USO DO BERT NA EXPANSÃO DE CONSULTAS E DOCUMENTOS PARA BUSCA E RECUPERAÇÃO DE INFORMAÇÃO

Elaine da Costa Tady Marques

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia de Sistemas e Computação.

Orientador: Geraldo Bonorino Xexéo

Rio de Janeiro
Setembro de 2022

USO DO BERT NA EXPANSÃO DE CONSULTAS E DOCUMENTOS PARA BUSCA E RECUPERAÇÃO DE INFORMAÇÃO

Elaine da Costa Tady Marques

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Orientador: Geraldo Bonorino Xexéo

Aprovada por: Prof. Eduardo Bezerra da Silva
Prof. Jano Moreira de Souza

RIO DE JANEIRO, RJ – BRASIL
SETEMBRO DE 2022

da Costa Tady Marques, Elaine

Uso do BERT na Expansão de Consultas e Documentos para Busca e Recuperação de Informação/Elaine da Costa Tady Marques. – Rio de Janeiro: UFRJ/COPPE, 2022.

XII, 49 p. 29, 7cm.

Orientador: Geraldo Bonorino Xexéo

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia de Sistemas e Computação, 2022.

Referências Bibliográficas: p. 46 – 49.

1. Expansão de consultas. 2. Expansão de documentos. 3. BERT. I. Bonorino Xexéo, Geraldo. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

Dedico à minha família

Agradecimentos

Agradeço primeiramente a Deus por ter me sustentado ao longo deste trabalho em meio a tempos complicados de pandemia. Agradeço ao meu marido Jones Marques Augusto que esteve ao meu lado durante todo o período de mestrado e ao longo da dissertação.

E por fim, agradeço ao meu orientador Geraldo Bonorino Xexéo pelo acompanhamento e dedicação desde as disciplinas até a conclusão desta dissertação.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

USO DO BERT NA EXPANSÃO DE CONSULTAS E DOCUMENTOS PARA BUSCA E RECUPERAÇÃO DE INFORMAÇÃO

Elaine da Costa Tady Marques

Setembro/2022

Orientador: Geraldo Bonorino Xexéo

Programa: Engenharia de Sistemas e Computação

Métodos de Expansão de Consultas e Expansão de Documentos são utilizados para obter melhores resultados em modelos de Busca e Recuperação de Informação. Foram utilizados os discos 1 e 2 do *dataset* TREC, totalizando 741853 documentos, com as consultas 51-100 como conjunto de desenvolvimento e as consultas 101-200 como conjunto de testes para avaliação dos métodos de expansão.

Foram exploradas as etapas de Pré-processamento, Indexação, Expansão de Consultas e Expansão de Documentos, utilizando as funções de expansão RM3-IDF (ROY *et al.*, 2019) e expandindo os documentos do *dataset* utilizando a biblioteca OpenNMT com o método Doc2Query (NOGUEIRA *et al.*, 2019), adicionando consultas ao final de cada documento para expandi-lo. E então, o BERT foi utilizado para *re-rank* dos documentos retornados nas etapas de expansão (NOGUEIRA e CHO, 2019). Os melhores resultados nos experimentos foram utilizando a Expansão de Consultas $RM3_3^+$ + BM25 + *Re-Rank* com o BERT LARGE, considerando P@10 e NDCG e mantendo o RECALL por utilizar a mesma lista e reordenar.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

USE OF BERT FOR QUERY EXPANSION AND DOCUMENT EXPANSION FOR INFORMATION SEARCH AND RETRIEVAL

Elaine da Costa Tady Marques

September/2022

Advisor: Geraldo Bonorino Xexéo

Department: Systems Engineering and Computer Science

Query Expansion and Document Expansion methods are used to obtain better results in Information Retrieval models. Disks 1 and 2 of the TREC dataset were used, totaling 741853 documents, with queries 51-100 as the development set and queries 101-200 as a test set to evaluate the expansion methods.

The steps of Pre-processing, Indexing, Query Expansion and Document Expansion were explored, using the RM3-IDF expansion functions (ROY *et al.*, 2019) and expanding the dataset documents using the OpenNMT library with the Doc2Query method (NOGUEIRA *et al.*, 2019), adding queries at the end of each document to expand it. And then, BERT was used to re-rank the documents returned in the expansion steps (NOGUEIRA e CHO, 2019). The best results in the experiments were using the Query Expansion $RM3_3^+$ + BM25 + *Re-Rank* with BERT LARGE, considering P@10 and NDCG and keeping RECALL by using the same list and reordering.

Sumário

1 Introdução **1**
1.1 Motivação . . . 1
1.2 Objetivo . . . 2
1.3 Organização . . . 3

2 Revisão Bibliográfica **4**
2.1 Recuperação de Informação (RI) . . . 4
2.2 Redes Neurais . . . 5
2.2.1 Deep Feedforward Neural Network . . . 7
2.2.2 Redes Neurais Convolucionais . . . 9
2.2.3 Redes Neurais Recorrentes . . . 9
2.3 Modelos *sequence-to-sequence* . . . 10
2.4 Transformadores . . . 10
2.5 Modelos para o *ranking* consulta-documento . . . 13
2.5.1 BM25 . . . 13
2.5.2 Jelinek-Mercer *smoothing* e Dirichlet *smoothing* . . . 15
2.6 Expansão de Consultas . . . 16
2.7 Expansão de Documentos . . . 19
2.8 BERT . . . 21
2.9 TREC . . . 27

3 Método Proposto **32**
3.1 Passo a passo . . . 32

4 Resultados e Discussões **35**
4.1 Configuração da máquina utilizada . . . 35
4.2 Métricas de avaliação . . . 35
4.2.1 Precision . . . 35
4.2.2 Recall . . . 36
4.2.3 MAP . . . 36
4.2.4 NDCG . . . 36
4.3 Resultados . . . 37

4.3.1 *Re-rank* BERT - Expansão de Consultas 37
4.3.2 *Re-rank* BERT - Expansão de Documentos 40
4.3.3 *Re-rank* BERT - Expansão de Consultas e Documentos 42

5 Conclusões **44**

Referências Bibliográficas **46**

Lista de Figuras

2.1 Exemplo que não pode ser resolvido por uma transformação linear, adaptado de NEUBIG (2017) 6
2.2 Função de ativação sigmoide, adaptado de (SALEHINEJAD *et al.*, 2017) 7
2.3 Perceptron, adaptado de (SENGUPTA *et al.*, 2020) 7
2.4 Rede *Feedforward*, adaptado de (SENGUPTA *et al.*, 2020) 8
2.5 *Deep Neural Network*, adaptado de (SENGUPTA *et al.*, 2020) 8
2.6 Transformador, adaptado de (ALAMMAR, 2018b) 11
2.7 Codificador e decodificador, adaptado de (ALAMMAR, 2018b) 12
2.8 Codificador e decodificador, adaptado de (ALAMMAR, 2018b) 13
2.9 Etapas da Expansão de Documentos, adaptado de (NOGUEIRA *et al.*, 2019) 20
2.10 Classificação de sentença com o BERT, adaptado de (ALAMMAR, 2018a) 24
2.11 Mascaramento de palavras, adaptado de (ALAMMAR, 2018a) 24
2.12 Previsão da próxima frase, adaptado de (ALAMMAR, 2018a) 25

3.1 Etapas da Expansão de Consultas 33
3.2 Etapas da Expansão de Documentos 33

Lista de Tabelas

2.1 Vetor da palavra *apple* na primeira sentença com Word2vec 21
2.2 Vetor da palavra *apple* na primeira sentença com o BERT 21
2.3 Vetor da palavra *apple* na segunda sentença com Word2vec 22
2.4 Vetor da palavra *apple* na segunda sentença com o BERT 22
2.5 Frase de exemplo, com os *tokens* especiais [CLS] e [SEP], adaptado de (ALAMMAR, 2018a) 23
2.6 Representação da frase anterior com 768 linhas x 5 colunas, adaptado de (ALAMMAR, 2018a) 23
2.7 Representação completa da frase apenas com o vetor [CLS] (ALAMMAR, 2018a) 23
2.8 Conteúdo dos discos 1, 2 e 3, adaptado de (HARMAN, 1995) 29
2.9 Estatísticas do Documento, adaptada de (HARMAN, 1995). 29

4.1 BM25 sem expandir as consultas 38
4.2 BM25 com Expansão de Consultas 38
4.3 Dirichlet Similarity com Expansão de Consultas 38
4.4 Jelinek Mercer Similarity com Expansão de Consultas 38
4.5 Re-rank com o BERT 39
4.6 Re-rank com o BERT 39
4.7 BM25 (Documentos Expandidos) 40
4.8 BERT - BM25 (Documentos Expandidos) 40
4.9 BM25 (Documentos Expandidos) 41
4.10 BERT: BM25 (Documentos Expandidos) 41
4.11 BERT: BM25 (Documentos Expandidos) 42
4.12 BM25 (Consultas Expandidas e Documentos Expandidos) 43
4.13 BERT: BM25 (Consultas Expandidas e Documentos Expandidos) . . 43

Capítulo 1

Introdução

Neste capítulo, será apresentada a motivação para o desenvolvimento do trabalho, o objetivo com as Expansões de Consulta e Documentos e a organização do que foi desenvolvido em cada capítulo.

1.1 Motivação

A área de Recuperação de Informação tem como objetivo principal prover fácil acesso à informações de interesse de um usuário, como em documentos ou páginas web. Indo além da indexação de textos e busca por documentos úteis em uma coleção. Lidando também com problemas como a classificação de textos e visualização de imagens (BAEZA-YATES e RIBEIRO-NETO, 2013).

Sistemas de Recuperação de Informação classificam os documentos de acordo com a sua importância para uma consulta de um usuário, atribuindo uma pontuação para cada documento e classificando de acordo com essa pontuação (SINGHAL *et al.*, 2001).

Para tal pontuação relacionada a proximidade entre os documentos e consultas, o modelo mede a semelhança entre o vetor da consulta e o vetor do documento, ou seja, o texto de ambos convertidos em vetores. O ângulo para semelhança utilizado pode ser uma medida como o cosseno do ângulo (SINGHAL *et al.*, 2001).

Em 1992, com o início do TREC (Text Retrieval Conference) possuindo grandes coleções de textos, novas técnicas foram desenvolvidas visando melhorar a eficácia da recuperação nessas grandes coleções. Ramificando em campos como: recuperação de informações de fala e recuperação de idiomas diferente de Inglês (SINGHAL *et al.*, 2001).

No início dos anos 90, novas técnicas como o *pseudo-feedback* foram desenvolvidas devido à ausência de termos de *feedback* do usuário, gerando novos termos para consulta e a expandindo baseado nos documentos relevantes recuperados, sendo eficaz principalmente para consultas curtas de um usuário (SINGHAL *et al.*, 2001).

A consulta de um usuário é expandida (ROY *et al.*, 2019), adicionando termos a fim de retornar documentos mais precisos em uma coleção, melhorando então, os resultados utilizando métricas de avaliação como P@10, RECALL, MAP e NDCG.

Outra maneira de melhorar a Recuperação de Informação é expandindo os documentos a partir da adição de consultas geradas, utilizando o modelo Doc2query, em que o objetivo é prever para cada documento um conjunto de consultas relevantes. E então, o BERT (Bidirectional Encoder Representations from Transformers) é utilizado como *re-rank* das consultas relevantes para cada documento (NOGUEIRA *et al.*, 2019).

O treinamento do BERT foi feito utilizando TPU (Tensor Processing Unit), que é um desenvolvimento em ASICs (Circuito Integrado Específico de Aplicação) pelo Google, projetado para treinamento de aprendizado profundo. Estes processadores usados nos *data centers* do Google Cloud são capazes de lidar com uma enorme quantidade de dados (SHAHID e MUSHTAQ, 2020).

ASIC é um circuito integrado que pode ser personalizado para qualquer uso específico, como para aprendizado de máquina ou mineração de dados. Esses chips usam a biblioteca matemática *tensorflow* (SHAHID e MUSHTAQ, 2020).

1.2 Objetivo

Esse trabalho tem como objetivo utilizar a metodologia proposta por (ROY *et al.*, 2019) para Expansão de Consultas, testando as melhores abordagens para documentos dos discos 1 e 2 (TREC1 e TREC2) do TREC (Text Retrieval Conference), utilizando as consultas 51-100 (TREC1 *ad-hoc*) como conjunto de desenvolvimento (ajuste de parâmetros) e as consultas 101-200(TREC2,3 *ad-hoc*) como conjunto de teste (avaliação de resultado).

Além das consultas serem expandidas, também será avaliado o método proposto por NOGUEIRA *et al.* (2019) para expandir documentos, ou seja, adicionando n consultas ao final de cada documento, utilizando a biblioteca OpenNMT.

E então, o BERT é utilizado para a partir do *ranking* (NOGUEIRA e CHO, 2019) retornado em cada expansão, gerar um *re-rank* para melhorar os resultados obtidos, gerando um novo *top* n de documentos relevantes para cada consulta.

Algumas questões de pesquisa foram levantadas para este trabalho, se baseando no descrito até aqui e considerando os métodos adotados e passo a passo utilizado, como pode ser visto a seguir:

- Após os ajustes de parâmetros com a Expansão de Consultas ou a Expansão de Documentos no *dataset* TREC foi possível obter melhores resultados utilizando o BERT ao reordenar a lista inicial obtida pelo BM25 (Modelo para o *ranking*

de similaridade)?

- Existe vantagem dos métodos de Expansão de Consultas utilizados sob o método de Expansão de Documentos Doc2query?
- A utilização de GPU/TPU nas etapas de treinamento melhora significativamente o desempenho nos testes realizados no *dataset* TREC?

1.3 Organização

No capítulo 2 será apresentada a revisão da literatura, apresentando os conceitos de Recuperação de Informação, Redes Neurais, Expansão de Consultas, Expansão de Documentos, *Dataset* utilizado e BERT. No capítulo 3 será apresentada a metodologia utilizada para expandir as consultas, expandir os documentos e *re-rank* com o BERT, No capítulo 4 serão apresentados os resultados dos experimentos com as métricas: P@10, RECALL, MAP e NDCG. E então, no capítulo 5 será apresentada a conclusão do trabalho e possíveis trabalhos futuros.

Capítulo 2

Revisão Bibliográfica

Neste capítulo, será apresentada a revisão da literatura com os conceitos relacionados à Recuperação de Informação, Redes Neurais, modelos de similaridade, Expansão de Consultas, Expansão de Documentos, BERT(Bidirectional Encoder Representations from Transformers) e o *dataset* TREC.

2.1 Recuperação de Informação (RI)

A RI trata da recuperação de registros não estruturados, como textos em linguagem natural, imagens, áudio e vídeo (GREENGRASS, 2000). Encontrando documentos não estruturados que satisfaça uma necessidade de informação dentro de grandes coleções (MANNING *et al.*). Os documentos em um repositório geralmente são chamados de corpus e a "biblioteca" de coleção, que pode ser mensagens de e-mails, notícias, entre outros (GREENGRASS, 2000).

Os registros estruturados são componentes nomeados de acordo com uma sintaxe bem definida, como em um banco de dados que os registros de um mesmo tipo tem a mesma sintaxe. Se considerarmos as linhas de uma tabela em uma banco de dados relacional, elas terão as mesmas colunas e cada componente de um registro tem um significado bem definido (GREENGRASS, 2000).

Um mecanismo de busca pode encontrar registros a partir do valor fornecido. Em um SGBD(Sistema de Gerenciamento de Banco de Dados) pode-se recuperar o valor da coluna "idade" em uma tabela "funcionário" em que cada coluna "idade" dentro da tabela "funcionário" terá a mesma semântica (GREENGRASS, 2000).

Em uma coleção de documentos não estruturados (dados que não possuem uma estrutura clara (MANNING *et al.*)) em um mecanismo de pesquisa, não há posição sintática para encontrar dados de acordo com uma semântica, como no exemplo anterior da idade dos funcionários. Mesmo que os documentos sejam sobre funcionários, não há garantias que todos especifiquem a idade (GREENGRASS, 2000).

