UNIVERSIDADE FEDERAL DO CEARÁ

CAMPUS DE QUIXADÁ

CURSO DE GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

ROBSON TEIXEIRA PAULA

GERAÇÃO DE DADOS SINTÉTICOS PARA TREINAMENTO DE CHATBOTS BASEADOS NA CARTA DE SERVIÇOS DO GOVERNO DO CEARÁ

QUIXADÁ

2021

ROBSON TEIXEIRA PAULA

GERAÇÃO DE DADOS SINTÉTICOS PARA TREINAMENTO DE CHATBOTS BASEADOS NA CARTA DE SERVIÇOS DO GOVERNO DO CEARÁ

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da computação do Campus de Quixadá da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Ciência da computação.

Orientador: Prof. Dr. Paulo de Tarso Guerra Oliveira

Coorientador: Me. Décio Gonçalves de Aguiar Neto

QUIXADÁ

2021

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca Universitária
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

P349g Paula, Robson Teixeira.
Geração de dados sintéticos para treinamento de chatbots baseados na carta de serviços do Governo do Ceará / Robson Teixeira Paula. – 2021.
75 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Quixadá, Curso de Ciência da Computação, Quixadá, 2021.
Orientação: Prof. Dr. Paulo de Tarso Guerra Oliveira.
Coorientação: Prof. Me. Décio Gonçalves de Aguiar Neto.

1. Chatterbot . 2. Processamento de linguagem natural (Computação). 3. Ceará - Governo do Estado. I. Título.

CDD 004

ROBSON TEIXEIRA PAULA

GERAÇÃO DE DADOS SINTÉTICOS PARA TREINAMENTO DE CHATBOTS BASEADOS NA CARTA DE SERVIÇOS DO GOVERNO DO CEARÁ

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da computação do Campus de Quixadá da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Ciência da computação.

Aprovada em: ____/_____/_____

BANCA EXAMINADORA

Prof. Dr. Paulo de Tarso Guerra Oliveira (Orientador)
Universidade Federal do Ceará (UFC)

Me. Décio Gonçalves de Aguiar Neto (Coorientador)
Universidade Estadual de Campinas (UNICAMP)

Prof. Dr. Davi Romero de Vasconcelos
Universidade Federal do Ceará(UFC)

Profª. Ma. Lívia Almada Cruz
Universidade Federal do Ceará (UFC)

AGRADECIMENTOS

Agradeço primeiramente a minha mãe Valdete e minha irmã Jaqueline por todo o suporte, carinho e dedicação durante os anos de graduação.

Agradeço aos meus orientadores Paulo de Tarso Guerra Oliveira e Décio Gonçalves de Aguiar Neto, pela oportunidade, conselhos, ensinamentos e pela excelente orientação exercida.

Agradeço aos professores Lívia Almada Cruz e Davi Romero de Vasconcelos pela disponibilidade em compor a banca avaliadora, e por todas as dicas e conselhos passados que contribuíram significativamente na melhoria e engrandecimento deste trabalho.

Agradeço aos meus colegas de turma do curso de ciência da computação e aos amigos que fiz durante estes quatro anos, por todos os momentos de diversão e de estudos compartilhado.

Agradeço aos meus amigos Alan, David, Davi, Wenden e Wladson pelos momentos que passamos juntos e por todos esses anos de amizade.

Agradeço também a FUNCAP, uma vez que este trabalho é parcialmente financiado pelo projeto Governo Digital do Estado do Ceará - 04772314/2020 FUNCAP.

RESUMO

A carta de serviços do Governo do Ceará é um conjunto de informações sobre os serviços ofertados pelo Estado. Formas de facilitar o acesso a essas informações é um dos objetivos do projeto Governo Digital. Considerando esse objetivo, o presente trabalho realiza uma análise sobre formas de gerar bases de dados textuais para um domínio ainda não explorado, utilizando a carta de serviços como domínio. Utilizando técnicas de geração de dados baseado em modelos de textos e geração de texto automática utilizando o modelo de aprendizagem profunda PTT5, são geradas bases de dados considerando a aplicação dessas bases em um *chatbot* desenvolvido com o Rasa, tendo como foco a tarefa de classificação de intenções. O desempenho da tarefa de classificação é avaliada através de modelos baseados em *transformers*, explorando casos mais restritos e menos restritos. Os resultados obtidos utilizando o *chatbot* e os modelos de classificação de intenções desenvolvido nesse trabalho são promissores e mostram que as técnicas de geração artificial de dados podem ser bem exploradas a fim de gerar uma base de dados de qualidade.

Palavras-chave: Transformers. Chatbot. Geração de dados sintéticos. Processamento de Linguagem Natural. Governo Digital.

ABSTRACT

The list of services of the Government of Ceará is a set of information about the services offered by the Government. Finding ways to facilitate access to this information is one of the objectives of the Digital Government project. Considering this objective, the present work is an analysis of methods to generate databases for a domain not yet explored, using the list of services as a domain. Using data generation techniques based on text models and automatic text generation using the PTT5 deep learning model, databases are generated considering the application in these databases in a *chatbot* developed in Rasa, focusing on the task classification of intentions. The classification task is performed through models based on *transformers*, exploring more restricted and less restricted cases. The results obtained using the *chatbot* and the intention classification models developed in this work are promising and show that the data generation techniques can be well explored in order to generate a quality database.

Keywords: Transformers. Chatbot. Synthetic data generation. Natural Language Processing. Digital Government.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Responder perguntas de forma automática em linguagem natural é uma tarefa que é pesquisada desde a década de 60, o que tem gerado um grande número de sistemas que visam resolvê-la (MISHRA; JAIN, 2016). A tarefa de responder questões (*question answering*, ou QA) consiste em construir um sistema de informação capaz de obter perguntas e responde-las em linguagem natural(STUPINA *et al.*, 2016). Segundo Silveira e Mauá (2018) as principais dificuldades em projetar sistemas de QA são o custo para criar e manter uma base de conhecimento e determinar se as respostas geradas são satisfatórias.

Entre as diferentes aplicações de sistemas de QA está o *chatbot*, um sistema voltado para conversação com usuário, normalmente utilizado como um assistente virtual para auxiliar pessoas ou tirar dúvidas (AL-SINANI; AL-SAIDI, 2019).

Conforme o conceito de *chatbot* se popularizou, diversas ferramentas surgiram para auxiliar na criação desses sistemas (AL-SINANI; AL-SAIDI, 2019), como o *Dialogflow*, ferramenta desenvolvida pela *Google* e o *Microsoft bot*, desenvolvido pela *Microsoft*. Estas ferramentas podem ser escolhidas de acordo com a necessidade e a experiência do desenvolvedor, com algumas permitindo que até pessoas que não sabem programar possam desenvolver *chatbots* (AL-SINANI; AL-SAIDI, 2019).

Dentre o conjunto de ferramentas existentes temos o Rasa (BOCKLISCH *et al.*, 2017), uma ferramenta de código aberto que permite o desenvolvimento de *chatbots* por pessoas que não sejam especialistas nessa área. A ferramenta funciona identificando as intenções contidas nas mensagens do usuário, classificando-as de acordo com as intenções definidas pelo desenvolvedor, e retornando uma resposta com base na intenção, podendo essa resposta ser um texto já definido ou uma ação customizada, tais como acesso a base de dados.

A classificação de intenção do Rasa é feita a partir de algoritmos de aprendizado de máquina, consistindo de métodos de classificação voltados para processamento de linguagem natural (*natural language processing* ou NLP), podendo ser auxiliado por métodos de modelagem e tokenização, que facilitam o entendimento do texto pelo sistema.

A classificação de intenção é crucial para o *chatbot* construído no Rasa, uma vez que identifica qual o desejo do usuário e, consequentemente, qual a resposta a ser retornada pelo sistema. Por isso é fundamental a construção de um modelo de classificação de intenções eficaz.

Contudo, um modelo eficaz precisa ser treinado com dados relevantes para que possa aprender bem as características do domínio o qual está treinado. Porém, bases de dados de quali-

dade nem sempre estão disponíveis para a tarefa desejada e na língua desejada, principalmente, quando a tarefa envolve um domínio pouco explorado, o que dificulta o processo de criação do modelo.

O desafio de ter um modelo de classificação de intenções para um domínio ainda não explorado foi um dos desafios encontrados por nós durante a execução do projeto Governo Digital do Ceará, um projeto financiado pela FUNCAP e desenvolvido em parceria entre o Governo do Estado do Ceará e o *Insight Lab*. Esse projeto tem como objetivo facilitar o acesso a informações do Estado pelos cidadãos, através da construção de um *chatbot* capaz de informar o cidadão sobre os serviços presentes na carta de serviço do cidadão[1].

Para que este *chatbot* possa ser desenvolvido é necessária uma base de dados voltada para classificação de intenção, o qual possua exemplos de textos e as intenções desses textos, sendo essas intenções referentes a que informação sobre um serviço o usuário deseja. Porém todas as informações do serviço se encontram em formato de FAQ, não estando disponibilizado em um formato que possa facilmente ser transformado em um conjunto de dados apropriado para o treinamento de *chatbots*.

Considerando esse problema, o presente trabalho visa investigar métodos para desenvolvimentos de dados sintéticos de qualidade e desenvolver bases de dados com questões sobre a carta de serviços do Governo do Estado do Ceará classificadas em intenções que indicam qual tipo de informação a questão requisita.

Neste trabalho, será desenvolvido um *chatbot* com o Rasa como forma de validar se as bases desenvolvidas são de qualidades e podem ser utilizadas para o desenvolvimento de *chatbots*.

O Rasa possui suporte a uma grande quantidade de classificadores, incluindo redes baseadas em *transformers* (VASWANI *et al.*, 2017). O *transformer* é um modelo para NLP proposto em Vaswani *et al.* (2017). Ele é um modelo *sequence-to-sequence*, onde a partir de uma sequência de símbolos é produzida uma nova, e possui uma arquitetura codificador-decodificador, tendo um codificador responsável por lidar com os dados de entrada e um decodificador responsável por gerar a saída (VASWANI *et al.*, 2017).

Apesar de inicialmente desenvolvido para realizar tarefas de tradução de forma rápida e com qualidade, inclusive alcançando um novo estado da arte em qualidade de tradução (VASWANI *et al.*, 2017), o *transformer* pode ser adaptado para realizar outras tarefas de NLP.

[1] Disponível em: https://cartadeservicos.ce.gov.br/ConsultaCesec/pg_cs_servico.aspx

Assim diversos modelos baseados em *transformer* foram desenvolvidos visando executar algumas tarefas de NLP, tais como GPT-2 (RADFORD *et al.*, 2019), transformer-XL (DAI *et al.*, 2019) e BERT (DEVLIN *et al.*, 2018), que podem ser utlizados para tarefas como QA e tradução de textos.

Entre os modelos citados anteriormente o BERT (*Bidirectional Encoder Representation from Transformers*), desenvolvido em Devlin *et al.* (2018), é um modelo pré-treinado baseado em *transformer* que conseguiu obter elevados resultados em onze tarefas de NLP, incluindo inferência de linguagem natural (*natural language inference* ou NLI) e QA (DEVLIN *et al.*, 2018).

Dado o novo estado da arte obtido pelo BERT e pelo *transformer*, será utilizado o BERTimbau (SOUZA *et al.*, 2020), uma versão do BERT para português, o BERT multilíngue e o BERT padrão para a tarefa de classificação de intenção, gerando modelos que podem ser integrados ao Rasa e que visa validar as bases de dados geradas.

Para gerar as bases de dados, dois métodos de geração serão explorados. A geração de textos baseada em modelos de texto, onde será definido um modelo com estruturas de textos, resultando em uma base com textos que seguem essas estruturas. E também a geração de bases baseado em modelos de aprendizagem de máquina, onde será utilizado o PTT5 (CARMO *et al.*, 2020), uma versão do T5 treinada com uma base de dados em português, para gerar novos textos.

1.1 CONTRIBUIÇÕES

As contribuições do presente trabalho são o desenvolvimento de bases de dados textuais sintéticas para a tarefa de classificação de intenções, através de métodos baseado em modelo de textos e modelos de aprendizagem profunda.

As contribuições desse trabalho são:

- Construção de bases de dados sintéticas em português para classificação de intenção de um *chatbot*.
- Proposta e implementação de um método para geração de dados sintéticos a partir de uma base existente utilizando o PTT5 (CARMO *et al.*, 2020).
- Análise das bases de dados geradas, utilizando *chatbot* e modelos de classificação de intenções.

2 FUNDAMENTAÇÃO TEÓRICA

Nesta seção são apresentados os conceitos principais deste trabalho. Na Seção 2.1 é apresentado o conceito de *chatbots*. Na Seção 2.2.1 é apresentado o conceito de redes *transformer* e modelos baseados em *transformer*. Na Seção 2.3 é apresentado o conceito de geração de dados sintéticos. A Seção 2.4 apresenta algumas métricas para avaliar modelos de aprendizado de máquina.

2.1 Chatbot

Um *chatbot* é um sistema de conversação capaz de emular a capacidade humana de comunicação utilizando técnicas de inteligência artificial para o entendimento da linguagem natural, obtendo significado a partir de textos e gerando respostas úteis para o usuário (NURUZZAMAN; HUSSAIN, 2018).

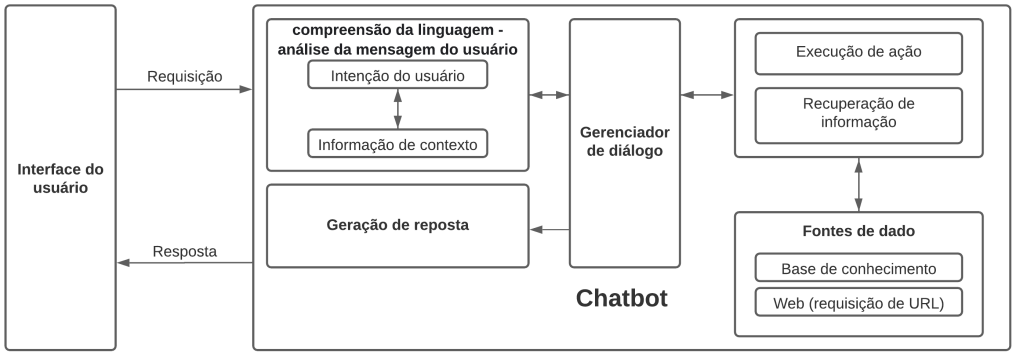
Segundo Nuruzzaman e Hussain (2018) um *chatbot* pode ser dividido em quatro grupos:

- Baseado em objetivo: *chatbots* que possuem um objetivo principal a ser alcançado, sendo configurado para ter uma conversa curta que ajuda o usuário a completar uma tarefa específica.
- Baseado em conhecimento: *chatbots* que acessam ou são treinados com uma determinada fonte de dados, gerando respostas a partir desses dados.
- Baseado em serviços: *chatbots* com o objetivo de auxiliar o usuário a requisitar um serviço, podendo ser de uso pessoal ou comercial.
- Baseado em geração de resposta: *chatbots* que possuem texto em linguagem natural como entrada e saída, onde a resposta retornada é decidida por um gerenciador de diálogo que gera modelos de respostas e retorna uma resposta baseada da prioridade de cada uma.

A Figura 1 ilustra o fluxo normalmente seguido por um *chatbot* baseado em geração de resposta. O processo inicia com uma requisição do usuário para o *chatbot*, que utilizará modelos de NLP para processar o texto e obter as informações, tais como intenções e entidades. Intenções são classes predefinidas que representam os tipos de consultas que o *chatbot* consegue entender e entidades são representações de objetos do mundo real ou suas propriedades específicas (SRIVASTAVA; PRABHAKAR, 2020). Um gerenciador de dialogo recebe essas informações e decide qual ação tomar, podendo resultar, por exemplo, na busca de informações

em alguma base de dados. Por último uma resposta é gerada e devolvida para o usuário. Esse é o tipo de *chatbot* desenvolvido nesse trabalho.

Figura 1 – Fluxo comumente seguido por um *chatbot*



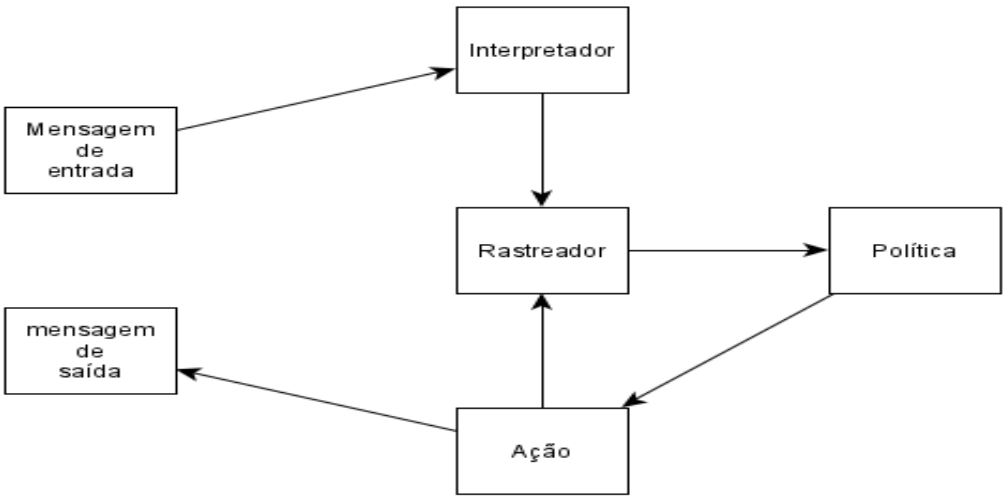
Fonte: (ADAMOPOULOU; MOUSSIADES, 2020)

Na próxima seção é apresentado o Rasa, um *framework* para desenvolvimento de conversações automáticas baseadas em texto e voz e que permite a implementação do tipo de *chatbot* desejado.

2.1.1 Rasa

O Rasa (BOCKLISCH *et al.*, 2017) é uma ferramenta que auxilia na construção de *chatbots*, permitindo que pessoas não especialistas possam construir um *chatbot* sem dificuldade. A Figura 2 mostra o fluxo seguido pelo Rasa, desde o recebimento da mensagem até o retorno da resposta.

