Uma Investigação sobre a Aplicabilidade de Redes *Transformers* no Contexto de Tradução Automática para Língua Brasileira de Sinais

Renan Paiva Oliveira Costa

CENTRO DE INFORMÁTICA
UNIVERSIDADE FEDERAL DA PARAÍBA

João Pessoa, Fevereiro de 2024

Renan Paiva Oliveira Costa

Uma Investigação sobre a Aplicabilidade de Redes *Transformers* no Contexto de Tradução Automática para Língua Brasileira de Sinais

Dissertação de Mestrado em Informática apresentada ao Programa de Pós-Graduação em Informática do Centro de Informática, da Universidade Federal da Paraíba, como requisito para a obtenção do grau de Mestre em Informática.

Orientador: Prof. Dr. Tiago Maritan Ugulino de Araújo
Coorientador: Prof. Dr. Daniel Faustino Lacerda

Fevereiro de 2024

Catalogação na publicação
Seção de Catalogação e Classificação

C838i Costa, Renan Paiva Oliveira.
Uma investigação sobre a aplicabilidade de Redes Transformers no contexto de tradução automática para Língua Brasileira de Sinais / Renan Paiva Oliveira Costa. - João Pessoa, 2024.
98 f. : il.

Orientação: Tiago Maritan Ugulino de Araújo.
Coorientação: Daniel Faustino Lacerda.
Dissertação (Mestrado) - UFPB/CI.

1. Informática. 2. Tradução automática neural. 3. Língua Brasileira de Sinais. 4. Redes Transformers. 5. Busca automatizada de hiperparâmetros. 6. Linguagens de poucos recursos. I. Araújo, Tiago Maritan Ugulino de. II. Lacerda, Daniel Faustino. III. Título.

UFPB/BC CDU 004(043)

Elaborado por Larissa Silva Oliveira de Mesquita - CRB-15/746

UNIVERSIDADE FEDERAL DA PARAÍBA
CENTRO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

Ata da Sessão Pública de Defesa de Dissertação de Mestrado de Renan Paiva Oliveira Costa, candidato ao título de Mestre em Informática na área de Sistemas de Computação, realizada em 28 de fevereiro de 2024.

Aos vinte e oito dias do mês de fevereiro do ano de dois mil e vinte e quatro, às nove horas, no Centro de Informática da Universidade Federal da Paraíba, reuniram-se os membros da Banca Examinadora constituída para julgar o Trabalho Final do discente Renan Paiva Oliveira Costa, vinculado a esta Universidade sob a matrícula nº 20221004967, candidato ao grau de Mestre em Informática, na área de *"Sistemas de Computação"*, na linha de pesquisa *"Computação Distribuída"*, do Programa de Pós-Graduação em Informática. A comissão examinadora foi composta pelos professores: Tiago Maritan Ugulino de Araújo, Orientador e Presidente da banca; Thais Gaudencio do Rego, Examinadora Interna; Daniel Faustino Lacerda de Souza, Examinador Externo ao Programa; Yuri de Almeida Malheiros Barbosa, Examinador Externo ao Programa. Dando início aos trabalhos, o Presidente da Banca cumprimentou os presentes, comunicou a finalidade da reunião e passou a palavra ao candidato para que ele fizesse a exposição oral do trabalho de dissertação intitulado "**Uma Investigação sobre a Aplicabilidade de Redes Transformers no Contexto de Tradução Automática para Língua Brasileira de Sinais**". Concluída a exposição, o candidato foi arguido pela Banca Examinadora que emitiu o seguinte parecer: "***aprovado***". Do ocorrido, eu, Maria Alice Ferreira Bezerra, secretária do Programa de Pós-Graduação em Informática, lavrei a presente ata que vai assinada por mim e pelos membros da Banca Examinadora. João Pessoa, 28 de fevereiro de 2024.

gov.br Documento assinado digitalmente
MARIA ALICE FERREIRA BEZERRA
Data: 11/07/2024 12:23:40-0300
Verifique em https://validar.iti.gov.br

Maria Alice Ferreira Bezerra
SIAPE 2329447

gov.br Documento assinado digitalmente
TIAGO MARITAN UGULINO DE ARAUJO
Data: 02/04/2024 21:30:41-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Tiago Maritan Ugulino de Araújo
Orientador (PPGI-UFPB)

gov.br Documento assinado digitalmente
THAIS GAUDENCIO DO REGO
Data: 02/04/2024 16:43:33-0300
Verifique em https://validar.iti.gov.br

Prof. Drª. Thais Gaudencio do Rego
Examinadora Interna (PPGI-UFPB)

gov.br Documento assinado digitalmente
Daniel Faustino Lacerda de Souza
Data: 02/04/2024 13:51:41-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Daniel Faustino Lacerda de Souza
Examinador Externo ao Programa (UFPB)

gov.br Documento assinado digitalmente
YURI DE ALMEIDA MALHEIROS BARBOSA
Data: 02/04/2024 13:57:54-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Yuri de Almeida Malheiros Barbosa
Examinador Externo ao Programa (UFPB)

"O homem teme a escuridão e, por isso, raspa as bordas dela com fogo. Ele cria vida diminuindo a escuridão."

Rei Ayanami

Dedico este trabalho a todos os que me ajudaram ao longo desta caminhada.

AGRADECIMENTOS

A presente dissertação de mestrado não poderia ter atingido um resultado adequado sem o precioso apoio e contribuição de várias pessoas.

Em primeiro lugar, não posso deixar de agradecer aos meu orientadores, Professor Doutor Tiago Maritan Ugulino de Araújo e Professor Doutor Daniel Faustino Lacerda, por toda a paciência, empenho e sentido prático com que sempre me orientaram neste trabalho e em todos aqueles que realizei durante a condução desta pesquisa. Muito obrigado por me ter corrigido quando necessário, sem nunca me desmotivar.

Desejo igualmente agradecer a todos os meus colegas do Laboratório de Aplicações de Vídeo Digital e, em particular, aos pesquisadores e bolsistas do Projeto VLibras, especialmente a Diego Bezerra da Silva e Samuel Moreira, cujo apoio e prestatividade estiveram presentes em todos os momentos que precisei.

Por último, quero agradecer à minha família e amigos pelo apoio incondicional que me deram, especialmente aos meus pais pelas sugestões e revisões ao longo da elaboração deste trabalho.

RESUMO

Segmentos significativos da população mundial, incluindo a comunidade surda, não conseguem se beneficiar plenamente dos recursos da tradução automática neural (ou NMT, do inglês *Neural Machine Translation*), devido a vários desafios que os desenvolvedores enfrentam ao construir tais sistemas para linguagens de poucos recursos, como as línguas de sinais. Algumas pesquisas recentes de processamento de linguagem natural (PLN), com poucos recursos, focam na criação de novos mecanismos linguísticos e *benchmarks*, enquanto outra corrente busca personalizar soluções de NMT existentes, para novos idiomas e domínios. Adicionalmente, modelos recentes de PLN podem ser igualmente aplicáveis tanto para linguagens de poucos recursos, quanto para domínios sem tais limitações, e algumas correntes começam a investigar se novas técnicas de NMT também podem ser generalizadas para diferentes recursos - em termos de disponibilidade de dados e de recursos computacionais. Neste contexto, o foco deste estudo foi prospectar modelos de Redes *Transformers* e analisar a sua potencial aplicabilidade em contextos de poucos recursos, como é o caso das línguas de sinais. Para uma melhor avaliação, alguns dos modelos de Redes *Transformers* mais promissores foram adaptados e utilizados no componente tradutor da Suíte VLibras e os resultados obtidos comparados com os fornecidos atualmente pela arquitetura *LightConv*. Nesse contexto, o primeiro conjunto de experimentos foi focado em avaliar se tal adequação poderia também ser aplicada em contextos de PLN com poucos recursos (*low-resources NLP*), que é o caso das línguas de sinais. Os resultados obtidos indicam que a adoção de uma das duas arquiteturas melhor classificadas (*Transformer Básico* ou *ByT5*) ajudaria a aumentar a precisão e qualidade do componente de tradução da Suíte VLibras, trazendo um aumento percentual máximo de até 12,73% na métrica *BLEU*. A partir de prospecção e avaliação dos modelos em evidência, considerando que o processo de seleção de modelos candidatos teve um espaço de busca mais amplo, foi realizado um estudo mais aprofundado para tentar otimizar o modelo *Transformer Básico*. Na segunda fase de experimentação, foi realizada uma busca e uma varredura de hiperparâmetros relevantes considerando os seguintes hiperparâmetros: *bpe tokens*, *warmup updates*, *relu dropout*, *attention dropout*, *dropout*, *augmentation* e *learning rate*, que apontou que os três primeiros respondiam por cerca de 80% da capacidade de predição do modelo. Os resultados médios obtidos para a métrica *BLEU* usando o modelo *Transformer Básico*, com hiperparâmetros refinados, foram superiores em cerca de 17,45% do que média do modelo de referência e 8,95% melhor do que a média obtida pelo modelo melhor classificado na fase de prospecção, *ByT5*, o que indica um bom potencial da estratégia de refinamento da configuração de hiperparâmetros.

Palavras-chave: línguas de baixos recursos, tradução automática neural, língua de sinais, *Transformers*, busca aleatória, busca por hiperparâmetros.

ABSTRACT

Significant segments of the world population, including the deaf community, can not fully benefit from Neural Machine Translation (NMT) resources due to various challenges developers faced when building such systems for low-resource languages. Some recent research in Natural Language Processing (NLP) with low resources focuses on creating new linguistic mechanisms and benchmarks. At the same time, another approach aim to customize existing NMT solutions for new languages and domains. Additionally, recent NLP models may apply to low-resource languages and domains without limitations. Some works investigate whether new NMT techniques can also be generalized to different resources regarding data availability and computational resources. In this context, the general objective of this study is to explore Transformer models and analyze their potential applicability in low-resource contexts, which is the case for sign languages. We identified that transformer-based solutions are state-of-the-art for most NLP problems, becoming a new industry standard for various practical problems. For a better evaluation, we adapted and used some promising identified current Transformer models in the machine translation component of the VLibras Suite, and the obtained results were compared with those currently provided by the current LightConv architecture. The first set of experiments evaluated whether such adaptation could also be applied in machine translation from Brazilian Portuguese into Libras. The results indicate that adopting one of the two top-performing architectures (Vanilla Transformer or ByT5) would help increase the accuracy and quality of the translation component of the VLibras Suite, with a maximum percentage increase of up to 12.73% considering the BLEU metric. Through prospecting and evaluation of evident models, considering that the candidate model selection process had a broader search space, a more in-depth study was conducted to try to optimize one of the top-performing models, the Vanilla Transformer. In the second phase of experimentation, a random search and a sweep of relevant hyperparameters were conducted, considering the following hyperparameters in the search space: bpe tokens, warmup updates, relu dropout, attention dropout, dropout, augmentation, and learning rate, which indicated that the first three accounted for approximately 80% of the model's prediction capacity. The average results obtained for the BLEU metric using the reconfigured Vanilla Transformer model with optimized hyperparameters, the performance was superior by approximately 17,45% compared to the average of the reference model, and 8,95% better than the average achieved by the top-ranked model in the exploration phase, ByT5, which indicates a good potential for the hyperparameter configuration refinement strategy.

Key-words: low-resources languages, neural machine translation, sign language, Transformers, random search, hyperparameter search.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

ASL – *American Sign Language*

BART - *Bidirectional and Auto-Regressive Transformer*

BERT - *Bidirectional Encoder Representations from Transformers*

BLEU - *BiLingual Evaluation Understudy*

BPE - *Byte Pair Encoding*

CNN - *Convolutional Neural Networks*

DL - *Deep Learning*

EBMT - *Example-Based Machine Translation*

GRU - *Gated Recurrent Units*

IWSLT - *International Conference on Spoken Language Translation*

LAVID - Laboratório de Aplicações de Vídeo Digital

LIBRAS – Língua Brasileira de Sinais

LO - Língua Oral

LS – Língua de Sinais

LSTM - *Long Short-Term Memory*

MT - *Machine Translation*

NLP - *Natural Language Processing*

NMT – *Neural Machine Translation*

PLN - *Processamento de Linguagem Natural*

RBMT – *Rule-Based Machine Translation*

RNN – *Recurrent Neural Network*

Seq2Seq - *Sequence to Sequence*

SMT – *Statistical Machine Translation*

SMTC - *Spatial Temporal Multi Cue*

T5 - *Text-to-Text Transfer Transformer*

UFPB - Universidade Federal da Paraíba

VT - *Vanilla Transformer*

WMT - *Workshop on Statistical Machine Translation*

Conteúdo

1 INTRODUÇÃO

1.1 Contextualização

A comunidade surda, que representa uma parcela relevante da população brasileira e mundial, enfrenta diversos desafios no acesso à informação, normalmente disponibilizada através de língua escrita ou falada. Isso se deve principalmente ao fato de que a maioria dos surdos passam vários anos na escola, mas não conseguem atingir proficiência na leitura e escrita da língua oral de seu país [Souza et al. 2017].

O principal motivo para essa dificuldade é que os surdos comunicam-se naturalmente através de *línguas de sinais* (LS), sendo as *línguas orais* apenas uma espécie de segunda língua. Cada LS, por sua vez, é uma língua natural com léxico e gramática próprios, desenvolvida por cada comunidade de surdos ao longo do tempo, assim como cada comunidade de ouvintes desenvolveu a sua língua oral. Essa característica, própria de formação da língua, faz com que não exista uma língua de sinais única praticada em todo o mundo. Embora existam muitas similaridades entre todas essas línguas, cada país normalmente tem a sua própria LS, e alguns países possuem até mais de uma [Quadros 2006].

Para permitir o acesso adequado, o ideal, portanto, é que os conteúdos em línguas orais sejam traduzidos ou interpretados para a LS associada [Westin 2019]. Contudo, considerando o volume e dinamismo de informações em alguns ambientes e plataformas, como, por exemplo, na Web, fazer isso usando intérpretes humanos é uma tarefa inviável, mesmo se consideramos apenas o conteúdo que é adicionado diariamente na Internet. Para endereçar de forma pragmática essa questão, uma das abordagens mais promissoras atualmente é a utilização de ferramentas para tradução automática (*machine translation*) de uma língua oral para uma língua de sinais [Corrêa e Cruz 2019].

A **Tradução Automática** (ou tradução por máquina) é o processo de tradução entre uma língua fonte e uma língua alvo realizado exclusivamente por métodos automatizados assistidos por computador. Considerando a TA para LS no Brasil, a língua fonte seria a Língua Portuguesa, ou seja, o componente de inteligência recebe um texto em português para, a partir dele, fazer a tradução para glosas[1] na Língua Brasileira de Sinais (Libras), a língua alvo [Almeida 2013].

Um dos principais desafios dos sistemas de tradução automática para língua de sinais é garantir que o conteúdo disponibilizado aos surdos chegue com a mesma consistência e qualidade do original, permitindo assim o entendimento adequado da mensagem [Farooq et al. 2021]. A **Tradução Automática Neural**, por exemplo, que é geralmente baseada em **Aprendizagem Profunda** (ou DL, do inglês *Deep Learning*), normalmente

[1] Glosa é uma forma de tradução simplificada dos morfemas de uma língua oralizada para uma língua sinalizada.

utiliza bases de dados com exemplos de sentenças, tanto na língua de origem quanto na língua de destino, para aprender a realizar as traduções.

Para construir tradutores automáticos neurais para para qualquer idioma, um dos aspectos mais importantes é dispor de dados neste idioma [Koehn e Knowles 2017]. Existem mais de 7.000 idiomas falados em todo o mundo, mas desses idiomas, apenas cerca de 20 tem corpo (ou *corpus*[2]) de texto de centenas de milhões de palavras [Dryer e Haspelmath 2011]. O inglês é um dos idiomas com maior quantidade de dados, seguido do chinês e do espanhol. Outros idiomas com grandes conjuntos de dados incluem os idiomas da Europa Ocidental e também o idioma Japonês [Lewis 2014].

Por outro lado, a maioria dos idiomas falados na Ásia e na África não possuem os dados de treinamento necessários para construir sistemas NLP precisos. Essas linguagens são chamadas de **linguagens de baixos recursos** [Magueresse, Carles e Heetderks 2020]. Esse também é o caso da maioria das línguas de sinais, com uma quase que total inexistência de material oralizado natural (escrito ou falado) em LS e quase sempre com poucos *corpus* bilíngue e, quase sempre, de pequeno porte e produzidos artificialmente.

1.2 Justificativa e Motivação

Uma parcela significativa da população mundial, incluindo a comunidade surda, ainda é mal atendida pelos sistemas NLP, devido a vários desafios que os desenvolvedores enfrentam ao construir sistemas NLP para linguagens de poucos recursos, como as línguas de sinais [Haque, Liu e Way 2021, Magueresse, Carles e Heetderks 2020]:

- ***Falta de conjuntos de dados anotados para treinamento dos modelos***: conjuntos de dados anotados são necessários para treinar modelos DL de maneira supervisionada. Esses modelos são comumente usados para resolver tarefas específicas com muita precisão, como, por exemplo, na detecção de discurso de ódio. No entanto, a criação de conjuntos de dados anotados requer intervenção humana, rotulando exemplos de treinamento um por um, tornando o processo geralmente demorado e muito caro, considerando que modelos DL requerem milhares (ou milhões) de exemplos de treinamento [Munappy et al. 2019]. Assim, pode ser inviável contar apenas com a criação manual de dados a longo prazo.
- ***Falta de conjuntos de dados não rotulados***: conjuntos de dados não rotulados, como *corpus* de texto, são os precursores de suas versões anotadas. Eles são essenciais para treinar modelos básicos (pré-treinamento), que possam posteriormente ser ajustados ou refinados, para tarefas específicas. Portanto, abordagens

[2]Quando um *corpus* possui um conjunto de sentenças equivalentes em mais de uma língua são chamados *corpus* bilíngue. Conteúdos padrão em várias línguas, como, por exemplo, a bíblia, são uma ótima referência para a construção de *corpus* bilíngues.

para contornar a falta de conjuntos de dados não rotulados também se tornam muito importantes.