Já uma coleção semiestruturada, pode ser pensada como documentos que compartilham uma estrutura e semântica. Em uma coleção sobre países, o primeiro parágrafo pode conter o nome, localização e população, seguindo uma forma consistente e o segundo parágrafo pode listar indústrias e exportação, seguindo também um padrão. Então, apesar de não possuir colunas definidas, os dados ocupam posições padronizadas (GREENGRASS, 2000).

Quando um documento possui um cabeçalho estruturado e um corpo não estruturado, ele também é considerado um documento parcialmente estruturado, que é o caso geralmente encontrado em RI. Esse cabeçalho costuma conter metadados como o nome do autor, data de publicação, entre outros (ex.: Z39.50) (GREENGRASS, 2000).

Uma solicitação de RI é normalmente chamada de consulta e especifica as características à serem recuperadas dos documentos. Exemplo: "Os documentos devem ser sobre 'Recuperação de informações' e seu autor deve ser 'Smith"'. Nesse caso, a consulta precisa de documentos com corpo(parte não estruturada) sobre um tópico e com nome do autor(parte estruturada). Esse tópico ou necessidade de informação, é representado por uma consulta de um usuário (GREENGRASS, 2000).

O conceito de documentos relevantes e não relevantes é referente aos documentos que satisfazem a necessidade de um usuário, sendo ou não sobre um tópico em questão. Outro conceito importante em RI é a relevância, que está relacionado ao quanto os documentos retornados satisfazem a necessidade de um usuário. Como esse conceito é subjetivo considerando juízes humanos, então é adotado o grau de relevância (GREENGRASS, 2000).

Algumas métricas baseadas em relevância são importantes em RI, como *precision* e *recall*. *Precision* é a proporção de itens recuperados para todos os itens recuperados. Já o *recall* é a proporção de itens relevantes recuperados para todos os itens relevantes. A maioria dos sistemas de RI retornam uma lista ordenada de documentos, já que o usuário começará com o primeiro documento, que é o considerado como melhor documento na lista (GREENGRASS, 2000).

Ao utilizar *datasets* do TREC, uma medida comum adotada é a precisão média, que é bastante utilizada em pesquisas. O objetivo é refletir o fato de que a precisão varia, geralmente diminuindo à medida que o *recall* aumenta. Sendo uma forma de resumir o tipo de curva de um gráfico de *precision* X *Recall* (GREENGRASS, 2000).

2.2 Redes Neurais

Uma Rede Neural Artificial compreende muitas redes interconectadas, unidades funcionais simples, ou neurônios que atuam em conjunto como processadores de informação paralelos, sendo capaz de resolver problemas de Classificação ou Regressão.

A camada de neurônios recebe informações através de unidades conhecidas como camadas de entrada (SENGUPTA *et al.*, 2020).

Redes Neurais são capazes de aprender alterando a distribuição de pesos para que seja possível aproximar a função representativa dos padrões de entrada, até que haja uma boa representação alcançada. As informações são passadas entre as camadas considerando a função objetivo e regras de aprendizagem, através de unidades conhecidas como camadas ocultas. E então, retransmitindo as informações através de unidades da camada de saída (SENGUPTA *et al.*, 2020).

Cada neurônio dentro de uma camada oculta está conectado às saídas dos neurônios da camada anterior ou a um subconjunto, sendo multiplicado por um peso. O neurônio calcula a soma dos produtos de suas entradas(saídas da camada anterior) e seus pesos correspondentes, cálculo chamado de produto escalar entre o vetor de entrada e o de peso (medida de semelhança entre os dois) (SENGUPTA *et al.*, 2020).

Cada neurônio adiciona um valor de polarização (*bias*) ao resultado de sua soma de produtos de entrada e passa por uma função de ativação não linear (Caso de classificação não linear). Os pesos são calculados por otimização sobre os pares de entrada-saída para minimizar a função de erro (SENGUPTA *et al.*, 2020).

Funções de ativação moldam as saídas dos neurônios artificiais, sendo parte integrante das redes neurais (LEDERER, 2021). Uma das funções de ativação mais populares é a função sigmoide dada pela equação 2.1 e representada pela figura 2.2, que é adequada para redes onde a saída está na faixa [0,1]. Funções de ativação não lineares são mais poderosas do que as funções lineares por serem capazes de desenhar limites não lineares, como seria necessário na figura 2.1. E as funções de perda avaliam o desempenho da rede comparando a saída com o alvo correspondente (SALEHINEJAD *et al.*, 2017).

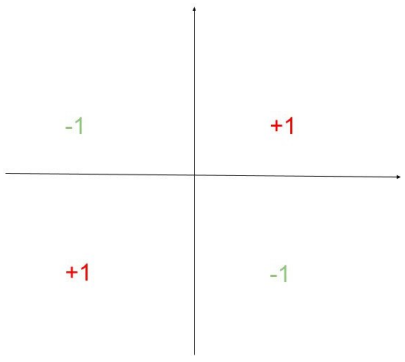


Figura 2.1: Exemplo que não pode ser resolvido por uma transformação linear, adaptado de NEUBIG (2017)

$$\sigma(x) = \frac{1}{1 + e^{-}x} \tag{2.1}$$

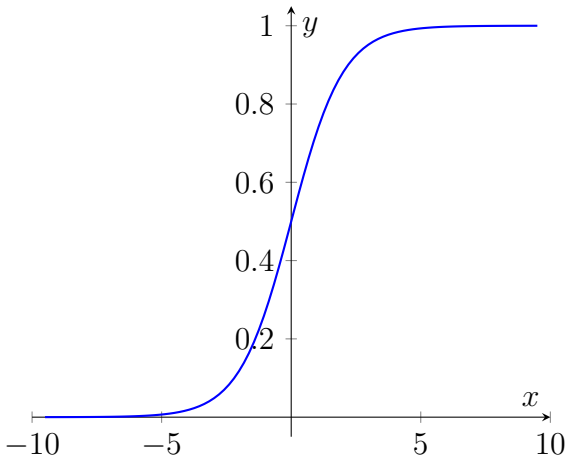


Figura 2.2: Função de ativação sigmoide, adaptado de (SALEHINEJAD *et al.*, 2017)

A figura 2.3 apresenta uma rede neural Perceptron, com as entradas sendo representadas por $x_1, ..., x_p$, os pesos sendo representados por $w_0, ..., w_p$, o somatório do produto das entradas e pesos correspondentes, a função de ativação e a saída do modelo (SENGUPTA *et al.*, 2020).

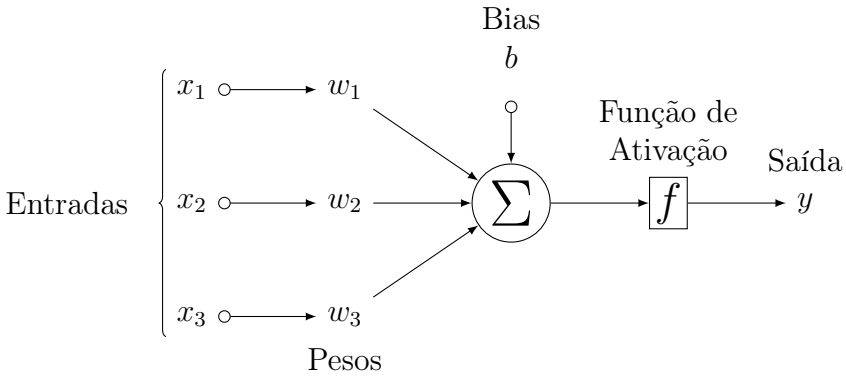


Figura 2.3: Perceptron, adaptado de (SENGUPTA *et al.*, 2020)

2.2.1 Deep Feedforward Neural Network

Redes Neurais *Feedforward* funcionam com um algoritmo de classificação de aprendizado de máquina composto por unidades (nós) organizadas em camadas, em que cada unidade se relaciona com outras unidades, podendo cada conexão entre as camadas e as unidades ter pesos diferentes, como pode ser visto na figura 2.4. Esses pesos medem a quantidade potencial de conhecimento da rede (ABIODUN *et al.*, 2018).

O processamento de informações envolve a unidade de entrada (entrada dos dados) fluindo até a camada de saída, sendo transmitidas em apenas uma direção, ou seja, sem *feedback* entre as camadas. Transmitindo então, as informações dos nós de entrada para os nós ocultos até os nós de saída (ABIODUN *et al.*, 2018).

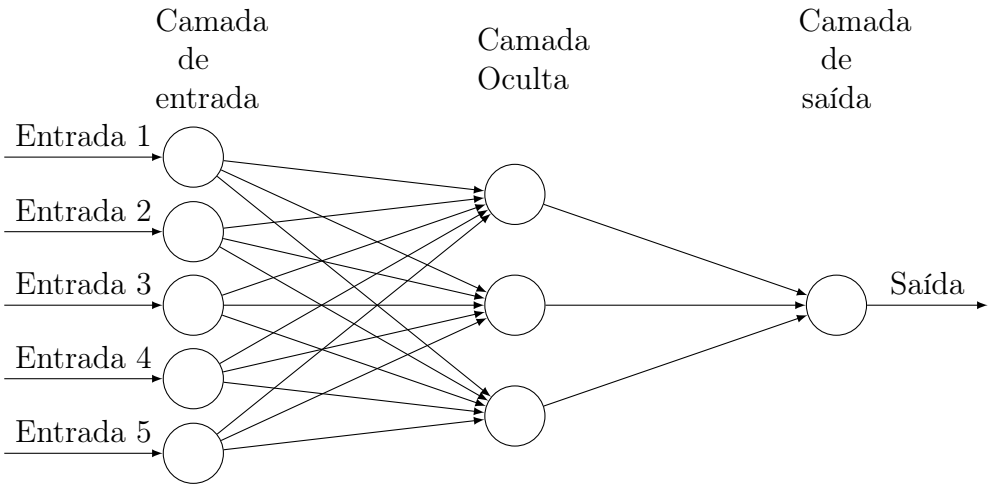


Figura 2.4: Rede *Feedforward*, adaptado de (SENGUPTA *et al.*, 2020)

Quando uma Rede Neural Multicamadas contém várias camadas ocultas, é chamada de Rede Neural Profunda, como pode ser visto na figura 2.5, ajudando na modelagem de relações não lineares complexas com mais eficiência. Uma função complexa pode ser modelada com menor número de unidades computacionais devido ao aprendizado hierárquico possível com os múltiplos níveis de não linearidade. O algoritmo de aprendizado mais comum para treinar esse modelo é o de *Backpropagation* com gradiente descendente (SENGUPTA *et al.*, 2020).

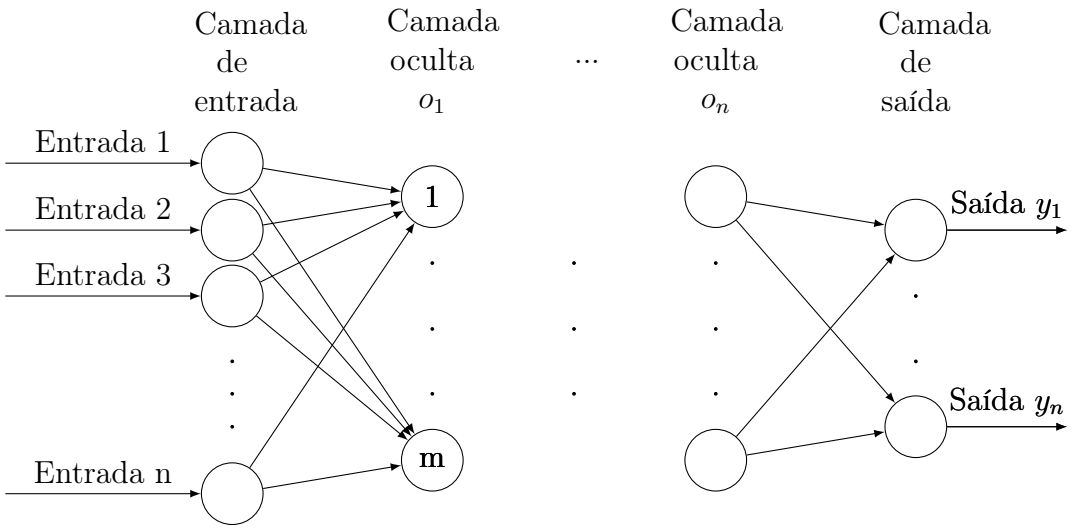


Figura 2.5: *Deep Neural Network*, adaptado de (SENGUPTA *et al.*, 2020)

Deep Learning é um subconjunto de *Machine Learning* inspirado nos padrões de processamento de informações do cérebro humano. Utilizando uma grande quan-

tidade de dados para mapear uma entrada fornecida para rótulos. É projetada usando várias camadas de algoritmos como Redes Neurais Artificiais. Pode ser aplicada para problemas como: reconhecimento de fala, previsão de preço, previsão do tempo, biometria, análise de sentimentos e classificações (ALZUBAIDI *et al.*, 2021).

Podem ser classificadas em três categorias, sendo (ALZUBAIDI *et al.*, 2021):

- Aprendizado supervisionado: Lida com dados rotulados, em que há uma coleção de entradas e saídas resultantes, como ao tentar prever um valor de entrada obtendo um valor de perda. Alguns exemplos são: Redes Neurais Recorrentes e Redes Neurais Convolucionais (ALZUBAIDI *et al.*, 2021).

- Aprendizado semi-supervisionado: Esse processo de aprendizagem é baseado em dados semi-rotulados. Um exemplo é um classificador de documentos de textos, em que pode ser mais complicado de se obter uma grande quantidade de dados rotulados, sendo ideal o aprendizado semi-supervisionado (ALZUBAIDI *et al.*, 2021).

- Aprendizado não supervisionado: Técnica que permite a aprendizagem na ausência de dados rotulados. Aprendendo os recursos significativos não identificados nos dados de entrada. Alguns exemplos são: Agrupamentos e redução da dimensionalidade (ALZUBAIDI *et al.*, 2021).

2.2.2 Redes Neurais Convolucionais

Rede Neural Convolucional é um algoritmo de *Deep Learning* (ALZUBAIDI *et al.*, 2021) que pode ser utilizado para um conjunto de tarefas relacionadas à visão computacional, como detecção e classificação de imagens (SENGUPTA *et al.*, 2020). Um dos principais benefícios relacionados à visão computacional é o compartilhamento de peso, reduzindo o número de parâmetros treináveis, ajudando a rede a melhorar a generalização e evitar *overfitting* (ALZUBAIDI *et al.*, 2021).

Fazendo parte de uma classe de Redes Neurais muito boas para o processamento de imagens, por possuírem propriedade espacial e características como bordas, contornos, cor, entre outros, sendo então essas redes úteis para o tratamento. E em um nível de aprendizado profundo, as representações complexas são dividas em um conjunto de representações mais simples (SENGUPTA *et al.*, 2020).

2.2.3 Redes Neurais Recorrentes

Redes Neurais Recorrentes são classes de modelos de aprendizado de máquina supervisionado com um ou mais *loops de feedback*. Esses *loops* são ciclos recorrentes ao longo do tempo ou sequência. Essas redes têm uma pilha de unidades não lineares

em que pelo menos uma conexão entre as unidade forma um ciclo (SALEHINEJAD *et al.*, 2017).

Utilizam dados sequenciais, já que a estrutura embutida na sequência de dados fornece informações importantes. Pode ser utilizada quando é importante entender o contexto da frase para determinar o significado de uma palavra contida nela (ALZUBAIDI *et al.*, 2021).

2.3 Modelos *sequence-to-sequence*

A tradução automática é uma tecnologia utilizada para traduzir a linguagem humana, como ao ver um filme ou acessar sites de tradução online. Em um sistema de tradução automática o idioma de origem é chamado de idioma de entrada e o idioma de destino é chamado de idioma de saída (NEUBIG, 2017).