Figura 2 – Arquitetura do RASA



Fonte: (BOCKLISCH *et al.*, 2017)

Uma vez que uma mensagem é recebida, o interpretador é responsável por extrair informações da mensagem, tais como intenções e entidades. Essas informações são repassadas para o rastreador, responsável por manter o estado da conversação. A política define uma ação a ser executada a partir do estado atual do rastreador. A ação executada irá gerar, normalmente, alguma saída e a retornará para o usuário. Se mais de uma ação deve ser tomada, após a execução de uma ação o rastreador é atualizado e volta a ser repassado para política.

O Rasa é *open source* e permite a integração com os meios de comunicação mais utilizados atualmente, como *facebook*, *whatsapp* e *telegram*. Devido a facilidade em customizar um *chatbot* baseado em Rasa e integrá-lo nos meios de comunicação e redes sociais, diversos trabalhos utilizam *chatbots* baseados em Rasa, como em Windiatmoko *et al.* (2021), onde um *chatbot* para dúvidas sobre uma universidade é desenvolvido e integrado ao *facebook*. Outro *chatbots* baseado do Rasa é apresentado em Mamatha *et al.* (2021), sendo desenvolvido um *chatbot* para o domínio de *e-commerce*.

2.2 Modelos de NLP baseados em *transformer*

2.2.1 Transformer

Redes neurais recorrentes (RNN), como *long short-term memory* (LSTM), foram por muito tempo o estado da arte para a solução de problemas de modelagem de sequência e tradução (SUTSKEVER *et al.*, 2014; VASWANI *et al.*, 2017).

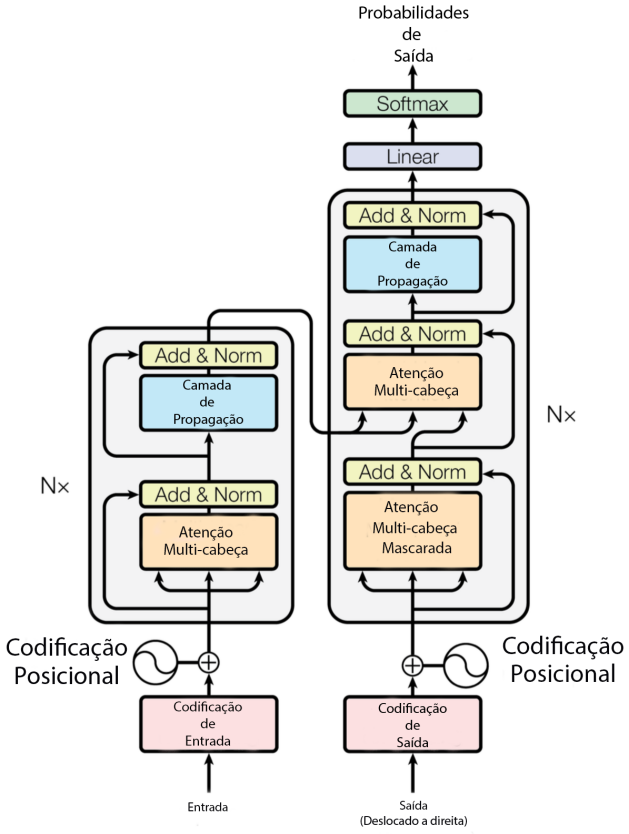
Modelos recorrentes fatoram a computação ao longo das posições dos símbolos das sequências de entrada e saída, gerando uma série de estados ocultos $h_1, h_2, \ldots, h_n$, onde cada estado h_t depende do estado h_{t-1} anterior (VASWANI *et al.*, 2017). Essa dependência dificulta a paralelização durante a etapa de treinamento, por conta das limitações de memória, uma vez que requer mais memória conforme a sequência de entrada cresce (VASWANI *et al.*, 2017).

Visando desenvolver um novo modelo que permitisse paralelização e fosse tão eficaz quanto redes neurais recorrentes para tradução de sequências, em Vaswani *et al.* (2017) foi desenvolvido uma arquitetura de rede denominada *transformer*. O *transformer* é o primeiro modelo a utilizar apenas auto atenção (*self-attention*, em inglês), um mecanismo de atenção responsável por determinar a relevância relativa de um *token* com relação aos demais *tokens* da sequência, não utilizando recorrência ou convolução alinhados por sequência (VASWANI *et al.*, 2017). No trabalho de Vaswani *et al.* (2017), após serem treinadas por 20 horas em oito GPUs

P100, essas redes alcançaram um novo estado da arte em qualidade de tradução (VASWANI *et al.*, 2017).

A rede *transformer* é composta de uma arquitetura do tipo codificador-decodificador, ilustrada na Figura 3. O codificador é responsável por transformar uma sequência de palavras, dado como entrada, para uma sequência de valores, que ao ser passada para o decodificador é gerado uma sequência de palavras como saída (VASWANI *et al.*, 2017).

Figura 3 – Arquitetura do modelo *transformer*



Fonte: (VASWANI *et al.*, 2017).

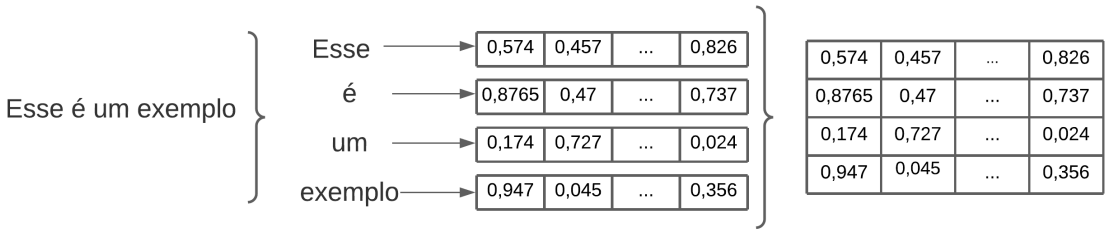
As próximas seções detalham alguns aspectos da arquitetura *transformer*.

2.2.1.1 Camada de codificação

A camada de codificação (*embedding*, em inglês) é utilizada no início do codificador e decodificador para converter cada *token* de entrada para um vetor de tamanho definido pelo desenvolvedor, tendo como resultado final uma matriz, onde cada linha da matriz é equivalente a um *token*.

Na Figura 4 é possível visualizar um exemplo de saída da camada de codificação. No exemplo é utilizado a frase “Esse é um exemplo”, que é dividido em 4 *tokens*: “Esse”, “é”, “um” e “exemplo”. Após passar pela camada de codificação, cada *token* é transformado em um vetor e no fim é retornada uma matriz.

Figura 4 – Exemplo de saída da camada de codificação



Fonte: Elaborado pelo autor

2.2.1.2 Codificação posicional

Uma vez que o modelo não utiliza recorrência ou convolução, é necessário adicionar informações sobre a posição absoluta ou relativa dos *tokens* da sequência, para assim o modelo poder fazer uso da ordem da sequência.

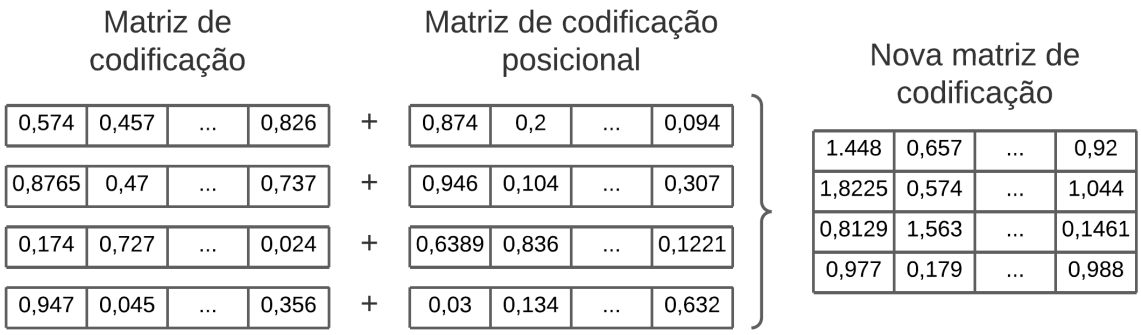
Assim, ao final, é gerado um vetor contendo a codificação posicional que é somado ao vetor de codificação que representa o *token*. Uma vez que esse cálculo seja feito para todos os *tokens* passados como entrada, a nova matriz de codificação obtida é passada para o codificador ou decodificador.

A Figura 5 apresenta um exemplo do funcionamento da camada de codificação posicional. Para cada linha da matriz, um vetor com a codificação posicional referente ao *token* é gerado. Cada linha da matriz de codificação é somada ao vetor de codificação posicional equivalente, gerando, ao final, a nova matriz de codificação.

2.2.1.3 Mecanismo de atenção

Em Hu (2019), mecanismo de atenção é definido como um método utilizado para codificar dados de sequências com base na pontuação de importância atribuída a cada elemento. Assim, dada uma sentença, é possível saber quais as palavras mais relevantes para resolver determinado problema.

Figura 5 – Exemplo de saída da camada de codificação posicional



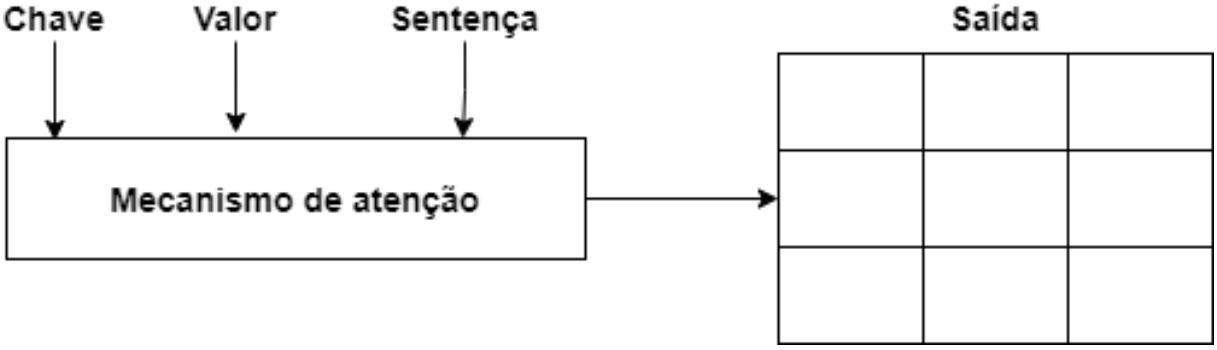
Fonte: Elaborado pelo autor

Formalmente podemos definir um mecanismo de atenção como a função:

$$f : Q \times K \times V \rightarrow Z \tag{2.1}$$

onde Q, K e V são, respectivamente, o conjunto de sentenças, chaves e valores. A saída é calculada como uma soma ponderada dos valores, onde o peso atribuído a cada valor é calculado com base em uma função de compatibilidade da sentença com a chave correspondente (VASWANI *et al.*, 2017). Ao final, o resultado do mecanismo de atenção é a correlação existente entre um conjunto de palavras, como apresentado na Figura 6.

Figura 6 – Mecanismo de atenção

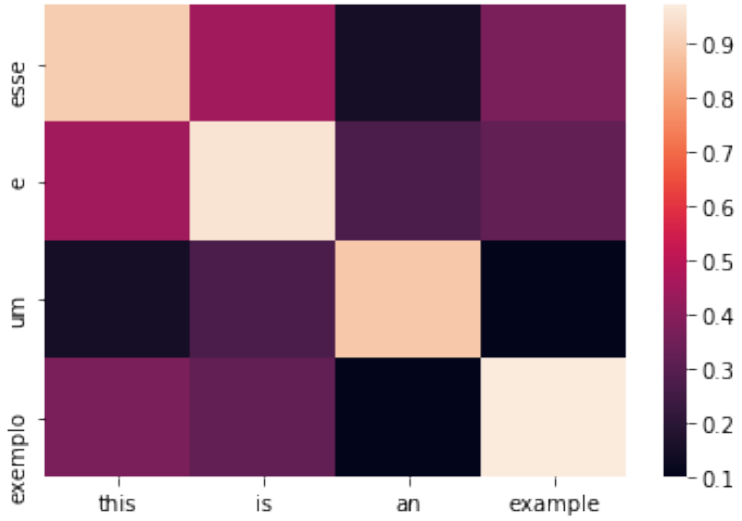


Fonte: Elaborado pelo autor

Na Figura 7 é possível ver um exemplo de matriz de correlação entre palavras. Na horizontal está um texto em inglês e na vertical está o mesmo texto em português. A matriz mostra o valor da correlação entre cada palavra em português com cada palavra em inglês. No exemplo, quanto mais claro a posição da matriz, maior o valor de correlação. Assim é possível saber quais as principais palavras necessárias para entender o contexto de determinada palavra.

O mecanismo de atenção utilizado pelo *transformers* é o produto escalar normalizado (*scaled dot-product*, em inglês) e pode ser visto na Figura 8. Inicialmente são informados

Figura 7 – Matriz de correlação entre palavras.



Fonte: Elaborado pelo autor

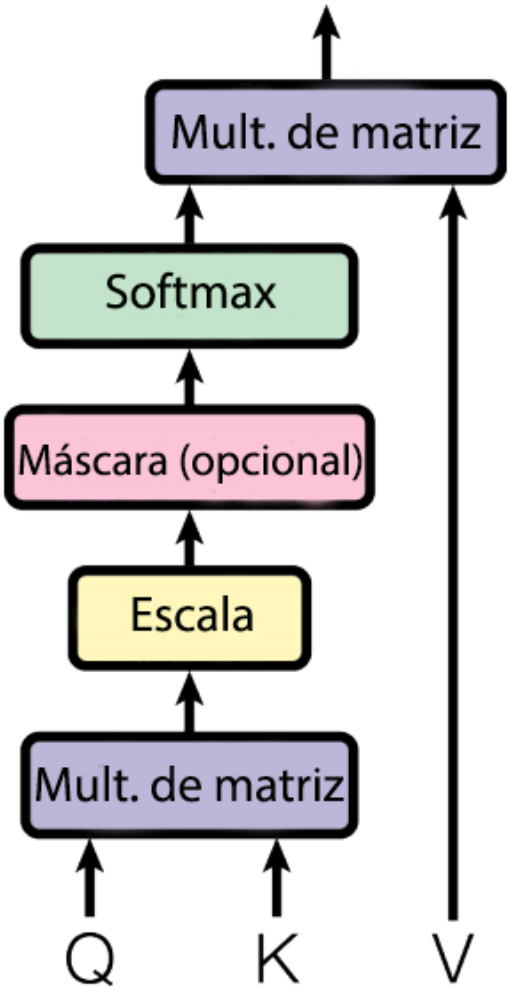
sentenças Q e chaves K, ambos de dimensão d_K, e valores V. Após informado as entradas é feito o produto escalar entre Q e K, o resultado é dividido por $\sqrt{d_K}$, realizado pela etapa escala na Figura 8. Essa etapa visa melhorar a performance para valores altos de d_K, pois sem a normalização os produtos escalares aumentam e fazem com que o *softmax* possua valores de gradientes extremamente pequenos (VASWANI *et al.*, 2017), causando uma perda de performance. Depois é aplicada uma função *softmax* para obter os pesos, sendo que esses pesos ajudam a definir quais palavras necessitam de mais atenção. Por último é feito o produto escalar dos pesos com V (VASWANI *et al.*, 2017). Assim a matriz resultante pode ser definida como:

$$\text{Attention}(Q,K,V) = \text{softmax}(\frac{QK^t}{\sqrt{d_K}})V, \text{ onde } d_K \text{ é a dimensão de } K \tag{2.2}$$

Assim, como a função *softmax* produz valores entre 0 e 1, as palavras mais importantes receberão valores próximos de 1. Isso garante que os valores das palavras principais permaneçam praticamente intactos, enquanto as demais palavras são multiplicados por valores extremamente pequenos, fazendo com que o modelo entenda que não são relevantes.

Para permitir que o máximo de informações possa ser obtido a partir da sentença é utilizado a atenção multi-cabeça. Com esse mecanismo os valores de Q, K e V são divididos em n partes. Em cada uma dessas partes é aplicado o produto escalar normalizado. Ao final o resultado é concatenado, permitindo obter as informações que foram adquiridas em cada aplicação do produto escalar normalizado.

Figura 8 – atenção do produto escalar normalizado



Fonte: (VASWANI *et al.*, 2017).