- ***Suporte a vários dialetos de um idioma***: os idiomas que possuem vários dialetos também são um problema complicado de resolver, especialmente para modelos de fala. Um modelo treinado em um idioma geralmente não terá um ótimo desempenho em seus diferentes dialetos. Por exemplo, a maioria dos conjuntos de dados não rotulados e anotados disponíveis para árabe estão em árabe padrão moderno. No entanto, para uma sensação humana ao interagir com assistentes de voz ou bate-papo para uso diário, é muito formal para muitos falantes de árabe. Assim, dialetos de suporte tornam-se necessários para casos de uso prático.

Algumas pesquisas recentes de NLP com poucos recursos (*low-resources NLP*) focam na criação de novos recursos linguísticos e *benchmarks*, enquanto outra corrente busca personalizar soluções de NLP existentes para novos idiomas e domínios [Magueresse, Carles e Heetdel 2020]. Adicionalmente, modelos modernos de NLP podem ser igualmente aplicáveis tanto para linguagens de poucos recursos quanto para domínios sem tais limitações, e algumas pesquisas começam a investigar se novas técnicas de NLP também podem ser generalizadas para diferentes recursos - em termos de disponibilidade de dados e disponibilidade de recursos computacionais [Haque, Liu e Way 2021].

Neste trabalho, nós investigaremos a aplicabilidade de modelos mais recentes de tradução automática neural para o contexto de Língua Brasileira de Sinais (Libras). Neste sentido, utilizaremos a Suíte VLibras [Araújo 2012] como ferramenta para essa investigação. A Suíte VLibras é o resultado de uma parceria entre o Ministério de Gestão e Inovação em Serviços Públicos (MGISP), através da Secretaria de Governo Digital (SGD), o Ministério de Direitos Humanos e da Cidadania (MDHC), via a Secretaria Nacional dos Direitos da Pessoa com Deficiência (SNDPD), e a Universidade Federal da Paraíba (UFPB), com execução pelo Laboratório de Aplicações de Vídeo Digital (LAVID). Ela consiste em um conjunto de ferramentas gratuitas e de código aberto para tradução automática de Português Brasileiro (texto, áudio e vídeo) para Libras, tornando computadores, dispositivos móveis e plataformas Web acessíveis para os surdos. Atualmente, o VLibras é usado em mais de 120.000 websites públicos e privados, dentre eles os principais sites do Governo Brasileiro (brasil.gov.br), da Câmara dos Deputados (camara.leg.br) e do Senado Federal (senado.leg.br) e está presente na vida cotidiana da comunidade surda através de milhões de traduções mensais[3]. Em seu núcleo, a Suíte VLibras utiliza um tradutor automático neural que traduz textos em Português Brasileiro para a glosa em Libras. O resultado da tradução posteriormente é utilizado por uma interface para a apresentação dos sinais em Libras através da renderização de um avatar 3D.

[3]Mais informações podem ser obtidas em http://www.vlibras.gov.br.

O componente principal de um sistema de tradução automática é seu componente de inteligência. O VLibras atualmente usa a arquitetura *LightConv* [Wu et al. 2019], sendo uma arquitetura de rede neural utilizada para tarefas de PLN. Ela é baseada em uma arquitetura de **Rede Neural Convolucional** (CNN, sigla em inglês para *Convolutional Neural Network*) e foi projetada para lidar com textos de comprimento variável. Essa arquitetura tem sido aplicada em diversas tarefas de PLN como: classificação de texto, tradução automática, entre outras.

O foco desta pesquisa é avaliar se, dentre os modelos mais destacados que surgiram desde este período, qual(is) pode(m) proporcionar resultados melhores do que o modelo usado atualmente na Suíte VLibras.

1.3 Objetivo Geral

Neste contexto, tem-se como objetivo geral investigar a viabilidade e potencial aplicabilidade das arquiteturas *Transformers* no contexto de tradução automática para Libras, e se elas poderiam superar os resultados atuais da Suíte VLibras.

Para uma melhor avaliação, alguns modelos *Transformers* promissores identificados na literatura foram adaptados e utilizados no componentes tradutor da Suíte VLibras e os resultados obtidos comparados com os fornecidos atualmente pela plataforma para avaliar se as arquiteturas baseadas em Redes Transformadoras podem representar alternativas para a melhoria na qualidade da tradução Português-Libras disponível atualmente.

1.4 Objetivos Específicos

Para atingir o objetivo geral deste trabalho, os seguintes objetivos específicos foram desenvolvidos:

- Identificar um conjunto de modelos *Transformers* relacionados com processamento de linguagem natural e tradução automática;
- Selecionar os modelos *Transformers* potencialmente aplicáveis em tradução automática, em geral, e a tradução automática de línguas de sinais, em particular;
- Avaliar os modelos *Transformers* candidatos selecionados usando, como referência, o componente tradutor da Suíte VLibras;
- Refinar o modelo *Transformer* com melhor desempenho e/ou aplicabilidade através de busca automática de hiperparâmetros relevantes;
- Reconfigurar o modelo *Transformer* refinado, via varredura dos hiperparâmetros relevantes;

- Avaliar o uso outras técnicas complementares, como *data augmentation* e *back translation*.

1.5 Organização da Dissertação

O restante da dissertação está organizada como segue. No Capítulo 2 é apresentada uma fundamentação teórica básica sobre tradução automática, em geral, e tradução automática neural, em particular, que ajudam no entendimento de conceitos abordados nesta pesquisa. No Capítulo 3 é descrito como foi realizada a prospecção de modelos baseados em Redes *Transformers* de tradução automática, aplicáveis em contextos de *low-resources NLP* e língua de sinais. No Capítulo 4 é detalhada como foi realizada a seleção dos modelos *Transformers* candidatos à avaliação. O Capítulo 5, por sua vez, descreve a metodologia usada na fase experimental de avaliação dos modelos *Transformers* candidatos e apresenta e discute os resultados obtidos. No Capítulo 6 é discutida a segunda fase experimental, voltada para a busca e varredura de hiperparâmetros relevantes objetivando otimizar o modelo *Transformer* melhor classificado na etapa anterior, o *Transformer Básico*[4]. Finalmente, no Capítulo 7, são feitas as considerações e trabalhos futuros acerca dos resultados obtidos na pesquisa.

[4]Neste documento, os termos *Transformer Básico* e *Vanilla Transformer* serão usados igualmente e alternadamente para se referir a arquitetura de rede neural que utiliza atenção multi-cabeça e camadas empilhadas de autoatenção e redes de alimentação direta para processar sequências de dados, sendo fundamental em diversas aplicações de processamento de linguagem natural, incluindo tradução automática e geração de texto.

2 FUNDAMENTAÇÃO TEÓRICA

NLP é uma área interdisciplinar que estuda o processamento por máquinas da linguagem humana com vistas à resolução de vários desafios que envolvem compreensão, tradução e interpretação de dados baseados em linguagens naturais [Nadkarni, Ohno-Machado e Chapman 2011]. NLP pode ser utilizado para diversos propósitos como: tradução automática, reconhecimento de fala, síntese de fala e de texto, análise de sentimento, entre outros [Hirschberg e Manning 2015]. Dentre os temas estudados no contexto de NLP, a tradução de linguagens por máquinas (*Machine Translation* - MT) é um dos mais ativos, por sua relevância e aplicabilidade em vários aspectos práticos da vida cotidiana. Neste campo, investiga-se o uso de *softwares* para tradução de texto e voz de uma linguagem natural para outra. Esta abordagem vem sendo tratada por pesquisadores há algum tempo e vários avanços puderam ser alcançados nos últimos anos. O restante do capítulo vai apresentar os principais sistemas, técnicas e modelos usados em tradução automática.

2.1 Sistemas de Tradução Automática

A TA ou MT (do inglês, *Machine Translation*) é uma vertente da computação linguística que define a automação da tradução de textos de uma língua natural (fonte) para outra (destino), de tal forma que essa prática não altere o sentido original da informação [Kituku, Muchemi e Nganga 2016, Okpor 2014, Russel e Norvig 2004].

Nos anos 40 [Weaver 1949] já se vislumbrava a utilização de computadores para a realização de processos de tradução textual, mas apenas nas últimas décadas a tecnologia teve seu uso ampliado, principalmente por causa de uma evolução considerável do poder computacional tecnológico [Wang et al. 2022]. Isto possibilitou o aperfeiçoamento das técnicas de tradução automática de tal forma que, em muitos cenários, é difícil diferenciar a tradução realizada por um computador da tradução realizada por um intérprete humano [Farooq et al. 2021].

Algumas das dificuldades enfrentadas por tradutores humanos também se aplicam à TA. Boa parte delas se dá pelo fato da maioria das palavras serem polissêmicas e por sentenças possuírem grande tamanho e complexidade estrutural, o que impacta diretamente na qualidade e tempo de tradução [Mishra, Bhattacharyya e Carl 2013]. A TA, de forma geral, requer um conhecimento profundo do texto [Russel e Norvig 2004].

Num sistema de TA, primeiramente o texto na língua de origem (LO) é analisado e criada uma representação interna do mesmo. Em seguida, essa representação é manipulada e transferida para a forma da língua de destino (LD) e, só então, o texto é gerado na LD [Okpor 2014]. A escolha da estratégia de tradução que o sistema utilizará reflete

a profundidade e diversidade linguística necessária para as línguas e a complexidade que o sistema deve ter [Chéragui 2012].

Os diferentes sistemas de TA, explicados em mais detalhes nas próximas seções, podem ser classificados pelas metodologias utilizadas para realizar a tradução [Okpor 2014, Kituku, Muchemi e Nganga 2016]:

- Tradução Automática Baseada em Regras (do inglês, *Rule-Based Machine Translation* - RBMT), onde regras que descrevem a tradução entre uma LO e uma LD são feitas por humanos especialistas nas línguas;
- Tradução Automática Baseada em *Corpus* (do inglês, *Corpus-Based Machine Translation* - CBMT), também referenciada como *Data Driven Machine Translation* (DDMT), onde um conjunto de dados, *corpus* bilíngue ou *corpus* paralelo, é utilizado para extrair o conhecimento da língua;
- Tradução Automática Neural (do inglês, *Neural Machine Translation* - NMT), uma abordagem de TA que, embora também seja baseada em *corpus*, engloba os modelos que usam uma rede neural artificial para prever a probabilidade de uma sequência de palavras, normalmente modelando e depois traduzindo frases inteiras em um único modelo integrado.

Em Kahlon e Singh (2021) é feito um levantamento sistemático das principais técnicas utilizadas para endereçar o problema de tradução. Neste sentido, os autores apresentam uma classificação dos métodos de MT conforme ilustrado na Figura 1.

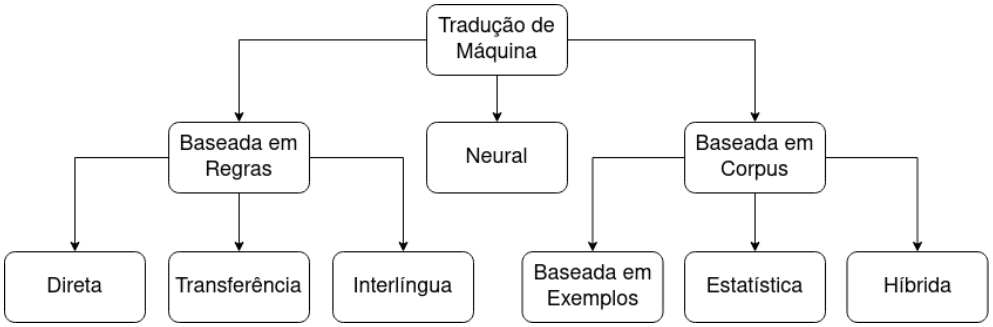

Figura 1: Classificação de Métodos de Tradução Automática
Fonte: [Kahlon e Singh 2021]

2.1.1 Tradução Automática Baseada em Regras – RBMT

A RBMT é tida como a "abordagem clássica" da TA e ainda pode ser encontrada em sistemas comerciais [Stein 2018]. Esse tipo de sistema de tradução consiste em uma coleção de regras, chamadas de regras gramaticais, um léxico bilíngue ou multilíngue, e um software que processa as regras [Antony 2013].

As regras gramaticais basicamente consistem na análise da LO e na geração da tradução na LD obedecendo o nível estrutural da gramática, principalmente no que diz respeito à sintaxe, semântica e morfologia de ambas as línguas. Essa abordagem depende fortemente de um amplo conhecimento da língua trabalhada e algumas soluções contam com o auxílio de linguistas nesse processo [Kituku, Muchemi e Nganga 2016, Cheragui 2012].

Dentro da abordagem RBMT existem três sub-abordagens que se diferem quanto à profundidade e quanto à forma em que tentam alcançar uma linguagem de representação, independente de significado ou intenção, entre os idiomas de origem e destino [Okpor 2014]:

- **Tradução Direta**: Realiza um tradução básica, a partir da troca de palavras do texto na língua de origem, pelas palavras com o mesmo significado na língua de destino, para isso, é utilizado um dicionário contendo essas palavras. Essa estratégia também pode incluir algumas regras para alterar a posição das palavras. Em geral, essa técnica é utilizada em mensurações teóricas para demonstrar os benefícios e avanços dos sistemas de tradução, apesar de historicamente ter sido a abordagem utilizada pelos primeiros sistemas de TA[Rehm et al. 2018];

- **Tradução por Transferência**: Nessa abordagem são necessárias as definições das regras morfológicas, sintáticas e semânticas para a tradução entre uma LO e uma LD [Rehm et al. 2018]. Essa estratégia de tradução pode ser dividida nos estágios de análise, transferência e geração. O primeiro estágio consiste em converter o texto na LO numa representação intermediária, geralmente uma representação sintática em forma de árvore [Chéragui 2012, Okpor 2014]. No próximo estágio, as regras sintáticas e semânticas são utilizadas no resultante do estágio anterior, a fim de converter essa representação intermediária nos moldes da LD. No último estágio é gerada a representação final do texto na LD, pela aplicação de regras e substituições morfológicas [Okpor 2014];

- **Interlíngua**: É uma estratégia de TA na qual se define uma língua universal intermediária, de tal forma que a mesma seja abstrata, homogênea, independente e sem ambiguidade [Kituku, Muchemi e Nganga 2016]. Primeiramente, o texto na LO será analisado e então seu conteúdo semântico extraído e representado na forma da língua universal. A partir do texto nessa língua intermediária, o texto na LD será gerado [Chéragui 2012]. Apesar de ser uma ótima estratégia para sistemas de tradução multilíngua, onde se tem várias LO e várias LD, a construção da língua universal intermediária acaba sendo um grande desafio [Rehm et al. 2018].

A dissimilaridade dessas sub-abordagens pode ser observada na Figura 2, ilustrada

pelo *Triângulo de Vauquois*, que representa os níveis de análise que cada uma dessas estratégias exerce sobre a língua [Vauquois 1968].

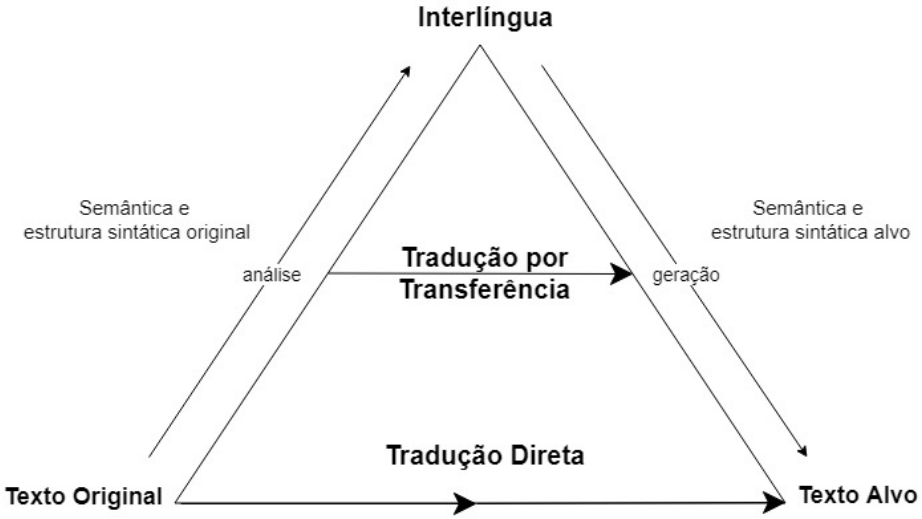

Figura 2: Triângulo de Vauquois
Fonte: Adaptado de [Vauquois 1968]

As abordagens de tradução baseadas em regras se apoiam na análise morfológica, sintática e semântica das linguagens fonte e alvo. Este processo pode ser feito de forma: i) **direta**, traduzindo palavra por palavra, utilizando um dicionário bilingual após a aplicação de análise morfológica; e ii) **interlíngua**, traduzindo o texto fonte para uma representação abstrata independente e a partir desta para a linguagem alvo. Um terceiro processo, denominado de **transferência**, também usa uma abordagem similar a interlíngua, contudo a linguagem intermediária deve possuir certa dependência em relação às linguagens fonte e alvo.

Okpor (2014) relata alguns problemas relacionados à estratégia RBMT, como: quantidade insuficiente de bons dicionários; a construção de novos dicionários é custosa; algumas informações linguísticas ainda precisam ser definidas manualmente; é difícil lidar com a relação entre regras e sistemas de grande porte; questões de ambiguidade e expressões idiomáticas, e; falhas para se adaptar a novos contextos. Entretanto, essa abordagem tem um bom desempenho de tradução, no que diz respeito a fluência, fidelidade, pós-processamento e precisão, principalmente em casos onde não se dispõe de uma grande disponibilidade de dados, que permitam a experimentação de outras estratégias [Kituku, Muchemi e Nganga 2016].