Modelos *sequence-to-sequence*, referem-se à classe de modelos que mapeiam uma sequência para outra. O objetivo é converter uma sequência de palavras na origem para uma sequência de palavras no destino (NEUBIG, 2017). Podendo ser treinado para gerar rótulos de relevância como "palavras-alvo" (NOGUEIRA *et al.*, 2020)

2.4 Transformadores

VASWANI *et al.* (2017) Propõe uma arquitetura de modelo (*Transformer*) que depende de um mecanismo de atenção para representações globais de entrada e saída, sem utilizar Redes recorrentes ou Convoluções. A auto-atenção é um mecanismo de atenção que relaciona diferentes posições de uma sequência para representar a sequência. Tal mecanismo vem sendo utilizado para tarefas como: compreensão da leitura e representações de sentenças independentes da tarefa de aprendizagem. (VASWANI *et al.*, 2017)

Muitos modelos de transdução de sequência neural tem a estrutura codificador-decodificador e em (VASWANI *et al.*, 2017) o codificador mapeia uma sequência de entrada $(x_1, ..., x_n)$ para uma sequência de representações contínuas z = $(z_1, ..., z_n)$ (VASWANI *et al.*, 2017), como pode ser visto na figura 2.6.

Dado z, o decodificador gera uma sequência de saída $(y_1, ..., y_m)$, sendo um elemento por vez. A cada passo o modelo consome os símbolos gerados anteriormente como entrada adicional ao gerar o próximo. O Transformador utiliza essa arquitetura usando camadas de auto-atenção empilhadas e conectadas por pontos para o codificador e o decodificador (VASWANI *et al.*, 2017), como pode ser visto na figura 2.7.

O codificador é composto por uma pilha de N = 6 camadas, sendo cada camada composta de duas subcamadas. A primeira é o mecanismo de auto-atenção multi-

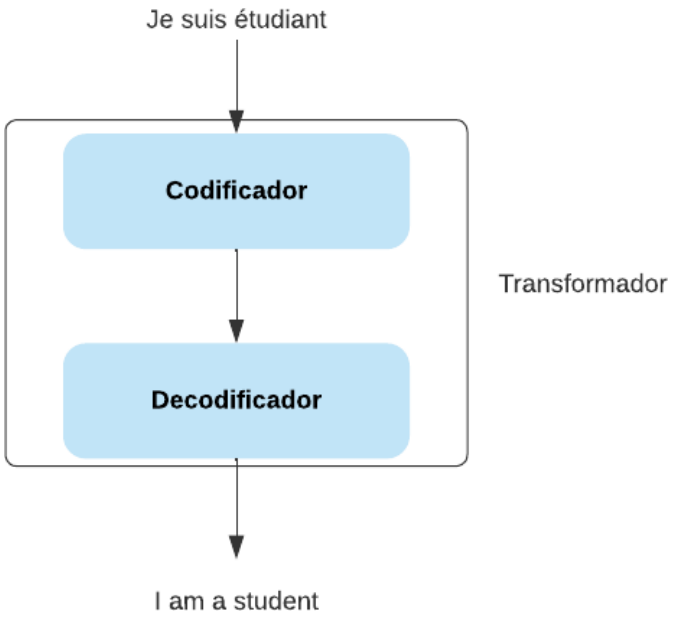


Figura 2.6: Transformador, adaptado de (ALAMMAR, 2018b)

cabeças e a segunda é uma rede *feed-forward* simples e totalmente conectada em termos de posição. A saída de cada subcamada é normalizada como $LayerNorm(x + Sublayer(x))$, em que $Sublayer(x)$ é implementada pela própria subcamada. Por facilitação, as camadas de incorporação e as subcamadas do modelo produzem saída de dimensão = 512 (d_{model}) (VASWANI *et al.*, 2017).

Da mesma forma que o codificador, o decodificador também é composto por uma pilha de N = 6 camadas idênticas. Inserindo uma terceira subcamada além das duas subcamadas em cada camada do codificador. A terceira subcamada realiza a atenção de várias cabeças sobre a saída da pilha do codificador (VASWANI *et al.*, 2017), como pode ser visto na figura 2.8.

O codificador processa a sequência de entrada, e a saída do codificador superior é transformada em um conjunto de vetores de atenção K e V, que serão utilizados por cada decodificador em sua camada de "atenção do codificador-decodificador", que auxilia o decodificador a focar em locais apropriados na sequência de entrada (ALAMMAR, 2018b).

Essa saída é calculada como uma soma ponderada dos valores, tendo o peso relativo a cada valor obtido por uma função de compatibilidade da consulta com a chave correspondente. O mapeamento de uma consulta e um conjunto de pares chave-valor para uma saída, ambos vetores, é chamado de atenção (VASWANI *et al.*, 2017).

A entrada consiste em consultas e chaves de dimensão d_k e valores de dimensão d_v. A função de atenção no conjunto de consultas é calculada simultaneamente em uma matriz Q, assim como as chaves e os valores em matrizes K e V , sendo a matriz de saída calculada a seguir, pela equação 2.2 (VASWANI *et al.*, 2017):

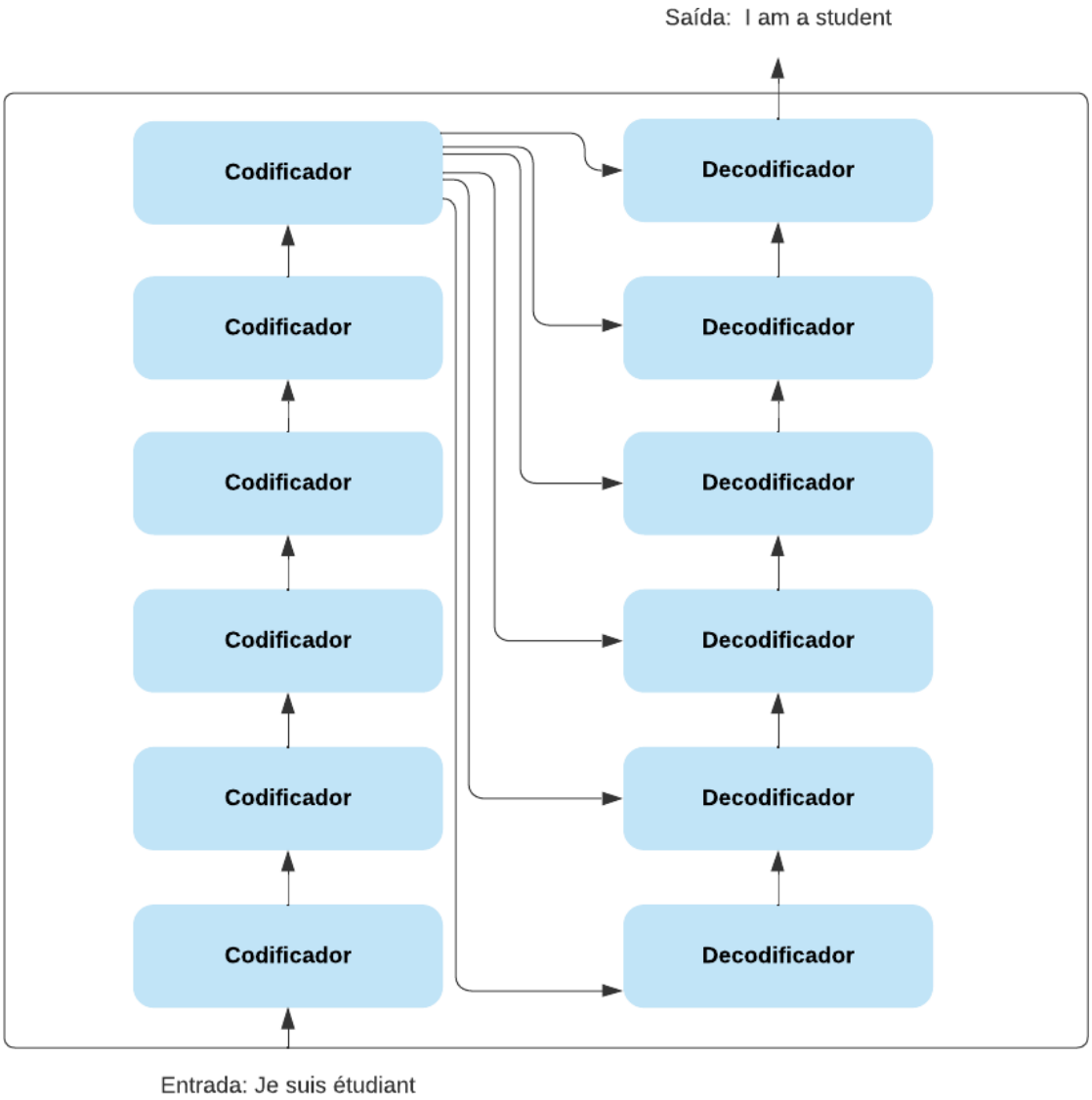


Figura 2.7: Codificador e decodificador, adaptado de (ALAMMAR, 2018b)

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{2.2}$$

Em que Q, K, V são matrizes vetoriais e d_k é a dimensão do vetor de entrada, já $T_1, T_2, ...T_N$ é a representação semântica de cada palavra (XIE *et al.*, 2021).

Já na atenção de várias cabeças, a ideia é projetar linearmente as consultas, chaves e valores h vezes com diferentes projeções aprendidas. Para cada uma dessas versões projetadas, a função de atenção é executada em paralelo, produzindo os valores de saída, que serão concatenados e projetados novamente, resultando nos valores finais (VASWANI *et al.*, 2017).

Os *embeddings* aprendidos são utilizados para converter os *tokens* de entrada em vetores de dimensão d_{model}. Também é utilizada a transformação linear aprendida e função *softmax* para converter a saída do decodificador em probabilidades previstas do próximo *token* (VASWANI *et al.*, 2017).

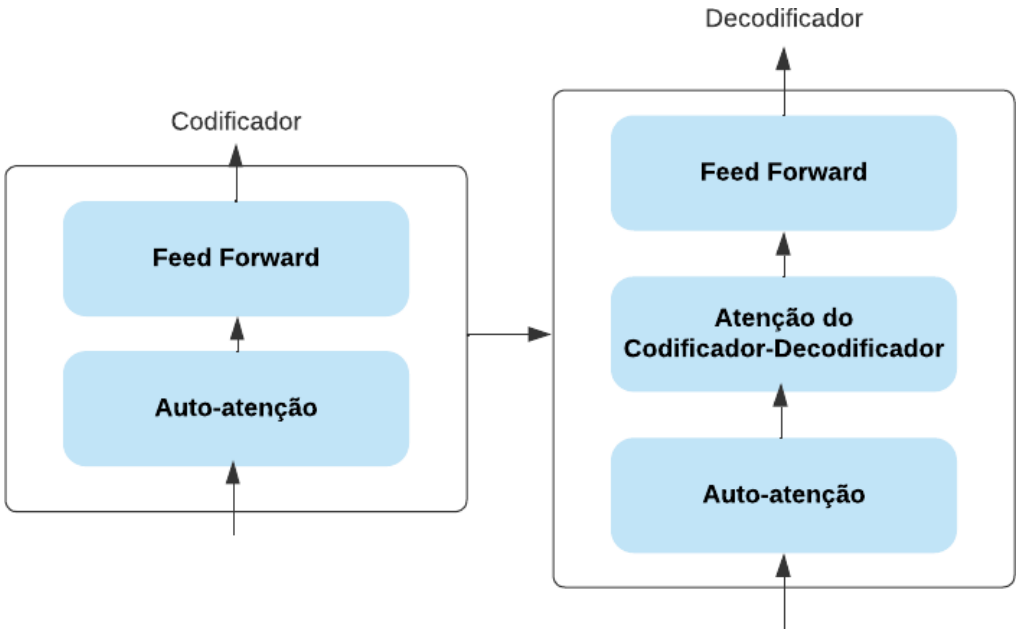


Figura 2.8: Codificador e decodificador, adaptado de (ALAMMAR, 2018b)

Para que o modelo faça uso da ordem da sequência, algumas informações sobre a posição são injetadas, sendo adicionadas "codificações posicionais" aos *embeddings* de entrada, tendo dimensão d_{model} como os *embeddings* para que possam ser somados (VASWANI *et al.*, 2017). Da mesma forma que outros modelos de tradução de sequência, são utilizados *embeddings* aprendidos para converter os *tokens* de entrada e *tokens* de saída para vetores de dimensão d_{model}. (VASWANI *et al.*, 2017).

Os vetores ocultos de origem influenciam a distribuição por meio de uma camada de concentração de atenção que pondera cada palavra de origem em relação à sua contribuição esperada para a previsão do destino (alvo). Então, o modelo é treinado para minimizar a probabilidade de *log* negativo (*negative loglikelihood*) do corpus de treinamento (KLEIN *et al.*, 2017).

2.5 Modelos para o *ranking* consulta-documento

2.5.1 BM25

Em um modelo vetorial, cada documento é representado como um vetor e cada elemento geralmente é correspondente à uma palavra. Considerando um vocabulário V de termos que aparecem em cada documento d no *corpus* (Coleção), um objeto do texto, como sentença e documento é representada por um vetor de números reais(v), sendo cada termo $w \in V$ (GHAWI e PFEFFER, 2019).

Os valores no vetor podem ser binários, sendo 1 para presença de palavra e 0 para ausência. Também é comum usar a frequência do termo dentro do documento (TF). Para medida de semelhança existem algumas medidas como a medida do cosseno e

o produto escalar. Para o produto escalar, a similidade é dada pela equação abaixo, sendo q a consulta e o documento d (GHAWI e PFEFFER, 2019).

$$sim(q,d) = v_q.v_d = \sum_{w \in q \cap d} c(w,q)c(w,d) \tag{2.3}$$

A magnitude de um componente está relacionada à importância de w no documento. O TF normalmente é combinado com a frequência inversa do documento (IDF), dado por (GHAWI e PFEFFER, 2019):

$$idf(w) = log\frac{M - df(w) + 0.5}{df(w) + 0.5} \tag{2.4}$$

M é o número de documentos na coleção e $df(w)$ é a frequência (contagem de documentos contendo w) (GHAWI e PFEFFER, 2019).

Outro método popular de ponderação de termos é o BM25, sendo utilizado para atribuir pesos aos termos ao representar um documento como um vetor de termos ponderados. Esse método estende o TF.IDF(*Term Frequency – Inverse Document Frequency*) em dois aspectos, com a transformação de frequência de termo e normalização do comprimento do documento (GHAWI e PFEFFER, 2019).

A transformação de frequência de termo é utilizada para controlar a influência de pesos altos, sendo dado por (GHAWI e PFEFFER, 2019):

$$tf = \frac{(k+1).c(w,d)}{k + c(w,d)} \tag{2.5}$$

O TF depende do número de *tokens* em um documento e deve ser normalizado pelo mesmo termo ocorrer em documentos de maior comprimento, de foma repetida. E também, um documento longo tem um longo tamanho de vocabulário (GHAWI e PFEFFER, 2019).

Um dos métodos de normalização do comprimento do documento é abordagem de normalização pivotada, sendo dada pela equação a seguir, em que $|d|$ é o comprimento do documento d (em palavras), $avgdl$ é o comprimento médio do documento sobre a coleção e $b \in [0,1]$ é um parâmetro de controle livre (GHAWI e PFEFFER, 2019).

$$1 - b + b\frac{|d|}{avgdl} \tag{2.6}$$

Modelos probabilísticos de Recuperação de Informação avaliam a probabilidade de relevância de um documento para uma consulta. Vários modelos probabilísticos foram propostos, tendo o BM25 uma função de similaridade com bons resultados (BENNETT *et al.*, 2008).