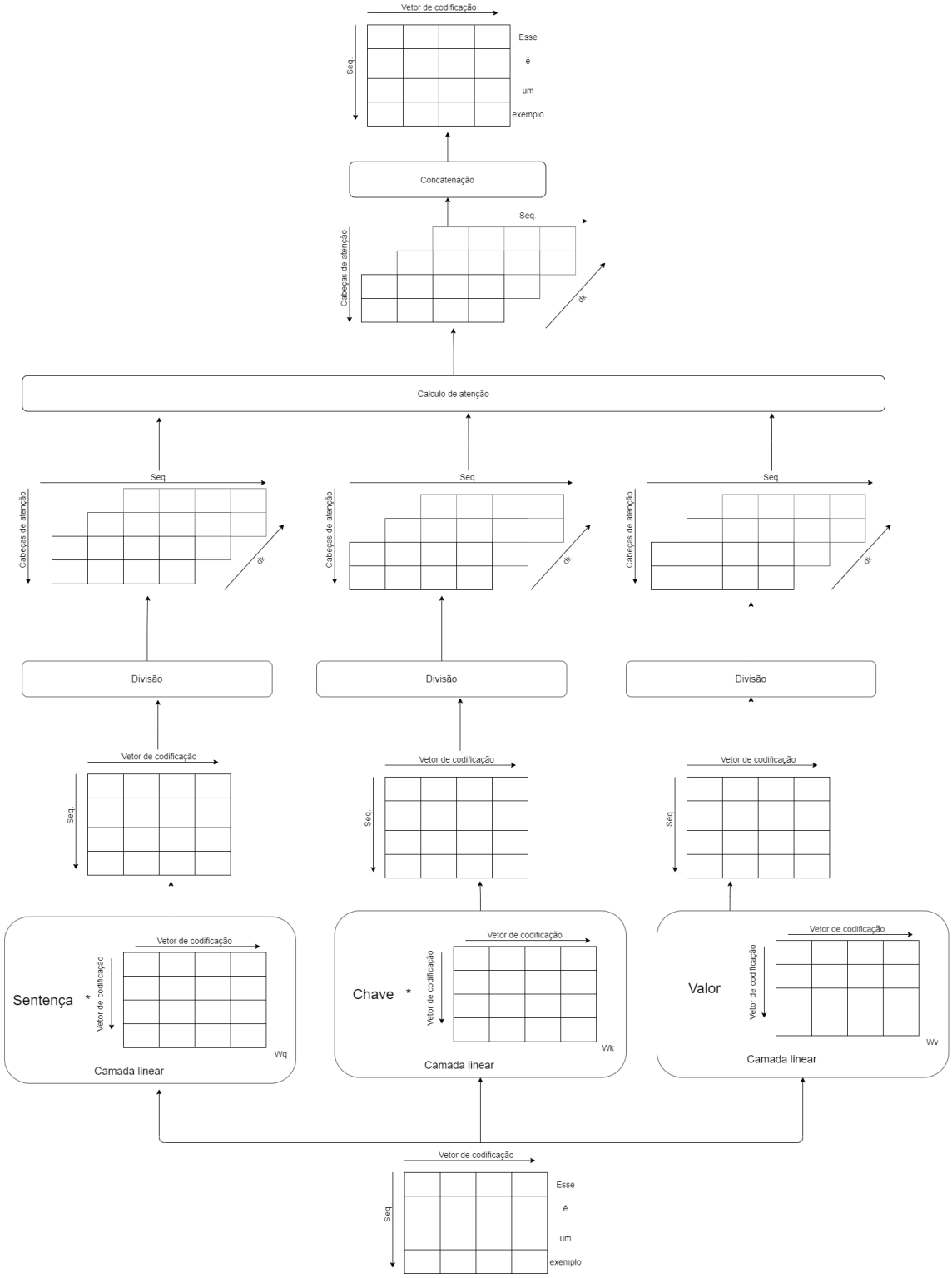
2.2.1.3.1 Atenção multi-cabeça

O cálculo de atenção utilizando o produto escalar normalizado é executado h vezes paralelamente, com cada execução sendo chamado de *cabeça de atenção*. Os valores de Q, K e V são divididos em quantidade equivalente ao número de cabeças de atenção. Cada divisão é passada para uma cabeça de atenção e o cálculo de produto escalar normalizado é realizado.

Para que os valores de Q, K e V sejam obtidos, a matriz de codificação é passada para o mecanismo de atenção. As três matrizes são inicialmente, iguais, porém cada uma dessas matrizes passa por uma camada linear diferente com os pesos QW, KW e VW, respectivamente.

Após os valores de Q, K e V serem gerados, o produto escalar normalizado é aplicado em cada cabeça de atenção e os resultados são concatenados, formando a matriz final. A Figura 9 ilustra o processo do mecanismo de atenção multi-cabeça.

Figura 9 – Processo realizado pelo mecanismo de atenção multi-cabeça



Fonte: Desenvolvido pelos autores.

Como cada cabeça de atenção foca em uma parte dos dados, cada resultado permitirá ao modelo aprender determinados relacionamento e detalhes sobre o comportamento das palavras dentro das frases. Asim o resultado final permite ao modelo obter o máximo de informações sobre uma dada sentença.

2.2.2 BERT

O BERT (DEVLIN *et al.*, 2018) é um modelo pré-treinado baseado nas redes *transformers* e é utilizado no presente trabalho durante a construção do classificador de intenções. O BERT consiste de um modelo treinado com o *BooksCorpus*[1] e a Wikipédia em inglês[2], ignorando listas, tabelas e cabeçalhos, tendo assim utilizado mais de oitocentos milhões de palavras , no total (DEVLIN *et al.*, 2018).

Além de ser treinado com uma grande base de dados, o BERT também permite que seja ajustado (*fine-tune*) facilmente para tarefas especificas (DEVLIN *et al.*, 2018). Segundo os autores, esse modelo alcançou um novo estado da arte em 11 tarefas de NLP, entre elas a de QA, como pode ser visto na Figura 10, que apresenta alguns dos modelo que obtiveram os melhores resultados para a tarefa de QA usando o conjunto de dados *QA SQuAD*. Os modelos são avaliado com as métricas de *exact match* (EM) e F1. É possível observar que alguns dos melhores resultados são obtidos combinando alguma versão do BERT com outros algoritmos, como em Zhang *et al.* (2020) que utilizam ELECTRA e ALBERT (uma versão do BERT).

Figura 10 – Ranking que mostra as soluções que obtiveram os melhores resultados sobre o dataset de QA SQuAD

Rank	Model	EM	F1
	Human Performance *Stanford University* (Rajpurkar & Jia et al. '18)	86.831	89.452
1 Apr 06, 2020	SA-Net on Albert (ensemble) *QIANXIN*	**90.724**	**93.011**
2 May 05, 2020	SA-Net-V2 (ensemble) *QIANXIN*	90.679	92.948
2 Apr 05, 2020	Retro-Reader (ensemble) *Shanghai Jiao Tong University* http://arxiv.org/abs/2001.09694v2	90.578	92.978
3 May 04, 2020	ELECTRA+ALBERT+EntitySpanFocus (ensemble) *SRCB_DML*	90.442	92.839
4 Jun 21, 2020	ELECTRA+ALBERT+EntitySpanFocus (ensemble) *SRCB_DML*	90.420	92.799
5 Mar 12, 2020	ALBERT + DAAF + Verifier (ensemble) *PINGAN Omni-Sinitic*	90.386	92.777

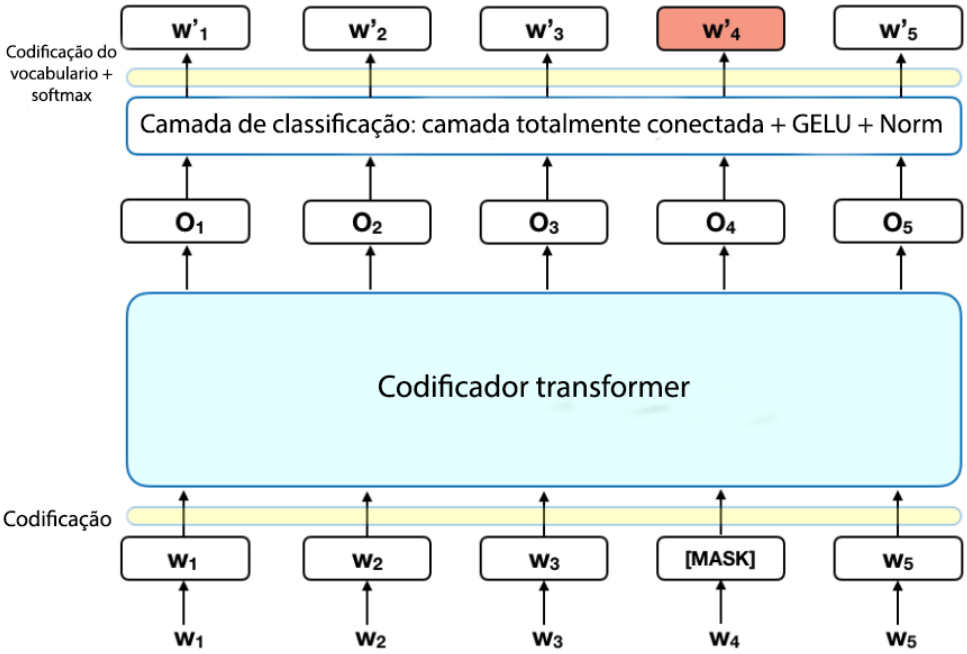
Fonte: (STANFORD, 2020)

[1] Ainda é possível achar cópias nesse dataset online, mas a versão original não está mais disponível.
[2] Disponível em: https://www.english-corpora.org/wiki/

2.2.2.1 *Arquitetura do modelo*

O modelo BERT é baseado em redes *transformer* e utiliza o mecanismo de codificação desse tipo de rede em sua arquitetura (DEVLIN *et al.*, 2018). A arquitetura consiste de 3 camadas: uma para processar as entradas e transforma-las em uma matriz de codificações; uma que possui um conjunto de codificadores *transformers*; e uma camada que permite realizar o ajuste das saídas, o que permite adaptar o BERT para resolver diversos problemas de NLP. A ilustração da arquitetura pode ser vista na Figura 11.

Figura 11 – Arquitetura do modelo BERT.



Fonte: (TOWARDS DATA SCIENCE, 2018)

Existem dois modelos BERT desenvolvidos em Devlin *et al.* (2018), o BERT$_{BASE}$ e o BERT$_{LARGE}$, os quais possuem três diferenças principais que são definidas durante a passagem dos parâmetros. Essas diferenças são o número de camadas (codificadores *transformer*), a dimensão da matriz gerada na etapa de codificação e o número de cabeças de auto atenção, pois como apresentado na Seção 2.2.1.3.1 o mecanismo de atenção consiste na execução e concatenação de *h* mecanismos de atenção de produto escalar, onde cada mecanismo recebe uma projeção linear das entradas.

O BERT$_{BASE}$ possui 12 codificadores *transformers*, uma matriz de codificação com 768 colunas e 12 cabeças de auto atenção. Já o BERT$_{LARGE}$ possui 24 codificadores *transformers*, uma matriz de codificação com 1024 colunas e 16 cabeças de auto atenção (DEVLIN *et al.*,

2018). Cada modelo deve ser escolhido de acordo com o problema a ser resolvido. Com o BERT_{BASE} já é possível obter um bom resultado, porém quando um modelo mais robusto for necessário é possível usar o BERT_{LARGE}.

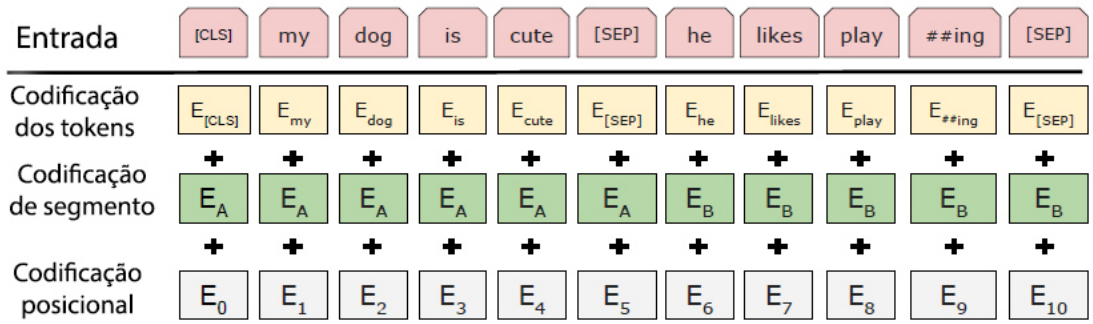
2.2.2.1.1 Entradas e saídas

Para permitir uma maior flexibilização do BERT em diversos problemas de NLP, a entrada é representada de forma que tanto pode ser apenas uma sentença como um par de sentenças.

Assim, a entrada que um modelo BERT recebe é uma sentença **A** e, opcionalmente, uma sentença **B**. Após recebida as entradas elas são tokenizadas e então é calculado a matriz de codificação.

A tokenização é feita utilizando o *wordpiece embedding* com um vocabulário de 30000 *tokens* (DEVLIN *et al.*, 2018). Uma vez que uma sentença é passada cada palavra é transformada em *token* se existe um *token* equivalente dentre os 30000 especificados. Caso não seja encontrada, é verificado se parte da palavra está do vocabulário, podendo dividir a palavra em vários *tokens*. Um exemplo pode ser visto na Figura 12. Considere apenas a frase "*he was playing*", perceba que *playing* é dividido em dois tokens: "*play*" e "##ing". Isso ocorre porque "*playing*" não faz parte do vocabulário, mas "*play*" faz.

Figura 12 – Entradas do modelo BERT.



Fonte: (DEVLIN *et al.*, 2018)

Além das palavras presentes da sentença, dois *tokens* especiais também são adicionados durante a tokenização, sendo representados por [CLS] e [SEP] na Figura 12. O *token* [CLS] é adicionado para tarefas de classificação e o [SEP] é adicionado ao fim de cada sentença, para permitir saber se determinado *token* é da sentença A ou B.

Uma vez feita a tokenização é necessário gerar a matriz de codificação. Para cada

token é adicionado um valor único que visa identificar esse *token*. A matriz com os valores dos *tokens* é chamada matriz de codificação.

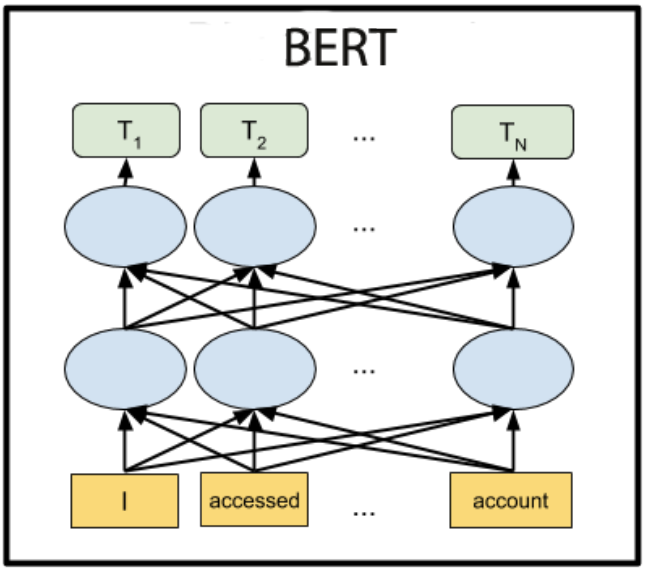
Uma vez gerada a matriz de codificação dos *tokens*, a representação da entrada é feita somando a matriz com dois novos valores, como pode ser visto na Figura 12, onde cada valor de codificação é representado por **E**. Um dos valores é usado para indicar a qual sentença o *token* pertence (sentença A ou sentença B), chamados de *segment embeddings* , enquanto o outro valor é usado para representar sua posição (semelhante a codificação posicional das redes *transformers*).

2.2.2.2 Codificador

Após gerada a matriz de codificação, esta é passada para o codificador. O codificador é uma pilha de codificadores como aqueles apresentados nas redes *transformers* (porém alguns parâmetros diferem dos utilizados em Vaswani *et al.* (2017)) (DEVLIN *et al.*, 2018).

Apesar de utilizar os mesmos codificadores que o *transformer*, o BERT os implementa com uma representação bidirecional. Isso significa que diferente do codificador do *transformer* que processa as entradas da esquerda para a direita, o BERT processa nos dois sentidos, como pode ser visto na Figura 13. Isso permite que o modelo aprenda mais sobre as relações presente das sentenças.

Figura 13 – Arquitetura do BERT



Fonte: (DEVLIN *et al.*, 2018)

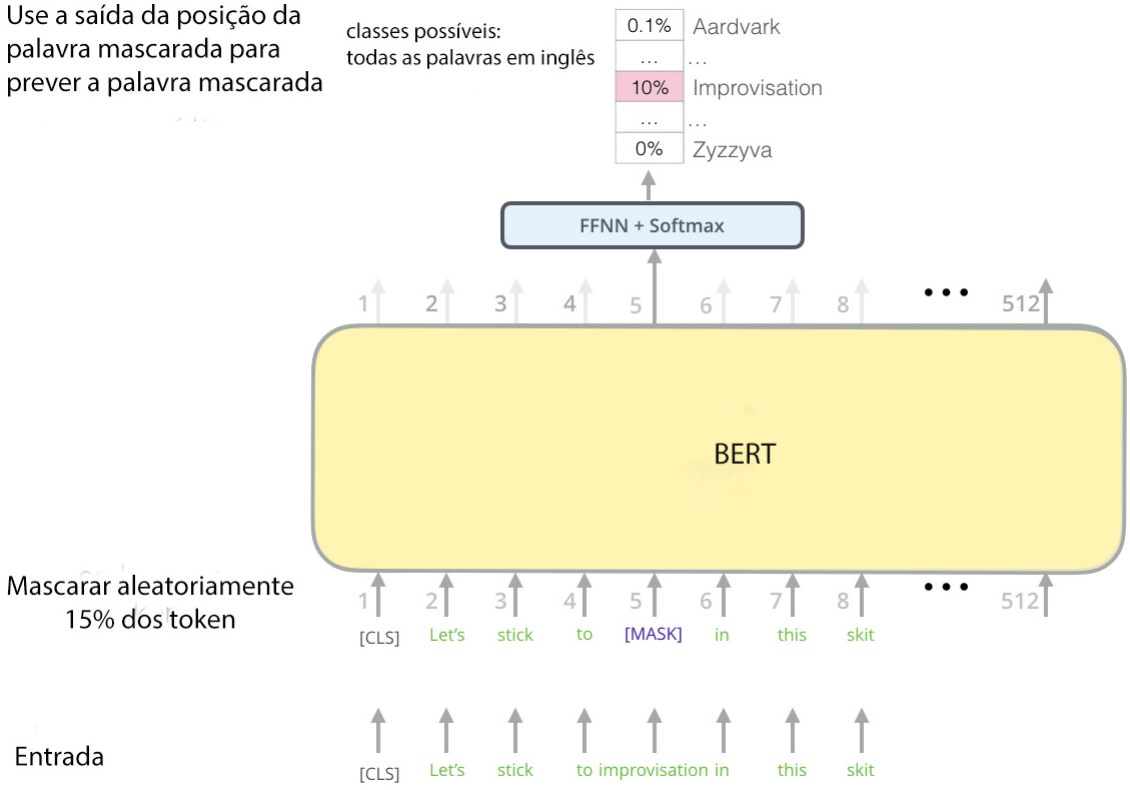
2.2.2.3 *Pré-treinamento*

O pré-treinamento do BERT foi feito utilizando dados da Wikipédia inglesa e utilizando o *dataset BooksCorpus*, que contém textos de livros de diversas áreas (DEVLIN *et al.*, 2018). Assim foram utilizados cerca de oitocentos e três milhões de palavras durante o pré-treinamento.

Durante o pré-treinamento, o modelo é treinado para executar duas tarefas: o modelo de linguagem mascarada e a predição de próxima sentença, que são representados por *masked LM* e NSP, respectivamente, na Figura 15.