2.1.2 Tradução Automática Baseada em *Corpus* – CBMT

As abordagens baseadas em *corpus* requerem a existência de grandes conjuntos bilíngues (bancos de sentenças equivalente de ambas as línguas). Uma das classes de

técnicas derivadas desta abordagem é a tradução baseada em exemplos, onde é feita a identificação e alinhamento de sentenças correspondentes, em ambas as partes do *corpus* paralelo. Técnicas desta natureza usam tradução por analogia, a partir da comparação entre os *corpus* de cada linguagem. Outra classe de técnicas baseadas em *corpus* é a tradução estatística. Em tal abordagem, a tradução se vale de grandes *corpus* bilíngues para construção de modelos estatísticos, que podem ser usados para inferência de sentenças da linguagem alvo, a partir da linguagem fonte. Ainda no contexto das abordagens baseadas em *corpus*, temos os métodos híbridos que derivam sistemas de tradução, a partir da combinação de múltiplas técnicas/abordagens.

Tradutores do tipo CBMT se utilizam do *corpus* bilíngue para adquirir conhecimento sobre sobre a língua e utilizá-lo em novas traduções. Essa é uma alternativa para o maior problema das RBMT: a aquisição de conhecimento sobre a língua de forma manual. Usualmente, o *corpus* é composto de textos na LO e uma tradução equivalente para a LD [Okpor 2014].

A abordagem CBMT se desdobra em diferentes estratégias, as principais são: **Tradução Automática Estatística** (do inglês, *Statistical Machine Translation* - SMT); **Tradução Automática Baseada em Exemplos** (do inglês, *Example-Based Machine Translation* - EBMT); **Tradução Automática Neural** (do inglês, *Neural Machine Translation* - NMT).

As duas primeiras serão detalhadas a seguir, já a estratégia NMT, pelo fato de ser a estratégia de maior destaque atualmente e também por ser o foco maior desse trabalho, será explorada separadamente na Seção 2.1.3:

- **Tradução Automática Estatística**: Os sistemas do tipo SMT realizam a tradução baseada em modelos estatísticos extraídos a partir de *corpus* bilíngues ou multilíngues [Modh e Saini 2018]. Nessa estratégia, uma sentença S na LO será traduzida para uma sentença T na LD, de acordo com uma função de distribuição probabilística P indicada por $P(S|T)$ [Antony 2013]. Essa abordagem de tradução ainda se divide em outras três: **SMT Baseada em Palavras** (do inglês, *Word Based SMT*), onde as sentenças são divididas em unidades (palavras) que serão traduzidas individualmente - posteriormente é utilizado um algoritmo para a ordenação das palavras traduzidas [Kituku, Muchemi e Nganga 2016]; **SMT Baseada em Frases** (do inglês, *Phrase Based SMT*), onde o texto de entrada será dividido em sentenças que serão traduzidas individualmente - essa estratégia traz resultados melhores que a anterior; **SMT Baseada em Frases Hierárquicas** (do inglês, *Hierarchical Phrases Based SMT*), que combina a SMT Baseada em Frases e uma tradução baseada em sintaxe. As frases serão consideradas um segmento de tradução, enquanto que a tradução baseada em sintaxe traz algumas regras de tradução. A vantagem dessa

estratégia é que as frases são consideradas estruturas recursivas, em vez de somente texto [Kituku, Muchemi e Nganga 2016, Antony 2013].

- **Tradução Automática Baseada em Exemplo**: A ideia principal da EBMT é a tradução por similaridade [Modh e Saini 2018]. O texto de entrada na LO é dividido em fragmentos, que dependem da granularidade do sistema (palavra, frase, etc). Esses fragmentos são comparados com os contidos no *corpus*. O fragmento do *corpus* que mais se assemelhar ao texto de entrada será escolhido e a tradução desse fragmento, que também está contido no *corpus*, será colocado no texto final de saída [Kituku, Muchemi e Nganga 2016].

2.1.3 Tradução Automática Neural – NMT

Outra abordagem utilizada para tratar o problema de tradução é a NMT. O termo **Tradução Automática Neural** (do inglês, *Neural Machine Translation*) foi cunhado por Cho [Cho et al. 2014] para classificar as novas abordagens de SMT que estavam surgindo [Kalchbrenner e Blunsom 2013, Sutskever, Vinyals e Le 2014, Cho et al. 2014] e que propunham tradutores automáticos puramente compostos de Redes Neurais (do inglês, *Neural Networks* - NN). As NMT trazem um significativo avanço sobre as SMT, de forma prática e teórica. Se torna uma estratégia atrativa aos pesquisadores por necessitarem de apenas uma fração da memória requerida pelas SMT tradicionais [Cho et al. 2014].

Uma NMT utiliza redes neurais artificiais para predizer as chances de uma sequência de palavras a partir de um modelo integrado. Esta abordagem pode utilizar DL ou *Aprendizado de Representação* (do inglês *Representation Learning*) para o processo de tradução. O mecanismo central é baseado na codificação de sequências variáveis de palavras em vetores, que representam a sentença completa. A partir deste vetor, uma sentença alvo é obtida a partir da produção de um vetor correspondente. Por fim, o decodificador gera a sentença traduzida, a partir do vetor correspondente à luz do modelo definido para a linguagem alvo [Brour e Benabbou 2021].

Numa NMT, ao contrário das outras abordagens, todos os seus componentes devem ser treinados juntos a fim de maximizar a performance de tradução [Cho et al. 2014]. Nesse contexto surge a arquitetura *Encoder-Decoder*, proposta por Cho et al. (2014). Nessa arquitetura, o *encoder* é responsável por aprender a realizar a codificação de um vetor de tamanho variável em um vetor de tamanho fixo. O *decoder*, por sua vez, dado um vetor de tamanho fixo, deve aprender a representá-lo novamente em um vetor de tamanho variável [Cho et al. 2014]. No contexto de NLP, o *encoder* processa o texto na língua fonte e o codifica em um vetor que representa a informação, e o *decoder* tem a função de decodificar esse vetor representando-o na língua destino.

A Figura 3 ilustra o funcionamento dos módulos *encoder* e *decoder* na arquitetura

original de Cho et al (2014). Na prática, o *encoder* é uma NN que lê cada símbolo de entrada x sequencialmente. A cada leitura, o estado interno da rede muda e o *encoder* utiliza o estado oculto anterior e o símbolo no tempo t atual para gerar o próximo estado. A leitura acaba quando o símbolo de fim de *string* (*end of string* - EOS) é encontrado. Por fim, toda a sequência de entrada está sumarizada no vetor de tamanho fixo c. O *decoder* é treinado para gerar a sequência de saída através da predição do símbolo y_t. Diferentemente do *encoder*, o *decoder* a cada passo utiliza o estado oculto anterior, o símbolo gerado em $t-1$ (y_{t-1}) e o vetor sumarizado c.

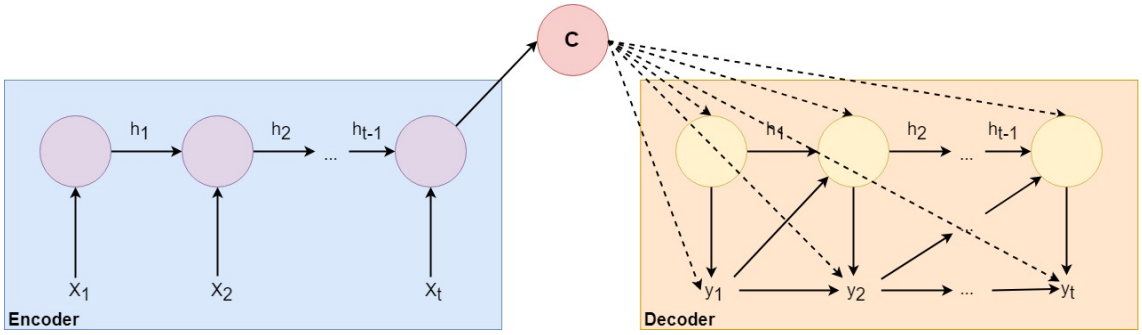

Figura 3: Arquitetura *Encoder-Decoder, onde* x_t *é a entrada no tempo t e* h_{t-1} *o estado oculto anterior.*
Fonte: [Cho et al. 2014]

2.2 Evolução dos Mecanismo de Tradução baseados em NMT

Os mecanismos de tradução baseados em Redes Neurais (NMT) têm evoluído significativamente desde sua introdução. Além disso, técnicas como o uso de pré-treinamento de modelos em grandes conjuntos de dados e a incorporação de informações linguísticas adicionais, como estruturas de dependência, têm sido exploradas para melhorar ainda mais o desempenho e a generalização dos modelos de NMT. Essa evolução contínua visa superar desafios como a TA de idiomas de baixo recurso e a melhoria da qualidade das traduções em várias línguas e domínios.

2.2.1 Redes Neurais Recorrentes

No contexto de NMT, o problema de tradução é comumente endereçado com o emprego de *Redes Neurais Recorrentes* (RNNs). Esta classe de NN inclui o conceito de memória a partir do uso de conexões de entrada, que consideram estados derivados de entradas anteriores. Esta característica permite que RNNs consigam capturar comportamento dinâmico temporal e consequentemente, possam manipular dados sequenciais. Na Figura 4 é ilustrada a arquitetura de uma RNN, considerando x_t como sua entrada no tempo **t** e h_t seu estado no tempo **t**. A Figura 4 ilustra ainda a visão ao longo do tempo de como a RNN pode ser retroalimentada por estados anteriores.

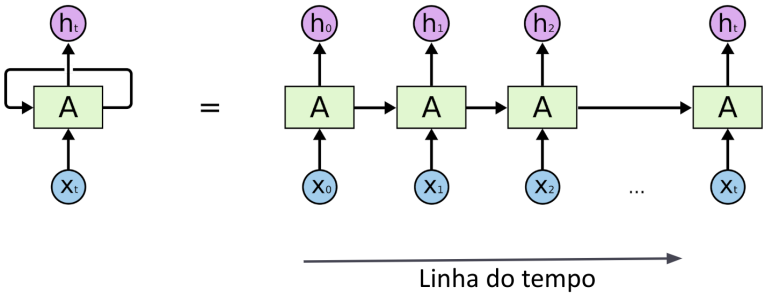

Figura 4: Dinâmica de Funcionamento de uma RNN, onde x_t é a entrada no tempo t e h_t o seu estado no tempo t.
Fonte: [Doell 2020]

As RNNs podem ser aplicadas a vários problemas relacionados à área de processamento de linguagens naturais e reconhecimento de fala. A depender do problema, diferentes tipos de RNNs podem ser utilizadas. Do ponto de vista de entradas e saídas, as RNNs podem ser classificadas em *um-para-um*, *um-para-muitos*, *muitos-para-um* e *muitos-para-muitos* [Doell 2020] (Figura 5).

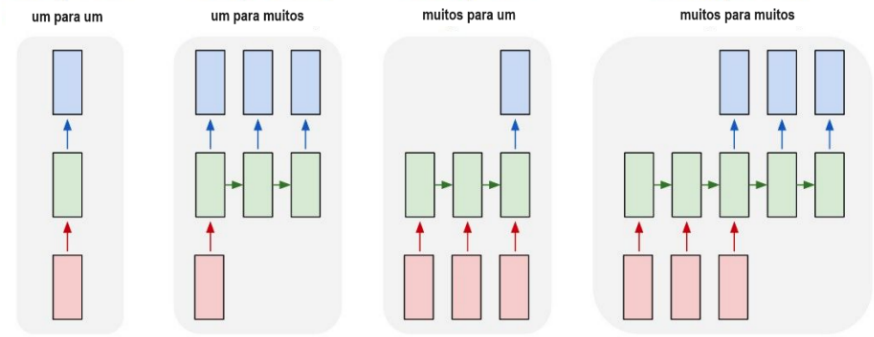

Figura 5: Tipos de RNN
Fonte: [Doell 2020]

A Tabela 1 relaciona cada um dos tipos de RNNs a um exemplo de aplicação relacionada às áreas de NLP e reconhecimento de fala.

Tabela 1: Tipos de RNNs e Exemplos de Área de Aplicação

Tipo de RNN	Exemplo de Aplicação
um-para-um	Rede neural tradicional [Coelho 2020]
um-para-muitos	Síntese de música [Santos 2019]
muitos-para-um	Análise de sentimento [Barnes 2019]
muitos-para-muitos	Tradução automática [Santos 2021]

As RNNs clássicas têm como desvantagem a dificuldade de lidar com dependências de longo prazo. Sentenças muito longas podem afetar a acurácia do mecanismo de classi-

ficação [Ribeiro et al. 2020]. No contexto do problema de tradução, isto é um desafio, uma vez que em muitos casos deseja-se lidar com a tradução de sentenças muito longas. Esta limitação é devida a um problema de desvanecimento do gradiente (*vanishing gradient*), algo intrínseco à natureza de redes neurais que precisam propagar o erro retroativamente até o início da sequência para realizar a predição da saída [Hochreiter 1998].

A arquitetura *Encoder-Decoder* de Cho et al. (2014) foi originalmente projetada para a utilização com RNNs, mas, com o passar do tempo, novos modelos neurais foram surgindo e essa arquitetura foi sendo modificada para alcançar a máxima eficiência dessas redes. Uma das primeiras adaptações a se popularizar foi a utilização de redes LSTM (*Long-Short Term Memory*) [Hochreiter e Schmidhuber 1997], um tipo de RNN que possui a capacidade de aprender dados com dependências temporais de longo alcance [Sutskever, Vinyals e Le 2014], como textos. Um texto nada mais é que um conjunto de elementos que possuem dependências temporais entre si, ou seja, uma palavra no tempo t pode influenciar o sentido de uma palavra no tempo $t + n$, assim como ela pode ser influenciada por uma palavra no tempo $t - n$.

As redes LSTM possuem a capacidade de lidar com o problema de dependências de longo prazo [Hochreiter e Schmidhuber 1997]. Neste sentido, o conceito de célula de memória foi introduzido com o objetivo de lidar com o problema de desaparecimento do gradiente (Figura 6). Estas células de memória são compostas por portões (ou *gates*) responsáveis por determinar quais informações devem ser mantidas ou descartadas, quando do processo de propagação das informações de estados anteriores.

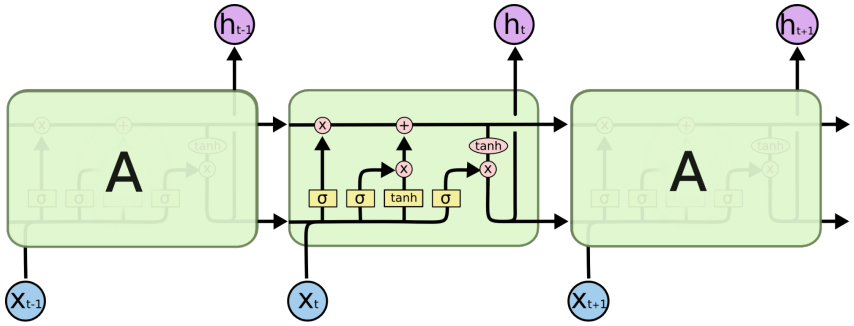

Figura 6: Representação de uma rede LSTM com destaque para a estrutura da célula de memória, onde h_t são vetores de camada oculta, x_t são vetores de entrada e σ e tanh **são funções de ativação.**
Fonte: [Doell 2020]

Outra abordagem similar às células de memória usadas nas redes LSTM foi proposta por Cho et al. (2014). Os autores propuseram o que atualmente é denominado de *Gated Recurrent Units* (GRUs). O mecanismo de memória das RNNs baseadas em GRU é mais simples do que o apresentado pela LSTM, uma vez que usa apenas dois tipos de *gates* (*reset gate* e *update gate*) para lidar com o problema de transferência de

informação. Na Figura 7 as diferenças entre as estruturas das redes recorrentes clássicas, LSTM e GRU são ilustradas.

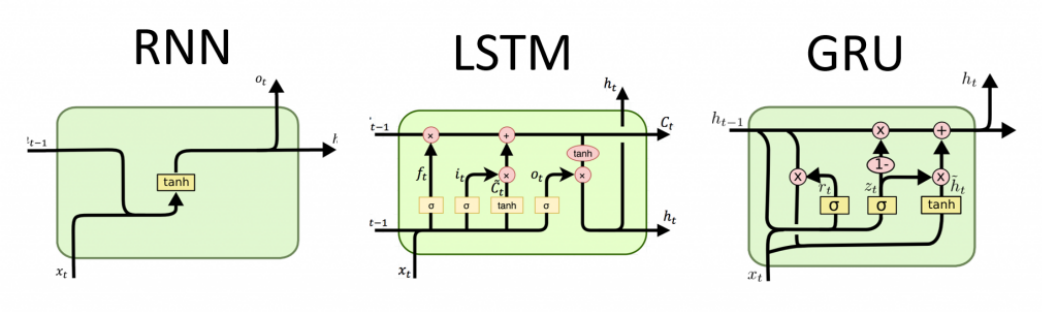

Figura 7: Diferenças entre os mecanismos de memória de RNNs clássicas (sem mecanismo de memória), LSTM e GRUs, , onde h_t são vetores de camada oculta, x_t são vetores de entrada e σ e tanh são funções de ativação. Fonte: [Doell 2020]

Como pode ser visto na Figura 7, as RNNs clássicas têm apenas uma camada oculta que processa a entrada e atualiza seu estado interno em cada etapa de tempo. Elas tendem a sofrer do problema de desvanecimento do gradiente, que dificulta o aprendizado de dependências de longo prazo. As LSTMs foram projetadas para superar o problema de desvanecimento do gradiente e aprender dependências de longo prazo. Para isso, possuem três portas principais: porta de esquecimento (*forget gate*), porta de entrada (*input gate*) e porta de saída (*output gate*), que regulam o fluxo de informações na célula de memória. As GRUs são uma variante mais simples das LSTMs, com menos portas e, portanto, menos parâmetros. Elas combinam as funções de atualização e esquecimento das LSTMs em uma única "porta de atualização", tornando-as computacionalmente mais eficientes.

