os *embeddings* de caracteres ajudaram no tratamento de palavras desconhecidas. Em trabalhos futuros pretende-se fazer uso de *embeddings* pré-treinados, baseados em palavras e caracteres e empregando técnicas como Mask LM em modelos *transformer*.

*Referências*


[1] N. Bach e S. Badaskar, "A Review of Relation Extraction", *Literature review for Language and Statistics II*, vol. 2, 2007.

[2] E. Shahab, "A Short Survey of Biomedical Relation Extraction Techniques", *arXiv preprint arXiv:1707.05850*, 2017.

[3] K. Gabor, D. Buscaldi, A.-K. Schumann, B. QasemiZadeh, H. Zargayouna e T. Charnois, "Semeval-2018 Task 7: Semantic relation extraction and classification in scientific papers", pp. 679–688, 2018.

[4] K. Swampillai e M. Stevenson, "Extracting Relations Within and Across Sentences", em *Proceedings of the International Conference Recent Advances in Natural Language Processing 2011*, jan. de 2011, pp. 25–32.

[5] C. Quirk e H. Poon, "Distant Supervision for Relation Extraction beyond the Sentence Boundary", *arXiv preprint arXiv:1609.04873*, 2016.

[6] D. Q. Nguyen e K. Verspoor, "Convolutional neural networks for chemical-disease relation extraction are improved with character-based word embeddings", *arXiv preprint arXiv:1805.10586*, 2018.

[7] N. Peng, H. Poon, C. Quirk, K. Toutanova e W.-t. Yih, "Cross-Sentence N-ary Relation Extraction with Graph LSTMs", *arXiv preprint arXiv:1708.03743*, 2017.

[8] P. Verga, E. Strubell e A. McCallum, "Simultaneously Self-Attending to All Mentions for Full-Abstract Biological Relation Extractionn", *arXiv preprint arXiv:1802.10569*, 2018.

[9] A. Mandya, D. Bollegala, F. Coenen e K. Atkinson, "Combining Long Short Term Memory and Convolutional Neural Network for Cross-Sentence n-ary Relation Extraction", *arXiv preprint arXiv:1811.00845*, 2018.

[10] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov e L.-C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks", *arXiv preprint arXiv:1801.04381*, 2018.

[11] M. Dehghani, S. Gouws, O. Vinyals, J. Uszkoreit e Ł. Kaiser, "Universal Transformers", *arXiv preprint arXiv:1807.03819*, 2018.

[12] Y. Cui, Z. Chen, S. Wei, S. Wang, T. Liu e G. Hu, "Attention-over-Attention Neural Networks for Reading Comprehension", *arXiv preprint arXiv:1607.04423*, 2016. DOI: 10.18653/v1/P17-1055.

[13] J. L. Ba, J. R. Kiros e G. E. Hinton, "Layer normalization", *arXiv preprint arXiv:1607.06450*, 2016.

[14] A. Graves, "Adaptive Computation Time for Recurrent Neural Networks", *arXiv preprint arXiv:1603.08983*, 2016.

[15] J. Gu, L. Qian e G. Zhou, "Chemical-induced disease relation extraction with various linguistic features", *Database*, vol. 2016, 2016.

[16] H. Zhou, H. Deng, L. Chen, Y. Yang, C. Jia e D. Huang, "Exploiting syntactic and semantics information for chemical–disease relation extraction", *Database*, vol. 2016, 2016. DOI: 10.1093/database/baw048.

[17] J. Gu, F. Sun, L. Qian e G. Zhou, "Chemical-induced disease relation extraction via convolutional neural network", *Database*, vol. 2017, 2017. DOI: 10.1093/database/bax024.

[18] F. Wilcoxon, "Individual comparisons by ranking methods", em *Breakthroughs in statistics*, Springer, 1992, pp. 196–202.