O BM25 utiliza a seguinte função de similaridade combinando os pontos anteri-

ores com os seguintes pontos e sendo representado pela equação a seguir (GHAWI e PFEFFER, 2019):

- $c(w, d)$ e $c(w, q)$ são as frequências do termo w na consulta q e no documento d.
- $avgdl$ é o comprimento médio do documento.
- $idf(w)$ é a frequência inversa no documento do termo w.
- k controla a taxa de saturação da frequência do termo e b controla a normalização do comprimento do documento.

$$sim(q, d) = \sum_{w} c(w, q) \frac{(k+1)c(w, d)}{c(w, d) + k(1 - b + b\frac{|d|}{avgdl})} idf(w) \tag{2.7}$$

2.5.2 Jelinek-Mercer *smoothing* e Dirichlet *smoothing*

Métodos de suavização foram propostos em contextos de tarefas como o reconhecimento de fala, e no geral descontam as probabilidades das palavras vistas no texto e atribuem a massa de probabilidade extra às palavras não vistas de acordo com algum modelo de "retorno" (GHAWI e PFEFFER, 2019).

A suavização(*smoothing*) Jelinek-Mercer combina a frequência relativa de um termo da consulta em um documento D com a frequência relativa do termo na coleção C completa. Sendo calculada pela equação abaixo, em que $\lambda = 0.1$ é utilizado para consultas pequenas, como títulos, e valores maiores como $\lambda = 0.7$ para consultas com um maior número de termos (BENNETT *et al.*, 2008).

$$P(t|M_D) = (1 - \lambda)\frac{f_{d,t}}{|D|} + \lambda P(t|M_c) \tag{2.8}$$

Para se obter uma lista inicial (documentos pseudo-relevantes para uma consulta) e para gerar uma lista ordenada de documentos para cada consulta após a expansão, ROY *et al.* (2019) utiliza *Dirichlet smoothing* (ZHAI e LAFFERTY, 2004). A pontuação é obtida por (ROY *et al.*, 2019):

$$Score(d, Q) = \sum_{q \in Q} NFW_{rm3_1^+}(q) \times log\frac{\mu.P(q|C) + |d|.P(q|d)}{P(q|C)(\mu + |d|)} \tag{2.9}$$

Sendo, NFW (q) a representação do peso normalizado de q ($q \in Q$) na consulta expandida, μ é o parâmetro de suavização e | d | representa o número de palavras indexadas em d (ROY *et al.*, 2019).

2.6 Expansão de Consultas

A Expansão de Consultas é um método utilizado para superar de forma automática a limitação em consultas, em que muitas vezes há imprecisão nas palavras-chave de um usuário. Tal método aumenta a consulta original utilizando novos termos de significado semelhante (CARPINETO e ROMANO, 2012).

Considerando que sistemas de Busca e Recuperação de Informação costumam conter um campo de busca (caixa de entrada) que o usuário preenche com palavras-chave, tais palavras são comparadas com índices em uma coleção, para encontrar documentos referentes a essas palavras. Dessa forma, quanto mais precisas forem as palavras-chave, maior a chance do sistema retornar boas correspondências. Porém, como a linguagem natural é ambígua, podem ocorrer erros (CARPINETO e ROMANO, 2012).

O comum em uma consulta de um usuário é que seja uma lista curta de termos, já que o usuário não é um especialista nos termos que deseja consultar. Uma abordagem adotada pelos mecanismos de RI é refinar e expandir a consulta original baseado nos documentos retornados por ela (GREENGRASS, 2000).

Existem várias abordagens para expansão automática de consultas, dentre elas, uma comum é expandir as consultas utilizando os documentos mais relevantes, extraindo os termos desses documentos. Aplicando à coleção termos de documentos relevantes para um usuário, método que é chamado de *feedback* de relevância (GREENGRASS, 2000).

Os pesos dos termos que aparecem nos documentos relevantes podem ser aumentados e os que não aparecem podem ser reduzidos, reponderando os termos na consulta. Algumas questões são importantes quando termos de documentos são adicionados à consulta, como: quantos documentos o usuário deve julgar? Quando esse limite for definido, ele se torna o conjunto de recuperação, em que o usuário julgará cada documento nesse conjunto como relevante ou não (GREENGRASS, 2000).

Um dos problemas enfrentados por indexadores é o das palavras fora do vocabulário (FURNAS *et al.*, 1987), que pode acontecer quando a mesma palavra tem significados diferentes (sinonímia). Nesse caso, juntamente com o uso de inflexões e plurais, pode resultar na diminuição da capacidade de recuperar todos os documentos relevantes (*recall*) (CARPINETO e ROMANO, 2012).

Já a homonímia, resulta na recuperação de documentos errados ou sem relevância, diminuindo então, a capacidade de recuperar apenas documentos relevantes (*precision*) (CARPINETO e ROMANO, 2012).

Originalmente sugerida por (MARON e KUHNS, 1960), a Expansão de Consultas foi posteriormente explorada e aprimorada, e outras abordagens foram propostas, como o refinamento da consulta e desambiguação de sentido da palavra. Podendo a

Expansão da Consulta conseguir capturar a intenção do usuário, aumentando assim, a probabilidade de recuperar documentos relevantes (CARPINETO e ROMANO, 2012).

A entrada para um mecanismo de Expansão de Consultas consiste na consulta original e uma fonte de dados para o cálculo e ponderação dos termos de expansão. A fonte de dados pode ser a própria coleção utilizando uma função para ponderação dos termos. Já a saída é formulada pela consulta com o conjunto de termos expandidos associados com seus respectivos pesos. Dessa forma, é possível calcular a similaridade entre a consulta expandida e os documentos (CARPINETO e ROMANO, 2012).

Como vantagem, espera-se que a Expansão de Consultas aumente as chances de recuperar documentos relevantes. Por exemplo, se a consulta 'Al-Qaeda' for expandida para 'Al-Qaeda al-Qaeda al-Qa'ida "Osama bin Laden" "organização terrorista sunita" "11 de setembro de 2001"', é possível que documentos que contenham não só os termos originais da consulta, mas também documentos que não tenham feito referência direta ao termo original sejam recuperados (CARPINETO e ROMANO, 2012).

Contudo, adicionar termos na consulta pode causar uma expansão inadequada, que então, prejudicaria a precisão. Um caso ocorre quando um termo de expansão tem relação à apenas um termo da consulta e não com toda a consulta ou quando o conjunto de termos de expansão não seja relevante como um todo à consulta original (CARPINETO e ROMANO, 2012).

Nos métodos de Expansão de Consulta baseados em *feedback* de pseudo-relevância, os documentos com melhor classificação recuperados a partir da consulta original são utilizados para selecionar os termos para expandir a consulta, não tendo então, a necessidade de utilização de fontes externas para expansão (ROY *et al.*, 2019).

Um método utilizado entre os baseados em *feedback* de pseudo-relevância é o RM3 (ABDUL-JALEEL *et al.*, 2004), que apesar de apresentar uma boa performance quando comparado a alguns outros métodos de Expansão de Consultas, pode acabar selecionando palavras genéricas(ruídos), que então, não ajudam a identificar documentos relevantes. Esse problema ocorre devido a presença de documentos não relevantes no conjunto de documentos pseudo-relevantes (ROY *et al.*, 2019).

ROY *et al.* (2019) propõe alguns métodos baseados em RM3, a fim de obter melhorias. No primeiro método $RM3^{+}_{1}$, o peso de *feedback* de um termo de expansão (w) é calculado por:

$$FW(w) \approx P(w|R) * r(w) \tag{2.10}$$

Sendo R o conjunto de documentos pseudo-relevantes e $r(w) \approx 1/P(w|C)$ que pode ser interpretado como uma medida de raridade de w, em que C é conjunto completo de documentos (ROY *et al.*, 2019).

Então, r(w) pode ser substituído por outra medida de raridade de um termo, nesse caso, é substituído pela frequência inversa do documento (IDF), já que produz uma melhor eficácia de recuperação (ROY *et al.*, 2019).

Os pesos são normalizados pela soma e os N termos com FW(w) mais altos são selecionados como termos de expansão. Esses pesos de *feedback* normalizados(NFW) são combinados com o modelo de probabilidade da consulta Q (ROY *et al.*, 2019).

Considerando a equação representada por RM3 (ABDUL-JALEEL *et al.*, 2004):

$$FW_{rm3_1^+}(w) = \alpha P(w|Q) + (1-\alpha)P(w|R) \tag{2.11}$$

E a alteração proposta, o peso final de um termo w na consulta expandida é dado pela seguinte equação (ROY *et al.*, 2019):

$$FW_{rm3_1^+}(w) = \alpha P(w|Q) + (1-\alpha) * NFW(w) \tag{2.12}$$

Sendo FW (*feedback weight*) o peso de *feedback* de um termo de expansão(w), $\alpha \in [0, 1]$ o parâmetro de interpolação e Q o conjunto de consultas (ROY *et al.*, 2019).

Já no método $RM3_2^+$ duas funções são utilizadas, sendo a primeira para selecionar os termos de expansão, ordenando os termos de expansão candidatos e selecionando N termos no topo da lista, com a seguinte equação (ROY *et al.*, 2019):

$$FW_{rm3_2^+}(w) \approx \alpha P(w|Q) * r(w) + (1-\alpha)P(w|R) * r(w) \tag{2.13}$$

Nesta equação, a informação de raridade é incorporada duas vezes (r(w)), e essa ênfase pode promover termos ruidosos(altamente raros) ao custo de se obter termos mais úteis. Porém, o objetivo é evitar o problema de selecionar palavras com IDF baixo como termos de expansão (ROY *et al.*, 2019).

Já a terceira função para ponderação dos termos proposta é a $RM3_3^+$, sendo representada a seguir pela equação 2.14, evitando o uso de um fator IDF "duplo" durante a recuperação final, obtendo geralmente então, uma melhor representação (ROY *et al.*, 2019):

$$FW_{rm3_3^+}(w) = \alpha P(w|Q) + (1-\alpha)P(w|R) \tag{2.14}$$

2.7 Expansão de Documentos

O problema da incompatibilidade de vocabulário é um dos desafios na Recuperação de Informação, sendo muitas vezes abordado por técnicas de Expansão de Consultas, fornecendo termos úteis para aumentar a consulta original (NOGUEIRA *et al.*, 2019).

A Expansão de Consultas mantém a representação do documento estática e enriquece a representação da consulta. Então NOGUEIRA *et al.* (2019) propõe uma abordagem baseada no enriquecimento da representação do documento, antes da indexação (NOGUEIRA *et al.*, 2019).

Um modelo *sequence-to-sequence* é treinado, para que dado um documento, possíveis perguntas que o documento pode responder sejam geradas. Esses modelos são obtidos com kit de ferramentas de código aberto existentes (NOGUEIRA *et al.*, 2019).

Técnicas de Expansão de Documentos costumam ser menos populares entres pesquisadores, devido ao tempo de experimentação, já que o *corpus* precisa ser reindexado quando a técnica de expansão for alterada (NOGUEIRA *et al.*, 2019).

O método proposto é chamado de Doc2query, em que o objetivo é prever para cada documento um conjunto de consultas relevantes. É utilizado um modelo transformador sequência para sequência em um *dataset* que tem como entrada os termos dos documentos, produzindo consultas (NOGUEIRA *et al.*, 2019).

Os documentos e as consultas alvo são tokenizados usando *Moses tokenizer* [1] e segmentados usando BPE (*Byte Pair Encoding*). Este último, é um algoritmo para segmentação de palavras que foi utilizado por (SENNRICH *et al.*, 2015) para mesclar caracteres ou sequências de caracteres.

Dessa forma, o tamanho final do vocabulário de símbolos é igual ao tamanho inicial mais o número de operações de mesclagem, considerando que os caracteres são mesclados em caracteres maiores e conhecidos, ou seja, as novas sequências de caracteres gerados precisam ser de palavras reais. Sendo aplicável a qualquer palavra e permitindo redes de vocabulário aberto com vocabulários de símbolos fixos (SENNRICH *et al.*, 2015).

Para prever n consultas, é utilizada uma amostragem aleatória (FAN *et al.*, 2018), concatenando-as a cada documento do *corpus* sem nenhuma marcação especial. Os documentos expandidos são indexados e uma lista classificada destes documentos para cada consulta é recuperada utilizando o BM25. Sendo então, reclassificada utilizando o BERT. A figura 2.9 mostra a representação da Expansão de Documentos (NOGUEIRA *et al.*, 2019).

Para o método Doc2query, NOGUEIRA *et al.* (2019) usa como implementação a

[1] http://www.statmt.org/moses/

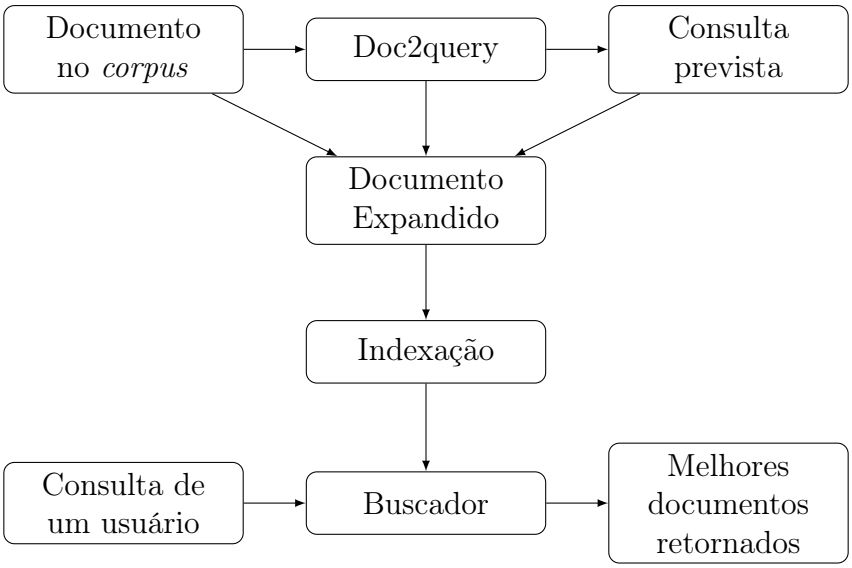


Figura 2.9: Etapas da Expansão de Documentos, adaptado de (NOGUEIRA *et al.*, 2019)

biblioteca OpenNMT por ser completa para treinar e implantar modelos de tradução automática neural. Sendo sucessora da seq2seq-attn desenvolvida em Harvard, e então reescrita para melhorias como: eficiência, legibilidade e generalização (KLEIN *et al.*, 2017).

OpenNMT é um kit de ferramentas de código aberto para tradução automática neural(Neural Machine Translation - NMT) que possui uma visão de modelagem de linguagem condicional de tradução, modelando a probabilidade de uma sentença alvo $w_{1:T}$ dada uma sentença fonte $x_{1:S}$ como (KLEIN *et al.*, 2017):

$$p(w_{1:T|x}) = \prod_{1}^{T} p(w_t|w_{1:t-1}, x; \theta) \tag{2.15}$$

Sendo θ os parâmetros do modelo.

Esta distribuição é estimada usando uma arquitetura de codificador-decodificador baseada em atenção (*attention-based encoder-decoder*), ou seja, sem recorrência ou convoluções (KLEIN *et al.*, 2017).

O exemplo a seguir presente do *dataset* MS MARCO apresenta um documento do *corpus*, uma consulta prevista e a consulta de um usuário (NOGUEIRA *et al.*, 2019):

Input Document: The Delaware River flows through Philadelphia into the Delaware Bay. It flows through and aqueduct in the Roundout Reservoir and then flows through Philadelphia and New Jersey before emptying into the Delaware Bay.

Predicted Query: what river flows through delaware

Target Query: where does the delaware river start and end

NOGUEIRA *et al.* (2019) utiliza o *dataset* MS MARCO[2], que é um conjunto de dados de reclassificação de passagens com 8,8 milhões de passagens. O conjunto de treinamento contém aproximadamente 500 mil pares de consulta e documentos relevantes, tendo em média cada consulta uma passagem relevante. Os conjuntos de desenvolvimento e teste contêm aproximadamente 6.900 consultas cada, já os rótulos de relevância são públicos apenas para o conjunto de desenvolvimento NOGUEIRA *et al.* (2019).