Na próxima seção, será apresentado o problema de lidar com dados sequenciais usando redes neurais e os modelos que endereçam os desafios associados.

2.2.2 Modelos *Sequence to Sequence* (*Seq2Seq*)

Lidar com o tratamento de sequências sempre foi um desafio para NN, uma vez que esta tarefa envolve o tratamento de sentenças com tamanhos que não são conhecidos *a priori*. Métodos baseados em redes neurais têm como requisito que a dimensionalidade das entradas e saídas sejam conhecidas e tenham tamanho fixo. Os trabalhos de Kalchbrenner e Blunsom (2013), Cho et al. (2014), Sutskever, Vinyals e Le (2014) deram passos importantes no sentido de permitir endereçar o problema de lidar com dados sequenciais usando NN. A partir disto, surgiram os modelos *Seq2Seq* para lidar com o desafio de tradução automática entre linguagens.

A tradução de textos pode ser mapeada como um problema *Seq2Seq*, onde uma sequência de caracteres, ou palavras (dependendo da granularidade trabalhada), deve ser

mapeada para uma outra sequência de caracteres, ou palavras [Sutskever, Vinyals e Le 2014]. Esse trabalho foi um dos que ajudou a popularizar a alta performance das DNNs (do inglês *Deep Neural Networks*) em tarefas de TA. Além disso, juntamente com a pesquisa de Cho et al. (2014), ajudou a consolidar a arquitetura *Encoder-Decoder* como a mais adequada para a construção de tradutores neurais.

Conforme ilustrado na Figura 8, no modelo *Seq2Seq* são usadas duas RNNs diferentes. Esta abordagem consiste em mapear a sequência de entrada para um vetor, utilizando a primeira RNN (*encoder*), e então mapear este vetor resultante para uma sequência alvo, utilizando a segunda RNN (*decoder*).

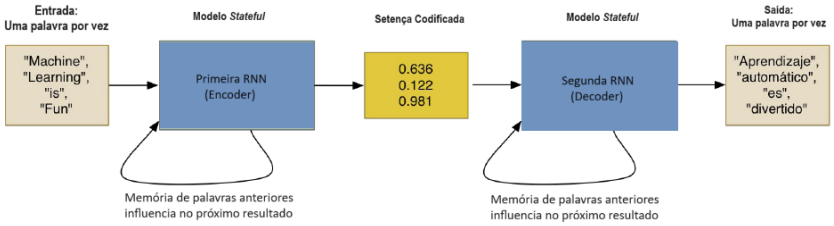

Figura 8: Visão Geral de um Modelo Seq2Seq
Fonte: [Sutskever, Vinyals e Le 2014]

Na Figura 9, por sua vez, está ilustrada a operação de um modelo Seq2Seq usando LSTMs como RNNs no *encoder* e *decoder*, conforme proposto por Sutskever, Vinyals e Le (2014).

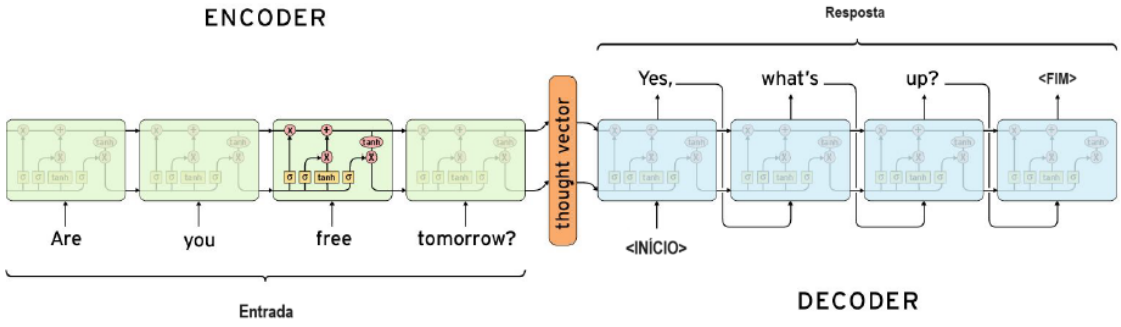

Figura 9: Modelo *Seq2Seq* Aplicado ao Problema de Pergunta-Resposta usando LSTMs, onde x são vetores de entrada e σ e $\tanh$ são funções de ativação.
Fonte: [Sutskever, Vinyals e Le 2014]

Esta abordagem utiliza uma arquitetura composta por um codificador LSTM para processar a pergunta e gerar uma representação latente, e um decodificador LSTM que gera a resposta com base nessa representação. Durante o treinamento, pares de pergunta-resposta são utilizados para ajustar os pesos do modelo, visando aprimorar a capacidade

de gerar respostas coerentes. Durante a inferência, o modelo codifica a pergunta e, utilizando o decodificador, gera a resposta *token* a *token* até atingir um textittoken de fim de sequência. Tal estratégia permite a geração automatizada de respostas para perguntas em tempo real.

2.2.3 Mecanismo de Atenção

A área de NMT conseguiu grandes avanços com o uso de RNNs com memória e uso de arquiteturas *Seq2Seq*. Este tipo de arquitetura, contudo, ainda sofria com problemas relacionados a sentenças longas, como é o caso do **Problema do Gargalo** (do inglês, *Bottleneck Problem*). O *encoder*, o qual gera o vetor com a sentença codificada, precisa capturar todas as informações necessárias para representar a sentença fonte. Isto gera um problema de representação, principalmente para os casos de sentenças longas [Cho et al. 2014, Bahdanau, Cho e Bengio 2014]. A Figura 10 ilustra o problema do gargalo. Nela, o último vetor (**h4**) produzido como saída da RNN codificadora serve como entrada para RNN decodificadora e toda a informação de contexto da sentença está representada neste vetor, o que pode levar a uma menor qualidade no processo de tradução.

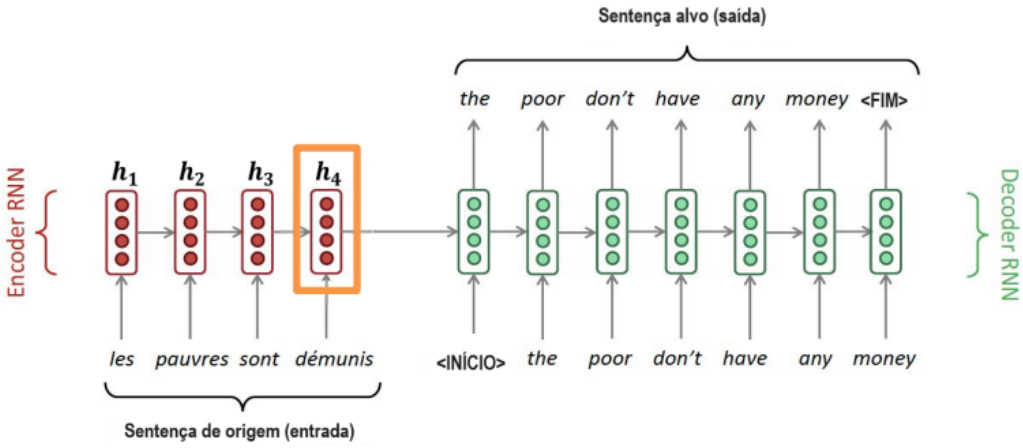

Figura 10: Problema do Gargalo, no qual o último vetor (h4) produzido como saída da RNN codificadora serve como entrada para RNN decodificadora e toda a informação de contexto da sentença está representada neste vetor.
Fonte: [Bahdanau, Cho e Bengio 2014]

Outra evolução significativa foi a adição de um mecanismo de atenção (do inglês, *attention mechanism*) proposta por Bahdanau et al. (2014), que era uma estratégia utilizada em outros contextos, como, por exemplo, na visão computacional. Ele se baseia na ideia de que o ser humano rastreia e reconhece padrões através da utilização de mecanismos de atenção, os quais aprendem a escolher pontos de fixação que leve a baixa incerteza na

localização do objeto alvo [Denil et al. 2012]. Numa NMT, o mecanismo de atenção auxilia a rede a "prestar atenção" ao texto, na prática, em vez de apenas se utilizar do último estado do *encoder* para gerar o vetor de contexto C[Luong, Pham e Manning 2015]. Cada estado oculto da última camada do *encoder* será utilizado par gerar C. A Figura 11 ilustra essa modificação sobre a arquitetura *Encoder-Decoder* original.

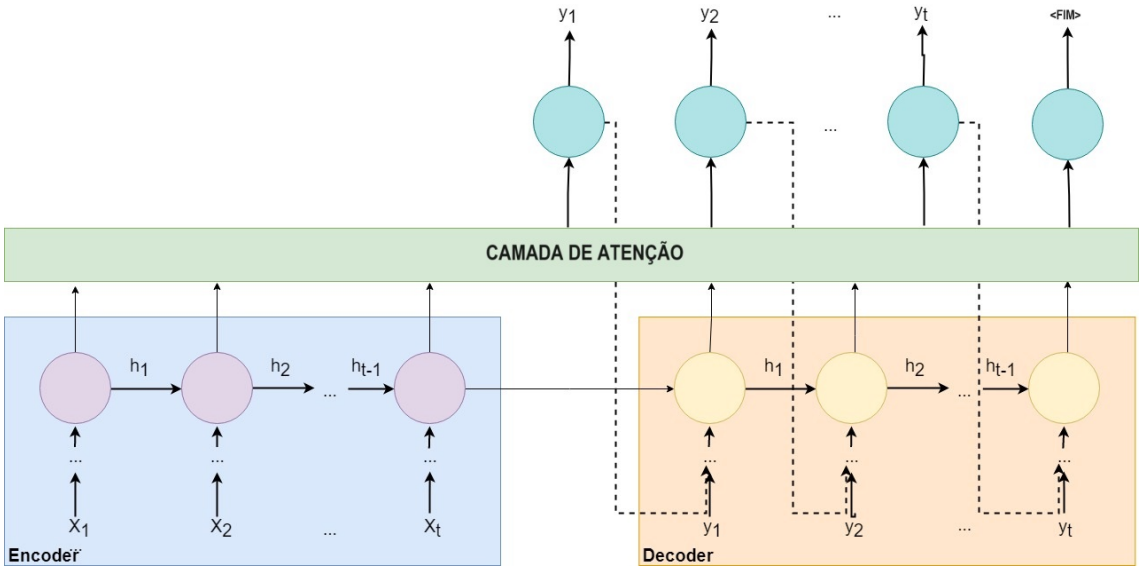

Figura 11: Arquitetura *Encoder-Decoder* com mecanismo de atenção (*Attention Mechanism*)
Fonte: Adaptado de [Luong, Pham e Manning 2015]

A ideia básica do uso do mecanismo de atenção é que todos os estados ocultos do codificador possam contribuir para o vetor de contexto que será utilizado no decodificador. Para tanto, *scores* de atenção são gerados durante o processo de tradução da sentença e uma distribuição probabilística é derivada a partir destes *scores*. Tal distribuição é utilizada para geração de um vetor de contexto, o qual será usado pelo decodificador para derivação da próxima palavra da sentença. Este processo é feito de forma interativa até a finalização do processo de derivação da sentença alvo (ver Figura 12).

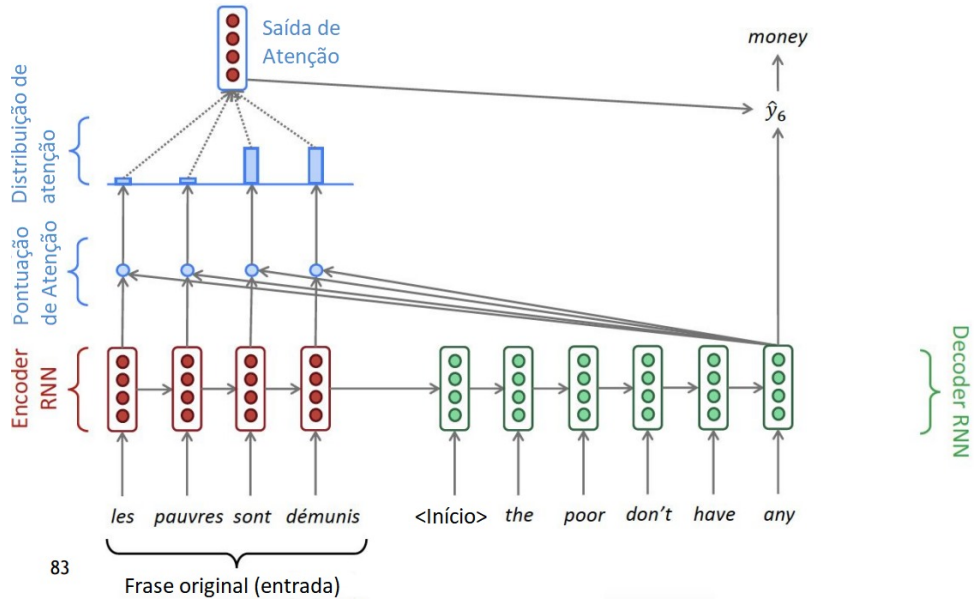

Figura 12: Mecanismo de atenção proposto por Bahdanau et al. (2014)
Fonte: [Bahdanau, Cho e Bengio 2014]

O mecanismo de atenção [Bahdanau, Cho e Bengio 2014] melhorou a performance dos métodos de tradução automática, uma vez que permite que os decodificadores foquem em certas partes da sentença de origem, a fim de derivar as palavras da sentença alvo. Tal mecanismo ainda auxilia no tratamento de problemas como o desvanecimento do gradiente e o alinhamento de sentenças.

2.2.4 Transformers

Em certo ponto, o uso das LSTM na construção de tradutores NMT começou a ser questionado, pois a sua arquitetura é computacionalmente pesada e suas operações são difíceis de serem paralelizadas. Seguindo essa ideia, Vaswani et al. (2017) propuseram a utilização do modelo *Transformer* (*self-attention*), que utilizava essencialmente o mecanismo de atenção para realizar a aprendizagem e tradução. A Figura 13 ilustra uma camada da proposta de Vaswani et al. (2017), quase que totalmente composta de mecanismos de atenção.

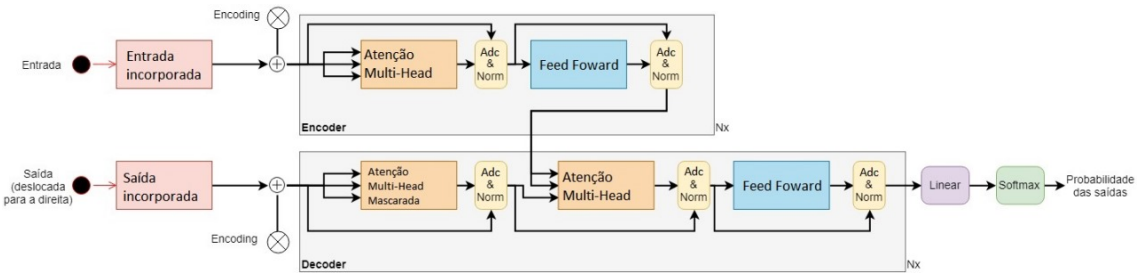

Figura 13: Arquitetura Encoder-Decoder com *Self-Attention*
Fonte: [Vaswani et al. 2017]

Vaswani et al. (2017) descrevem *attention mechanism* como um mapeamento de uma consulta (*query*) e um conjunto de pares de chave-valor para uma saída, onde consulta, chave, valor e saída são todos vetores. A saída é calculada como uma soma ponderada dos valores, em que o peso atribuído a cada valor é calculado por uma função de compatibilidade da consulta com a chave correspondente.

RNNs clássicas e LSTM demandam alto processamento para treinamento, dada a sua natureza sequencial. Esta demanda está diretamente associada à quantidade de informação que a rede precisa processar para extração de um contexto para uma sequência. O fluxo sequencial destas redes não explora bem o poder do processamento paralelo, principalmente no contexto de GPUs. Em 2017, Vaswani et al. criaram um novo modelo de redes neurais denominado ***Transformers***. Os *Transformers* permitem que sequências de entrada possam ser processadas em paralelo.

Uma rede *Transformer* é computacionalmente mais performática que métodos anteriores baseados em RNNs. Os *Transformers* contam com um mecanismo de *self-attention* e com o uso de redes *feed-forward* [Svozil, Kvasnička e Pospichal 1997]. Esta nova técnica modela a relação entre as palavras de forma independente do posicionamento destas nas sentenças, por meio do que é chamado de *Codificação Posicional* (do inglês *Positional Encoding*). Partindo da premissa de que uma palavra pode ter diferentes significados em sentenças diferentes, a etapa de *positional encoding* gera um vetor que representa o contexto da palavra, de acordo com a sua posição na sentença. A técnica de *positional encoding* é empregada em conjunto com a técnica de *embedding* para gerar um vetor numérico, com informações de contexto para as palavras codificadas. A Figura 14 apresenta a arquitetura básica de uma rede *Transformer*.