A tarefa de modelo de linguagem mascarada consiste em tentar prever qual o valor do *token* que está mascarado. Para isso 15% dos dados de entrada são mascarados e o modelo tenta prever qual a palavra. Voltando para o exemplo apresentado na Figura 12, suponha que a frase passada é "*my dog is cute*". A palavra *dog* pode ser mascarada e então o modelo irá avaliar os demais *tokens*: *my*, *is* e *cute* e tentar prever qual a palavra que deveria vir depois de *my*, retornando ao final um vetor com as possíveis palavras e a probabilidade de cada uma ser a correta.

Figura 14 – exemplo de modelo de linguagem mascarada

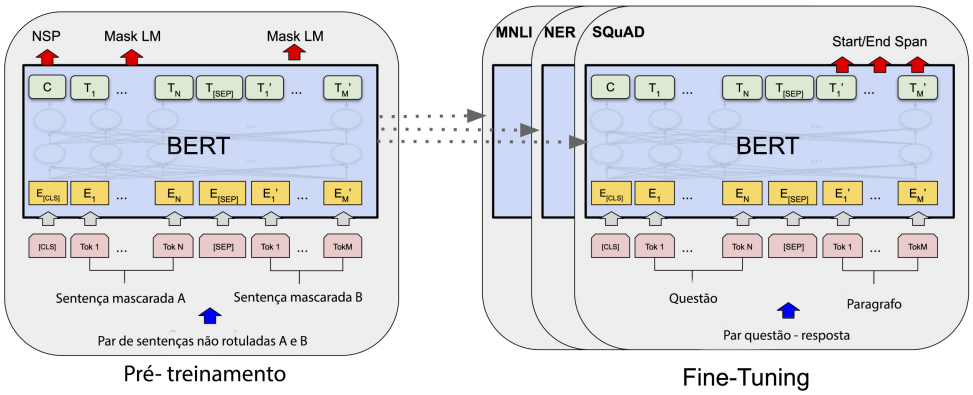


Fonte: (JUMUTC, 2019)

Na Figura 14 é possível visualizar outro exemplo, com a frase "*Let's stick to improvisation in this skit*". No exemplo, a palavra *improvisation* é mascarada e o modelo tentar prever essa palavra. Após a sentença passar pelo modelo é gerado um vetor com as possíveis palavras e probabilidades de cada uma. Entre as palavras resultantes está a palavra *improvisation* com 10% de chance de ser a correta. Assim o modelo consegue prever corretamente, já que *improvisation* possui probabilidade maior que as demais palavras.

Já a tarefa de predição de próxima sentença consiste em tentar prever se a sentença B é a sentença que vem após a sentença A em um texto (DEVLIN *et al.*, 2018). Para essa tarefa 50% das sentenças passadas para o modelo estavam relacionadas com a sentença B e 50% não estavam. Essa tarefa é bastante útil para tarefas de QA e tarefas de NLI, uma vez que permite ao modelo aprender a relação entre as sentenças.

Figura 15 – Pré-treinamento e *fine-tuning* do modelo BERT.



Fonte: (TOWARDS DATA SCIENCE, 2018)

2.2.2.4 Fine-tuning

O *fine-tuning* do modelo BERT, mostrado do lado direito da Figura 15, consiste em ajustar as entradas e saídas para o problema que deseja resolver (DEVLIN *et al.*, 2018).

Por exemplo, para o problema de QA é possível colocar a sentença A como a questão e a sentença B como o texto em que a resposta deve ser procurada. Para o problema de NLI, a sentença A pode ser definida como a premissa e a sentença B como a hipótese. Uma vez definidas as entradas, basta definir como é a última camada responsável por retornar a saída. Essa camada poderia ser alterada, para a tarefa de NLI para prever se a sentença B pode ser inferida a partir da sentença A.

2.2.2.5 BERTimbau

A primeira versão do BERT foi desenvolvida visando o inglês, utilizando para treino e avaliação textos desse idioma. Outras versões do BERT foram desenvolvidas com o passar do tempo, tendo outros idiomas como foco, como o BERTje (VRIES *et al.*, 2019) e o BERT para espanhol (CANETE *et al.*, 2020).

O BERTimbau (SOUZA *et al.*, 2020) é uma versão do BERT desenvolvida para lidar com textos em português brasileiro. Assim como o BERT original, essa versão possui as variantes *base* e *large* e permite que o modelo seja ajustado facilmente para diversas tarefas. Como o foco desse trabalho são textos em português, o BERTimbau foi um dos modelos escolhidos para o desenvolvimento do classificador de intenções, visando avaliar as bases de dados geradas.

2.3 Geração de dados sintéticos

Conforme o conceito de *deep learning* foi se popularizando, a dificuldade em encontrar bases de dados para criação de modelos foi ficando mais evidente (NIKOLENKO, 2019). Assim surge a necessidade de criação de bases de dados, podendo ou não utilizar dados reais.

Desenvolver bases de dados é uma tarefa difícil e demorada, tendo uma complexidade ainda maior quando os dados precisam ser rotulados (NIKOLENKO, 2019). Assim surgiram diferente sistemas visando semi-automatizar a criação de dados, tais como o *synner* (MANNINO; ABOUZIED, 2020) e o *Chatette* [3], uma das ferramentas que será utilizado neste trabalho para geração de dados sintéticos.

Esses sistemas permitem que dados sejam gerados a partir de *templates* e especificações definidas pelo desenvolvedor responsável por criar a base de dados. Então cabe ao desenvolvedor pensar da melhor forma de criar a base de dados e garantir que os dados sintéticos representem os dados reais.

Segundo Mannino e Abouzied (2019) quando desenvolvendo uma base de dados é importante garantir que os dados gerados representem os dados do mundo real. É necessário termos cuidado para que inserção de erros e ruídos não tenham um efeito negativo e garantir que conforme a base de dados seja redimensionada suas propriedades permaneçam inalteradas. Isso se torna um verdadeiro desafio quando um modelo de geração de texto é utilizado, visando gerar

[3] Disponivel em: https://github.com/SimGus/Chatette

os dados de forma automática.

A geração de dados de forma automática pode também ser feita através de modelos de *deep learning*, como GAN (*Generative Adversarial Network*) e T5 (NIKOLENKO, 2019). Esses métodos são comumente utilizados para aumentar uma base de dados inicial, como em Bird *et al.* (2020) e em Amin-Nejad *et al.* (2020).

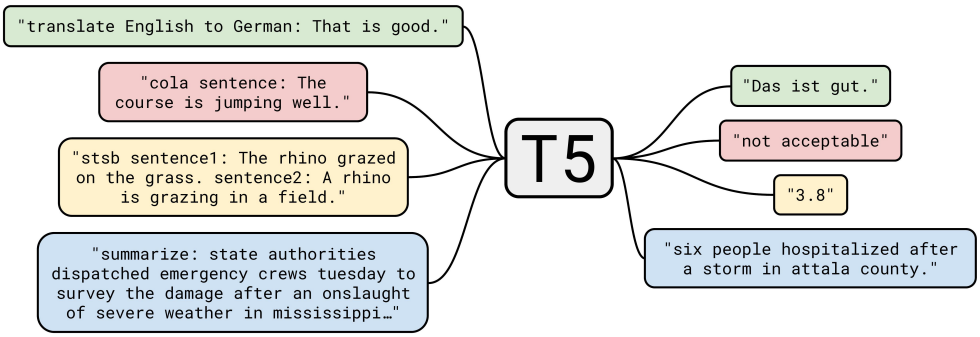
A próxima seção irá detalhar o T5 e o *Chatette* e como é possível criar conjunto de dados utilizando essas ferramentas.

2.3.1 Text-To-Text Transfer Transformer (T5) e PTT5

Assim como o BERT, o *Text-To-Text Transfer Transformer* (T5) é um modelo pré-treinado baseado em *transformers*, proposto em Raffel *et al.* (2019), tendo obtido resultados estado da arte em várias tarefas de NLP, como sumarização (*summarization*), que consiste em criar uma versão reduzida de um texto passado como entrada, e QA. Esse modelo utiliza uma arquitetura bastante semelhante a apresentada na Figura 3, apenas alterando a codificação posicional senoidal do *transformers* por uma codificação posicional relativa e mudando a posição da camada de normalização.

Assim como o *transformers*, esse modelo recebe e gera texto. Assim sua entrada consiste de um *label* indicando a tarefa e o texto de entrada, no formato "tarefa: texto", tendo como saída outro texto, como mostrado na Figura 16.

Figura 16 – Funcionamento do T5.



Fonte: (RAFFEL *et al.*, 2019)

Então para que seja possível fazer o ajuste do modelo é necessário apenas um conjunto de dados com a entrada formatada como esperada pelo modelo e tendo um texto como saída esperada.

Assim como o BERT, também foi desenvolvido uma versão em português do T5.

Apresentado em Carmo *et al.* (2020) e nomeado de PTT5, o modelo foi pré-treinado com dados em português e permite que seja ajustado para tarefas de NLP que utilizem textos nesse idioma.

2.3.2 Chatette

O *Chatette* é uma ferramenta desenvolvida em Python que permite a criação de diversos exemplos a partir de um único *template* definido pelo usuário.

Figura 17 – Exemplo de arquivo Chatette.

```
%[&ask_toilet](3)
    ~[sorry?] ~[tell me] where the @[toilet#singular] is [please?]?
    ~[sorry?] ~[tell me] where the @[toilet#plural] are [please?]?

~[sorry]
    sorry
    excuse me

~[tell me]
    ~[can you?] [tell|show] me
~[can you]
    [can|could|would] you

@[toilet#singular]
    toilet
    loo
@[toilet#plural]
    toilets
```

Fonte: https://github.com/SimGus/Chatette

Como ilustrado na Figura 17, podemos definir uma entidade através da sintaxe "`@[label]`", seguido dos valores que a entidade pode assumir, como por exemplo a entidade "`toilet#singular`". Trechos de textos são representados por "`~[label]`" seguida da lista de textos que são representados por essa *label*. Por último temos a representação das sentenças que serão geradas, "`%[intenção]`", que combina textos com os trecho e entidades já definidas. É a partir da representação de sentenças que é gerado exemplos como o da Figura 18, contendo texto, lista de entidades e intenção.

Figura 18 – Exemplo de resultado gerado pelo Chatette.


```
"rasa_nlu_data": {
  "common_examples": [
    {
      "entities": [
        {
          "end": 45,
          "entity": "toilet",
          "start": 38,
          "value": "toilets"
        }
      ],
      "intent": "ask_toilet",
      "text": "Excuse me could you show me where the toilets are please?"
    },
```


Fonte: https://github.com/SimGus/Chatette

2.4 Métricas

Essa seção se dedica a apresentar as métricas que são utilizadas ao longo do presente trabalho, detalhando as métricas utilizadas para avaliar textos gerados por um modelo de aprendizagem de máquina e as métricas utilizadas para avaliar modelos classificatórios.

2.4.1 Métricas para avaliar geração de textos

Existe diversas métricas para avaliar textos gerados por um modelo de aprendizagem de máquina. Essa seção irá focar em três métricas especificas, sendo elas:

- BLEU: Proposto inicialmente em Papineni *et al.* (2002) como uma métrica para avaliar modelos de tradução, atualmente é utilizada para avaliar textos gerados por modelos em geral. O cálculo é apresentado na equação 2.3. Essa métrica é utilizada para avaliar a qualidade dos dados gerados, onde quanto maior o valor mais próximo a resposta gerada está da resposta esperada.

$$BLEU = \frac{\sum_{C \in \{\text{possíveis respostas}\}} \sum_{\text{n-grama} \in C} Count_{clip}(\text{n-grama})}{\sum_{C' \in \{\text{possíveis respostas}\}} \sum_{\text{n-grama'} \in C'} Count(\text{n-grama'})} \tag{2.3}$$

  Onde $count(\text{n-grama})$ é o número de vezes que um n-grama ocorre em uma possível resposta e $count_{clip}(\text{n-grama})$ é o mínimo entre o resultado de *count* e o maior valor de número de vezes que um n-grama ocorre em uma das reposta;

- F1: Esta métrica mede a diferença média entre a previsão e a resposta correta (RAJPURKAR *et al.*, 2016). O cálculo é apresentado na equação 2.4. Quanto mais próximo de 1 melhor o valor de F1.

$$F1 = \frac{2 \times \frac{tp}{tp+fp} \times \frac{tp}{tp+fn}}{\frac{tp}{tp+fp} + \frac{tp}{tp+fn}} \tag{2.4}$$

Onde tp é a quantidade de *tokens* que estão presentes tanto na sentença gerada quanto na sentença esperada, fp é o número de *tokens* presente no texto gerado e fn é o número de *tokens* na sentença esperada;

- *exact math*: Calcula a porcentagem de respostas que são idênticas as repostas esperadas (RAJPURKAR *et al.*, 2016). O cálculo é apresentado na equação 2.5.

$$exact\_match = \frac{\text{quantidade de sentenças preditas idênticas as respostas esperadas}}{\text{quantidade de respostas total}} \tag{2.5}$$

2.4.2 Métricas para avaliar modelos classificatórios

Para avaliação de modelos classificatórios, as principais métricas são acurácia, precisão, F1 e revocação, apresentados, respectivamente nas equações 2.6, 2.7, 2.9 e 2.8. Onde, para uma intenção, tp são os casos em que o texto foi classificado corretamente como essa intenção, fp são os casos em que o texto foi classificado incorretamente como essa intenção e fn são os casos em que o texto foi classificado incorretamente como não sendo essa intenção.

$$acuracia = \frac{Quantidade\ de\ respostas\ preditas\ corretamente}{Quantidade\ de\ respostas\ totais} \tag{2.6}$$

$$precisao = \frac{tp}{tp+fp} \tag{2.7}$$

$$revocacao = \frac{tp}{tp+fn} \tag{2.8}$$

$$F1 = \frac{2 \times precisao \times revocacao}{precisao + revocacao} \tag{2.9}$$

A parir da acurácia temos quantas amostras foram preditas corretamente no total, enquanto que a precisão e a revocação nos permite analisar se o modelo tende a predizer corretamente mais uma classe que outra. Já o F1 é uma combinação de precisão e revocação. Assim, quanto maior o valor de precisão, revocação e F1, maior a garantia de que o modelo não está classificando corretamente apenas determinadas classes, mas que de fato está classificando corretamente para a maioria das classes presentes na base.

3 TRABALHOS RELACIONADOS

Nesta seção são apresentados trabalhos relacionados, descrevendo seus problemas e soluções.

3.1 Exploring transformer text generation for Medical dataset augmentation

Em Amin-Nejad *et al.* (2020), visando suprir a falta de dados acessíveis no domínio médico, os autores propõem duas abordagens para geração de dados utilizando arquiteturas baseadas em *transformers*. A primeiro arquitetura é o próprio *transformers*, que obteve resultado estado da arte na tarefa de tradução. O segundo é o GPT-2, que obteve resultados estado da arte em tarefas de QA e modelagem de linguagem.

Para fins de avaliação dos dados gerados, eles são utilizados nas tarefas de predição de readmissão não planejada e classificação de fenótipo. Em cada tarefa dois experimentos são realizados, um experimento com base totalmente sintética e outro em que os dados sintéticos são utilizados em conjunto com os dados originais, tendo como objetivo apenas incrementar a quantidade de dados do *dataset* original.

Os dados originais são provenientes da base de dados MIMIC-III (JOHNSON *et al.*, 2016), que possui uma boa variedade de dados de domínio médico. Para a geração de dados são utilizadas 55040 amostras, sendo divididas em 80% para treino, 10% para teste e 10% para validação. Os dados de treino são, então, divididos em dois *datasets*, um de poucos recursos chamado de MimicText-9 e outro com todos os recursos chamado de MimicText-98.

Para que os textos sejam gerados, o problema é tratado como uma tarefa de geração condicional, onde é passada uma série de informações chaves para o modelo (tais como dados demográficos, diagnósticos e procedimentos) e um texto é gerado.

A partir dos dados gerados as tarefas de predição de readmissão não planejada e classificação de fenótipo são avaliadas utilizando BERT e BioBERT (LEE *et al.*, 2020) em 3 cenários: com os dados originais, com duas vezes os dados originais (aumento do *dataset* com uma cópia dos dados existentes) e com a combinação dos dados originais com os dados gerados pelo modelo. Em cada experimento são coletados dados de acurácia, F1, precisão, *recall* e área sob a curva (area under the curve, em inglês ou AUC), que avalia a habilidade do modelo de distinguir entre as classes.

A partir dos resultados obtidos, os autores concluem que os dados gerados melhoram

a performance do bioBERT que é um modelo treinado com dados de domínio médico, mas não influenciam tanto no desempenho do BERT padrão.

3.2 Generation and evaluation of artificial mental health records for Natural Language Processing

Em Ive *et al.* (2020), visando suprir a falta de dados accessíveis no domínio médico, os autores propõem e implementam a construção de um modelo capaz de gerar dados. Os dados gerados são utilizados como um conjunto de teste para um modelo construído com dados reais, utilizando o resultado obtido para avaliar o *dataset*.

Os dados artificiais são produzidos a partir de dados reais. Para gerar os novos dados é utilizado o *dataset* Mimic-III e o *dataset* CRIS MHR, que contém resumos de altas de pacientes anônimos.

Dado um parágrafo real, palavras chaves são extraídas e combinadas com um conjunto de informações médicas (diagnóstico, idade e gênero), sendo então passadas para um modelo *transformer* que gerará o texto sintético.

Assim, são considerados três experimentos para a geração de dados, sendo eles: frases geradas com todas as palavras chaves extraídas e sem informações médicas (*all*); frases geradas com informações médicas e a melhor palavra-chave extraída (*one+meta*); frases geradas por um conjunto das melhores palavras chaves com informações médicas (*top+meta*).

A qualidade dos dados gerados é avaliada manualmente, verificando se os dados textuais gerados fazem sentido, se são relevantes, se são compreensíveis, se não contradiz o diagnóstico e se não é apenas uma cópia dos textos de treino.

Os dados gerados são então avaliados em tarefas de classificação utilizando os modelos *bag-of-words* (BoW), *Latent Dirichlet Allocation* (LDA) e rede neural convolucional (CNN), sendo analisado os valores de F1 obtidos.

A partir dos resultados os autores concluíram que o desempenho não altera significativamente quando os dados sintéticos são utilizados e que, tanto com os dados originais quanto com os dados sintéticos, o CNN continua como o melhor modelo entre os três utilizados.