2.8 BERT

O BERT (Bidirectional Encoder Representations from Transformers), ao contrário de modelos de representação de linguagem, é projetado para pré-treinar representações bidirecionais profundas de texto não rotulado, ou seja, é capaz de aprender uma palavra com base no contexto a direita e a esquerda dessa palavra. Como resultado, o modelo BERT pré-treinado pode ser ajustado com apenas uma camada de saída adicional para criar modelos para uma ampla gama de tarefas (DEVLIN *et al.*, 2018).

Modelos que utilizam *embeddings* como o Word2vec, mostraram a utilização de um vetor para representar palavras, capturando relações semânticas ou de significado, como dizer se palavras são opostas ou semelhantes ou capturar relações sintáticas ou gramaticais (DEVLIN *et al.*, 2018).

Porém nesses modelos, não era possível diferenciar palavras iguais em contextos diferentes, como no exemplo [3] a seguir, em que a palavra *apple* é representada nos dois modelos.

Sentença 1: *'apple released iphone 12 pro max in 2020'.*

-0.5985	-0.4632	...	0.3490

Tabela 2.1: Vetor da palavra *apple* na primeira sentença com Word2vec

-0.5985	-0.4632	...	0.2832

Tabela 2.2: Vetor da palavra *apple* na primeira sentença com o BERT

Sentença 2: *'an apple a day keeps the doctor away'.*

Se considerarmos o modelo de representação com Word2vec, a palavra *apple* nas duas sentenças seria representada pelo mesmo vetor, e por consequência a distância

[2] https://microsoft.github.io/msmarco/

[3] `https://github.com/bhattbhavesh91/word2vec-vs-bert/blob/main/Word2Vec_vs_BERT.ipynb`

-0.5985	-0.4632	...	0.3490

Tabela 2.3: Vetor da palavra *apple* na segunda sentença com Word2vec

-0.5985	-0.4632	...	0.2707

Tabela 2.4: Vetor da palavra *apple* na segunda sentença com o BERT

entre os vetores da palavra em cada frase vai ser zero. Já na representação com o BERT, a mesma palavra teria representações diferentes, ou seja, em cada frase a palavra *apple* terá um vetor diferente, e assim, a distância entre esses vetores vai ser diferente de zero, ou seja, obtém uma melhor representação para uma mesma palavra em contextos diferentes.

Duas estratégias para aplicar representações de linguagem pré-treinadas a tarefas posteriores são: A baseada em recursos, que utiliza arquiteturas específicas de tarefas que inclua as representações pré-treinadas como recursos adicionais. E a abordagem de ajuste fino, que introduz parâmetros específicos sendo treinada ajustando todos os parâmetros pré treinados (DEVLIN *et al.*, 2018).

Ambas compartilham a mesma função objetivo, em que usam modelos de linguagem unidirecionais para aprender representações gerais de linguagem. Um dos problemas desses modelos está em, por exemplo, quando se é utilizada uma arquitetura da esquerda para direita, em que cada *token* pode atender somente aos *tokens* anteriores nas camadas de auto atenção do transformador, não sendo ideal para tarefas de nível de frase, como responder perguntas, em que é crucial incorporar contexto em ambas as direções (DEVLIN *et al.*, 2018).

Um modelo tradicional de geração de vetores de palavras não é capaz de representar palavras polissêmicas de maneira distinta em um texto, então, o método BERT é apresentado para representar o vetor de características semânticas do texto, em seguida, o vetor de recursos pode ser inserido em um classificador *Softmax* para resolver um problema de classificação (XIE *et al.*, 2021).

O BERT possui modelos pré-treinados, conhecidos como BERT *base* e BERT *large*, ambos com um grande número de camadas de codificador(também conhecido como blocos transformadores), redes *feedforward* e cabeças de atenção. O modelo *base* possui 12 camadas de codificador, redes *feedforward* com 768 unidades ocultas e 12 cabeças de atenção. Já o modelo *large* possui 24 camadas de codificador, redes *feedforward* com 1024 unidades ocultas e 16 cabeças de atenção (ALAMMAR, 2018a).

O módulo central da camada de codificação do transformador do BERT é o mecanismo de auto-atenção, que modela frases de texto inteiramente usando tal mecanismo. A ideia central é calcular a relação entre cada palavra nas frases, usando a relação entre os pesos para conseguir ajustar cada palavra, afim de obter uma nova

expressão, que não conterá apenas a relação entre outras palavras, mas também a própria palavra (XIE *et al.*, 2021).

O modelo *Softmax* pode ser usado para construir o classificador, sendo a extensão do modelo de regressão logística na classificação multivariada. A fórmula de cálculo para um modelo de classificação de sentimentos é dada da seguinte forma (XIE *et al.*, 2021):

$$P(y = j|C, W_0, b_0) = softmax(W_0 * C + b_0) \tag{2.16}$$

Em que, C é vetor de representação semântica de toda a frase que pode ser utilizado como entrada do classificador. W_0 é a matriz de coeficiente de peso, b_0 é a matriz de viés e cada entrada x tem uma probabilidade $P(y = j|x)$ para cada categoria $j = 1, 2, 3, ..., k$, sendo a categoria de maior probabilidade a que x pertence (XIE *et al.*, 2021).

[CLS]	everybody	dance	now	[SEP]

Tabela 2.5: Frase de exemplo, com os *tokens* especiais [CLS] e [SEP], adaptado de (ALAMMAR, 2018a)

	CLS	everybody	dance	now	[SEP]
0	0.115728	-0.150873	0.076831	-0.661984	0.834172
1	0.223339	0.269943	0.009255	-0.858786	0.279408
...	...	...	...	...	...
768	0.454554	0.539476	0.672731	0.648410	-0.864230

Tabela 2.6: Representação da frase anterior com 768 linhas x 5 colunas, adaptado de (ALAMMAR, 2018a)

	CLS
0	0.115728
1	0.223339
...	...
768	0.454554

Tabela 2.7: Representação completa da frase apenas com o vetor [CLS] (ALAMMAR, 2018a)

Na exemplo acima podemos observar a frase iniciada pelo *token* especial [CLS], que a seguir veremos que representará classificação e finalizada pelo *token* especial [SEP] marcando o final da frase, que também poderia ser o separador entre frases. Na tabela 2.6 há a representação de cada *token* como um vetor de 768 posições (BERT *base*) e então a representação completa da frase na tabela 2.7.

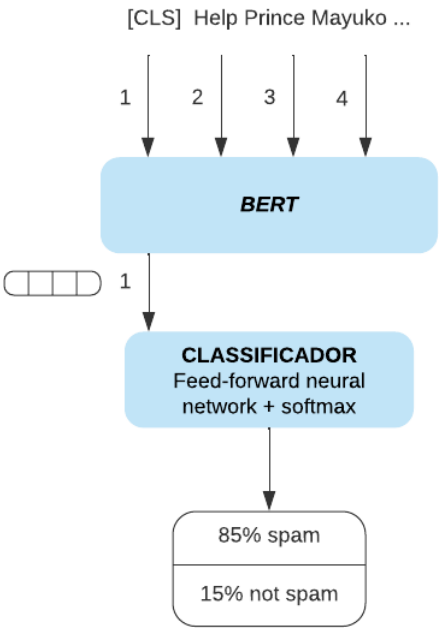


Figura 2.10: Classificação de sentença com o BERT, adaptado de (ALAMMAR, 2018a)

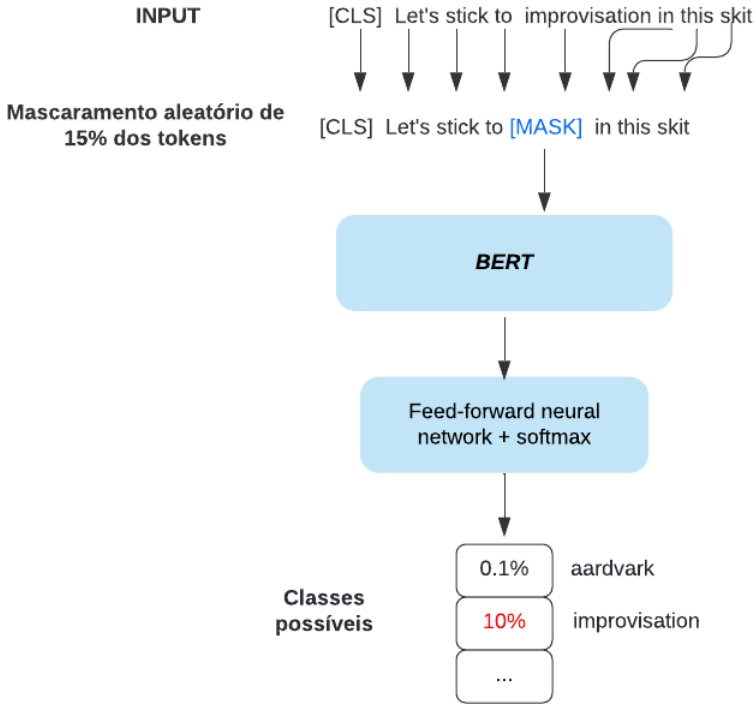


Figura 2.11: Mascaramento de palavras, adaptado de (ALAMMAR, 2018a)

Na figura 2.10 podemos observar que o primeiro *token* de entrada é fornecido por um *token* especial [CLS], que aqui significa classificação. Iremos nos concentrar apenas nessa saída da primeira posição, que produz um vetor de tamanho 768(BERT *base*). Esse vetor é utilizado como entrada para um classificador escolhido (ALAMMAR, 2018a).

Já a figura 2.11 mostra o mascaramento de palavras, em que o modelo irá prever

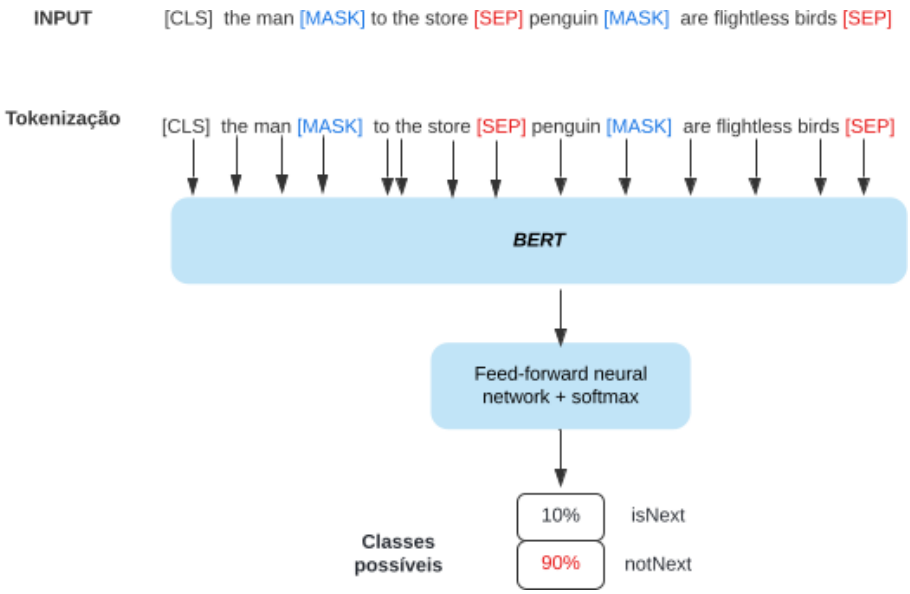


Figura 2.12: Previsão da próxima frase, adaptado de (ALAMMAR, 2018a)

a porcentagem relacionada às classes possíveis para palavra marcada com o *token* especial [MASK]. Na figura 2.12 a frase de entrada é tokenizada, passando pelo modelo e tendo como saída a porcentagem dessa frase ser a próxima frase dentro de uma coleção (ALAMMAR, 2018a).

O BERT utiliza uma sequência de palavras como entrada e em cada camada aplica auto-atenção, passando seus resultados por uma rede *feedforward* e, em seguida, passa para o próximo codificador (ALAMMAR, 2018a).

Modelos neurais pré-treinados em uma tarefa de modelagem de linguagem como o BERT, alcançaram ótimos resultados em linguagens naturais e tarefas de processamento, como responder perguntas e inferência de linguagem natural (NOGUEIRA e CHO, 2019).

Um reimplementação do BERT é proposta para re-classificação de passagem baseada em consulta, através de um *pipeline* de respostas à perguntas em três estágios. No primeiro estágio, um grande número de documentos possivelmente relevantes para uma questão são recuperados de um *corpus* por um mecanismo padrão, como o BM25 (NOGUEIRA e CHO, 2019).

No segundo (reclassificação da passagem), estes documentos serão pontuados e reclassificados por um método mais intensivo em computação. E no terceiro, os n primeiros documentos serão a fonte para as respostas por um módulo de geração de respostas (NOGUEIRA e CHO, 2019).

O objetivo do re-classificador (BERT) é estimar uma pontuação s_i de quão relevante uma passagem candidata d_i é para uma consulta q. A consulta é considerada como sentença A e o texto da passagem como sentença B. Então, a consulta é truncada para ter no máximo 64 *tokens* e o texto da passagem também é truncado para

que sua concatenação com a consulta não ultrapasse 512 *tokens* (NOGUEIRA e CHO, 2019).

O modelo BERT-LARGE é utilizado para classificação binária, ou seja, o vetor [CLS] é utilizado como entrada para uma Rede Neural de camada única para obter a probabilidade da passagem ser relevante. Essa probabilidade para cada passagem é calculada de forma independente, obtendo a lista final de passagens e classificando-as de acordo com essas probabilidades (NOGUEIRA e CHO, 2019).

O treinamento é iniciado a partir de um modelo BERT pré-treinado, sendo ajustado para tarefa de reclassificação utilizando entropia cruzada. Em que J_{pos} é o conjunto de índices das passagens relevantes e J_{neg} é o conjunto de índices de passagens não relevantes nos n (1.000) principais documentos recuperados com o BM25 (NOGUEIRA e CHO, 2019):

$$L = -\sum_{j \in J_{pos}} log(s_j) - \sum_{j \in J_{neg}} log(1 - s_j) \tag{2.17}$$

No exemplo (DEVLIN *et al.*, 2018) abaixo podemos observar que as palavras foram quebradas utilizando o modelo de *wordpieces*, que se trata de uma abordagem orientada a dados para gerar uma segmentação determinística para uma sequência de caracteres, em que palavras são divididas em fragmentos, dado um modelo de fragmentos de palavras treinado (WU *et al.*, 2016). Sendo que os *embeddings* de entrada são a soma dos *embeddings* de *token*, os *embeddings* de segmentação e os embeddings de posição (DEVLIN *et al.*, 2018).

- INPUT: [CLS] my dog is cute [SEP] he likes play ##ing [SEP]
- Token Embeddings: E_{CLS} E_{my} E_{dog} E_{is} E_{cute} $E_{[SEP]}$ E_{he} E_{likes} E_{play} $E_{\#\#ing}$ $E_{[SEP]}$
- Segment Emdeddings: E_A E_A E_A E_A E_A E_A E_B E_B E_B E_B E_B
- Position Emdeddings: E_0 E_1 E_2 E_3 E_4 E_5 E_6 E_7 E_8 E_9 E_{10}

Tal modelo é gerado a partir de uma abordagem orientada a dados para maximizar a probabilidade do modelo de linguagem dos dados de treinamento. Considerando um *corpus* e um número de *tokens* D, o problema de otimização consiste em selecionar D palavras para que o *corpus* resultante seja mínimo no número de *wordpieces* (WU *et al.*, 2016).

Porém o ideal é que tal número de caracteres seja gerenciável, podendo ser reduzido para 500 em idiomas ocidentais e mais para asiáticos. Também é observado que um vocabulário entre 8 mil e 32 mil *wordpieces* alcança uma boa precisão, assim como uma boa velocidade de decodificação (WU *et al.*, 2016).