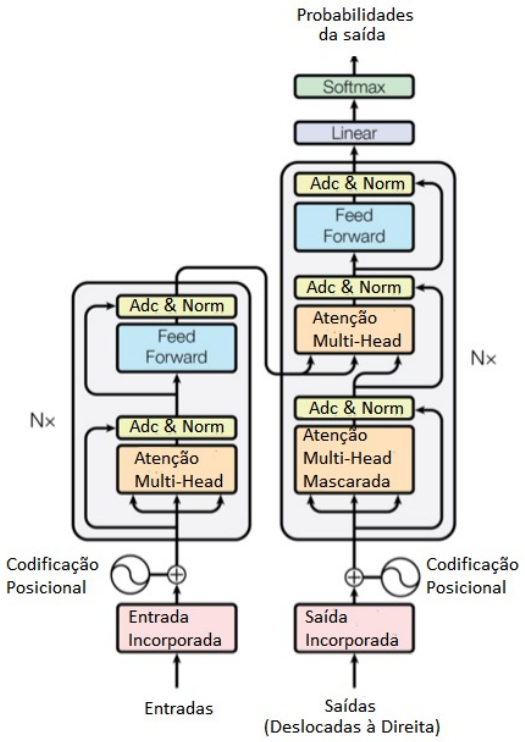

Figura 14: Arquitetura Básica das Redes Transformers
Fonte: [Vaswani et al. 2017]

A estrutura do *encoder*[Vaswani et al. 2017] é dividida em duas unidades: *Multi-Head Attention* e *Feed Forward* [Vaswani et al. 2017]. A unidade *Multi-Head Attention* aplica o conceito de *self-attention*, que tem como ideia aprender a relação entre os elementos da própria sentença, derivando vetores de atenção. A segunda unidade é uma rede *feed-forward* associada a cada vetor de atenção e tem como saída um conjunto de vetores, um para cada palavra. Cada rede *feed-forward* é independente, de forma que é possível paralelizar o processo de treinamento, passando todas as palavras ao mesmo tempo no bloco *encoder*.

O *decoder*, por sua vez, é composto de três unidades: *Masked Multi-Head Attention*, *Multi-Head Attention* e *Feed Foward*. As duas últimas unidades são semelhantes ao *encoder*. Em uma etapa inicial, passamos a sentença na linguagem alvo pelo processo de *embedding* e *positional encoding* e então os vetores são enviados ao *decoder*. Na primeira unidade do *decoder*, vetores de atenção são gerados com os dados de palavras posteriores mascarados com **-infinito** (*negative infinity*). A partir dos vetores de atenção da linguagem alvo, passamos para a segunda unidade de processamento: o *Multi-Head Attention*.

Neste ponto, o mecanismo realiza o mapeamento entre palavras na linguagem fonte e alvo e dá como saída vetores de atenção, os quais representam a relação com outras palavras em ambas as línguas[Vaswani et al. 2017]. Na etapa seguinte, de forma análoga

ao *encoder*, a rede *feed-forward* transforma os vetores de atenção em um formato mais apropriado para dar como entrada a camada linear. A camada linear também é uma rede *feed-forward*, a qual processa os vetores de saída do *decoder* para transformá-los em um vetor, onde cada unidade corresponde a uma das palavras do vocabulário da linguagem alvo com os respectivos *scores*. Este vetor é então passado por uma camada *softmax*, a qual transforma tais *scores* em probabilidades. A célula (palavra na linguagem alvo) com maior probabilidade é então escolhida e dada como saída.

2.2.5 Convolutional Neural Networks

Outras iniciativas em NMT buscaram a utilização de *Redes Neurais Convolucionais* (do inglês, *Convolutional Neural Networks* - CNNs). Apesar de serem pouco comum a utilização de CNNs em redes *Seq2Seq*, esse tipo de NN traz algumas vantagens sobre as RNNs. As CNNs possuem uma execução diferente das RNNs, pois elas não dependem da computação sobre o dado anterior da sequência, o que permite a paralelização sobre todo elemento na sequência [Gehring et al. 2017]. Mais recentemente, as redes LightConv (*Lightweight Convolutions*) [Wu et al. 2019] (um tipo de CNN) ganharam destaque por serem computacionalmente leves e terem resultados próximos dos das RNNs.

A LightConv é uma CNN *depth-wise separable* que usa uma convolução espacial sobre cada canal da informação de entrada, seguido de uma convolução *pointwise*, ou seja, uma convolução comum de janela 1x1. O resultado é uma convolução que possui menos pesos que uma convolução normal [Kaiser, Gomez e Chollet 2017, Wu et al. 2019]. Esta arquitetura consiste em uma camada de *embedding*, seguida por várias camadas de convolução com diferentes tamanhos de *kernel*, chamadas de camadas "*lightweight*", seguidas por uma camada totalmente conectada. A camada de *embedding* é responsável por mapear cada palavra em um vetor de características numéricas de alta dimensão, que é então passado para as camadas de convolução.

Um dos principais benefícios da arquitetura *LightConv* é que ela é capaz de lidar com textos de comprimento variável de forma eficiente, sem a necessidade de técnicas como *padding* (preenchimento) ou *truncation* (corte), para lidar com textos de comprimento variável [Elbayad 2020]. Além disso, é uma arquitetura relativamente enxuta com poucos parâmetros treináveis e com desempenho semelhante a arquiteturas baseadas em mecanismos de atenção, requerendo uma infraestrutura de inferência menos complexa e mais barata.

Em suma, as RNNs integram informações de contexto atualizando um estado oculto a cada etapa de tempo. As CNNs, por sua vez, resumem um contexto de tamanho fixo em várias camadas, enquanto que a *self-attention* sumariza diretamente todo o contexto. [Wu et al. 2019]. Essas arquiteturas estão sendo aplicadas em cenários de NLP e já estão

disponíveis diversas implementações e modelos pré-treinados em múltiplos contextos.

No próximo capítulo serão apresentados alguns trabalhos relacionados com esta pesquisa.

3 TRABALHOS RELACIONADOS

Este capítulo tem por objetivo levantar alguns dos trabalhos mais recentes na área de tradução entre línguas naturais e, em particular, na tradução entre línguas orais e línguas de sinais. Para tanto, foi feito um mapeamento dos principais métodos e estratégias utilizados atualmente nesta área, com vistas a mapear os modelos *Transformers* em evidência, que melhor representem o estado da arte e da técnica de NLP potencialmente aplicável ao contexto de interesse: linguagens de sinais.

Para o mapeamento destes trabalhos, foram definidas algumas regras para levantamento da literatura relacionada. Como fontes de pesquisa foram utilizadas as bases *Google Scholar* e *Arxiv*. Para uniformizar os parâmetros de busca nas diversas bases de dados foram definidas *strings* para pesquisa. As *strings* de busca utilizadas foram: *text to text*, *text to gloss*, *gloss to text*, *low resource languages*, *neural machine translation* e *automatic text translation* e foram priorizados os trabalhos em língua inglesa. Para fins deste levantamento da literatura, a busca foi limitada temporalmente para buscar trabalhos a partir de 2017. A limitação se dá com o objetivo de tentar capturar os trabalhos mais recentes e avanços importantes na área de tradução de máquina neural, com o objetivo de identificar métodos baseados em Redes *Transformers* mais eficientes para avaliação de aplicabilidade no VLibras.

No contexto de modelos baseados puramente em redes neurais profundas, Shazeer et al. (2017) propuseram um mecanismo denominado *Mixture of Experts (MoE)*, o qual consiste de um número, do que os autores denominaram de *experts*, que representa um conjunto de redes neurais *feed-forward* combinadas em uma rede de bloqueio. A rede de bloqueio seleciona combinações esparsas dos ditos *experts* para processar cada entrada. O mecanismo MoE é aplicado intermediariamente a uma pilha de redes LSTM. Entre outros avanços destacados no contexto de aplicação na tarefa de tradução, os autores conseguiram valores de BLEU de 40.56%, para tradução inglês-francês (En-Fr), usando a base **WMT'14/En-Fr**, e 26.03%, para tradução Inglês-Dinarmarquês (En-De), usando a base **WMT'14/En-De**.

No contexto de *linguagens de poucos recursos*, Ortega, Mamani e CHo (2020) propuseram um sistema NMT baseado em LSTM e com um mecanismo de segmentação morfológica baseado em **BPE** (*Byte Pair Encoding*) [Gage 1994].

Na linha de modelos que se utilizam do mecanismos de atenção, verificou-se uma quantidade crescente de trabalhos que seguem tal metodologia. Um mecanismo de reconhecimento de sinais em vídeo e posterior tradução para línguas orais foi proposto por Camgoz [Camgoz et al. 2018]. Para a etapa de tradução dos *tokens* provenientes do processamento de vídeo, os autores utilizaram RNN com mecanismo de atenção para realizar a etapa de tradução de glosa para texto.

O trabalho de Arvanitis, Constantinopoulos e Kosmopoulos (2019) trata do problema de tradução de glosa para texto partindo de **ASL** (*American Sign Language*) para o inglês. Os autores utilizaram três diferentes funções de atenção para construção da solução. No mesmo tema, o trabalho de Amin, Hefny e Ammar (2021) propõe uma abordagem bidirecional a partir de GRU, LSTM e mecanismo de atenção. Os autores aplicaram o modelo para tradução bidirecional entre a língua inglesa e ASL. Outros trabalhos na mesma temática também foram desenvolvidos [Zhang e Duh 2021, Abujar et al. 2021, Hamed, Helmy e Mohammed 2022, Yonglan e Wenjia 2022].

Na linha de soluções completamente baseadas em mecanismos de atenção, a arquitetura *Transformer* se apresenta como o estado da arte, em termos de melhores resultados para o problema de TA. Muitos trabalhos têm sido desenvolvidos à luz desta arquitetura e de modelos derivados [Camgoz et al. 2020, Devlin et al. 2018, Gomez, McGill e Saggion 2021, Liu et al. 2020, Mohamed A. & Hefny 2022, Xue et al. 2020, Xue et al. 2022, Xu, Durme e Murray 2021].

No contexto da tradução de línguas orais para línguas de sinais, Camgoz et al. (2020) propuseram o uso de *transformers* para atacar o problema de tradução de texto para glosa. Os trabalhos de Yin e Read (2020, 2020a) utilizam modelos baseados em *transformers* e *Spatial-Temporal Multi-Cue* (SMTC) para executar as tarefas de reconhecimento e tradução de sinais. Na mesma linha, uma arquitetura denominada *Progressive Transformers* foi apresentada por Saunders, Camgoz e Bowden (2020) com foco na tradução de texto para sequências contínuas de poses tridimensionais de sinais. Gómez, McGill e Saggion (2021) propuseram o uso de *transformers* para o processo de tradução de texto para glosa, com uma etapa de pré-processamento que leva em consideração informações de dependência léxica para o processo de tradução. Outros trabalhos também focaram no problema de reconhecimento e tradução de sinais para texto e texto para glosa [Angelova, Avramidis e Möller 2022, Mohamed A. & Hefny 2022].

Alguns dos novos modelos *transformers* se apresentam como ótimos candidatos para aplicação nos problemas de tradução de texto para glosa e que ainda foram pouco explorados ou não foram testados. Tais arquiteturas vão ser tratadas de forma mais aprofundada no Capítulo 4, a exemplo dos modelos *BERT* (*Bidirectional Encoder Representations from Transformers*), *BART* (*Bidirectional and Auto-Regressive Transformer*) e *T5* (*Text-to-Text Transfer Transformer*).

No contexto da tradução automática de texto entre línguas faladas, o *Bert* foi adaptado por Chen et al. (2021) em uma arquitetura *Seq2Seq*, denominada *Bert2Bert*, para ser aplicado ao problema de tradução. Os trabalhos de Lewis et al. (2019) e Liu et al. (2020) demonstraram o uso do modelo *Bart* aplicado, dentre outras áreas da PLN, ao problema de tradução. Uma versão modificada do modelo, denominada *mBart*, foi usada para desenvolvimento de um sistema de TA com foco na linguagem húngara

[Laki e Yang 2022]. Por sua vez, Liu et al. (2021) [Liu, Winata e Fung 2021] utilizou *mBart* aplicado ao problema de tradução de linguagens de poucos recursos. Em relação ao modelo *T5* e suas variações, o trabalho de Nagoudi [Nagoudi et al. 2021] propôs o *IndT5*, uma variação do modelo para tradução automática de 10 linguagens indígenas. Na mesma linha, um trabalho associado [Nagoudi, Elmadany e Abdul-Mageed 2021] apresentou um modelo denominado *AraT5* para tradução de texto para arábico. O trabalho de Xue et al. (2020) demonstrou a eficácia do modelo *mT5* aplicado ao problema de tradução e, em um trabalho posterior, Xue et al. (2021) apresentou uma variação do modelo, denominada *ByT5*, para tradução automática.

Dado o volume de trabalhos relacionados a aplicação de *Transformers* ao problema de TA, utilizamos a plataforma **PapersWithCode**[5] para realizar o mapeamento de modelos e arquiteturas pesquisadas e seus respectivos resultados quantitativos em função da métrica **BLEU**, (*BiLingual Evaluation Understudy*). Nas Tabelas 2 e 3 são apresentadas algumas das principais bases usadas para validação de sistemas de tradução e os cinco melhores modelos avaliados, quanto à qualidade de tradução. As bases escolhidas para mapeamento dos trabalhos foram a **WMT** [Bojar et al. 2014, Macháček e Bojar 2014, Bojar et al. 2016] e a **IWSLT** [Cettolo et al. 2014, Ha et al. 2015].

Como pode ser observado nas referidas tabelas, os modelos baseados em *transformers* são predominantes no estado da arte atual no contexto de TA. Entretanto, não localizamos nenhum trabalho que aplicasse este tipo de rede de forma específica para tradução para línguas de sinais, em geral, e para Libras, em particular.

Mesmo sem termos encontrado pesquisas com relação direta com este trabalho, os resultados obtidos com redes *Transformers* em outros contextos, sobretudo em cenários de baixos recursos, motivam a avaliação da sua aplicabilidade em TA de Português Brasileiro para Libras usando a Suíte VLibras.

[5]https://paperswithcode.com/task/machine-translation

Tabela 2: Valores de BLEU para métodos baseados em *deep learning* aplicados à base WMT2014 [Bojar et al. 2014, Macháček e Bojar 2014] nos pares Inglês-Alemão, Inglês-Francês, Alemão-Inglês, e à base WMT2016 [Bojar et al. 2016] para o par *Inglês-Alemão*.

Base	**Citação**	**BLEU**	**Arquiteturas**
WMT2014 English-German	[Edunov et al. 2018]	**35,00%**	Big Transformer
	[Takase e Kiyono 2021]	33,89%	Transformers (Base/Big)
	[Raffel et al. 2020]	32,10%	T5
	[Xu, Durme e Murray 2021]	31,26%	BERT
	[Li e Liang 2021]	30,91%	BART
WMT2014 English-French	[Liu et al. 2020, Liu et al. 2020]	**46,40%**	Very Deep Transformer
	[Edunov et al. 2018]	45,60%	Big Transformer
	[Liu et al. 2020]	44,30%	mRASP
	[Li e Liang 2021]	43,95%	BART
	[Liu et al. 2020]	43,80%	Very Deep Transformer
WMT2014 German-English	[Gao et al. 2022]	**35,15%**	Transformer, mBART
	[Xu, Durme e Murray 2021]	34,94%	BERT
	[Gao et al. 2022]	34,86%	Transformer, mBART
	[Ma et al. 2022]	33,12%	Mega
	[Kong, Zhang e Hovy 2020]	32,04%	Non-autoregressive
WMT2016 English-German	[Wang et al. 2019]	**40,68%**	Transformer
	[Sennrich, Haddow e Birch 2016]	34,20%	dl4mt-tutorial
	[Sennrich e Haddow 2016]	28,40%	dl4mt-tutorial
	[Wei et al. 2021]	27,00%	FLAN
	[Mehta et al. 2020]	28,00%	DeLighT

Tabela 3: Valores de BLEU para métodos baseados em *deep learning* aplicados à base IWSLT2014 [Cettolo et al. 2014] no par *Alemão-Inglês* e à base IWSLT2015 [Ha et al. 2015] nos pares *Inglês-Vietnamita* e *Alemão-Inglês*.

Base	**Citação**	**BLEU**	**Arquiteturas**
IWSLT2014 German-English	[Xu, Durme e Murray 2021]	**38,61%**	BERT
	[Gao et al. 2022]	38,37%	Transformer, mBART
	[Lohrenz, Möller B. e Fingscheidt 2022]	37,96%	Av-HuBERT
	[Li e Liang 2021]	37,90%	BART
	[Gao et al. 2022]	37,81%	Transformer, mBART
IWSLT2015 English-Vietnamese	[Ngo et al. 2022]	**40,20%**	T5
	[Ngo e Trinh 2021]	37,80%	Tall Transformer
	[Provilkov, Emelianenko e Voita 2019]	33,27%	Transformer
	[Nguyen e Salazar 2019]	32,80%	ScaleNorm + FixNorm Transformer
	[Xu et al. 2019]	31,40%	LayerNorm Transformer
IWSLT2015 German-English	[Kim et al. 2021]	**36,20%**	progressive self-knowledge distillation
	[Elbayad, Besacier e Verbeek 2018]	34,18%	2D Pervasive Attention
	[Gong et al. 2018]	33,97%	Frequency Agnostic Word Embedding
	[Edunov et al. 2017]	32,93%	Soft-Attention Encoder-Decoder
	[Lee, Mansimov e Cho 2018]	32.43%	Non-autoregressive

4 METODOLOGIA

Este capítulo é dedicado a elencar a metodologia e os critérios de elegibilidade adotados para seleção de modelos candidatos para avaliação e apresentar, dentre os modelos identificados na prospecção preliminar realizada, os que apresentaram a melhor aderência aos requisitos de viabilidade e adequação ao nosso contexto.

4.1 Critérios de Elegibilidade dos Modelos

Desde a introdução da arquitetura *LightConv* em 2019 (modelo atual do VLibras), novas técnicas e modelos têm sido propostos na literatura. A popularidade das arquiteturas baseadas em *transformers* tem aumentado e, atualmente, a maioria dos problemas e tarefas de PLN tem seu estado-da-arte baseado nessas redes.

Neste sentido, a revisão da literatura para a prospecção preliminar de modelos realizada durante esta pesquisa e descrita no Capítulo 3 teve, como objetivo, identificar quais os modelos foram mais aplicados e/ou referenciados em artigos recentes da área, publicados nos anos de 2017 a 2022 relacionados com o tema em pauta, sobretudo "low resource NLP" e/ou "sign language NMT".