3.3 Chatbot interaction with artificial intelligence: human data augmentation with T5 and language transformer Ensemble for Text classification

Em Bird *et al.* (2020), os autores desenvolvem um sistema de *chatbot* que executa tarefas a partir do texto recebido, sendo nomeado por eles de *Chatbot Interaction with Artificial Intelligence* (CI-AI).

O objetivo nesse *chatbot* é que a partir de um texto informado pelo usuário (seja por teclado, voz convertido para texto ou linguagens de sinal convertido para texto), seja identificada qual a tarefa esperada e a mesma seja executada.

Para o treinamento nesse *chatbot* é utilizado um *dataset* com 487 amostras e um que combina essas 487 amostras com amostras geradas por um T5. As amostras são divididas em 7 tarefas: reconhecimento de cena, classificação EEG, análise de sentimento, reconhecimento de linguagem de sinais, IA de conversação e gerador de piadas. Assim a tarefa do *chatbot* é classificar o texto recebido em uma destas 7 tarefas.

Para gerar as novas amostras é utilizado um modelo T5 treinado na tarefa de parafrasear, ou seja, a partir de um texto recebido são geradas novas formas de escrever esse texto. Com os novos dados gerados é obtido um *dataset* com um total de 13090 amostras.

Para a avaliação dos dados são utilizados 7 modelos para classificação, sendo eles: BERT; DistilBERT; DistilRoBERTa; RoBERTa; XLM; XLM-RoBERTa e XLNet. Os modelos são treinados apenas com os dados originais e treinados com a combinação dos dados originais com os gerados pelo T5.

A partir dos resultados os autores concluem que todos os modelos obtiveram uma melhora de desempenho, sendo o RoBERTa o modelo que obteve os melhores resultados.

3.4 Tabela comparativa

A Tabela 1 apresenta um comparativo deste trabalho com os trabalhos relacionados descritos. O presente trabalho visa resolver um problema semelhante ao abordado em Bird *et al.* (2020), focando em geração de dados para treinamento de *chatbots*, porém com técnicas e ferramentas diferentes, utilizando o BERTimbau para classificação e com um conjunto de dados diferente. O objetivo nesse trabalho é semelhante ao de Amin-Nejad *et al.* (2020) e de Ive *et al.* (2020), onde é necessário a criação de um conjunto de dados de qualidade, considerando a escassez de dados existente para o domínio, definido no domínio da aplicação.

Tabela 1 – Tabela comparativa

-	(AMIN-NEJAD *et al.*, 2020)	(IVE *et al.*, 2020)	(BIRD *et al.*, 2020)	Trabalho proposto
Abordagem	Utilização de *transformers* e GPT-2	Utilização de *transformers*	Utilização de T5 e modelos de classificação	Utilização de Rasa, BERT, PTT5 e chattete
Base de dados	Mimic-III	Mimic-III e CRIS MHR	*Dataset* desenvolvido pelos autores	*Dataset* desenvolvido pelos autores
Objetivo	Criar modelos de classificação com dados gerados artificialmente	Criar modelos de classificação com dados gerados artificialmente	Desenvolvimento de um conjunto de dados de qualidade para desenvolvimento de um *chatbot*	Desenvolvimento de um conjunto de dados de qualidade para desenvolvimento de um *chatbot*

Fonte: Elaborado pelo autor.

4 DESENVOLVIMENTO

O desenvolvimento deste trabalho foi dividido em três etapas. A primeira etapa foi a criação das bases de dados artificiais de maneira manual, sem auxílio de modelos de inteligência artificial para geração de texto. A segunda etapa foi a criação de uma base de dados artificiais gerada por um modelo de inteligência artificial. A última etapa foi a realização de experimentos com diferentes modelos para classificação de intenções visando analisar a influência das bases de dados geradas nesses modelos.

4.1 Criação das bases de dados baseada em modelos de perguntas

O domínio utilizado para a criação das bases de dados foi a carta de serviços do cidadão[1]. A carta de serviços tem como objetivo informar o usuário sobre os serviços prestados pelo estado, as formas de acesso a esses serviços e seus compromissos e padrões de qualidade de atendimento ao público. Por isso ela consiste de um conjunto de informações para cada serviço prestado pelo estado.

A criação das primeiras bases de dados foi baseada em modelos de perguntas. Esse tipo de método de criação de bases possui um baixo custo e comumente utilizado durante as etapas iniciais do processo de criação de bases de dados. Nele é possível que o desenvolvedor defina um modelo de perguntas, combinando trechos de texto com valores de entidades, resultando em um conjunto de exemplos de texto relacionados a determinadas intenções.

Considerando que a tarefa utilizada nesse trabalho é a classificação de intenções, as bases de dados geradas necessitam de um conjunto de exemplos, onde cada exemplo é composto por um rótulo, indicando a intenção, um texto, representando uma questão relacionada a intenção e informações sobre as entidades contidas no texto. O *chatette*, ferramenta para geração de dados para o Rasa apresentado na Seção 2.3.2, foi utilizado para a criação das bases.

A carta de serviços é gerida pela Controladoria e Ouvidoria Geral do Estado e possui, atualmente, serviços ofertados por 68 órgãos e entidades públicas. A partir do site é possível acessar os serviços por categoria, assim como o conjunto de serviços ofertados por determinado órgão. A partir dos dados da carta de serviços é possível saber informações sobre o órgão prestador do serviço, formas de acesso, custo, requisitos e documentos necessários para a solicitação do serviço, principais etapas e informações sobre o atendimento.

[1] Disponível em: https://cartadeservicos.ce.gov.br/ConsultaCesec/pg_cs_servico.aspx

Dentre o conjunto de informações contidas na carta de serviço, quatro foram selecionadas para a criação da base de dados desse trabalho: documentos necessários, o que é o serviço, horários de atendimento e onde fazer de forma presencial. A partir das informações selecionadas, foram gerados nove tipos de perguntas, onde cada tipo de pergunta é definido como sendo uma intenção. Para cada intenção foi definido um rótulo, com o intuito de identificar qual a intenção de cada pergunta. As intenções geradas são as apresentadas na Tabela 2.

Tabela 2 – Possíveis intenções das perguntas geradas

Rótulo	Descrição	Exemplo
dsc_servico	O que é o serviço	O que é audiometria?
dsc_local_presencial	Onde o serviço pode ser realizado de forma presencial	Onde audiometria está disponível?
dsc_local_presencial_objetivo	Se o serviço está disponível em uma determinada cidade	Audiometria pode ser realizado presencialmente em Fortaleza?
dsc_documentos	Quais os documentos requeridos pelo serviço	Quais os documentos necessários para audiometria?
dsc_documentos_objetivo	Se o serviço requer um determinado documento	RG é necessário para audiometria?
dsc_documentos_estado_civil	Quais os documentos requeridos pelo serviço para determinado estado civil	Quais os documentos necessários para audiometria, quando se é casada?
dsc_documentos_estado_civil_objetivo	Se o serviço requer um determinado documento para determinado estado civil	RG é necessário para audiometria, quando se é solteiro?
dsc_horarios_funcionamento	Horário de funcionamento do serviço	Em quais horários posso realizar audiometria?
dsc_horarios_funcionamento_objetivo	Se o serviço está disponível em determinado dia e horário	Posso realiza audiometria na segunda às 14h?

Fonte: Desenvolvido pelos autores

Cada texto contém um conjunto de entidades, onde, na base de dados, existem informações sobre essas entidades, tais como rótulo e posicionamento. A Tabela 3, apresenta os tipos de entidades e alguns exemplos de cada tipo. Para cada tipo de entidade foi definido um conjunto de valores, consistindo de termos diferentes e de sinônimos de alguns termos já definidos, como apresentado na tabela para a entidade documento.

Tabela 3 – Tipo de entidades definidas

Rótulo	Exemplo
servico	obter cnh; solicitação de RG; emissão de licenciamento anual (CRLV).
documento	RG; Carteira de identidade; certidão de casamento.
estado_civil	solteiro; casada.
dia	segunda; quarta-feira; sábado.
horario	13h; 16:30; 9 da manhã.
cidade	Fortaleza; Canindé; Quixadá.

Fonte: Desenvolvido pelos autores

Definido as intenções e entidades, a próxima etapa foi a construção de modelos de perguntas com o *chatette* para gerar as bases de dados. O modelo para o *chatette* segue uma estrutura semelhante ao apresentado na Figura 19, para cada intenção. "`%[&dsc_servico]`" define os exemplos que serão gerados para a intenção dsc_servico. "`@[servico]`" define possíveis valores que a entidade serviço pode assumir. Já "`~[dsc_servico_questao_inicio]`" define trechos de texto que podem ser utilizados durante a geração dos textos. Como "`%[&dsc_servico]`" é uma combinação de "`~[dsc_servico_questao_inicio]`" e "`@[servico]`", é gerado uma nova frase para cada valor de" dsc_servico_questao_inicio" e para cada valor de "servico". Ao final, o modelo *chatette* gera um arquivo JSON com um conjunto de exemplos seguindo o formato apresentado na Figura 20. O modelo completo utilizado para geração das bases pode ser visto no apêndice A.

Visando obter um conjunto de dados de qualidade e mais próximo de dados reais nos atentamos quanto a capitalização das palavras, o posicionamento das entidades dentro do texto e pontuação[2]. Considerando a capitalização das palavras, os textos comumente começam com letras maiúscula e possuem siglas também em maiúsculas. Como as frases geradas não são muito longas, o foco quanto a pontuação diz respeito ao fim das frases, normalmente terminando com ponto final ou interrogação. Já a variação das posições das entidades foi feita gerando frases onde as entidades aparecem no início, no meio ou no fim do texto.

As bases de dados resultantes são descritas na Tabela 4, sendo essas bases de dados as que foram utilizadas nos experimentos que são detalhados na Seção 4.4.

Mais dados poderiam ter sido gerados a partir do *chatette*, porém isso poderia causar um *overfitting* nos modelos, dado que mais frases semelhantes já existentes seriam geradas. Além disso, por algumas estruturas possuírem uma maior quantidade de entidades, não impor um

[2] Experimentos iniciais que não levaram esses detalhes em consideração resultaram em bases de dados simples e redução da performance dos modelos para a tarefa avaliada.

Figura 19 – Exemplo de modelo para geração de dados com *chatette*.

```
%[&dsc_servico]
    ~[dsc_servico_questao_inicio] @[servico]?

~[dsc_servico_questao_inicio]
    O que é
    O que significa
    Do que se trata

@[servico]
    emissão de certidão nada consta
    solicitação de carteira de habilitação popular (CNH popular)
    emissão de certidão
    solicitar cnh
    solicitação de carteira de habilitação definitiva (CNH definitiva)
    carteira de habilitação definitiva
    cnh final
    carteira permanente
    emissão de atestado de antecendentes criminais e identificação criminal
    identificação criminal
    atestado de antecedentes criminais
    antecedentes criminais
    emissão licenciamento anual (CRLV)
    solicitação carteira de identidade civil
    solicitação de RG
    solicitação de carteira de identidade (RG)
```

Fonte: Desenvolvido pelos autores

Figura 20 – Estrutura dos dados contidos nas bases de dados.


```
{
  "entities": [
    {
      "end": 25,
      "entity": "servico",
      "start": 8,
      "value": "solicitação de RG"
    }
  ],
  "intent": "dsc_servico",
  "text": "o que é solicitação de RG?"
},
```


Fonte: Desenvolvido pelos autores

limite ao número de textos gerados levaria a geração de mais textos para intenções especificas, como *dsc_documentos_objetivo* e *dsc_documentos_estado_civil_objetivo*, deixando a base desbalanceada.

Tabela 4 – Bases de dados criadas

Base de dados	Descrição	número de amostras	número de intenções	número de entidades
base_treino_maior_variacao	*Dataset* criado com *chatette*, onde todos os valores da entidade "servico" foram utilizando e contem textos onde as entidades aparecem no início, meio ou fim.	5324	9	128
base_treino_menor_variacao	*Dataset* criado com *chatette*, onde apenas alguns valores da entidade "servico" foram utilizando (foi excluído aleatoriamente alguns valores que eram sinônimos de termos já utilizados) e contem textos onde as entidades aparecem no início ou fim.	4844	9	102

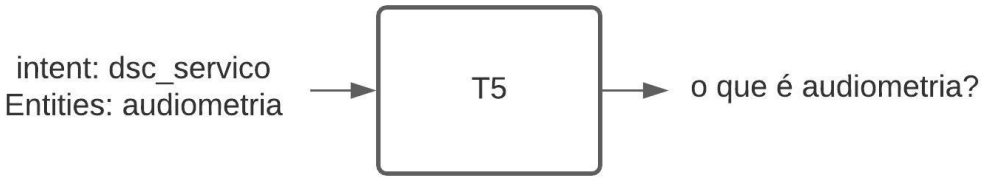
Fonte: Desenvolvido pelos autores

4.2 Criação de bases de dados com modelo PTT5

Outro método avaliado neste trabalho para a criação de bases de dados sintéticos é baseado no uso de modelos de aprendizado profundo capazes de gerar texto a partir de uma informação de entrada. Utilizamos esse método para a criação de duas novas bases de dados. O modelo utilizado foi o PTT5 (CARMO *et al.*, 2020), um modelo baseado no *transformers* capaz de gerar texto a partir de uma entrada em texto e com vocabulário em português, como explicado na Seção 2.3.1.

Para que o modelo fosse capaz de gerar os textos esperados, foi necessário fazer o *fine-tune* do modelo para que fossem geradas frases relacionadas as entidades e intenções relacionadas ao domínio. Esse *fine-tune* foi feito fornecendo uma intenção e um conjunto de entidades como entrada e gerando uma frase com essas informações, como representado na Figura 21.

Figura 21 – Exemplo de funcionamento do T5.



Fonte: Elaborado pelo autor

Para o *fine-tune* do modelo foi utilizado as bases apresentadas na Seção 4.1. A Tabela 5 apresenta como foi realizada a divisão dos dados existentes em dados de treino, para fazer o *fine-tune*, e dados para gerar a nova base. Os dados foram divididos de modo que ficasse uma

pequena quantidade para treino e uma maior quantidade para geração das novas bases, visando avaliar a qualidade dos textos gerados pelo PTT5 quando poucos dados estão disponíveis para treino.

Durante a geração dos novos dados, os dados são tokenizados e passados para o PTT5, que irá gerar uma resposta também tokenizada. A resposta é então decodificada e *tokens* especias são removidos, resultando no texto final. As informações sobre as bases de dados resultantes são apresentadas na Tabela 6.

Tabela 5 – Configuração dos dados para utilização do PTT5

Base de dados resultante	Base de dados utilizada	Porcentagem de dados para treino	porcentagem de dados para geração de nova base
t5_maior_variacao_menos_exemplos	*base_treino_maior_variacao*	30% (1597 amostras)	70% (3727 amostras)
t5_maior_variacao_mais_exemplos	*base_treino_maior_variacao*	16% (852 amostras)	84% (4472 amostras)
t5_menor_variacao	*base_treino_menor_variacao*	30% (1453 amostras)	70% (3391 amostras)

Fonte: Desenvolvido pelos autores

Tabela 6 – Bases de dados geradas com PTT5

Base de dados	número de amostras	número de intenções
t5_maior_variacao_menos_exemplos	3727	9
t5_maior_variacao_mais_exemplos	4472	9
t5_menor_variacao	3391	9

Fonte: Desenvolvido pelos autores

Para avaliar a qualidade das novas bases de dados três métricas foram utilizadas o BLEU, F1 e *exact match*, métricas utilizadas para avaliar a qualidade de textos gerados por modelo, como descrito na Seção 2.4. A Tabela 7 apresenta os resultados obtidos na geração de dados com o T5.

Tabela 7 – resultado obtido para geração de dados com PTT5

Base	BLEU	F1	exact match
t5_maior_variacao_menos_exemplos	56,6332	0,7273	0,3565
t5_maior_variacao_mais_exemplos	49,3629	0,6868	0,2517
t5_menor_variacao	62,2126	0,7670	0,4638

Fonte: Desenvolvido pelos autores

A partir dos valores de *exact match* para *t5_maior_variacao_menos_exemplos* e

t5_maior_variacao_mais_exemplos é possível perceber que quanto menor a quantidade de dados utilizado para treinar o modelo, mais difícil é para o modelo gerar textos idênticos aos esperados. Enquanto que o *exact match* para a base *t5_menor_variacao* mostra que o T5 tende a acertar mais quando os textos de treino e os textos esperados como resultado possuem uma menor diversidade de estruturas.A Tabela 8 apresenta alguns exemplos de textos gerados pelo PTT5.

Tabela 8 – Textos gerados pelo PTT5

Intenção	Texto
dsc_documentos_estado_civil	documentos necessário para obter cnh definitiva, para quem é casada.
dsc_horarios_funcionamento_objetivo	nos dias de sabado posso realizar antecedentes criminais no período da tarde?
dsc_servico	Necessito saber o que carteira permanente significa.
dsc_horarios_funcionamento	cnh final periodo de funcionamento.

Fonte: Desenvolvido pelos autores

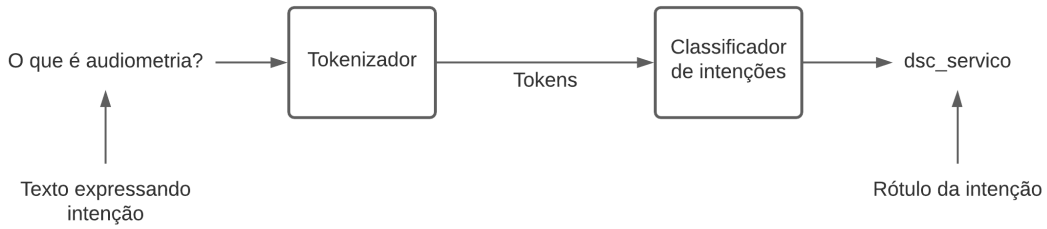
Analisando a estrutura dos textos gerados, foi possível perceber que, de maneira geral, eles são compreensíveis e bem estruturados, o que indica que apesar de alguns textos não saírem como esperado, eles ainda possuem uma boa qualidade. Essa afirmação também é sustentada pelos valores de BLEU e F1, que indicam que apesar de uma frase gerada não ser idêntica a esperada, ela possui muitas palavras que estão presentes no texto esperado, o que faz com que a ideia geral do texto seja mantida.