2.9 TREC

A primeira conferência de recuperação de texto (TREC-1) foi realizada em novembro de 1992 co-patrocinada pela ARPA e NIST. Reunindo pesquisadores de Recuperação de Informação para discutir pontos sobre a coleção TIPSTER, que se trata de uma coleção com um grande volume de dados (HARMAN, 1995).

A iniciativa visa incentivar a Pesquisa em RI utilizando grandes coleções. Já que ao fornecer um conjunto de testes em larga escala e incentivando a interação entre grupos é gerado um novo impulso (HARMAN, 1995).

Nessa conferência, apesar da variedade de abordagens para RI em grandes coleções, contava com pontos como prazos apertados, tendo então, os resultados vistos como preliminares (HARMAN, 1995).

Já na segunda conferência (TREC-2), que ocorreu em agosto de 1993, 9 novos grupos participaram, além dos 22 da primeira conferência. Muitos desses grupos melhoraram significativamente os resultados em comparação ao TREC-1 (HARMAN, 1995).

Dois grupos de RI são examinados, com consulta "adhoc", como um pesquisador que pode usar em um ambiente de biblioteca, e uma consulta de "roteamento", como um arquivo de perfil para filtrar um fluxo de documento de entrada (HARMAN, 1995).

Algumas diretrizes foram desenvolvidas para o TREC (HARMAN, 1995):

1. O conjunto inicial são referentes aos discos 1 e 2 e aos tópicos 1-100, podendo ser modificados com base nos documentos de teste do disco 3. Esse conjunto é utilizado para costrução das estruturas de dados do sistema (HARMAN, 1995).

2. As tags SGML são utilizadas em partes da coleção de testes, como o Wall Street Journal e o Ziff, que contêm os termos de índices controlados ou não controlados atribuídos manualmente e não devem ser utilizados (HARMAN, 1995).

3. Como os documentos de teste geralmente são indexados como um conjunto, o roteamento deve ser simulado sem nenhuma informação baseada em tal conjunto, já que no mundo real um único documento seria indexado e comparado aos tópicos de roteamento (HARMAN, 1995).

Também havia diretrizes para a construção de consultas a partir dos tópicos fornecidos e três categorias foram definidas a partir da quantidade e tipo de intervenção manual (HARMAN, 1995):

1. Automático: Relacionado à construção de consulta inicial automática.

- Consultas adhoc: São informações extraídas a partir do tópico para construir a consulta e não deve haver intervenção manual afetando os resultados (HARMAN, 1995).
- Consultas de roteamento: As consultas são construídas de forma automática à partir dos tópicos, relevância e documentos de treinamento (HARMAN, 1995).

2. Manual: Relacionado à construção de consulta inicial manual.

   - Consultas adhoc: São construídas utilizando tópicos manualmente ou com assistência à máquina. E não deve haver intervenção manual após a construção inicial que afete os resultados (HARMAN, 1995).
   - Consultas de roteamento: Da mesma que na etapa manual, porém utilizando os tópicos, julgamentos e documentos de treinamento (HARMAN, 1995).

3. FEEDBACK: Relacionado à construção de consulta de forma automática ou manual com *feedback* (HARMAN, 1995).

   - Consultas adhoc: A consulta é enviada ao sistema e um humano faz os julgamentos sobre a relevância dos documentos no conjunto recuperado. As sentenças podem ir ao sistema para modificar automaticamente a consulta ou um humano pode fazer essa modificação. Ao terminar o *feedback* de relevância, a consulta é aceita como final (HARMAN, 1995).
   - Consultas de roteamento: Essas consultas não aceitam *feedback*, devido ao sistemas de roteamento não aceitarem (HARMAN, 1995).

Os *workshops* são patrocinados pelo Instituto Nacional de Padrões e Tecnologia e pelo Departamento de Defesa dos EUA e fornecem grandes coleções de testes (VOORHEES, 2001). Na coleção de testes chamada TIPSTER, há três partes distintas, sendo os documentos, consultas ou tópicos e julgamento de relevância ou respostas corretas. Os documentos são divididos conforme a tabela 2.8 (HARMAN, 1995).

A tabela 2.9, mostra a estrutura dos documentos, mostrando que embora alguns documentos sejam semelhantes em tamanho (megabytes), há uma variedade em relação ao comprimento de documentos, como nos documentos muito curtos (DOE) e muito longos (FR) (HARMAN, 1995).

A linha A é relativa ao tamanho da coleção em megabytes, a linha B ao número de registros, a linha C à mediana de termos por registro e a linha D ao número médio de termos por registro (HARMAN, 1995).

Disco 1	Disco 2	Disco 3
• WSJ – Wall Street Journal (1987, 1988, 1989) • AP – AP Newswire (1989) • ZIFF – Artigos de discos de seleção de computador (Ziff Editora Davis) • FR – Registro Federal (1989) • DOE – Resumos curtos de publicações do DOE	• WSJ – Wall Street Journal (1990, 1991, 1992) • AP – AP Newswire (1988) • ZIFF – Artigos de discos de seleção de computador (Ziff Editora Davis) • FR – Registro Federal (1988)	• SJMN – San Jose Mercury News (1991) • AP – AP Newswire (1990) • ZIFF – Artigos de discos de seleção de computador (Ziff Editora Davis) • PAT – Patentes dos EUA (1993)

Tabela 2.8: Conteúdo dos discos 1, 2 e 3, adaptado de (HARMAN, 1995)

	Subconjunto da coleção	WSJ (Disco 1 e 2) SJMN (Disco 3)	AP	ZIFF	FR (Disco 1 e 2) PAT (Disco 3)	DOE
A	Disco 1 Disco 2 Disco 3	295 255 315	266 248 248	251 188 358	258 211 251	190
B	Disco 1 Disco 2 Disco 3	98,736 74,520 90,257	84,930 79,923 78,325	75,180 56,920 161,021	26,207 20,108 6,711	226,087
C	Disco 1 Disco 2 Disco 3	182 218 279	353 346 358	181 167 119	313 315 2896	82
D	Disco 1 Disco 2 Disco 3	329 377 337	375 370 379	412 394 263	1017 1073 3543	89

Tabela 2.9: Estatísticas do Documento, adaptada de (HARMAN, 1995).

Os documentos são formatados utilizando uma estrutura semelhante ao SGML, como no exemplo a seguir. Como pode ser observado, os documentos possuem marcadores no início e fim, além do campo de id DOCNO. (HARMAN, 1995):

<DOC>
<DOCNO> WSJ880406−0090 </DOCNO>

<HL> AT&T Unveils Services to Upgrade Phone Networks Under Global Plan </HL>
<AUTHOR> Janet Guyon (WSJ Staff) </AUTHOR>
<DATELINE> NEW YORK </DATELINE>
<TEXT>
American Telephone & Telegraph Co. introduced the first of a new generation of phone services with broad implications for computer and communications equipment markets.
AT&T said it is the first national long-distance carrier to announce prices for specific services under a world-wide standardization plan to upgrade
phone networks. By announcing commercial services under the plan, which the industry calls the Integrated Services Digital Network, AT&T will influence evolving communications standards to its advantage, consultants said,
just as International Business Machines
Corp. has created de facto computer standards favoring its products.
.
.
.
</TEXT>
</DOC>

Para imitar a necessidade de um usuário real, os tópicos foram escritos por usuários que utilizam Sistemas de Informação. Como ser visto no exemplo a seguir (HARMAN, 1995):

<top>
<head> Tipster Topic Description
<num> Number: 066
<dom> Domain: Science and Technology
<title> Topic: Natural Language Processing
<desc> Description:
Document will identify a type of natural language processing technology which is being developed or
marketed in the U.S.
<smry> Summary:
Document will identify a type of natural language processing technology which is being developed or marketed in the U.S.

<narr> Narrative:
A relevant document will identify a company or institution developing or marketing a natural language processing technology, identify the technology, and identify one or more features of the companys product.
<con> Concept(s):
1. natural language processing
2. translation, language, dictionary, font
3. software applications
<fac> Factor(s):
<nat> Nationality: U.S.
</fac>
<def> Definition(s):
</top>

Capítulo 3

Método Proposto

Neste capítulo será apresentado o passo a passo detalhado da metodologia utilizada, explicando as metodologias de expansão, o *dataset* utilizado e o processo de *re-rank* com o BERT.

3.1 Passo a passo

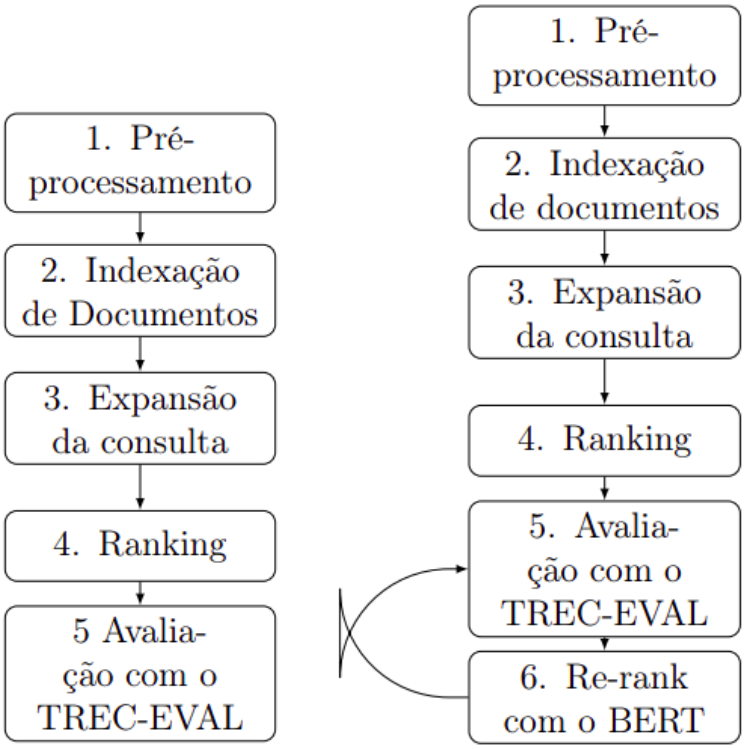
Para o treinamento do modelo Doc2query(NOGUEIRA *et al.*, 2019), foi utilizado GPU (Unidade de Processamento Gráfico), que além de ser eficiente para manipular imagens e computação gráfica, por serem capazes de realizar processamento paralelo, também tornam mais rápido o treinamento de uma rede neural em comparação às CPUs (SHAHID e MUSHTAQ, 2020).

Para este trabalho, algumas abordagens foram utilizadas para avaliação de qual combinação gera melhores resultados. A primeira abordagem é referente ao *baseline*, que é dado através da utilização de somente o *ranking* inicial gerado por um modelo sem nenhuma Expansão de Consultas ou documentos.

A figura 3.1a mostra as etapas da Expansão de Consultas, partindo da etapa 1 com o pré-processamento, a etapa 2 com a indexação de 741853 documentos dos Discos (TREC1 e TREC2), a etapa 3 com a utilização das funções de expansão $RM3_1^+$, $RM3_2^+$ ou $RM3_3^+$. Na etapa 4 um modelo será escolhido para o *ranking*, sendo o BM25, Dirichlet *smoothing* ou Jelinek-Mercer *smoothing* e na etapa 5 o resultado obtido na etapa 4 será avaliado.

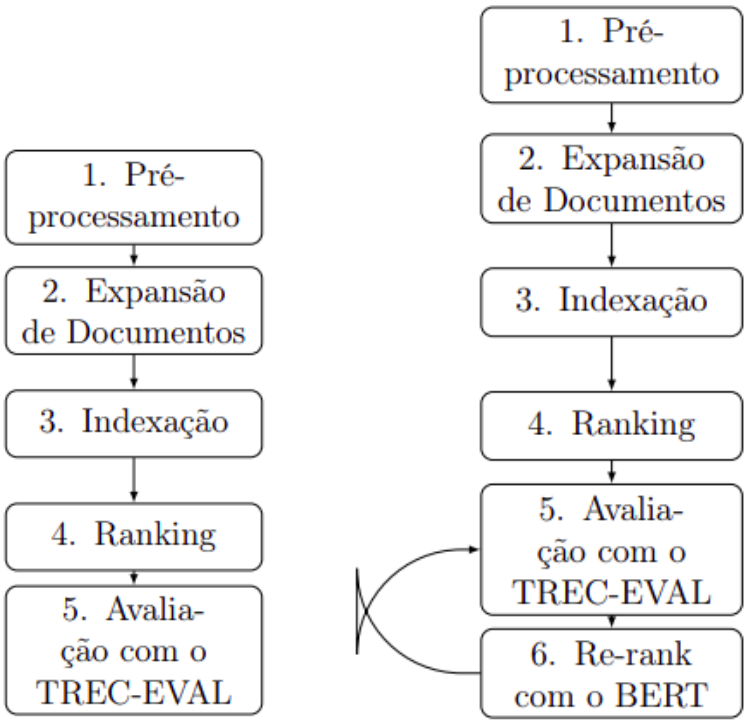
A figura 3.1b é referente à utilização da etapa anterior em que as consultas são expandidas, porém, nesta etapa o *ranking* que foi obtido será reordenado com o BERT, para uma nova avaliação.

A figura 3.2a é relativa à Expansão de Documentos, em que os documentos presentes no *dataset* serão expandidos com o método Doc2query. As etapas são como na Expansão de Consultas, porém os documentos só são indexados após a etapa de expansão.



(a) Expansão de Consultas (b) Expansão de Consultas + BERT

Figura 3.1: Etapas da Expansão de Consultas



(a) Expansão de Documentos (b) Expansão de Documentos + BERT

Figura 3.2: Etapas da Expansão de Documentos

Assim como na Expansão de Consultas, o *ranking* obtido na Expansão de Documentos também pode ser reordenado com o BERT (etapa 6), como na figura 3.2b.

Para este trabalho também foi avaliada a alternativa de combinar funções de Expansão de Consultas com o método Doc2query, gerando um novo ranking para ser reordenado com o BERT, finalizando as fases de testes. A seguir são apresentados os pontos utilizados em cada etapa de forma mais detalhada.

1. São utilizados os documentos referentes aos Discos 1 e 2 do TREC, totalizando 741853 documentos. Cada documento é indexado usando o Apache Lucene.

2. Os documentos e consultas são pré-processados, sendo removidas as *tags* do XML.

3. A consulta(título) é expandida utilizando a função de expansão escolhida: $RM3_1^+$, $RM3_2^+$ ou $RM3_3^+$. Cada termo é adicionado à consulta com um peso.

4. É armazenado em arquivo um ranking de documentos com um score de similaridade para cada consulta.

5. Os documentos são expandidos utilizando o método Doc2query, implementado na biblioteca OpenNMT.

6. Para escolha de parâmetros(Conjunto de Desenvolvimento) são utilizadas as consultas de 51-100.

7. Para avaliação do resultado(Conjunto de Teste) são utilizadas as consultas de 101-200.

8. Os resultados obtidos pelo algoritmo são comparados com as respostas(qrels), avaliando os documentos retornados.

9. O resultado obtido é reordenado com o BERT e reavaliado.

Capítulo 4

Resultados e Discussões

Neste capítulo serão apresentados os resultados com as Funções de Expansão. Na Expansão de Consultas, para gerar o conjunto de documentos pseudo-relevantes e o ranking final para avaliação, foi utilizado BM25, Dirichlet Similarity ou Jelinek Mercer Similarity.

Na etapa de Expansão de Documentos foi utilizada a biblioteca OpenNMT para adicionar consultas geradas aos documentos contidos na base, e para o treinamento do modelo foi utilizado GPU. Já a última etapa é referente a reordenar os *rankings* iniciais com o BERT LARGE (*batch* = 128), que parte de um modelo LARGE pré-treinado no *dataset* MSMARCO e continuando o treinamento no *dataset* TREC com os textos originais das consultas e documentos, e para treinamento do modelo foi utilizado TPU [1].