Para esta fase de experimentação, alguns critérios de inclusão e exclusão adicionais foram definidos para seleção dos modelos candidatos para uma avaliação mais detalhada. Assim, além do possível ganho potencial de qualidade na tradução, outros fatores também foram considerados ao escolher os modelos candidatos, incluindo:

- Custo de infraestrutura de treinamento;
- Reprodutibilidade;
- Viabilidade de expansão e customização dos modelos;
- Ausência de restrições para uso e licenciamento.

Dentre as muitas arquiteturas e variações da arquitetura *transformer*, algumas são geralmente consideradas mais promissoras para problemas de TA. Partindo dos modelos mais referenciados nos trabalhos e considerando os critérios retro citados, foram pré-selecionados e tiveram a viabilidade da experimentação verificada os disponíveis no portal *PapersWithCode*. Este portal reúne um acervo de trabalhos de pesquisa reprodutíveis, disponibilizando tanto *datasets*, código fonte e *benchmarks* comparáveis, obtidos sobre *corpus* públicos relevantes.

Após essa fase de confirmação de viabilidade da experimentação, foram selecionados os seguintes modelos para serem avaliados de forma mais criteriosa:

- **Transformer Básico** (*Vanilla Transformer*);
- **BERT**, da Google;
- **BART**, da Meta;
- **T5 e ByT5**, do Google.

As arquiteturas *BART* e *T5* foram escolhidas devido à disponibilidade de modelos pré-treinados em grandes *corpus*, facilitando tarefas de PLN como TA. Em especial, a *T5* tem versões treinadas no *corpus* **BrWac** [Filho et al. 2018], um grande *corpus* de português brasileiro. Apesar de não haver versões generalistas para português brasileiro, a arquitetura *BART* tem versões treinadas para múltiplos idiomas, incluindo português [Liu, Winata e Fung 2021]. A arquitetura *ByT5*, por sua vez, herda as características da *T5*, além de ter um processo de "tokenização" mais agnóstico e resiliente a ruídos. A seguir, esses modelos serão avaliados de forma sucinta.

Além disso, os novos modelos também serão avaliados quanto ao seu custo computacional e de infraestrutura, sendo assim, só serão considerados modelos que consigam ser executados em ambientes (servidores) baseados apenas em CPUs. O valor de referência para o tempo de processamento de uma tradução na infraestrutura atual de nuvem do VLibras foi estimado, em média, em **1,2** segundos. Para o contexto dessa avaliação, um modelo candidato será considerado inviável se seu tempo de inferência em CPU for superior a **2** segundos.

4.2 Modelos Selecionados

A seguir, serão apresentados os modelos identificados e selecionados nas fases anteriores para experimentação.

4.2.1 *Transformer* Básico

A arquitetura *Transformer Básico* [Vaswani et al. 2017], também conhecida como *Vanilla Transformer* (VT), é uma arquitetura de DNN muito utilizada para tarefas de PLN, como TA e compreensão de linguagem natural. Ela foi introduzida pela primeira vez em um artigo de 2017 chamado "Attention Is All You Need". Essa arquitetura encontra-se disponível em vários *frameworks* de DL, incluindo a plataforma *Fairseq* [Ott et al. 2019].

A arquitetura básica do VT consiste nos seguintes componentes principais:

- **Encoder**: O *encoder* é responsável por processar a entrada, que pode ser uma sequência de palavras ou *tokens*. Ele consiste em várias camadas empilhadas, cada uma com duas partes principais:

  - **Camada de Autoatenção**: Esta camada permite que o modelo capture as relações de dependência entre as palavras na sequência, atribuindo pesos diferentes a diferentes partes da entrada.
  - **Rede de Alimentação Direta** (*Feed-Forward*): Após a autoatenção, há uma camada de *feed-forward*, que ajuda a refinar as representações intermediárias da sequência.

- **Decoder**: O *decoder* também consiste em várias camadas empilhadas, mas tem uma tarefa ligeiramente diferente. Ele gera a saída com base nas representações intermediárias produzidas pelo *encoder*. Cada camada do *decoder* também possui uma autoatenção e uma camada de *feed-forward*.

- **Atenção Multi-Cabeça**: A atenção multi-cabeça é um componente crítico que permite ao modelo considerar diferentes partes da sequência em paralelo. Ela permite ao *Transformer* capturar relacionamentos de longo alcance e é essencial para o seu desempenho em tarefas de PLN.

- **Normalização por Camada**: A normalização por camada é aplicada após cada uma delas, ajudando a estabilizar o treinamento e a melhorar o fluxo de gradientes durante a retropropagação.

- **Conexões Residuais**: As conexões residuais permitem que os gradientes fluam mais facilmente através das camadas, facilitando o treinamento de modelos profundos.

- **Máscaras de Atenção**: No *decoder*, são usadas máscaras de atenção para garantir que, durante a geração da saída, o modelo não olhe para *tokens* futuros, garantindo uma geração autônoma e causal.

- **Embeddings de Posição**: Para permitir que o *Transformer* leve em consideração a ordem das palavras em uma sequência, ele usa *embeddings* de posição para codificar informações de posição nas representações.

O modelo VT é altamente personalizável e pode ser adaptado para diferentes tarefas de PLN, tornando-se uma arquitetura amplamente utilizada em aplicações como TA, sumarização de texto, *chatbots* e muito mais.

Essa arquitetura é muito bem sucedida em várias tarefas de PLN devido à sua capacidade de lidar com dependências longas e contexto global na entrada, além de ser treinável paralelamente, ao contrário das redes recorrentes tradicionais (redes LSTM ou GRU), fazendo com que os tempos de treinamento sejam menores que outras arquiteturas. De maneira geral, essa arquitetura é o componente base para as demais arquiteturas.

Essa arquitetura também foi avaliada no contexto do **Marian**[6] [Junczys-Dowmunt et al. 2018], um *framework* de MT construído em C++. A sua principal vantagem é a sua performance de treinamento e inferência, sendo altamente escalável, com capacidade de ser treinado em grandes conjuntos de dados de idiomas, e sendo apresentado como um dos melhores modelos de TA no idioma inglês e outros idiomas.

4.2.2 BERT

BERT [Devlin et al. 2018] é um modelo de aprendizado profundo do Google muito utilizado para PLN. Desenvolvido no ano de 2018, o *BERT* é um modelo pré-treinado, de código aberto. O *BERT* treina os modelos de linguagem com base no conjunto completo de palavras, ou frase, como treinamento bidirecional, enquanto os modelos de PLN tradicionais treinam os modelos de linguagem na ordem da sequência de palavras (da direita para a esquerda ou da esquerda para a direita). Isso facilita os modelos de linguagem *BERT* a discernir o contexto das palavras, com base nas palavras circundantes, em vez de palavras que o seguem ou precedem.

Um ponto importante de diferença entre o *BERT* e outros modelos de NLP é que é a primeira tentativa do Google de um modelo pré-treinado, que é profundamente bidirecional e faz pouco uso de qualquer outra coisa, além de um corpo de texto simples. Ele tem alcançado resultados inovadores em tarefas NLP, como classificação de sentimentos, TA e funções de perguntas e respostas. Um dos principais benefícios dele é a sua capacidade de reconhecer o contexto bidirecional de palavras em uma frase, o que lhe permite entender a relação entre uma palavra e seus predecessores e sucessores. Para isso, o *BERT* emprega um procedimento de pré-treinamento no qual é alimentado com grandes quantidades de texto não rotativo, aprendendo a representar palavras de forma mais eficaz como resultado.

É pertinente clarificar que a arquitetura *BERT* não é uma arquitetura *Seq2Seq* e não pode ser usada diretamente para TA. Para isso, foi proposta a arquitetura *BERT2BERT* [Chen et al. 2021], que tem como principal objetivo carregar modelos *BERT* pré-treinados em arquiteturas *Seq2Seq*, ou seja, o conhecimento dessas modelos é reaproveitado no codificador e decodificado, que constituem uma arquitetura *Seq2Seq*.

4.2.3 BART

A arquitetura **BART** [Wang et al. 2019] é uma arquitetura de NN desenvolvida para tarefas de geração de texto, como TA, resumo de texto, e geração de respostas. Ela foi introduzida em um artigo de 2019 chamado "BART: Denoising Autoencoder Pre-training for Sequence Generation".

[6]https://marian-nmt.github.io/

A ideia por trás do *BART* é que ele possa ser treinado para realizar tarefas de geração de texto, usando um processo de pré-treinamento baseado em um *autoencoder denoising*, que tenta reconstruir uma entrada original, a partir de uma versão ruidosa dela. O modelo é treinado primeiramente para *denoising*, e depois para a tarefa de geração de texto. Isso permite que o modelo aprenda uma representação interna sólida e robusta da linguagem, o que o torna mais eficiente na realização de tarefas de geração de texto.

A arquitetura do *BART* é baseada na arquitetura *transformer* básica, consistindo de uma camada de codificação, composta de múltiplas camadas de auto-atenção, e uma camada de decodificação, composta por múltiplas camadas de auto-atenção e uma camada densa de saída. Ele também inclui o processo de pré-treinamento de *autoencoder denoising*, que é realizado antes do treinamento da tarefa específica. Além disso, BART usa pré-treinamento com um grande volume de dados, permitindo que o modelo aprenda informações gerais sobre o idioma e ajudando a melhorar a performance para tarefas de geração de texto.

4.2.4 T5

A arquitetura **T5** [Raffel et al. 2020] é uma arquitetura de redes neurais profundas desenvolvida para tarefas de PLN generalistas, como geração de texto, TA, e compreensão de linguagem natural. Ela foi introduzida em um artigo de 2020 chamado "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer".

A ideia por trás do *T5* é que ele possa ser treinado para realizar uma variedade de tarefas de PLN usando a mesma arquitetura, simplesmente mudando o texto de entrada para incluir uma tarefa específica na forma de prefixo no início da entrada, como "traduza de português para libras:"ou "gerar uma resposta para a pergunta:". Isso permite que o modelo aproveite a transferência de conhecimento entre tarefas diferentes, tornando o treinamento mais eficiente.

A arquitetura *T5* é baseada na arquitetura *transformer* básica, com algumas adições e modificações. Ele consiste de uma camada de codificação, composta de múltiplas camadas de auto-atenção, e uma camada de decodificação, composta por múltiplas camadas de auto-atenção e uma camada densa de saída. Ele utiliza também um sistema de prefixo de tarefa na entrada, o que permite ao modelo saber qual tarefa ele deve realizar e aplicar o conhecimento aprendido durante o treinamento para realizá-la.

Além disso, *T5* usa pré-treinamento em um grande volume de dados, como o **C4**[7] (*colossal clean crawl corpus*) [Raffel et al. 2020], que contém milhões de documentos,

[7]https://github.com/google-research/text-to-text-transfer-transformer

permitindo que o modelo aprenda informações gerais sobre o idioma, o que o torna muito mais eficiente na realização de tarefas de PLN.

4.2.5 ByT5

ByT5 [Xue et al. 2022] é um modelo de linguagem baseado em *transformers* desenvolvido pela Google em 2019. Como outros modelos *transformers*, ele usa mecanismos de auto-atenção para processar o texto de entrada. No entanto, ele foi projetado para operar no nível de *bytes*, e não no nível de *tokens*. Isso significa que ele processa o texto bruto como uma sequência de *bytes* individuais, em vez de palavras ou subpalavras pré-"tokenizadas".

Além disso, o *ByT5* usa um "tokenizador" de subpalavras, que é um algoritmo que segmenta o texto bruto em unidades menores, chamadas "bytes". O "tokenizador" de nível de *bytes* funciona convertendo o texto de entrada em caracteres *unicodes*, em seguida treinando um modelo de linguagem nas sequências de *bytes*. Durante a inferência, o "tokenizador" usa esse modelo de linguagem para segmentar o texto de entrada em subpalavras. Isso permite que o modelo manipule informações de nível de caracteres, enquanto também consegue lidar com palavras fora do vocabulário, similarmente ao mecanismo tradicional de "tokenização".

Esse tipo de "tokenização" torna a arquitetura *ByT5* mais resiliente a dados de baixa qualidade, ou a situações onde existe um baixo volume de dados. Além disso, alguns resultados na literatura mostram que usar uma "tokenização" com um espaço reduzido traz ganhos em situações de escassez de dados [Zhang e Duh 2021, Ding, Renduchintala e Duh 2019].

4.3 Considerações Complementares

BART, ***Transformer Básico*** e **T5** e suas variações são três arquiteturas de NN muito utilizadas em tarefas de PLN. Embora todos eles sejam baseados na arquitetura *Transformer* básica, eles têm algumas diferenças importantes em relação ao treinamento, arquitetura e aplicações.

A principal diferença entre *BART* e *T5* é o objetivo do treinamento. O *BART* é projetado para tarefas de geração de texto, como TA e resumo de texto, enquanto o *T5* é projetado para tarefas de PLN gerais, como geração de texto, TA, e compreensão de linguagem natural. Isso é refletido na arquitetura dos modelos, onde *BART* inclui um processo de pré-treinamento de *autoencoder denoising*, que é realizado antes do treinamento da tarefa específica, enquanto o *T5* inclui o uso de prefixo de tarefa na entrada, o que permite que o modelo saiba qual tarefa ele deve realizar e aplicar o conhecimento aprendido durante o treinamento para realizá-la.

A principal diferença entre *T5* e *vanilla transformer* é que *T5* é projetado para tarefas de PLN gerais, enquanto a *vanilla transformer* é projetada para tarefas de PLN específicas, como TA. Além disso, a *T5* e *ByT5* usam pré-treinamento em um grande volume de dados, o que lhe permite aprender informações gerais sobre o idioma, enquanto a *vanilla transformer* é treinada apenas para uma tarefa específica e não se beneficia de pré-treinamento.

Em geral, *T5* e *BART* apresentam melhores resultados para suas tarefas específicas, devido a seus processos de pré-treinamento e sua capacidade de lidar com contexto global na entrada, mas a *vanilla transformer* tem uma estrutura mais simples e pode ser mais fácil de implementar e aplicar para tarefas específicas, o que também resulta em um menor custo computacional.

Quando se trabalha com um *corpus* pequeno, principalmente no cenário de Libras, uma língua com poucos dados disponíveis (*low resource languages*), é importante considerar que o modelo pode ter dificuldade para aprender padrões de linguagem e generalizar bem a tarefa desejada, devido a uma falta ou desbalanceamento de dados. Neste caso, a escolha da arquitetura do modelo pode ser crítica para obter bons resultados.

Uma opção seria usar a arquitetura *vanilla transformer*, pois ela tem uma estrutura mais simples e pode ser mais fácil de treinar e implementar do que outras arquiteturas mais avançadas, como *T5* (e suas variações) ou *BART*. Além disso, essa arquitetura apresenta um baixo custo computacional em função da sua simplicidade.

Outra opção seria utilizar modelos de transferência de conhecimento, onde se utilizam modelos pré-treinados em *corpus* massivos, que são adaptados para uma tarefa específica, processo conhecido na literatura como *downstream*[8], a qual pode representar uma boa estratégia, quando se trabalha com um volume de dados limitado.

Finalmente, é importante lembrar que escolher a melhor arquitetura pode ser uma questão de experimentação e que é importante testar várias arquiteturas e técnicas para encontrar a melhor configuração para o seu conjunto de dados específico. Portanto, foi realizado um estudo experimental com as arquiteturas *BART*, *T5* (e suas variações) e *Transformer Básico* com o mesmo *corpus* usado para gerar a versão de produção atual, ou seja, a única variável é a arquitetura de tradução.

4.4 Planejamento de Experimentos

O objetivo da experimentação é testar os modelos *Transformer Básico*, *BART*, *BERT*, *T5* e *ByT5* de forma direta, mediante adaptação e integração, nos componentes de tradução do VLibras. Os resultados obtidos serão comparados com os produzidos pela

[8]Processo que consiste na especialização de um modelo pré-treinado, muitas vezes de forma semi-supervisionada, em uma tarefa específica de processamento de linguagem natural.

versão atual do tradutor híbrido do VLibras, o qual usa o modelo *LightConv* do *framework Fairseq*. Para facilitar a comparação com os resultados já disponíveis do VLibras serão usados nos experimentos os mesmos conjuntos de dados de treinamento e validação e também calculadas as mesmas métricas de avaliação. O *dataset* de treinamento utilizado será o mesmo utilizado no treinamento do fluxo de tradução do VLibras atualmente em produção. Esse conjunto de dados possui 67.874 tuplas de português/glosa. As frases e traduções foram feitas manualmente por linguistas e intérpretes e, atualmente, é um dos maiores *corpus* desse tipo disponíveis.

Para facilitar a execução desta etapa de experimentação, será usada a biblioteca **Hugging Face**[9]. É uma biblioteca de aprendizado profundo para NLP que oferece acesso a vários modelos pré-treinados, assim como modelos de linguagem e tokenizadores. Esses modelos pré-treinados podem ser utilizados para tarefas comuns de NLP, como classificação de texto, extração de entidade, TA e muito mais. A biblioteca também fornece ferramentas para treinar e personalizar modelos para tarefas específicas e facilita a integração com outras bibliotecas e *frameworks*. Essa biblioteca foi selecionada devido à disponibilidade de todos os modelos prospectados, por ser de fácil integração e estar em constante desenvolvimento, garantindo, assim, uma boa manutenção para o projeto

4.4.1 Arquitetura de Tradução da Suíte VLibras

O componente de tradução do VLibras atualmente adota uma arquitetura híbrida, baseada em um tradutor de regras (RBMT) [Oliveira et al. 2019] e um tradutor baseado em inteligência artificial (NMT) (modelo *LightConv* [Wu et al. 2019]). Nesse contexto, o tradutor de regras faz o papel de um componente de pré-processamento da sentença em português, que por sua vez alimenta o modelo *LightConv*. Esse processo tem como objetivo normalizar a entrada e ajudar o modelo durante o treinamento. Essa etapa é fundamental em função do relativo baixo volume de dados disponíveis.