4.3 Modelo para classificação de intenção

Para avaliar as bases de dadas geradas, cinco métodos de classificação de intenções foram utilizados combinando tokenizadores com modelos de classificação de intenções. Os cinco métodos foram escolhidos com o intuito de avaliar métodos estado da arte e que possuem grande probabilidade de serem escolhidos para o desenvolvimento de um *chatbot*. Um método de classificação a combinação de um tokenizador e de um classificador de intenções como ilustrado na Figura 22.

Para fazer a tokenização três tipos de tokenizadores são utilizados o BERTimbau, o BERT multilíngue e o BERT. Enquanto que para o classificador de intenções foi utilizado o BERTimbau, o BERT multilíngue e o DIET (*Dual Intent Entity Transformer*), o modelo padrão

Figura 22 – Exemplo de funcionamento do classificador de intenções.



Fonte: Elaborado pelo autor

do Rasa para a tarefa de classificação de intenções e reconhecimento de entidades.

Tanto para todo os tokenizadores quanto para os classificadores baseados do BERTimbau e do BERT multilíngue foram utilizados a versão base e sensível à capitalização das palavras. Os classificadores de intenção baseado do BERTimbau e do BERT multilíngue foram implementados utilizando a biblioteca *transformers* e o *tensorflow*[3].

Para que os modelos baseados do BERTimbau e do BERT multilíngue fossem capazes de classificar a intenção de um texto foi realizado um *fine-tune* no modelo, tendo sido treinado por 3 épocas com uma taxa de aprendizado de $2e^{-5}$. O treinamento foi realizado utilizando um *k-fold* estratificado, com k=5. Para cada subconjunto o modelo foi treinado utilizando um subconjunto como validação e os subconjuntos restantes para treino. No fim foi analisado qual iteração gerou o melhor resultado de validação, sendo esse escolhido para a realização dos experimentos.

Já o DIET foi utilizado com sua configuração padrão definido pelo Rasa, apenas alterando seu funcionamento para realizar apenas a tarefa de classificação de intenções. Assim o modelo foi treinado por 100 épocas, uma vez que não é um modelo pré-treinado como BERT.

A Tabela 9 apresenta um resumo dos métodos de classificação de intenções definidos nesse trabalho, apresentado as combinações de tokenizadores e classificadores utilizados.

Tabela 9 – Tipo de métodos de classificação de intenções

Método	Tokenizador	Classificador
BERTimbau	BERTimbau	BERTimbau
BERTimbau + DIET	BERTimbau	DIET
BERT multilíngue	BERT multilíngue	BERT multilíngue
BERT multilíngue + DIET	BERT multilíngue	DIET
BERT + DIET	BERT	DIET

Fonte: Desenvolvido pelos autores

[3] Disponível em: https://www.tensorflow.org

Para avaliação das bases de dados com relação a tarefa de classificação de intenções, uma série de experimentos foram realizados, obtendo os resultados gerados pelos modelos descritos nessa seção.

4.4 Experimentos

A base de testes para os experimentos [4] foi gerada a partir de um modelo do *chatette*, com a mesma estrutura de textos e valores de entidades que a base *base_treino_maior_variacao*, porém sem intersecção de exemplos, sendo utilizada em todo os experimentos para gerar os resultados de avaliação.

Os experimentos foram avaliados utilizando as métricas F1, acurácia, revocação e precisão, descritas na Seção 2.4.2, utilizando os métodos de classificação de intenções descrito na Seção 4.3, através da ferramenta de testes do Rasa. Para cada modelo e base de dados foram executados três testes, onde em cada teste foi definido um valor mínimo de confiança para que a reposta gerada pelo modelo fosse considerada correta. Os testes foram executados com 0%, 70% e 90% de confiança, sendo utilizado o componente *fallback classifier* do Rasa para definição dos valores de confiança de 70% e 90%. O *fallback classifier* retorna uma mensagem padrão sempre que a intenção gerada pelo modelo tiver um nível de confiança menor que o definido.

A seguir é apresentado os resultados obtidos em cada tipo de experimento, bem como uma seção detalhando o que podemos implicar a partir da análise desses experimentos.

4.4.1 Experimentos para avaliar a base de dados gerada a partir de modelos de questões

Neste experimento os modelos foram treinados com a base de dados *base_treino_maior_variacao*, descrita na Seção 4.1. Os resultados de teste obtidos são apresentados na Tabela 10.

No geral os resultados gerados pelos modelos foram altos, ficando acima de 99%, exceto pelo teste com BERTimbau com 90% de confiança, que apesar de ter conseguido um resultado abaixo dos demais modelo, ainda obteve valores próximos a 88% para F1, revocação e precisão e 86% para acurácia. Considerandos esses valores é possível que o modelo tenha aprendido apenas as estruturas dos textos e estar classificando corretamente devido a base de

[4] Esses experimentos foram realizados através do *Google Colab*, um ambiente de *notebooks Jupyter* executado na nuvem, disponível em https://colab.research.google.com/drive/1sOU46y3Sc1PyQxTKFSoq8kNjZpyX67PN?usp=sharing.

Tabela 10 – Resultados obtido nos experimentos com a base *base_treino_maior_variacao*

Experimento	F1			Acurácia			Revocação			Precisão		
	0%	70%	90%	0%	70%	90%	0%	70%	90%	0%	70%	90%
BERTimbau	0,993	0,993	0,885	0,997	0,997	0,869	0,998	0,998	0,885	0,994	0,994	0,885
BERTimbau + DIET	0,999	0,997	0,997	0,999	0,998	0,998	0,998	0,998	0,998	0,997	0,996	0,996
BERT multilíngue	0,998	0,998	0,909	0,997	0,997	0,908	0,997	0,997	0,909	1,000	1,000	0,909
BERT multilíngue + DIET	0,999	0,997	0,995	0,999	0,997	0,995	0,998	0,997	0,995	0,998	0,997	0,995
BERT + DIET	0,997	0,997	0,996	0,998	0,997	0,996	0,997	0,997	0,997	0,996	0,995	0,994

Fonte: Desenvolvido pelos autores

teste possuir a mesma estrutura.

4.4.2 Experimentos para avaliar como as estruturas das frases influenciam no desempenho do modelo

Neste experimento os modelos foram treinados com a base de dados *base_treino_menor_variacao*, descrita na Seção 4.1. Neste experimento os dados de teste possuem estruturas não vistas pelos modelos durante o treinamento.

O objetivo nesses experimentos é verificar o desempenho dos modelos de classificação quando a base de treinamento não reflete a exata estrutura das frases contidas no conjunto de teste. A base de dados *base_treino_menor_variacao* é semelhante ao *base_treino_maior_variacao*, porém exemplos específicos foram removidos, garantindo que o modelo classificasse exemplos na base de teste que não possuem exemplos semelhantes na base de treino. Os resultados de teste obtidos são apresentados na Tabela 11.

Tabela 11 – Resultados obtidos nos experimentos para determinar se o modelo está aprendendo bem a partir das bases de dados

Experimento	F1			Acurácia			Revocação			Precisão		
	0%	70%	90%	0%	70%	90%	0%	70%	90%	0%	70%	90%
BERTimbau	0,988	0,965	0,849	0,988	0,977	0,818	0,988	0,943	0,818	0,988	0,986	0,885
BERTimbau + DIET	0,949	0,903	0,874	0,949	0,901	0,874	0,949	0,901	0,874	0,955	0,919	0,885
BERT multilíngue	0,902	0,794	0,742	0,905	0,826	0,695	0,905	0,763	0,693	0,921	0,834	0,818
BERT multilíngue + DIET	0,931	0,892	0,888	0,931	0,891	0,891	0,931	0,891	0,891	0,937	0,913	0,903
BERT + DIET	0,881	0,876	0,846	0,883	0,872	0,849	0,881	0,872	0,849	0,918	0,915	0,882

Fonte: Desenvolvido pelos autores

A partir da Tabela 11 é possível perceber que há redução significativa dos resultados quando comparado com o experimento anterior, chegando a mais de 20% de diferença como no caso do BERT multilíngue, onde o F1 cai de 0,998 para 0,794, quando definindo uma confiança mínima de 70%. Essa redução é mais significativa quando o nível de confiança é de 90%, como no caso do BERT multilíngue e em alguns casos em que o nível de confiança é de 70%, como no

BERT + DIET. Porém, é possível perceber que o treinamento com *base_treino_menor_variacao* é bom o suficiente para permitir que os modelos classifiquem corretamente estruturas de textos não vistas.

4.4.3 Experimentos para avaliar as bases de dados geradas com T5

Esses experimentos foram conduzidos com o objetivo de verificar se o T5 poderia ser uma boa ferramenta para geração de bases de dados artificiais. Foi feito um experimento para cada base de dados apresentado na Seção 4.2 e os resultados são apresentados nas tabelas 12, 13 e 14.

Tabela 12 – Resultados obtidos nos experimentos com *t5_maior_variacao_menos_exemplos*

Experimento	F1			Acurácia			Revocação			Precisão		
	0%	70%	90%	0%	70%	90%	0%	70%	90%	0%	70%	90%
BERTimbau	0,971	0,954	0,853	0,972	0,966	0,813	0,972	0,938	0,813	0,972	0,972	0,909
BERTimbau + DIET	0,916	0,910	0,899	0,915	0,912	0,767	0,921	0,912	0,904	0,920	0,918	0,910
BERT multilíngue	0,959	0,945	0,901	0,959	0,939	0,936	0,959	0,918	0,836	0,962	0,951	0,944
BERT multilíngue + DIET	0,908	0,900	0,897	0,909	0,902	0,899	0,909	0,902	0,899	0,916	0,910	0,909
BERT + DIET	0,840	0,793	0,784	0,847	0,803	0,801	0,846	0,803	0,795	0,870	0,829	0,824

Fonte: Desenvolvido pelos autores

Tabela 13 – resultados obtido nos experimentos com *t5_menor_variacao*

Experimento	F1			Acurácia			Revocação			Precisão		
	0%	70%	90%	0%	70%	90%	0%	70%	90%	0%	70%	90%
BERTimbau	0,998	0,897	0,696	0,998	0,946	0,639	0,998	0,864	0,639	0,998	0,989	0,797
BERTimbau + DIET	0,861	0,839	0,817	0,873	0,850	0,834	0,873	0,850	0,845	0,867	0,858	0,834
BERT multilíngue	0,996	0,839	0,753	0,996	0,857	0,674	0,996	0,787	0,674	0,996	0,954	0,885
BERT multilíngue + DIET	0,810	0,808	0,768	0,816	0,814	0,776	0,816	0,814	0,776	0,840	0,819	0,801
BERT + DIET	0,750	0,729	0,724	0,758	0,736	0,715	0,759	0,736	0,725	0,767	0,753	0,749

Fonte: Desenvolvido pelos autores

Tabela 14 – Resultados obtido nos experimentos com *t5_maior_variacao_mais_exemplos*

Experimento	F1			Acurácia			Revocação			Precisão		
	0%	70%	90%	0%	70%	90%	0%	70%	90%	0%	70%	90%
BERTimbau	0,942	0,930	0,769	0,941	0,925	0,782	0,941	0,893	0,682	0,943	0,924	0,895
BERTimbau + DIET	0,886	0,868	0,863	0,886	0,868	0,866	0,886	0,866	0,866	0,895	0,888	0,875
BERT multilíngue	0,913	0,899	0,777	0,910	0,812	0,764	0,910	0,864	0,696	0,920	0,842	0,768
BERT multilíngue + DIET	0,940	0,927	0,922	0,939	0,926	0,921	0,939	0,924	0,921	0,942	0,934	0,930
BERT + DIET	0,830	0,829	0,827	0,829	0,829	0,826	0,843	0,843	0,841	0,837	0,834	0,832

Fonte: Desenvolvido pelos autores

É possível perceber que os resultados obtidos nesses experimentos são um pouco inferiores ao dos experimentos anteriores, tendo valores, no geral, acima de 80%. Isso significa que o T5 foi capaz de extrair bem as informações dos dados de treino e gerar bases de dados semelhantes as bases geradas pelo *chatette*, porém com alguns pequenos erros em alguns textos, diminuindo o desempenho do modelo.

Considerando os resultados, das Tabelas 13 e 14, podemos ver que o T5 é capaz de gerar bases de dados satisfatórias, mesmo quando treinado em uma base pequena(850 amostras).

A próxima seção apresenta perspectivas e opiniões com base dos experimentos realizados.

4.4.4 Discussões

Considerando os experimentos realizados e os resultados obtidos, essa seção apresenta algumas analises comparativas e algumas perspectivas quanto aos resultados.

A Tabela 15 apresenta os resultados gerados pelo BERTimbau, modelo que obteve os melhores resultados na maioria dos testes, para a base *base_treino_maior_variacao* e as bases geradas pelo PTT5 a partir dela.

Tabela 15 – Comparação entre os resultados obtidos com o BERTimbau usando as bases de dados geradas com *chatette* e com PTT5

Base de dados	F1			Acurácia			Revocação			Precisão		
	0%	70%	90%	0%	70%	90%	0%	70%	90%	0%	70%	90%
base_treino_maior_variacao	0,993	0,993	0,885	0,997	0,997	0,869	0,998	0,998	0,885	0,994	0,994	0,885
t5_maior_variacao_menos_exemplos	0,971	0,954	0,853	0,972	0,966	0,813	0,972	0,938	0,813	0,972	0,972	0,909
t5_maior_variacao_mais_exemplos	0,942	0,930	0,769	0,941	0,925	0,782	0,941	0,893	0,682	0,943	0,924	0,895

Fonte: Desenvolvido pelos autores

É possível perceber que os resultados gerados para cada base são próximos, exceto quando a confiança é de 90%, onde a base *t5_maior_variacao_mais_exemplos* tem resultados inferiores quando comparados aos demais. A base *t5_maior_variacao_mais_exemplos* já é esperado possuir desempenho inferior aos demais, visto que o PTT5 foi treinado com menos amostras (852 amostras), gerando textos de qualidade mais baixa que *t5_maior_variacao_menos_exemplos*, como apresentado na Tabela 16. A quantidade de amostras presente em cada base também aparenta interferir nos resultados do modelo, apesar de não aparentar ser o grande responsável pela perda de desempenho. Algumas intenções possuem textos semelhantes, como as intenções *dsc_local_presencial* e *dsc_local_presencial_objetivo*. É esperado que quanto menos amos-

tras houver para treino, mais difícil é para o modelo classificar corretamente o texto recebido.

Tabela 16 – Exemplos de frases presentes das bases *base_treino_maior_variacao*, *t5_maior_variacao_mais_exemplos* e *t5_maior_variacao_menos_exemplos*

Base de dados	Frase	Intenção
base_treino_maior_variacao	Definição de solicitação de RG.	dsc_servico
	Consigo tirar CNH as 7h pm de quinta?	dsc_horarios_funcionamento_objetivo
	Carteira de habilitação definitiva precisa de fotos 3x4?	dsc_documentos_objetivo
t5_maior_variacao_menos_exemplos	Necessito saber o que carteira permanente significa.	dsc_servico
	Consigo emitir CRLV as 3 a tarde de quarta-feira?	dsc_horarios_funcionamento_objetivo
	fotos é um do documentos que solicitação carteira de identidade civil exige?	dsc_horarios_funcionamento_objetivo
t5_maior_variacao_mais_exemplos	emissão licenciamento anual (CRLV) significa o que?	dsc_servico
	Emissão de atestado de antecendentes criminais e identificação criminal esta disponivel nos dias de 16:30?	dsc_horarios_funcionamento_objetivo
	solicitação de carteira de identidade (RG) precisa de reservista) reservista?	dsc_documentos_objetivo

Fonte: Desenvolvido pelos autores

As tabelas 17 e 18 apresentam uma comparação entre as bases *t5_maior_variacao_menos_exemplos* e *base_treino_menor_variacao*, utilizando, respectivamente, o BERTimbau e o BERT multilíngue, com confiança de 90%.

A partir dos resultados é possível perceber que, apesar de possuir todas as variações de estruturas de textos, os resultados obtidos pelo modelo treinado com a base gerada por PTT5 são bastante próximos aos resultados obtidos pelo modelo treinado com a *base_treino_menor_variacao*. Enquanto que no BERT multilíngue o modelo treinado com *t5_maior_variacao_menos_exemplos* tem um resultado muito superior. Considerando que o BERT multilíngue foi treinado para lidar com múltiplos idiomas, esse modelo provavelmente sabe lidar melhor com diferentes estruturas de textos, mesmo que sejam incomuns no português. Acreditamos que por isso o BERT multilíngue obteve um melhor resultado com o *t5_maior_variacao_menos_exemplos*, uma vez que consegue lidar com os erros cometido pelo PTT5 e aprender o suficiente para classificar bem as amostras de teste. Isso, mais uma vez, corrobora com a hipótese de que a queda de performance apresentada na Tabela 15 pelos modelos

treinados com os dados gerados pelo PTT5 foi causada pela perda de qualidade dos dados.

Tabela 17 – Comparação dos resultados gerado pelo BERTimbau com 90% de confiança, onde as bases *base_treino_menor_variacao* e *t5_maior_variacao_menos_exemplos* geraram resultados semelhantes

Base de dados	F1	Acurácia	Revocação	Precisão
base_treino_menor_variacao	0,849	0,818	0,818	0,885
t5_maior_variacao_menos_exemplos	0,853	0,813	0,813	0,909

Fonte: Desenvolvido pelos autores

Tabela 18 – Comparação dos resultados gerado pelo BERT multilíngue com 90% de confiança, onde as bases *base_treino_menor_variacao* e *t5_maior_variacao_menos_exemplos* geraram resultados diferentes

Base de dados	F1	Acurácia	Revocação	Precisão
base_treino_menor_variacao	0,742	0,695	0,693	0,818
t5_maior_variacao_menos_exemplos	0,901	0,936	0,836	0,944

Fonte: Desenvolvido pelos autores

Já a Tabela 19, apresenta os resultados para as bases que possuem uma menor variação de estruturas de textos, também com o modelo BERTimbau. Nesse caso, a quantidade de amostras nas bases de dados parece ter influenciado diretamente nos resultados.