4.1 Configuração da máquina utilizada

Processador: AMD® Ryzen 5 3600 6-core processor × 12
Placa de vídeo: NVIDIA Corporation TU116 [GeForce GTX 1660]
Memória RAM: 16 GB
Sistema Operacional: Ubuntu 20.04.2 LTS
TPU: v3-8 (Google Cloud)

4.2 Métricas de avaliação

4.2.1 Precision

A precisão representa a probabilidade de um item selecionado ser relevante, dada pela fórmula abaixo, sendo a razão de itens relevantes selecionados (N_{rs}) para o número total de itens selecionados (N_s): (HERLOCKER *et al.*, 2004).

[1] https://cloud.google.com/tpu/docs/intro-to-tpu?hl=pt-br

$$P = \frac{N_{rs}}{N_s} \tag{4.1}$$

4.2.2 Recall

O *Recall* representa a probabilidade de que um item relevante seja selecionado, sendo a razão de itens relevantes selecionados (N_{rs}) por um número total de itens relevantes (N_r): (HERLOCKER *et al.*, 2004):

$$R = \frac{N_{rs}}{N_r} \tag{4.2}$$

4.2.3 MAP

A *Mean Average Precision* (MAP) é o AP(Average Precision) médio em todas as consultas (BAST *et al.*, 2018). Como pode ser visto no exemplo[2] abaixo:

Considerando duas consultas e um *ranking* de documentos para cada consulta, em que a primeira consulta possui 5 documentos relevantes e a segunda consulta possui 3 documentos relevantes, o AP médio é dado por:

- AP para consulta 1: (1.0+0.67+0.5+0.44+0.5)/5=0.62
- AP para consulta 2: (0.5+0.4+0.43)/3=0.44
- MAP: (0.62+0.44)/2=0.53

Ou seja, a média entre a razão do somatório de precisões para a primeira consulta pelo número de documentos relevantes + somatório de precisões para a segunda consulta pelo número de documentos relevantes, sob o número de consultas.

4.2.4 NDCG

NDCG é a normalização do DCG (Ganho Acumulado Descontado), que é a soma ponderada do grau de relevância dos itens classificados. O peso é uma função decrescente da classificação, sendo então considerado como desconto. Esse desconto é considerado devido a probabilidade de um usuário selecionar um documento diminuir de acordo com a classificação (LESTARI *et al.*, 2018).

O NDCG normaliza o DCG pelo ideal DCG (IDCG), que é a medida do DCG do melhor resultado de classificação. A equação abaixo representa o cálculo do NDCG, possuindo valores entre 0 e 1 LESTARI *et al.* (2018).

[2]https://web.stanford.edu/class/cs276/handouts/EvaluationNew-handout-1-per.pdf

$$DCG_D(f, S_n) = \sum_{r=1}^{n} y_{(r)}^{f} D(r) \tag{4.3}$$

$$NDCG_D(f, S_n) = \frac{DCG_D(f, S_n)}{IDCG_D(S_n)} \tag{4.4}$$

D(r) (r>=1) é a função de desconto, f é a função de *ranking* e S_n um *dataset*, o $IDCG_D = max f' \sum_{r=1}^{n} y_{(r)}^{f'} D(r)$ (LESTARI *et al.*, 2018).

4.3 Resultados

Nesta seção serão apresentados os resultados obtidos nos experimentos, variando as funções de Expansão de Consultas, utilizando a Expansão de Documentos e o *Re-rank* com o BERT LARGE no conjunto de consultas 101-200. As tabelas a seguir apresentam os resultados com os métodos de similaridade/*ranking* escolhidos: BM25, Dirichlet Similarity ou Jelinek Mercer Similarity, ambos utilizando o Apache Lucene.

São apresentados os resultados sem a Expansão de Consultas (4.1) e expandindo as consultas com as funções de Expansão $RM3_1^+$, $RM3_2^+$ ou $RM3_3^+$. Como são modelos baseados em IDF, as *stopwords* foram removidas, foi aplicado *lowercase* e Porter Stemming. Após a etapa de Expansão de Consultas, o melhor *ranking* obtido será reordenando (*re-rank*) com o BERT (4.5). Cada método de *ranking* retorna para cada consulta 1000 documentos relevantes com uma pontuação associada, então o RECALL apresentado é referente ao R@1000.

4.3.1 *Re-rank* BERT - Expansão de Consultas

Para Expandir as consultas foram utilizados 40 Documentos Pseudo-relevantes, ou seja, documentos que serão utilizados como ranking inicial com as consultas sem expansão, a fim de obter os melhores termos de expansão sem a necessidade de um *dataset* externo. Também foram utilizados 30 termos para expansão, ou seja, cada consulta pode ter até 30 termos adicionados, com o objetivo de aumentar a consulta de um usuário e trazer melhores resultados.

A tabela 4.1 abaixo mostra os resultados com as métricas P@10, RECALL, RPREC, MAP e NDCG obtidos com o BM25, a seguir serão apresentados os resultados com as funções de Expansão de Consultas.

Métrica	BM25 (k = 0.82 , b = 0.68)
P@10	0.4830
RECALL	0.6111
MAP	0.2295
NDCG	0.5139

Tabela 4.1: BM25 sem expandir as consultas

Na tabela a seguir são apresentados os resultados com BM25, em que os melhores resultados foram obtidos com a função de expansão $RM3^{+}_{3}$.

Métrica	BM25 (k = 0.82 , b = 0.68)		
	$RM3^{+}_{1}$	$RM3^{+}_{2}$	$RM3^{+}_{3}$
P@10	0.5510	0.5350	0.5460
RECALL	0.6835	0.6468	0.6840
MAP	0.3054	0.2720	0.3082
NDCG	0.5834	0.5505	0.5856

Tabela 4.2: BM25 com Expansão de Consultas

Na tabela a seguir são apresentados os resultados com Dirichlet Similarity, em que os melhores resultados foram obtidos com a função de expansão $RM3^{+}_{1}$.

Métrica	Dirichlet Similarity ($\mu = 1000$)		
	$RM3^{+}_{1}$	$RM3^{+}_{2}$	$RM3^{+}_{3}$
P@10	0.5610	0.5370	0.5530
RECALL	0.6855	0.6460	0.6810
MAP	0.3023	0.2713	0.3017
NDCG	0.5833	0.5478	0.5815

Tabela 4.3: Dirichlet Similarity com Expansão de Consultas

Métrica	Jelinek Mercer Similarity($\lambda = 0.1$)		
	$RM3^{+}_{1}$	$RM3^{+}_{2}$	$RM3^{+}_{3}$
P@10	0.4870	0.4480	0.4720
RECALL	0.6523	0.5972	0.6165
MAP	0.2646	0.2183	0.2392
NDCG	0.5464	0.4952	0.5163

Tabela 4.4: Jelinek Mercer Similarity com Expansão de Consultas

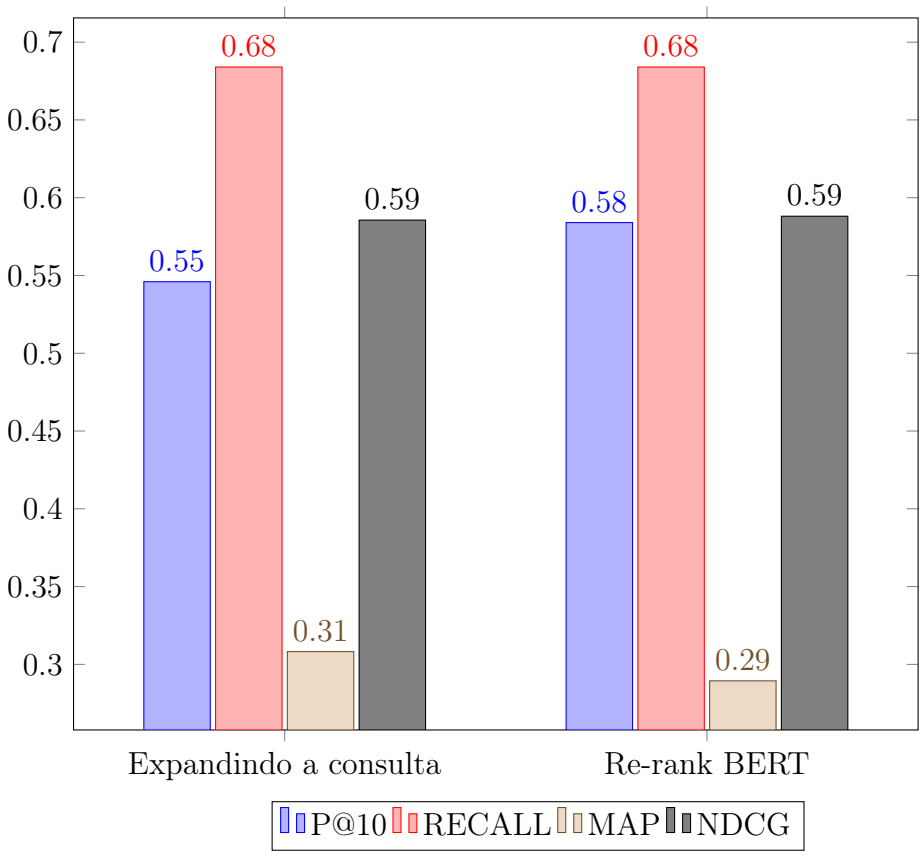
A tabela acima é referente os resultados com Jelinek Mercer Similarity, em que os melhores resultados foram obtidos com a função de expansão $RM3^{+}_{1}$.

No conjunto de desenvolvimento o BM25 com a função de expansão $RM3^{+}_{3}$ obteve melhores resultados. No conjunto de testes, bons resultados foram obtidos

com o BM25 com a função de expansão $RM3_3^+$ e com Dirichlet Similarity e função $RM3_1^+$, então a combinação BM25 + $RM3_3^+$ foi utilizada para o *re-rank* com o BERT, já que a função $RM3_3^+$ possui melhor generalização. No gráfico a seguir é possível ver a melhora em algumas métricas.

Métrica	BERT: BM25 (k = 0.82 , b = 0.68)
P@10	0.5840
RECALL	0.6840
MAP	0.2894
NDCG	0.5881

Tabela 4.5: Re-rank com o BERT



Métrica	BM25 (k = 0.82 , b = 0.68)
P@10	0.4550
RECALL	0.6840
MAP	0.2316
NDCG	0.5444

Tabela 4.6: Re-rank com o BERT

Os testes com o BERT foram realizados com o *ranking* inicial com o BM25, em que a consulta utilizada foi expandida. O BERT utilizou nas etapas anteriores o

texto original da consulta. A tabela 4.6, mostra os resultados em que o texto expandido da consulta foi utilizado além do *ranking* inicial com o BM25, mas também no BERT(representação do *embedding* da frase) e não gerou melhores resultados.

4.3.2 *Re-rank* BERT - Expansão de Documentos

Nas tabelas abaixo serão apresentados os resultados com a Expansão de Documentos utilizando o método Doc2query em que cada documento do *dataset* é expandido utilizando a biblioteca OpenNMT. Assim como na Expansão de Consultas, os documentos também serão reordenados com o BERT LARGE, melhorando os resultados após o *re-rank*.

Métrica	BM25 (k = 0.82 , b = 0.68)
P@10	0.4780
RECALL	0.5928
MAP	0.2255
NDCG	0.4989

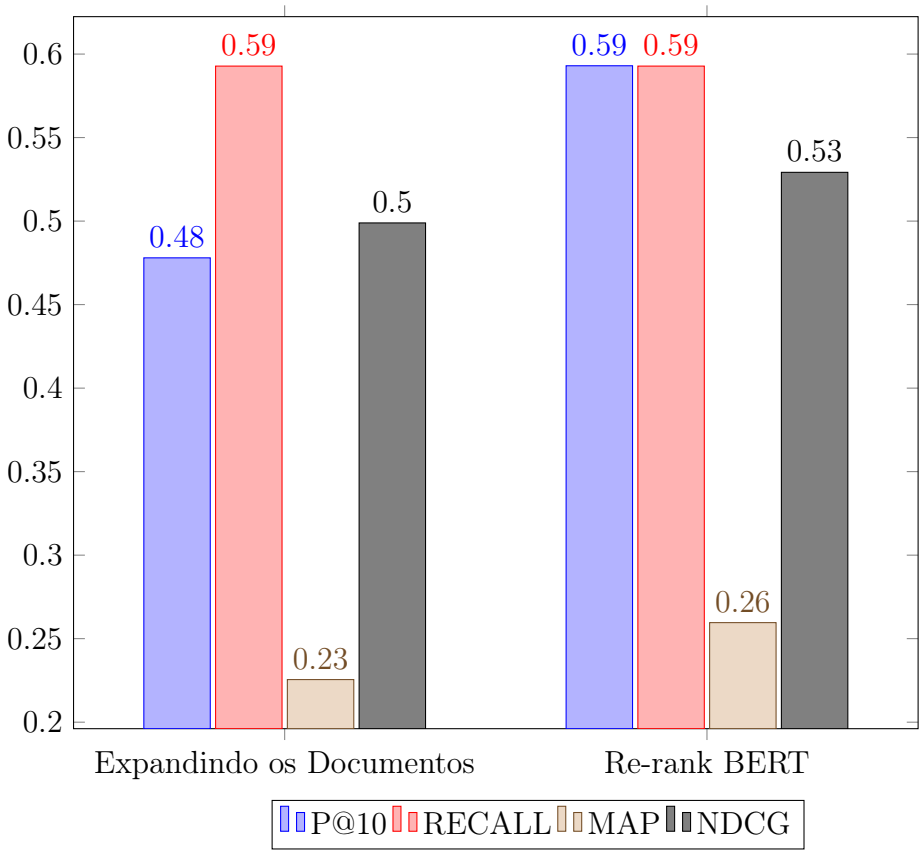
Tabela 4.7: BM25 (Documentos Expandidos)

A tabela 4.7 é referente aos resultados com BM25, em que os resultados foram obtidos adicionando 5 consultas a cada documento com *batch=8*. Para evitar o uso excessivo de memória (NOGUEIRA *et al.*, 2019) da GPU durante o treinamento do modelo, os documentos foram truncados em 400 *tokens* e as consultas em 100 *tokens*.

Métrica	BERT: BM25 (k = 0.82 , b = 0.68)
P@10	0.5930
RECALL	0.5928
MAP	0.2596
NDCG	0.5292

Tabela 4.8: BERT - BM25 (Documentos Expandidos)

A tabela 4.8 apresenta os resultados obtidos com o *ranking* da etapa anterior, após o *re-rank* com o BERT. No gráfico a seguir é possível ver a comparação.



A tabela 4.9 é referente os resultados com BM25, em que os resultados foram obtidos adicionando 10 consultas a cada documento com *batch=32*.