O fluxo de inferência do processo de tradução (Figura 15) é representado pelas etapas que uma frase em português percorre até ser convertida em uma representação traduzida em glosa[10], pronta para ser consumida por outra aplicação. As etapas que o Vlibras atualmente em produção utiliza para a inferência são:

- Receber a frase em português (PT);
- Inserir a frase no tradutor baseado em regras;

[9] https://huggingface.co/

[10] Glosa são palavras de uma determinada língua oral grafadas com letras maiúsculas que representam sinais manuais de sentido próximo. Wilcox, S. e Wilcox, P. P. (1997) definem glosa como sendo uma tradução simplificada de morfemas da língua sinalizada para morfemas de uma língua oral.

- Gerar uma glosa intermediária[11] (GR);
- Inserir a glosa intermediária no tradutor neural;
- Gerar a glosa final (GI), pronta para ser sinalizada.

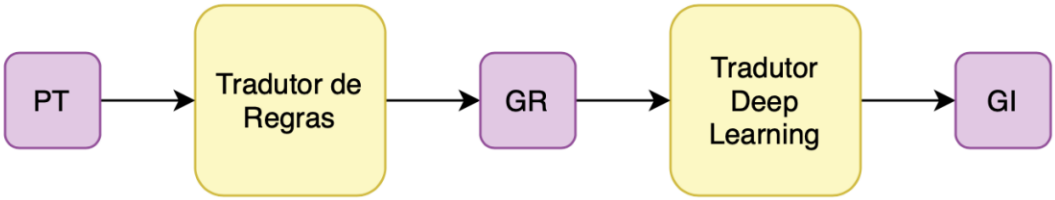

Figura 15: Fluxo do tradutor híbrido do VLibras
Fonte: Autor

O treinamento do tradutor DL do VLibras também possui um fluxo integrado ou *pipeline* (Ver Figura 16). O *corpus* bilíngue de treinamento, um conjunto de dados contendo diversas sentenças equivalente em português para glosa, é traduzido utilizando o tradutor de regras para glosa intermediária. Neste processo, as sentenças passam por várias etapas de pré-processamento. Em seguida, essas entradas são divididas em dois conjuntos distintos: um para ser usado no treinamento do modelo e outro para ser usado apenas na validação do processo de tradução. O conjunto de treinamento também passa por um processo de *data augmentation*[12] para ampliar a ocorrência de palavras raras em seu conjunto de sentenças.

[11] Frase simplificada e em estado intermediário para facilitar a tradução *deep learning*.

[12] *Data augmentation* é um recurso muito usado em *low-resource NLP* para ampliar, sinteticamente e usando técnicas específicas, a quantidade de sentenças em corpus usados em treinamento de modelos neurais.

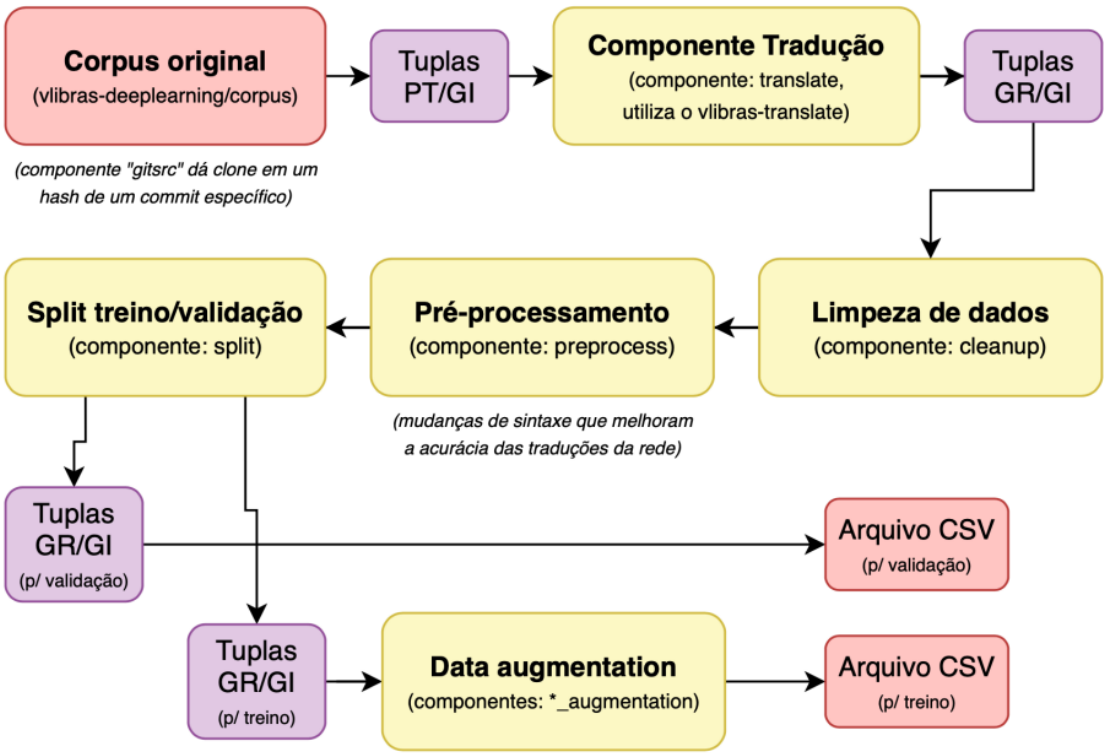

Figura 16: Arquitetura geral do fluxo de treinamento do VLibras
Fonte: Autor

Após o pré-processamento do *corpus*, esses dados passam por um componente de aprendizado para geração de *tokens* **BPE**[13] para serem aplicados nas tuplas de treino e validação. Em seguida, as tuplas são binarizadas para serem utilizadas para o treinamento do modelo usado no tradutor neural do VLibras (modelo *LightConv* do *framework Fairseq*[14]).

4.4.2 Métricas de Interesse

Os modelos de TA neural baseados em DL são, normalmente, treinados em grandes bases de dados denominadas *corpus*[15], sendo necessário realizar a avaliação dessas traduções, visando medir a eficiência do modelo de TA. Até o presente momento, a métrica de avaliação mais usada para avaliação de TA é conhecida como *BLEU*, a qual foi proposta pela primeira vez por Papineni et al. (2002). A ideia por trás da métrica *BLEU* é calcular a similaridade semântica entre a tradução gerada pelo computador e uma ou mais traduções humanas de referência, sendo projetada para substituir e automatizar, quando possível, a avaliação humana em cenários onde múltiplas avaliações são necessárias.

[13] *Byte pair encoding* (BPE) é um método de tokenização caracterizado por representar um texto com o menor número de bytes.
[14] Fairseq é um kit de ferramentas de modelagem de sequência para treinar modelos personalizados para tradução, resumo e outras tarefas de geração de texto produzido pela Meta.
[15] Coleção de documentos ou textos escritos em determinada língua.

A métrica *BLEU* é calculada usando a seguinte equação:

$$BLEU = \min\left(1, \frac{candidato_{n-gramas}}{referencia_{n-gramas}}\right) \cdot \exp\left(\frac{1}{N}\sum_{n=1}^{N} \log\left(\frac{candidato_{n-gramas}}{referencia_{n-gramas}}\right)\right)$$

onde:

- $candidato_{n-gramas}$ representa o número de *n-gramas* na tradução candidata.
- $referencia_{n-gramas}$ representa o número de *n-gramas* na(s) referência(s);
- N é o número de diferentes tamanhos de *n-gramas* considerados (geralmente, são utilizados unigramas, bigramas, trigramas, etc.).

Essa equação calcula a precisão dos *n-gramas* na tradução candidata em relação às referências e, em seguida, combina essas precisões usando uma média geométrica ponderada para obter a pontuação *BLEU* final.

O resultado da *BLEU* é normalmente expresso como um número entre **0** e **1**, onde **1** indica uma correspondência perfeita entre a tradução gerada e a tradução de referência. Valores mais próximos de **1** indicam melhores resultados. Alguns algoritmos de TA são avaliados com o *corpus* de dados de avaliação *BLEU*.

Também é possível avaliar tradução automática utilizando medidas de similaridade. A distância de *Levenshtein* [Levenshtein 1966], também conhecida como distância de edição, é uma medida de similaridade entre duas *strings* ou sequências de caracteres. Ela é baseada no número mínimo de operações de edição (inserção, deleção ou substituição de caracteres) necessárias para transformar uma *string* em outra.

Uma variação da distância de *Levenshtein* é a distância *Levenshtein* normalizada. Ele calcula a distância de *Levenshtein* de forma padrão, mas normaliza o resultado dividindo-o pelo comprimento da *string* mais longa. Dessa forma, a distância *Levenshtein* normalizada varia de **0** a **1**, onde **0** indica que duas sentenças não têm nenhuma palavra ou *token* em comum e **1** indica que as duas sentenças são idênticas.

A distância *Levenshtein* normalizada é frequentemente usada como uma métrica de similaridade para *strings*. Ela é utilizada em diversas áreas como por exemplo: detecção de plágios, PLN, reconhecimento de fala, TA, entre outros. A distância *Levenshtein* normalizada é útil porque permite comparar *strings* de tamanhos diferentes de maneira justa. Além disso, é uma forma de contornar a desvantagem de distância de *Levenshtein* que é altamente sensível às diferenças de tamanho, caso não seja normalizada.

Para avaliação do VLibras, uma tradução com distância *Levenshtein* normalizada de valor igual a **1** é considerada correta, ou seja, é uma tradução perfeita. Uma distância de *Levenshtein* normalizada, de valor menor do que **0,85**, é considerada uma tradução incorreta e um valor maior do que **0,85** e menor do que **1** é considerado uma tradução

parcial. Esses valores limiares foram ajustados de forma incremental durante o desenvolvimento do componente de inteligência, através de validações feitas com usuários. Resumidamente:

- **Perfeita** (similaridade **igual a** 1)
- **Parcial** (similaridade **maior que** 0,85 e **menor que** 1)
- **Incorreto** (similaridade **menor que** 0,85)

Portanto, a validação computacional usada atualmente no componente tradutor do VLibras usa duas métricas de interesse: uma métrica de tradução (*BLEU*) e uma métrica de similaridade (distância de *Levenshtein* normalizada). Para permitir uma comparação adequada, as mesmas métricas serão adotadas neste estudo.

4.4.3 Conjuntos de Avaliação

Tão importante quanto a definição das métricas de interesse é a definição do conjunto de dados que será usado para avaliação do modelo, também chamado de *conjunto de teste*. Um conjunto de teste é um subconjunto do conjunto de dados de treinamento que é separado e usado apenas para avaliar o desempenho de um modelo de DL. Ele é usado para medir quão bem o modelo é capaz de generalizar para dados que ele nunca viu antes.

Geralmente, os dados disponíveis (neste caso em particular, o *corpus* bilíngue de referência) são divididos em três partes: i) conjunto de treinamento; ii) conjunto de validação; e iii) conjunto de teste. O conjunto de treinamento é usado para treinar o modelo, o conjunto de validação é usado para selecionar o melhor modelo entre várias opções (por exemplo, selecionando a melhor configuração de hiperparâmetros) e o conjunto de teste é usado para avaliar o desempenho final do modelo selecionado.

É importante notar que o conjunto de teste deve ser completamente separado do conjunto de treinamento e validação, de forma que ele contenha dados que o modelo nunca viu antes. Isso é importante para evitar que o modelo "memorize" os dados de treinamento e validação, o que resultaria em uma sobre-estimação do desempenho do modelo. Este fenômeno é conhecido como **sobreajuste** dos dados de treinamento, do inglês *overfitting*.

Além disso, é importante avaliar o modelo em diferentes conjuntos de dados, tanto em termos de precisão, quanto em termos de generalização. Para isso, além de avaliar o modelo em dados de teste, também é importante avaliar o desempenho do modelo com diferentes tipos de dados, como dados desbalanceados, dados incompletos, dados diferentes

daqueles usados no treinamento e assim por diante. Dessa forma, é possível entender melhor como o modelo se comporta e identificar quaisquer problemas ou limitações.

No contexto do VLibras, essa avaliação é feita sobre diferentes conjuntos de teste, que procuram modelar cenários e pontos críticos encontradas no processo de tradução de português para Libras, sendo projetados com a supervisão de especialistas em Libras:

- *frases* **básicas**
- *frases contendo referências de* **contexto**
- *frases com referências* **direcionais**
- *frases com sentido de* **negação**
- *frases contendo nome de* **pessoas famosas**
- *frases com referências de* **lugares**
- *frases com indicadores de* **intensidade**
- *frases com* **números cardinais**
- *frases com* **números romanos**

Todos os modelos de TA de português para Libras gerados no contexto do VLibras são validados sobre cada um desses conjuntos, antes de serem movidos para uma etapa de homologação e, finalmente, serem disponibilizados para o usuário final, através dos componentes interativos da Suíte VLibras.

4.4.4 Configuração do Ambiente

Para a realização dos experimentos foram utilizados dois ambientes de processamento, para permitir uma paralelização de cada execução planejada, posto que cada ciclo de treinamento e validação durava cerca de 5 horas. Para a configuração de cada ambiente, foi preciso instalar diversos módulos *python* dos *frameworks* utilizados. Em seguida, foi baixado o código fonte do *pipeline* VLibras em produção e seus submódulos do serviço de versionamento GitLab, hospedado no LAVID. Os dois ambientes foram configurados de forma similar para o treinamento de cada modelo previsto, incluindo a adaptação do modelo e a integração do mesmo ao *pipeline* do VLibras.

Antes da execução dos experimentos, foram conferidos os hiperparâmetros com os usados em produção e foram feitos testes de sanidade para aferir se ambos os ambientes forneciam resultados compatíveis e sincronizados. Adicionalmente, foram executados treinamentos exploratórios e comparados com os resultados do modelo atual e diversos ajustes

como versão de dependências, variáveis de configuração, entre outros, foram realizados até que os resultados fossem equivalentes.

4.4.5 Realização dos Experimentos

O treinamento e validação de cada experimento foi executado de forma paralela em um dos dois ambientes e os resultados foram calculados e consolidados para cada modelo e para cada subconjunto de sentenças de avaliação. O objetivo do primeiro ciclo de experimentos foi gerar uma pontuação de referência do modelo atual, *LightConv*. Para os ciclos seguintes, foram utilizados os seguintes modelos: *BART*, *Transformer Básico*, *T5* e *ByT5*. Também foram combinadas com os modelos testados, algumas técnicas como *back translation*, aplicação de técnica de aumento de dados (*data augmentation*) antes da tradução para glosa intermediária no pré-processamento, e alteração na quantidade de *tokens BPE*[16].

Durante essa fase, foi identificado que o modelo *BERT* não estava produzindo resultados adequados no contexto de tradução Português/Libras. De maneira geral, o modelo não apresentou uma boa adequação para o problema de TA sendo investigado, o que ocasionou o seu descarte para as fases seguintes de avaliação.

[16] *Byte Pair Encoding.*

5 RESULTADOS DA AVALIAÇÃO

Neste capítulo serão apresentados os resultados obtidos em cada ciclo de experimentos. Em geral, as tabelas de resultados possuem uma coluna para um dos nove subconjunto de teste considerados e uma linha para cada uma das classificações possíveis para cada tradução (**Perfeita**, **Parcial** e **Incorreto**). Os valores em cada célula trazem o percentual de resultados de cada classificação, obtido em cada subconjunto pelo modelo/configuração sendo considerado.

O modelo que está em uso no VLibras atualmente, tradutor híbrido baseado no modelo *LightConv*, será usado como referência para validação de novos modelos, ou seja, será possível verificar se os novos modelos apresentam métricas computacionais melhores, ou piores, do que o modelo em produção. Neste sentido, as comparações que foram feitas com os modelos avaliados tomaram como base os resultados mostrados na Figura 17.

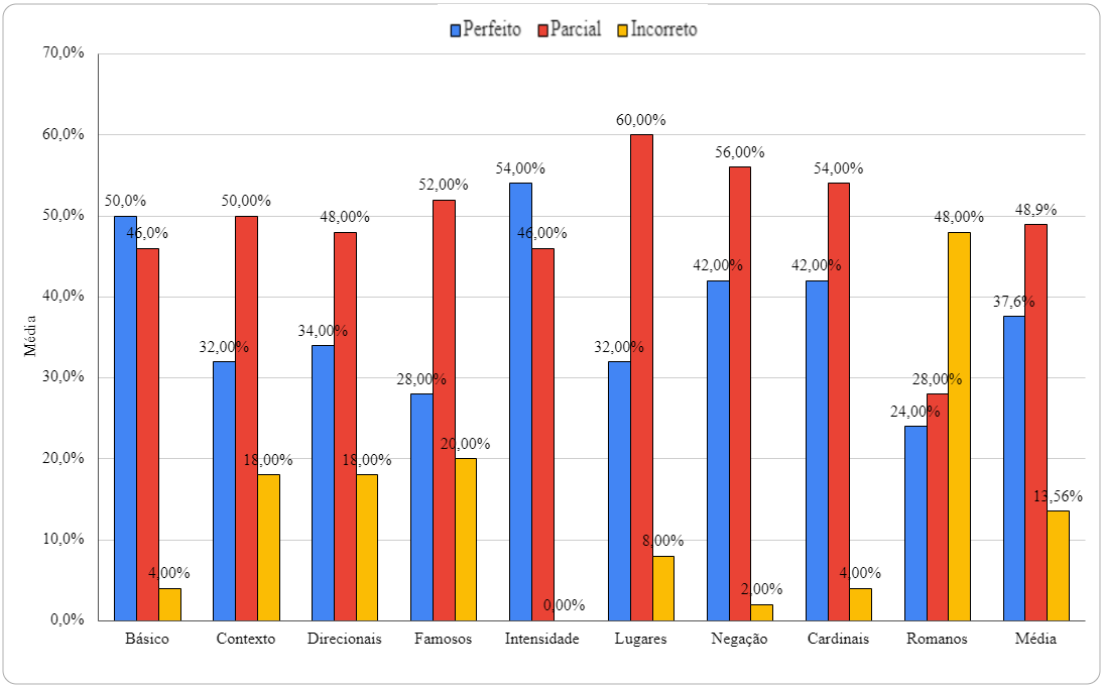

Figura 17: Resultados da Métrica de Similaridade por Tipo de Sentença obtidos com o Modelo de Referência (*LightConv*)
Fonte: Autor

Na Figura 18 são exibidos os resultados obtidos pelo modelo *BART*. Houve uma melhora nas traduções perfeitas e uma piora na ordem entre 1 e 2 pontos percentuais nos resultados parciais e incorretos, respectivamente. O principal subconjunto afetado foi o de sentenças com pessoas famosas. Esse baixo desempenho e seu alto custo de inferência torna esse modelo pouco viável para o contexto do projeto.