Tabela 19 – Comparação entre as bases de dados mais simples gerados com *chatette* e com PTT5, usando o BERTimbau

Base de dados	F1			Acurácia			Revocação			Precisão		
	0%	70%	90%	0%	70%	90%	0%	70%	90%	0%	70%	90%
base_treino_menor_variacao	0,988	0,965	0,849	0,988	0,977	0,818	0,988	0,943	0,818	0,988	0,986	0,885
t5_menor_variacao	0,998	0,897	0,696	0,998	0,946	0,639	0,998	0,864	0,639	0,998	0,989	0,797

Fonte: Desenvolvido pelos autores

Apesar de obter bons resultados nos testes menos restritos (0% e 70%), o modelo treinado com a base *t5_menor_variacao* obteve resultados bem abaixo que o modelo treinado com a base *base_treino_menor_variacao*, quando utilizado confiança de 90%. Como a base de teste possui estruturas de texto ainda não vistas pelo modelo, como mostrado na tabela 20, é comum uma performance não tão boa. Porém como a base *t5_menor_variacao* possui uma quantidade menor de dados, fica mais difícil para o modelo classificar corretamente. Considerando, ainda, que essa base foi a que obteve melhores resultados segundo as métricas para geração de texto, parece improvável que a qualidade dos textos da base tenha sido o principal responsável pela queda de desempenho do modelo.

Tabela 20 – Exemplos de frases presentes das bases *base_treino_menor_variacao*, *t5_menor_variacao* e *teste*

Base de dados	Frase	Intenção
base_treino_menor_variacao	o que significa atestado de antecedentes criminais?	dsc_servico
	As 13h de segunda feira é possível obter cnh definitiva?	dsc_horarios_funcionamento_objetivo
t5_menor_variacao	Carteira de habilitação definitiva definição.	dsc_servico
	Quinta-feira as 2 horas posso emitir RG?	dsc_horarios_funcionamento_objetivo
teste	Estou com dúvida sobre o que obter cnh definitiva representa.	dsc_servico
	Em que dias e horário solicitação de carteira de identidade (RG) pode ser realizado?	dsc_horarios_funcionamento_objetivo

Fonte: Desenvolvido pelos autores

Considerando o que foi observado nessa seção, bases de dados geradas a partir de modelos de textos são uma boa escolha quando uma base real não está disponível, sendo um método fácil e de custo menor do que gerar uma base com dados reais. Porém pode levar a geração de bases com muitos textos semelhantes, prejudicando a capacidade do modelo de classificar corretamente textos muito diferentes dos já vistos durante o treino. Já bases de dados geradas com PTT5 aparentam ser uma boa opção quando existe alguma base de qualidade, mesmo que pequena, para utilizar no treinamento do modelo, visto que a qualidade da base de treino irá influenciar diretamente da qualidade da base resultante. Apesar disso, uma análise da base resultante se faz necessária para garantir a remoção de textos sem sentido e mal formados, garantindo a qualidade dos dados que serão utilizados.

5 CONCLUSÕES E TRABALHOS FUTUROS

O presente trabalho teve como objetivo avaliar base de dados artificiais como uma alternativa viável para treinamento de modelos de classificação de intenções, quando uma base de dados real não está disponível. Assim como verificar se a utilização de dados gerados por um modelo T5 é capaz de ajudar a criar bons modelos quando poucos dados estão disponíveis.

Com base dos resultados obtidos é possível concluir que bases de dados artificiais podem ser boas bases de dados, porém deve-se ter cuidado para os exemplos não ficarem muito simples ou muito diferentes dos dados reais. As bases desenvolvidas para esse trabalho ainda precisam de melhorias, como a geração de frases mais diversificadas e eliminação das frases cuja a estrutura das palavras são bastante comuns entre si.

Os resultados obtidos para os modelos treinados com os dados gerados em PTT5, mostram que foi possível obter resultados acima de 80% na maioria dos testes com confiabilidades de 0% e 70%. E obteve resultados próximos ou maior que 80% quando a confiabilidade era de 90%, exceto em alguns onde os resultados ficaram entre 60% e 75%. Podemos concluir que dados gerados pelo modelo PTT5 aparentam ser uma boa fonte de dados para treinamento de modelos, quando não se tem acesso a uma grande quantidade de dados.

Mas para o T5 ser capaz de gerar bases de qualidade é esperado que a base de treino seja a mais diversificada possível e represente bem os dados reais. Além disso, os dados gerados pelo T5 tendem a ter uma qualidade um pouco abaixo dos dados gerados a partir de modelos de questões, uma vez que a probabilidade de gerar textos com uma má estrutura é maior. Então deve-se esperar que um modelo treinado com dados gerados pelo T5 tenha uma perda de eficácia quando comparada com modelos treinados com dados gerados por modelos de questões, apesar de não ser uma perda significativa.

Como trabalhos futuros pretendemos utilizar as bases de dado criadas para treinar um modelo com BERTimbau para classificação de entidades. Além disso, criar um único modelo com BERT capaz de realizar a classificação de intenção e a classificação de entidade em uma única execução. O modelo criado então será integrado ao Rasa para a criação do *chatbot* no projeto transformação digital.

REFERÊNCIAS

ADAMOPOULOU, E.; MOUSSIADES, L. An overview of chatbot technology. In: SPRINGER. **IFIP International Conference on Artificial Intelligence Applications and Innovations**. [*S. l.*], 2020. p. 373–383.

AL-SINANI, A. H.; AL-SAIDI, B. S. A survey of chatbot creation tools for non-coder. **Journal of Student Research**, 2019.

AMIN-NEJAD, A.; IVE, J.; VELUPILLAI, S. Exploring transformer text generation for medical dataset augmentation. In: **Proceedings of the 12th Language Resources and Evaluation Conference**. [*S. l.*: *s. n.*], 2020. p. 4699–4708.

BIRD, J. J.; EKÁRT, A.; FARIA, D. R. Chatbot interaction with artificial intelligence: Human data augmentation with t5 and language transformer ensemble for text classification. **arXiv preprint arXiv:2010.05990**, 2020.

BOCKLISCH, T.; FAULKNER, J.; PAWLOWSKI, N.; NICHOL, A. Rasa: Open source language understanding and dialogue management. **arXiv preprint arXiv:1712.05181**, 2017.

CANETE, J.; CHAPERON, G.; FUENTES, R.; PÉREZ, J. Spanish pre-trained bert model and evaluation data. **Pml4dc at iclr**, v. 2020, 2020.

CARMO, D.; PIAU, M.; CAMPIOTTI, I.; NOGUEIRA, R.; LOTUFO, R. Ptt5: Pretraining and validating the t5 model on brazilian portuguese data. **arXiv preprint arXiv:2008.09144**, 2020.

DAI, Z.; YANG, Z.; YANG, Y.; CARBONELL, J.; LE, Q. V.; SALAKHUTDINOV, R. Transformer-xl: Attentive language models beyond a fixed-length context. **arXiv preprint arXiv:1901.02860**, 2019.

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

HU, D. An introductory survey on attention mechanisms in nlp problems. In: SPRINGER. **Proceedings of SAI Intelligent Systems Conference**. [*S. l.*], 2019. p. 432–448.

IVE, J.; VIANI, N.; KAM, J.; YIN, L.; VERMA, S.; PUNTIS, S.; CARDINAL, R. N.; ROBERTS, A.; STEWART, R.; VELUPILLAI, S. Generation and evaluation of artificial mental health records for natural language processing. **NPJ Digital Medicine**, Nature Publishing Group, v. 3, n. 1, p. 1–9, 2020.

JOHNSON, A. E.; POLLARD, T. J.; SHEN, L.; LI-WEI, H. L.; FENG, M.; GHASSEMI, M.; MOODY, B.; SZOLOVITS, P.; CELI, L. A.; MARK, R. G. Mimic-iii, a freely accessible critical care database. **Scientific data**, Nature Publishing Group, v. 3, n. 1, p. 1–9, 2016.

JUMUTC, V. **Efficient training and usage of NLU and NLG models**. 2019. Disponível em: https://medium.com/@jumutc/efficient-training-and-usage-of-nlu-and-nlg-models-306eb28f8172. Acesso em: 25 maio 2020.

LEE, J.; YOON, W.; KIM, S.; KIM, D.; KIM, S.; SO, C. H.; KANG, J. Biobert: a pre-trained biomedical language representation model for biomedical text mining. **Bioinformatics**, Oxford University Press, v. 36, n. 4, p. 1234–1240, 2020.

MAMATHA, M. *et al.* Chatbot for e-commerce assistance: based on rasa. **Turkish Journal of Computer and Mathematics Education (TURCOMAT)**, v. 12, n. 11, p. 6173–6179, 2021.

MANNINO, M.; ABOUZIED, A. Is this real? generating synthetic data that looks real. In: **Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology**. [*S. l.*: *s. n.*], 2019. p. 549–561.

MANNINO, M.; ABOUZIED, A. Synner: Generating realistic synthetic data. In: **Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data**. [*S. l.*: *s. n.*], 2020. p. 2749–2752.

MISHRA, A.; JAIN, S. K. A survey on question answering systems with classification. **Journal of King Saud University-Computer and Information Sciences**, Elsevier, v. 28, n. 3, p. 345–361, 2016.

NIKOLENKO, S. I. Synthetic data for deep learning. **arXiv preprint arXiv:1909.11512**, 2019.

NURUZZAMAN, M.; HUSSAIN, O. K. A survey on chatbot implementation in customer service industry through deep neural networks. In: IEEE. **2018 IEEE 15th International Conference on e-Business Engineering (ICEBE)**. [*S. l.*], 2018. p. 54–61.

PAPINENI, K.; ROUKOS, S.; WARD, T.; ZHU, W.-J. Bleu: a method for automatic evaluation of machine translation. In: **Proceedings of the 40th annual meeting of the Association for Computational Linguistics**. [*S. l.*: *s. n.*], 2002. p. 311–318.

RADFORD, A.; WU, J.; CHILD, R.; LUAN, D.; AMODEI, D.; SUTSKEVER, I. Language models are unsupervised multitask learners. **OpenAI Blog**, v. 1, n. 8, p. 9, 2019.

RAFFEL, C.; SHAZEER, N.; ROBERTS, A.; LEE, K.; NARANG, S.; MATENA, M.; ZHOU, Y.; LI, W.; LIU, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. **arXiv preprint arXiv:1910.10683**, 2019.

RAJPURKAR, P.; ZHANG, J.; LOPYREV, K.; LIANG, P. Squad: 100,000+ questions for machine comprehension of text. **arXiv preprint arXiv:1606.05250**, 2016.

SILVEIRA, I. C.; MAUÁ, D. D. Advances in automatically solving the enem. In: IEEE. **7th Brazilian Conference on Intelligent Systems (BRACIS)**. SP, Brazil, 2018. p. 43–48.

SOUZA, F.; NOGUEIRA, R.; LOTUFO, R. BERTimbau: pretrained BERT models for Brazilian Portuguese. In: **9th Brazilian Conference on Intelligent Systems, BRACIS, Rio Grande do Sul, Brazil, October 20-23 (to appear)**. [*S. l.*: *s. n.*], 2020.

SRIVASTAVA, S.; PRABHAKAR, T. Intent sets: Architectural choices for building practical chatbots. In: **Proceedings of the 2020 12th International Conference on Computer and Automation Engineering**. [*S. l.*: *s. n.*], 2020. p. 194–199.

STANFORD, U. **SQuAD 2.0 The Stanford Question Answering Dataset**. 2020. Disponível em: https://rajpurkar.github.io/SQuAD-explorer/. Acesso em: 05 maio 2020.

STUPINA, A.; ZHUKOV, E.; EZHEMANSKAYA, S.; KARASEVA, M.; KORPACHEVA, L. Question-answering system. In: IOP PUBLISHING. **IOP Conference Series: Materials Science and Engineering**. [*S. l.*], 2016. v. 155, n. 1, p. 012024.

SUTSKEVER, I.; VINYALS, O.; LE, Q. V. Sequence to sequence learning with neural networks. In: **Advances in neural information processing systems**. [*S. l.*: *s. n.*], 2014. p. 3104–3112.

TOWARDS DATA SCIENCE. **BERT Explained**: State of the art language model for nlp. 2018. Disponível em: https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270. Acesso em: 20 maio 2020.

VASWANI, A.; SHAZEER, N.; PARMAR, N.; USZKOREIT, J.; JONES, L.; GOMEZ, A. N.; KAISER, Ł.; POLOSUKHIN, I. Attention is all you need. In: **Advances in neural information processing systems**. Long Beach, Califórnia: [*S. n.*], 2017. p. 5998–6008.

VRIES, W. de; CRANENBURGH, A. van; BISAZZA, A.; CASELLI, T.; NOORD, G. van; NISSIM, M. Bertje: A dutch BERT model. **CoRR**, abs/1912.09582, 2019. Disponível em: http://arxiv.org/abs/1912.09582.

WINDIATMOKO, Y.; RAHMADI, R.; HIDAYATULLAH, A. F. Developing facebook chatbot based on deep learning using rasa framework for university enquiries. In: IOP PUBLISHING. **IOP Conference Series: Materials Science and Engineering**. [*S. l.*], 2021. v. 1077, n. 1, p. 012060.

ZHANG, Z.; YANG, J.; ZHAO, H. Retrospective reader for machine reading comprehension. **arXiv preprint arXiv:2001.09694**, 2020.