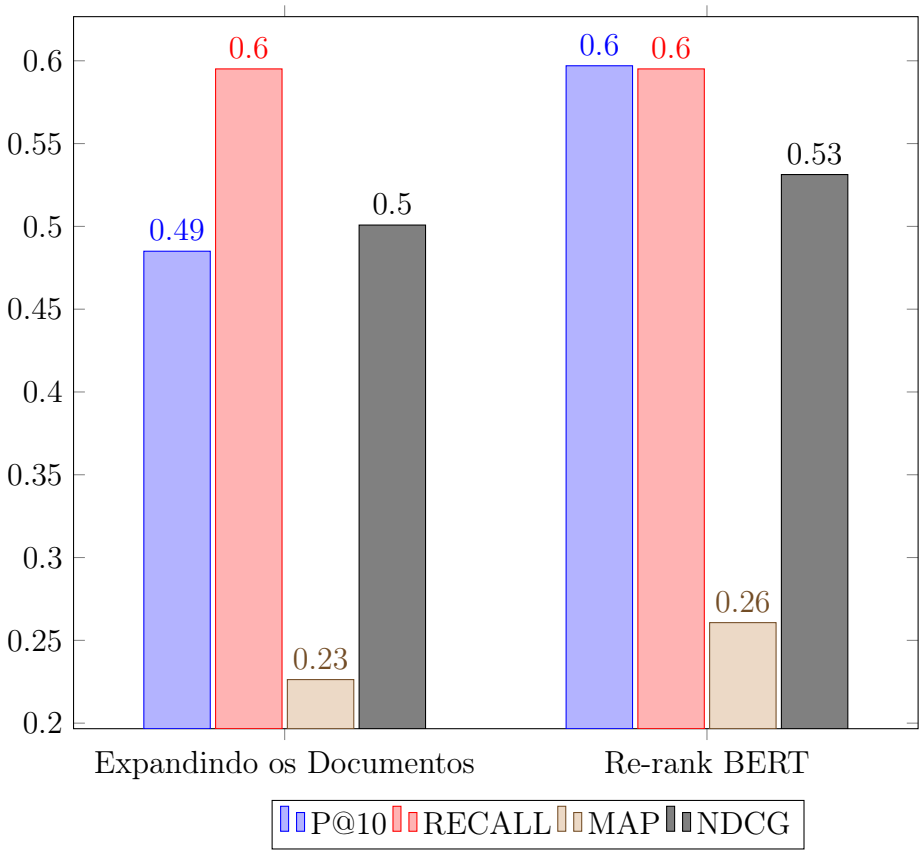
Métrica	BM25 (k = 0.82 , b = 0.68)
P@10	0.4850
RECALL	0.5951
MAP	0.2263
NDCG	0.5008

Tabela 4.9: BM25 (Documentos Expandidos)

A tabela 4.10 apresenta os resultados obtidos com o ranking da etapa anterior, após o re-rank com o BERT, em que houve melhora nos resultados.

Métrica	BM25 (k = 0.82 , b = 0.68)
P@10	0.5970
RECALL	0.5951
MAP	0.2607
NDCG	0.5313

Tabela 4.10: BERT: BM25 (Documentos Expandidos)



Também foi avaliado utilizar o texto expandido dos documentos, em que além de partir de um *ranking* inicial com a pontuação obtida com o BM25 que obteve bons resultados na etapa anterior, também será reordenado com os textos expandidos utilizando o BERT, conforme as tabelas abaixo, em que pode ser observado que não houve melhora nos resultados.

Métrica	BM25 (k = 0.82 , b = 0.68)
P@10	0.5930
RECALL	0.5951
MAP	0.2607
NDCG	0.5312

Tabela 4.11: BERT: BM25 (Documentos Expandidos)

4.3.3 *Re-rank* BERT - Expansão de Consultas e Documentos

Nessa seção, o BERT foi testado para representar o *embedding* da frase das consultas expandidas e o *embedding* da frase das documentos expandidos. A tabela 4.12 mostra os resultados iniciais com o BM25, sendo então utilizado $RM3_3^+$ para Expansão de Consultas, Doc2query para Expansão de Documento e o BERT como *re-rank*.

A tabela 4.13 mostra o *re-rank* do resultado anterior, obtido na tabela 4.12, em que houve melhora em relação ao P@10, RECALL(mantido, por ser a mesma lista)

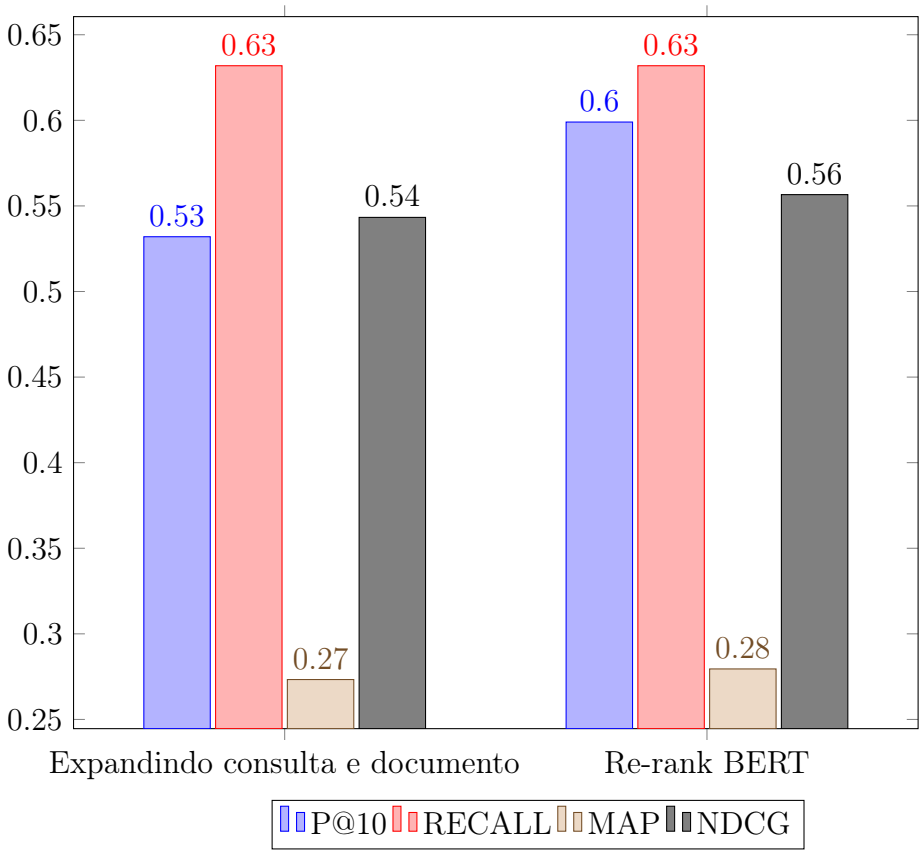
Métrica	BM25 (k = 0.82 , b = 0.68) /101-200
P@10	0.5320
RECALL	0.6319
MAP	0.2733
NDCG	0.5433

Tabela 4.12: BM25 (Consultas Expandidas e Documentos Expandidos)

e NDCG.

Métrica	BM25 (k = 0.82 , b = 0.68)
P@10	0.5990
RECALL	0.6319
MAP	0.2795
NDCG	0.5566

Tabela 4.13: BERT: BM25 (Consultas Expandidas e Documentos Expandidos)



Capítulo 5

Conclusões

Conforme apresentado ao longo do trabalho, alguns métodos para o *ranking* consulta-documento foram testados, sendo o BM25 um modelo probabilístico de Recuperação de Informação que avalia a probabilidade de que cada documento seja relevante para cada consulta (BENNETT *et al.*, 2008). E também os modelos de suavização: Jelinek-Mercer e Dirichlet *smoothing*, em que o primeiro combina a frequência relativa de um termo da consulta em um documento d com a frequência relativa do termo na coleção C completa (BENNETT *et al.*, 2008). Já o segundo, torna a suavização dependente do tamanho do documento, tendo geralmente em documentos mais longos menos necessidade de suavização, devido a melhor precisão (BENNETT *et al.*, 2008).

Também foram avaliadas as funções de Expansão $RM3^+_1$, $RM3^+_2$ o $RM3^+_3$, em que a terceira função proposta evita o uso de um fator IDF "duplo" como na função anterior durante a recuperação para obter uma melhor representação (ROY *et al.*, 2019). A Expansão de Documentos Doc2query(NOGUEIRA *et al.*, 2019) foi feita com a biblioteca OpenNMT que é um kit de ferramentas de código aberto para tradução automática neural(Neural Machine Translation - NMT) modelando a probabilidade de uma sentença alvo KLEIN *et al.* (2017).

E por último é feito o *re-rank* com o BERT LARGE (NOGUEIRA e CHO, 2019). Pode ser observado que o BERT obtém melhora em algumas métricas em relação ao resultado inicial, sendo então considerado uma boa opção, já que métodos menos robustos como um modelo de Expansão de Consultas em conjunto com o BM25 pode ser utilizado e o BERT ser utilizado em um conjunto de respostas já retornadas.

A combinação que gerou melhores resultados (P@10, RECALL(igual, devido a ser a mesma lista reordenada) e NDCG), foi a Expansão de Consultas $RM3^+_3$ + BM25 + *Re-Rank* com o BERT LARGE, apresentado na tabela 4.5, tendo então resultado superior à Expansão de Documentos, apresentando vantagem em relação ao resultado e por não ter a necessidade de treinamento de modelo neural, nem reindexação de documentos caso algum parâmetro da Expansão de Documentos

for alterado, já que os documentos na Expansão de Consultas não precisam ser reindexados por variação no método.

O uso de GPU foi essencial para o treinamento do modelo Doc2query, pois o treinamento seria muito mais demorado se fosse realizado em CPU, assim como o uso de TPU foi essencial para o treinamento do BERT, em que na máquina utilizada antes do BERT não era possível devido ao estouro de memória da GPU. Além de poder levar até 10x o tempo (NOGUEIRA *et al.*, 2019). Como trabalhos futuros outros modelos de similaridade podem ser avaliados para que o BERT obtenha melhor reordenação.

Referências Bibliográficas

ABDUL-JALEEL, N., ALLAN, J., CROFT, W. B., et al., 2004, "UMass at TREC 2004: Novelty and HARD", *Computer Science Department Faculty Publication Series*, p. 189.

ABIODUN, O. I., JANTAN, A., OMOLARA, A. E., et al., 2018, "State-of-the-art in artificial neural network applications: A survey", *Heliyon*, v. 4, n. 11, pp. e00938.

ALAMMAR, J., 2018a, "The illustrated bert, elmo, and co", *How NLP Cracked Transfer Learning*, p. 38.

ALAMMAR, J., 2018b, "The illustrated transformer", *The Illustrated Transformer–Jay Alammar–Visualizing Machine Learning One Concept at a Time*, v. 27.

ALZUBAIDI, L., ZHANG, J., HUMAIDI, A. J., et al., 2021, "Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions", *Journal of big Data*, v. 8, n. 1, pp. 1–74.

BAEZA-YATES, R., RIBEIRO-NETO, B., 2013, "Recuperação de Informação-: Conceitos e Tecnologia das Máquinas de Busca", .

BAST, H., BUCHHOLD, B., HAUSSMANN, E., 2018, "A quality evaluation of combined search on a knowledge base and text", *KI-Künstliche Intelligenz*, v. 32, n. 1, pp. 19–26.

BENNETT, G., SCHOLER, F., UITDENBOGERD, A., 2008, "A comparative study of probabilistic and language models for information retrieval". In: *Database Technologies 2008: Proceedings of the Nineteenth Australasian Database Conference (ADC 2008)*, pp. 65–74. CRPIT.

CARPINETO, C., ROMANO, G., 2012, "A survey of automatic query expansion in information retrieval", *ACM Computing Surveys (CSUR)*, v. 44, n. 1, pp. 1.

DEVLIN, J., CHANG, M.-W., LEE, K., et al., 2018, "Bert: Pre-training of deep bidirectional transformers for language understanding", *arXiv preprint arXiv:1810.04805*.

FAN, A., LEWIS, M., DAUPHIN, Y., 2018, "Hierarchical neural story generation", *arXiv preprint arXiv:1805.04833*.

FURNAS, G. W., LANDAUER, T. K., GOMEZ, L. M., et al., 1987, "The vocabulary problem in human-system communication", *Communications of the ACM*, v. 30, n. 11, pp. 964–971.

GHAWI, R., PFEFFER, J., 2019, "Efficient hyperparameter tuning with grid search for text categorization using kNN approach with BM25 similarity", *Open Computer Science*, v. 9, n. 1, pp. 160–180.

GREENGRASS, E., 2000, "Information retrieval: A survey", .

HARMAN, D., 1995, "Overview of the second text retrieval conference (TREC-2)", *Information Processing & Management*, v. 31, n. 3, pp. 271–289.

HERLOCKER, J. L., KONSTAN, J. A., TERVEEN, L. G., et al., 2004, "Evaluating collaborative filtering recommender systems", *ACM Transactions on Information Systems (TOIS)*, v. 22, n. 1, pp. 5–53.

KLEIN, G., KIM, Y., DENG, Y., et al., 2017, "Opennmt: Open-source toolkit for neural machine translation", *arXiv preprint arXiv:1701.02810*.

LEDERER, J., 2021, "Activation functions in artificial neural networks: A systematic overview", *arXiv preprint arXiv:2101.09957*.

LESTARI, S., ADJI, T. B., PERMANASARI, A. E., 2018, "WP-Rank: Rank Aggregation based Collaborative Filtering Method in Recommender System", *Int. J. Eng. Technol*, v. 7, pp. 193–197.

MANNING, C. D., RAGHAVAN, P., SCHUTZE, H., "Introduction to infor-mation retrieval? cambridge university press 2008", *Ch*, v. 20, pp. 405–416.

MARON, M. E., KUHNS, J. L., 1960, "On relevance, probabilistic indexing and information retrieval", *Journal of the ACM (JACM)*, v. 7, n. 3, pp. 216–244.

NEUBIG, G., 2017, "Neural machine translation and sequence-to-sequence models: A tutorial", *arXiv preprint arXiv:1703.01619*.

NOGUEIRA, R., CHO, K., 2019, "Passage Re-ranking with BERT", *arXiv preprint arXiv:1901.04085*.

NOGUEIRA, R., YANG, W., LIN, J., et al., 2019, "Document expansion by query prediction", *arXiv preprint arXiv:1904.08375*.

NOGUEIRA, R., JIANG, Z., LIN, J., 2020, "Document ranking with a pretrained sequence-to-sequence model", *arXiv preprint arXiv:2003.06713*.

ROY, D., BHATIA, S., MITRA, M., 2019, "Selecting Discriminative Terms for Relevance Model". In: *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1253–1256.

SALEHINEJAD, H., SANKAR, S., BARFETT, J., et al., 2017, "Recent advances in recurrent neural networks", *arXiv preprint arXiv:1801.01078*.

SENGUPTA, S., BASAK, S., SAIKIA, P., et al., 2020, "A review of deep learning with special emphasis on architectures, applications and recent trends", *Knowledge-Based Systems*, v. 194, pp. 105596.

SENNRICH, R., HADDOW, B., BIRCH, A., 2015, "Neural machine translation of rare words with subword units", *arXiv preprint arXiv:1508.07909*.

SHAHID, A., MUSHTAQ, M., 2020, "A Survey Comparing Specialized Hardware And Evolution In TPUs For Neural Networks". In: *2020 IEEE 23rd International Multitopic Conference (INMIC)*, pp. 1–6. doi: 10.1109/INMIC50486.2020.9318136.

SINGHAL, A., OTHERS, 2001, "Modern information retrieval: A brief overview", *IEEE Data Eng. Bull.*, v. 24, n. 4, pp. 35–43.

VASWANI, A., SHAZEER, N., PARMAR, N., et al., 2017, "Attention is All you Need". In: Guyon, I., Luxburg, U. V., Bengio, S., et al. (Eds.), *Advances in Neural Information Processing Systems*, v. 30. Curran Associates, Inc. Disponível em: <`https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf`>.

VOORHEES, E. M., 2001, "Question answering in TREC". In: *Proceedings of the tenth international conference on Information and knowledge management*, pp. 535–537.

WU, Y., SCHUSTER, M., CHEN, Z., et al., 2016, "Google's neural machine translation system: Bridging the gap between human and machine translation", *arXiv preprint arXiv:1609.08144*.

XIE, Y., WEN, H., YANG, Q., 2021, "Ternary Sentiment Classification of Airline Passengers' Twitter Text Based on BERT", *Journal of Physics: Conference Series*, v. 1813 (02), pp. 012017. doi: 10.1088/1742-6596/1813/1/012017.

ZHAI, C., LAFFERTY, J., 2004, "A study of smoothing methods for language models applied to information retrieval", *ACM Transactions on Information Systems (TOIS)*, v. 22, n. 2, pp. 179–214.