APÊNDICE A – ARQUIVOS CHATTETES PARA GERAÇÃO DAS BASE DE DADOS

Código-fonte 1 – modelo utilizado para gerar a base de dados com maior variação e a base de teste

```
%[&dsc_servico]( train : 600,  test : 200)
    ~[dsc_servico_questao_inicio] @[servico#tipo1]?
    @[servico#tipo1] ~[dsc_servico_questao_fim]?
    ~[dsc_servico_afirmacao_inicio] @[servico#tipo1].
    @[servico#tipo1] ~[dsc_servico_afirmacao_fim].
    Me diga o que @[servico#tipo1] significa.
    pe o que, por favor, me fale o que @[servico#tipo2]
       representa.
%[&dsc_documentos]( train : 600,  test : 200)
    ~[dsc_documentos_questao_inicio] @[servico#tipo2]?
    @[servico#tipo1] ~[dsc_documentos_questao_fim]?
    ~[dsc_documentos_afirmacao_inicio] @[servico#tipo1].
    Qual o conjunto de documentos que @[servico#tipo1] pede
       ?
    [Liste?] tudo que @[servico#tipo1] requer.
%[&dsc_documentos_objetivo]( train : 600,  test : 200)
    ~[dsc_documentos_objetivo_questao_inicio] @[servico#
       tipo2]?
    @[servico#tipo1] ~[dsc_documentos_objetivo_questao_fim
       ]?
    ser  que @[documento]  obrigatorio para @[servico#
       tipo1] ser executado?
    Considerando que existe a possibilidade de fazer @[
       servico#tipo1], @[documento]  necess rio?
    Para @[servico#tipo2]  necess rio @[documento]?
%[&dsc_documentos_estado_civil]( train : 600,  test : 200)
```

```
21      ~[dsc_documentos_estado_civil_questao_inicio] @[servico
          #tipo1]?
22      @[servico#tipo1] ~[
          dsc_documentos_estado_civil_questao_fim]?
23      ~[dsc_documentos_estado_civil_afirmacao_inicio] @[
          servico#tipo2].
24      quais coisas s o necess rias para @[servico#tipo2],
          sendo @[estado_civil]?
25      me d  uma lista de tudo que  exigido por @[servico#
          tipo2], para uma pessoa @[estado_civil].
26      Quais documento s o exigido para que quem  @[
          estado_civil] consiga @[servico#tipo2] livre de
          complica  es?
27  %[&dsc_documentos_objetivo_estado_civil](train: 600,
      test: 200)
28      ~[dsc_documentos_objetivo_estado_civil_questao_inicio]
          @[servico#tipo2]?
29      @[servico#tipo1] ~[
          dsc_documentos_objetivo_estado_civil_questao_fim]?
30      Para o servi o @[servico#tipo1]  necess rio @[
          documento], se sou @[estado_civil]?
31      @[documento] se faz necess rio para @[servico#tipo2],
          quando se  @[estado_civil]?
32      @[estado_civil] precisa de @[documento] para que @[
          servico#tipo1] seja realizado?
33  %[&dsc_horarios_funcionamento](train: 600, test: 200)
34      ~[dsc_horarios_funcionamento_questao_inicio] @[servico#
          tipo2]?
35      @[servico#tipo1] ~[
          dsc_horarios_funcionamento_questao_fim]?
36      ~[dsc_horarios_funcionamento_afirmacao_inicio] @[
          servico#tipo1].
```

```
37      @[servico#tipo1] ~[
            dsc_horarios_funcionamento_afirmacao_fim].
38      O servi o @[servico#tipo1] pode ser feito em quais
            hor rios?
39      Em que dias e hor rio @[servico#tipo1] pode ser
            realizado?
40  %[&dsc_horarios_funcionamento_objetivo](train: 600, test
        : 200)
41      ~[dsc_horarios_funcionamento_objetivo_questao_inicio] @
            [servico#tipo2]?
42      @[servico#tipo1] ~[
            dsc_horarios_funcionamento_objetivo_questao_fim]?
43      Nos dias de @[dia] posso realizar @[servico#tipo1] no
            per odo da @[horario#turno]?
44      Consigo @[servico#tipo2] as @[horario#horas] de @[dia]?
45      Consigo @[servico#tipo2] @[dia] as @[horario#horas]?
46  %[&dsc_local_presencial](train: 600, test: 200)
47      ~[dsc_local_presencial_questao_inicio] @[servico#tipo2
            ]?
48      @[servico#tipo1] ~[dsc_local_presencial_questao_fim]?
49      ~[dsc_local_presencial_afirmacao_inicio] @[servico#
            tipo2].
50      @[servico#tipo1] ~[dsc_local_presencial_afirmacao_fim].
51      Para @[servico#tipo2] preciso me deslocar para qual
            regi o?
52      Lista de locais em que @[servico#tipo1] est
            dispon vel.
53      Para onde devo me deslocar para que o servi o @[
            servico#tipo1] possa ser realizado?
54  %[&dsc_local_presencial_objetivo](train: 600, test:
        200)
```

```
55      ~[dsc_local_presencial_objetivo_questao_inicio] @[
            servico#tipo1]?
56      @[servico#tipo1] ~[
            dsc_local_presencial_objetivo_questao_fim]?
57      O servi o @[servico#tipo1] est  dispon vel em @[
            cidade]?
58      Consigo @[servico#tipo2] em @[cidade]?
59      @[cidade]    uma das cidades em que @[servico#tipo1]
            pode ser realizado?
60
61  ~[dsc_servico_questao_inicio]
62      O que
63      O que significa
64      Do que se trata
65
66  ~[dsc_servico_questao_fim]
67      pode ser descrito como
68      significa o que
69          o que
70
71  ~[dsc_servico_afirmacao_inicio]
72      Quero saber o que
73      Descreva
74      Defini  o de
75
76  ~[dsc_servico_afirmacao_fim]
77      defini  o
78      descri  o
79
80  ~[dsc_documentos_questao_inicio]
81      Quais os documentos necess rios para
82      Quais os documentos requeridos para
```

```
    Do que preciso para

~[dsc_documentos_questao_fim]
    precisa de quais documentos
    exige quais documentos
    requer o que

~[dsc_documentos_afirmacao_inicio]
    Tudo que    requerido para
    Lista de documentos para
    Documentos para

~[dsc_documentos_objetivo_questao_inicio]
    [O documento?] @[documento]    necess rio para
    @[documento]    um dos documentos necess rios para
       realizar
    Preciso de @[documento] para

~[dsc_documentos_objetivo_questao_fim]
    precisa de @[documento]
    tem @[documento] como um de seus documentos necess rio
    necessita de @[documento] para ser realizado

~[dsc_documentos_estado_civil_questao_inicio]
    [Quem   ?] @[estado_civil] precisa de quais documentos
       para
    @[estado_civil] deve levar quais documentos se quiser
       realizar

~[dsc_documentos_estado_civil_questao_fim]
    precisa de quais documentos se @[estado_civil]
    necessita de @[documento] para quem    @[estado_civil]
```

```
112
113 ~[dsc_documentos_estado_civil_afirmacao_inicio]
114     Lista de documentos que @[estado_civil] precisa para
115     Documentos necess rios para eu, que sou @[estado_civil
        ],
116
117 ~[dsc_documentos_objetivo_estado_civil_questao_inicio]
118     Quem    @[estado_civil] precisa de @[documento] para
119     Considerando que sou @[estado_civil], preciso levar @[
        documento] se quero
120     Preciso de @[documento], se sou @[estado_civil] e
        desejo
121
122 ~[dsc_documentos_objetivo_estado_civil_questao_fim]
123     exige @[documento] de quem    @[estado_civil]
124     requer que @[estado_civil] leve @[documento] para ser
        realizado
125     requisita @[documento] para algu m @[estado_civil]
126
127 ~[dsc_horarios_funcionamento_questao_inicio]
128     Quando posso
129     Em que momentos consigo
130     Quais os horarios de disponibilidade para
131
132 ~[dsc_horarios_funcionamento_questao_fim]
133     funciona em quais hor rios
134        possivel ser feito em quais hor rios
135     est  disponivel em quais hor rios
136
137 ~[dsc_horarios_funcionamento_afirmacao_inicio]
138     dias e hor rios de funcionamento de
139     hor rios de funcionamento de
```

```
140     periodo de funcionamento do servi o
141
142 ~[dsc_horarios_funcionamento_afirmacao_fim]
143     horarios de funcionamento
144     periodo de funcionamento
145
146 ~[dsc_local_presencial_questao_inicio]
147     Onde posso
148     Onde devo ir para
149     Para onde preciso me direcionar se quero
150     em quais locais posso
151
152 ~[dsc_local_presencial_questao_fim]
153     esta disponivel em quais locais
154     pode ser realizado em quais locais
155
156 ~[dsc_local_presencial_afirmacao_inicio]
157     lista de locais para
158     locais em que    possivel
159     regi es em que posso
160
161 ~[dsc_local_presencial_afirmacao_fim]
162     locais para ser realizado
163     locais disponivel
164
165 ~[dsc_horarios_funcionamento_objetivo_questao_inicio]
166     @[dia] as @[horario#horas] posso
167     as @[horario#horas] de @[dia]    poss vel
168     No dia de @[dia] no perido da @[horario#turno] consigo
169     @[dia] no hor rio de @[horario#minutos] existe a
          possibilidade de
170
```

```
171 ~[dsc_horarios_funcionamento_objetivo_questao_fim]
172     esta disponivel nos dias de @[dia] as @[horario#minutos
        ]
173     pode ser realizado @[dia] de @[horario#turno]
174
175 ~[dsc_local_presencial_objetivo_questao_inicio]
176     @[cidade] permite a realizar o servi o
177     consigo, em @[cidade], realizar
178     est  disponivel em @[cidade] o servi o
179
180 ~[dsc_local_presencial_objetivo_questao_fim]
181     esta disponivel em @[cidade]
182     pode ser realizado em @[cidade]
183     pode ser feito em @[cidade] ou n o
184
185 @[servico#tipo1]
186     emiss o de certid o nada consta
187     solicita  o de carteira de habilita  o popular (CNH
          popular)
188     emiss o de certid o
189     solicitar cnh
190     solicita  o de carteira de habilita  o definitiva (
        CNH definitiva)
191     carteira de habilita  o definitiva
192     cnh final
193     carteira permanente
194     emiss o de atestado de antecendentes criminais e
        identifica  o criminal
195     identifica  o criminal
196     atestado de antecedentes criminais
197     antecedentes criminais
198     emiss o licenciamento anual (CRLV)
```

```
199     solicita  o carteira de identidade civil
200     solicita  o de RG
201     solicita  o de carteira de identidade (RG)
202
203 @[servico#tipo2]
204     emitir certid o (nada consta)
205     emitir certid o nada consta
206     emitir certid o em que n o consta nada
207     tirar cnh
208     obter carteira de habilita  o
209     obter habilita  o
210     solicitar habilita  o
211     solicitar permiss o para dirigir
212     obter cnh definitiva
213     tirar cnh definitiva
214     emitir identifica  o criminal e atestando de
          antecedentes criminais
215     emitir licenciamento anual
216     emitir CRLV
217     obter crlv
218     emitir licenciamento
219     emitir RG
220     solicitar RG
221     obter identidade
222     obter carteira de identidade
223
224 @[documento]
225     [RG|carteira de identidade|c dula de identidade|
          identidade]
226     [certid o de nascimento original|certid o de
          nascimento|c pia da certid o de nascimento]
```

```
227     [fotos 3 4 |fotos|certid o de casamento original|
          c pia da certid o de casamento|certid o de
          casamento]
228     [averba  o de div rcio|carteira de trabalho|CTPS|
          reservista|carteira de reservista|certid o de
          nascimento dos filhos|certid o de nascimento dos
          pais|c pia do boleto|comprovante de pagamento da
          taxa|boletim de ocorr ncia|BO]
229
230 @[estado_civil]
231     [solteiro|casado]
232     [solteira|casada]
233
234 @[dia]
235     [segunda|segunda feira|segunda-feira]
236     [ter a|ter a feira|ter a -feira]
237     [quarta|quarta feira|quarta-feira]
238     [quinta|quinta feira|quinta-feira]
239     [sexta|sexta feira|sexta-feira]
240     sabado
241
242 @[horario#horas]
243     [13h|1h pm|1 hora]
244     [14h|2h pm|2 horas]
245     [15h|3h pm|3 a tarde]
246     [16h|4h pm|4 horas]
247     [17h|5h pm|5 da tarde]
248     [18h|6h pm|6 horas]
249     [19h|7h pm|7 da noite]
250     [8h|8h am|8 horas]
251     [9h|9h am|9 da manh ]
252     [10h|10h am|8 horas]
```

```
253
254 @[horario#minutos]
255     16:30
256     16h40m
257     8:48
258     10h 40 min
259     15h 27min
260     12h 06m
261
262 @[horario#turno]
263     [tarde|manh |noite]
264
265 @[cidade]
266     Fortaleza
267     Sobral
268     Quixad
269     Canind
270     Juazeiro do Norte
271     iguat
272     Crate s
273     Itapipoca
274     Quixeramobim
275     Crato
```

Código-fonte 2 – modelo utilizado para gerar a base de dados com menor variação

```
1 %[&dsc_servico]( train : 600)
2     ~[dsc_servico_questao_inicio] @[servico#tipo1]?
3     @[servico#tipo1] ~[dsc_servico_questao_fim]?
4     ~[dsc_servico_afirmacao_inicio] @[servico#tipo1].
5     @[servico#tipo1] ~[dsc_servico_afirmacao_fim].
6 %[&dsc_documentos]( train : 600)
```

```
    ~[dsc_documentos_questao_inicio] @[servico#tipo2]?
    @[servico#tipo1] ~[dsc_documentos_questao_fim]?
    ~[dsc_documentos_afirmacao_inicio] @[servico#tipo1].
%[&dsc_documentos_objetivo]( train : 600)
    ~[dsc_documentos_objetivo_questao_inicio] @[servico#
       tipo2]?
    @[servico#tipo1] ~[dsc_documentos_objetivo_questao_fim
       ]?
    @[documento]    um do documentos que @[servico#tipo1]
       exige?
%[&dsc_documentos_estado_civil]( train : 600)
    ~[dsc_documentos_estado_civil_questao_inicio] @[servico
       #tipo1]?
    @[servico#tipo1] ~[
       dsc_documentos_estado_civil_questao_fim]?
    ~[dsc_documentos_estado_civil_afirmacao_inicio] @[
       servico#tipo2].
%[&dsc_documentos_objetivo_estado_civil]( train : 600)
    ~[dsc_documentos_objetivo_estado_civil_questao_inicio]
       @[servico#tipo2]?
    @[servico#tipo1] ~[
       dsc_documentos_objetivo_estado_civil_questao_fim]?
%[&dsc_horarios_funcionamento]( train : 600)
    ~[dsc_horarios_funcionamento_questao_inicio] @[servico#
       tipo2]?
    @[servico#tipo1] ~[
       dsc_horarios_funcionamento_questao_fim]?
    ~[dsc_horarios_funcionamento_afirmacao_inicio] @[
       servico#tipo1].
    @[servico#tipo1] ~[
       dsc_horarios_funcionamento_afirmacao_fim].
%[&dsc_horarios_funcionamento_objetivo]( train : 600)
```

```
27      ~[dsc_horarios_funcionamento_objetivo_questao_inicio] @
          [servico#tipo2]?
28      @[servico#tipo1] ~[
          dsc_horarios_funcionamento_objetivo_questao_fim]?
29  %[&dsc_local_presencial]( train : 600)
30      ~[dsc_local_presencial_questao_inicio] @[servico#tipo2
          ]?
31      @[servico#tipo1] ~[dsc_local_presencial_questao_fim]?
32      ~[dsc_local_presencial_afirmacao_inicio] @[servico#
          tipo2].
33      @[servico#tipo1] ~[dsc_local_presencial_afirmacao_fim].
34  %[&dsc_local_presencial_objetivo]( train : 600)
35      ~[dsc_local_presencial_objetivo_questao_inicio] @[
          servico#tipo1]?
36      @[servico#tipo1] ~[
          dsc_local_presencial_objetivo_questao_fim]?
37
38  ~[dsc_servico_questao_inicio]
39      O que
40      O que significa
41      Do que se trata
42
43  ~[dsc_servico_questao_fim]
44      pode ser descrito como
45      significa o que
46          o que
47
48  ~[dsc_servico_afirmacao_inicio]
49      Quero saber o que
50      Descreva
51      Defini  o de
52
```

```
53 ~[dsc_servico_afirmacao_fim]
54     defini  o
55     descri  o
56
57 ~[dsc_documentos_questao_inicio]
58     Quais os documentos necess rios para
59     Quais os documentos requeridos para
60     Do que preciso para
61
62 ~[dsc_documentos_questao_fim]
63     precisa de quais documentos
64     exige quais documentos
65     requer o que
66
67 ~[dsc_documentos_afirmacao_inicio]
68     Tudo que    requerido para
69     Lista de documentos para
70     Documentos para
71
72 ~[dsc_documentos_objetivo_questao_inicio]
73     [O documento?] @[documento]    necess rio para
74     @[documento]    um dos documentos necess rios para
         realizar
75     Preciso de @[documento] para
76
77 ~[dsc_documentos_objetivo_questao_fim]
78     precisa de @[documento]
79     tem @[documento] como um de seus documentos necess rio
80     necessita de @[documento] para ser realizado
81
82 ~[dsc_documentos_estado_civil_questao_inicio]
83     [Quem   ?] @[estado_civil] precisa de quais documentos
```

```
      para
84    @[estado_civil] deve levar quais documentos se quiser
      realizar
85
86  ~[dsc_documentos_estado_civil_questao_fim]
87    precisa de quais documentos se @[estado_civil]
88    necessita de @[documento] para quem    @[estado_civil]
89
90  ~[dsc_documentos_estado_civil_afirmacao_inicio]
91    Lista de documentos que @[estado_civil] precisa para
92    Documentos necess rios para eu, que sou @[estado_civil
      ],
93
94  ~[dsc_documentos_objetivo_estado_civil_questao_inicio]
95    Quem    @[estado_civil] precisa de @[documento] para
96    Considerando que sou @[estado_civil], preciso levar @[
      documento] se quero
97    Preciso de @[documento], se sou @[estado_civil] e
      desejo
98
99  ~[dsc_documentos_objetivo_estado_civil_questao_fim]
100   exige @[documento] de quem    @[estado_civil]
101   requer que @[estado_civil] leve @[documento] para ser
      realizado
102   requisita @[documento] para algu m @[estado_civil]
103
104 ~[dsc_horarios_funcionamento_questao_inicio]
105   Quando posso
106   Em que momentos consigo
107   Quais os horarios de disponibilidade para
108
109 ~[dsc_horarios_funcionamento_questao_fim]
```

```
    funciona em quais hor rios
       possivel ser feito em quais hor rios
    est  disponivel em quais hor rios

~[dsc_horarios_funcionamento_afirmacao_inicio]
    dias e hor rios de funcionamento de
    hor rios de funcionamento de
    periodo de funcionamento do servi o

~[dsc_horarios_funcionamento_afirmacao_fim]
    horarios de funcionamento
    periodo de funcionamento

~[dsc_local_presencial_questao_inicio]
    Onde posso
    Onde devo ir para
    Para onde preciso me direcionar se quero
    em quais locais posso

~[dsc_local_presencial_questao_fim]
    esta disponivel em quais locais
    pode ser realizado em quais locais

~[dsc_local_presencial_afirmacao_inicio]
    lista de locais para
    locais em que   possivel
    regi es em que posso

~[dsc_local_presencial_afirmacao_fim]
    locais para ser realizado
    locais disponivel

```

```
142 ~[dsc_horarios_funcionamento_objetivo_questao_inicio]
143     @[dia] as @[horario#horas] posso
144     as @[horario#horas] de @[dia]    poss vel
145     No dia de @[dia] no perido da @[horario#turno] consigo
146     @[dia] no hor rio de @[horario#minutos] existe a
            possibilidade de
147
148 ~[dsc_horarios_funcionamento_objetivo_questao_fim]
149     esta disponivel nos dias de @[dia] as @[horario#minutos
            ]
150     pode ser realizado @[dia] de @[horario#turno]
151
152 ~[dsc_local_presencial_objetivo_questao_inicio]
153     @[cidade] permite a realizar o servi o
154     consigo, em @[cidade], realizar
155     est  disponivel em @[cidade] o servi o
156
157 ~[dsc_local_presencial_objetivo_questao_fim]
158     esta disponivel em @[cidade]
159     pode ser realizado em @[cidade]
160     pode ser feito em @[cidade] ou n o
161
162 @[servico#tipo1]
163     emiss o de certid o nada consta
164     solicita  o de carteira de habilita  o popular (CNH
            popular)
165     emiss o de certid o
166     solicitar cnh
167     carteira de habilita  o definitiva
168     cnh final
169     carteira permanente
170     identifica  o criminal
```

```
171     atestado de antecedentes criminais
172     antecedentes criminais
173     solicita  o carteira de identidade civil
174     solicita  o de RG
175     solicita  o de carteira de identidade (RG)
176
177 @[servico#tipo2]
178     emitir certid o (nada consta)
179     emitir certid o nada consta
180     emitir certid o em que n o consta nada
181     obter carteira de habilita  o
182     obter habilita  o
183     solicitar habilita  o
184     solicitar permiss o para dirigir
185     tirar cnh definitiva
186     emitir identifica  o criminal e atestando de
          antecedentes criminais
187     emitir CRLV
188     obter crlv
189     emitir licenciamento
190     emitir RG
191     obter identidade
192     obter carteira de identidade
193
194 @[documento]
195     [RG|carteira de identidade|c dula de identidade|
          identidade]
196     [fotos 3 4 |fotos|certid o de casamento original|
          c pia da certid o de casamento|certid o de
          casamento]
197     [averba  o de div rcio|carteira de trabalho|CTPS|
          reservista|carteira de reservista|certid o de
```

```
        nascimento dos filhos|certid o de nascimento dos
        pais]
198
199 @[estado_civil]
200     [solteiro|casado]
201     [solteira|casada]
202
203 @[dia]
204     [segunda|segunda-feira]
205     [ter a|ter a-feira]
206     [quarta|quarta feira|quarta-feira]
207     [quinta|quinta feira|quinta-feira]
208     [sexta|sexta-feira]
209     sabado
210
211 @[horario#horas]
212     [13h|1h pm]
213     [14h|2h pm|2 horas]
214     [15h|3 a tarde]
215     [16h|4h pm|4 horas]
216     [17h|5h pm|5 da tarde]
217     [18h|6h pm|6 horas]
218     [19h|7h pm]
219     [8h|8h am|8 horas]
220     [9h|9h am|9 da manh ]
221     [10h am|8 horas]
222
223 @[horario#minutos]
224     16:30
225     16h40m
226     8:48
227     10h 40 min
```

```
    12h 06m

@[horario#turno]
    [tarde|manh |noite]

@[cidade]
    Fortaleza
    Sobral
    Quixad
    Canind
    Juazeiro do Norte
    Itapipoca
    Quixeramobim
    Crato
```