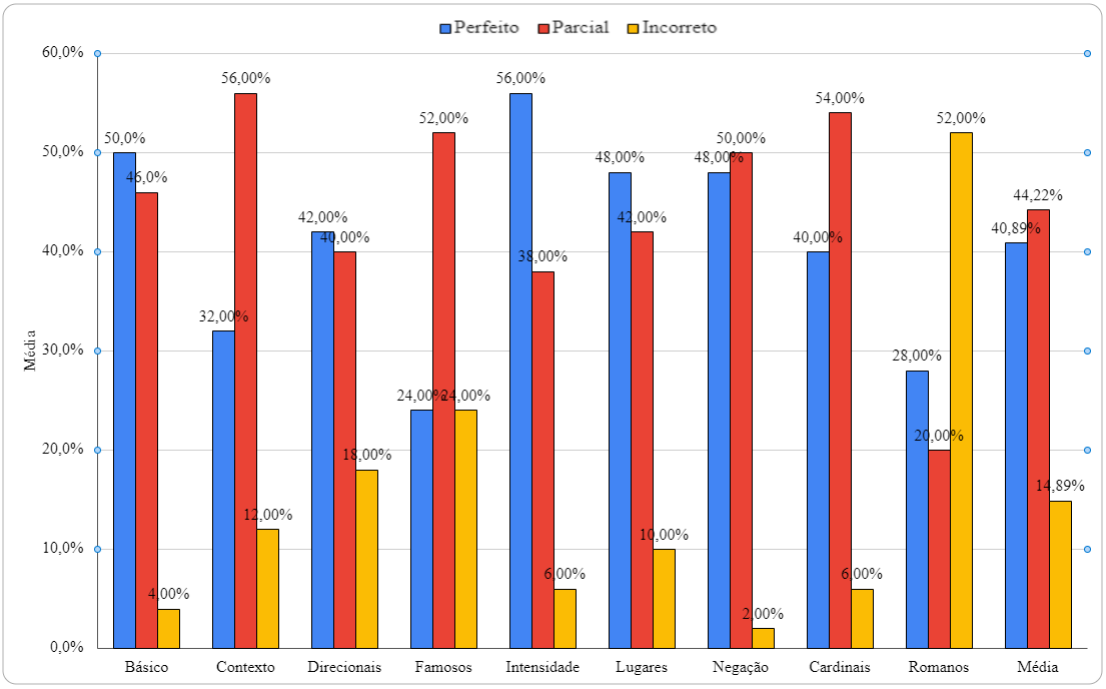

Figura 18: Resultados da Métrica de Similaridade por Tipo de Sentença obtidos com o Modelo *BART*
Fonte: Autor

Os resultados obtidos com o *Transformer Básico*, por sua vez, podem ser visualizados na Figura 19. Pela primeira vez, um modelo conseguiu uma melhora consistente de mais de 4,5 pontos percentuais nas traduções corretas. Tal melhora foi observada em quase todos os subconjuntos de validação e sempre com uma migração de traduções parciais para traduções perfeitas.

Além de um ganho na qualidade da tradução, essa arquitetura também tem um custo computacional próximo ao modelo atualmente em uso e uma complexidade relativamente baixa para integração com um ecossistema como o do VLibras.

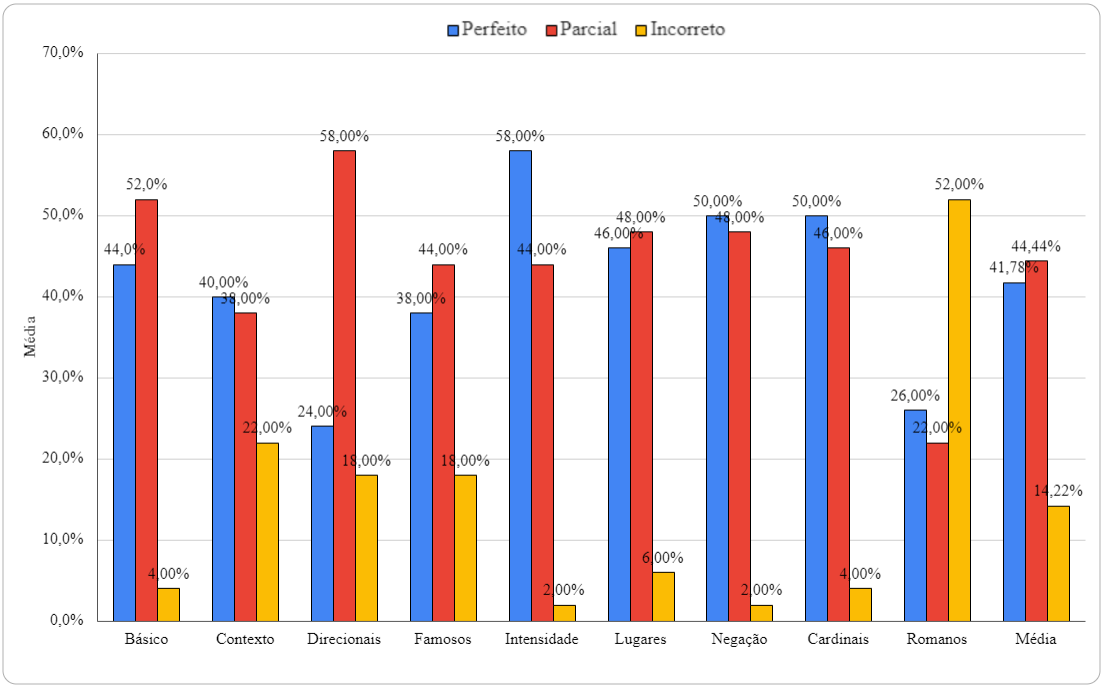

Figura 19: Resultados da Métrica de Similaridade por Tipo de Sentença obtidos com o Modelo *Transformer Básico*
Fonte: Autor

A Figura 20, por sua vez, apresenta os resultados que foram encontrados usando o modelo *T5*. Novamente houve uma melhora discreta das traduções corretas, de quase 1,5 pontos percentuais, enquanto que as traduções incorretas pioraram em mais de 2%. As categorias de sentenças mais afetadas foram as com indicativo de intensidade (melhora de 10%) e sentenças com pessoas famosas (piora de 12%).

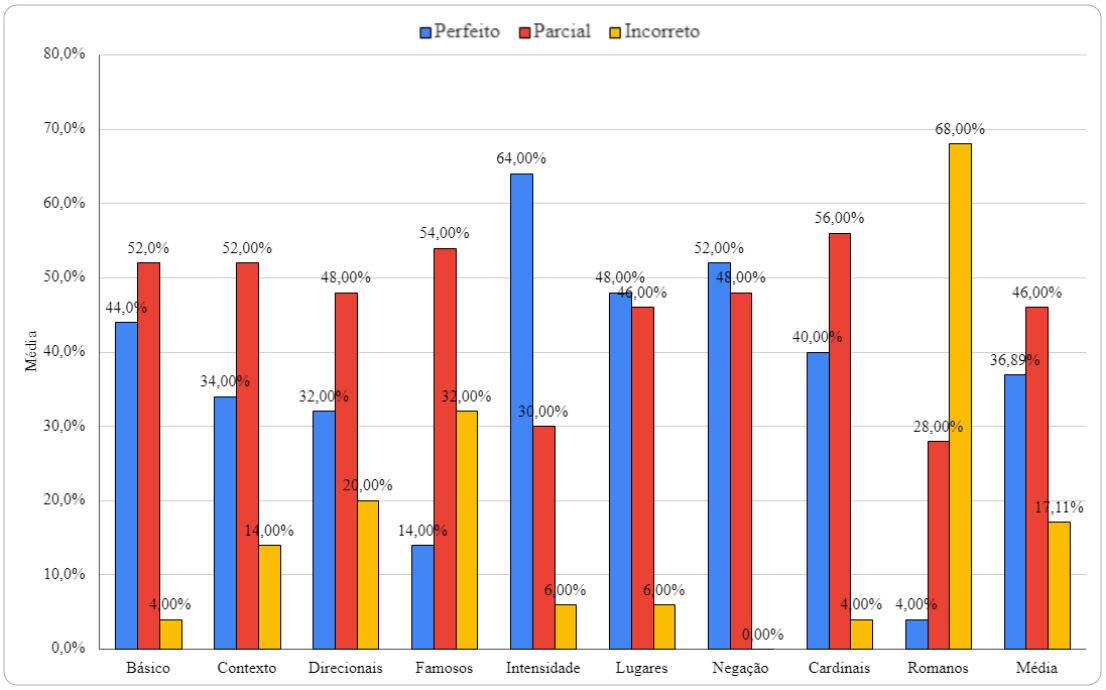

Figura 20: Resultados da Métrica de Similaridade por Tipo de Sentença obtidos com o modelo *T5*
Fonte: Autor

Após a conclusão do primeiro ciclo de experimentos com os modelos originais, uma variação do modelo *T5* também foi considerada, o modelo *ByT5*. A Figura 21 traz os resultados obtidos com *ByT5*. A principal diferença entre eles é o processo de "tokenização" utilizado, enquanto no *T5* é usada uma abordagem de subpalavras baseada na **SentencePiece** [Kudo e Richardson 2018], no *ByT5* a "tokenização" é baseada em *bytes* (ou caracteres *unicodes*). Essa variação apresentou uma melhora significativa em quase todas as faixas de resultados (com exceção do subconjunto de número romanos), com um acréscimo de quase 5%, nas traduções corretas, e uma redução de 1,5%, nas traduções incorretas.

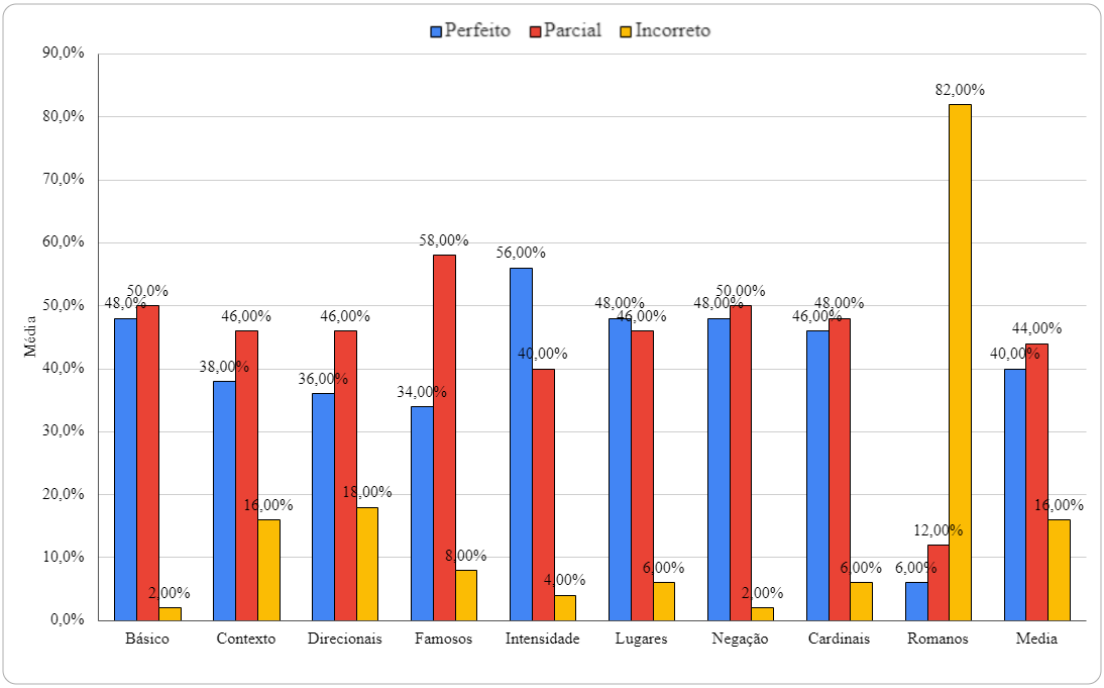

Figura 21: Resultados da Métrica de Similaridade por Tipo de Sentença obtidos com o modelo *ByT5*
Fonte: Autor

Outra variação testada foi a combinação do modelo *Transformer Básico*, de forma combinada com a técnica *Back Translation*. Esta técnica se baseia em fazer uma tradução adicional inversa durante a fase de treinamento. Essa abordagem se mostrou promissora, com melhoria de quase 6% na média das traduções corretas e diminuição discreta na média de traduções incorretas. Um ponto de atenção, a ser investigado posteriormente, foi a inversão entre os percentuais obtidos de traduções corretas e traduções parcialmente corretas e algumas categorias de sentenças com piora discreta, com exceção da tradução de sentenças com algarismos romanos, que piorou 4 pontos percentuais.

5.1 Discussão

Na Figura 22, onde estão sumarizados os resultados obtidos nos experimentos, é possível perceber que dois modelos candidatos conseguiram melhores resultados, do que o modelo de referência (em produção), sendo que o modelo *ByT5* apresentou as melhores médias, obtendo um aumento percentual de 12,74% nas traduções perfeitas e diminuindo as traduções incorretas em 16,22%. Em segundo lugar, ficou a arquitetura *Transformer Básico*, a qual obteve um aumento percentual de 11,46% para traduções corretas, porém as traduções incorretas aumentaram em 2,70%. No entanto, cada um desses modelos apresenta alguns prós e contras que precisam ser avaliados.

A arquitetura *BART*, por sua vez, foi descartada para o cenário em pauta, pois

apesar de obter resultados superiores ao modelo em produção, a mesma apresenta um alto custo de infraestrutura, o que poderia tornar inviável em um cenário como o projetado, com milhões de acessos mensais.

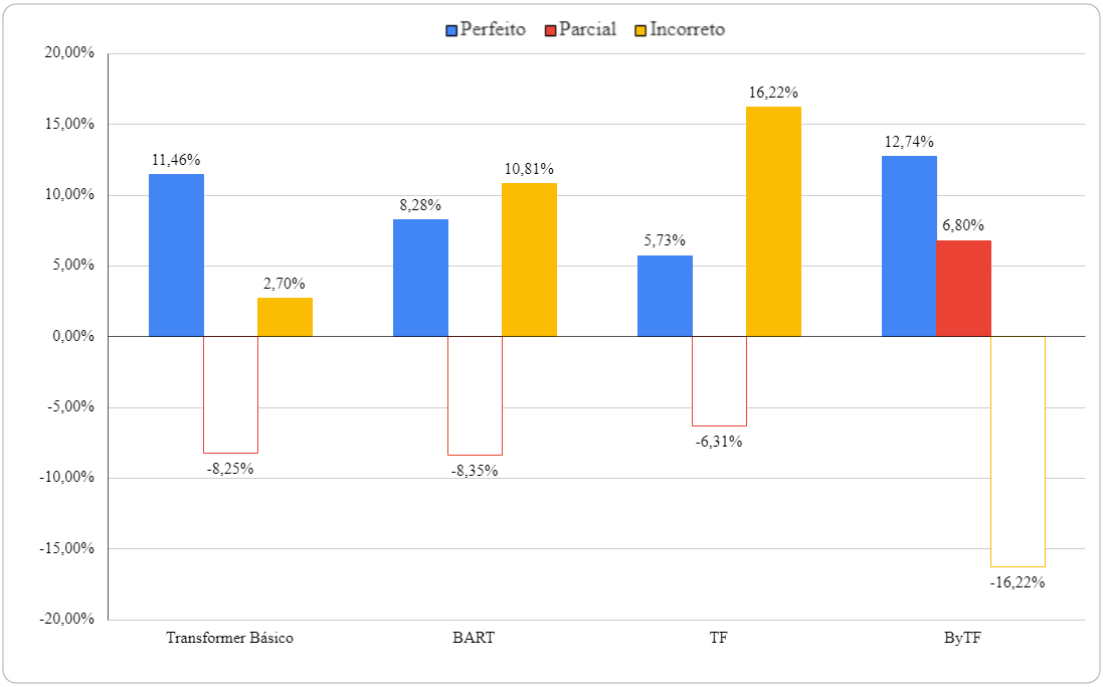

Figura 22: Consolidação dos Resultados Obtidos pelos Modelos Avaliados em Relação ao Modelo de Referência (*LightConv*).
Fonte: Autor

A arquitetura *Transformer Básico* foi considerada uma das arquiteturas mais eficientes do ponto de vista computacional, pois foi, dentre as testadas, uma das arquiteturas mais leves e de fácil integração com sistemas como o VLibras. Além disso, ela também apresenta um treinamento mais rápido, tornando-a uma opção atraente para projetos que requerem performance e eficiência computacional. É importante destacar que, embora o custo computacional e a facilidade de integração sejam aspectos vantajosos, a precisão e a qualidade do modelo, treinado com base na arquitetura *Transformer Básico*, tende a ser menos competitiva, pois não faz uso de modelos pré-treinados. Feitas tais considerações, entende-se que a arquitetura *Transformer Básico* pode ser considerada uma opção viável, em termos de custo e benefício, para ser adotada como uma solução de produção.

A arquitetura *T5* apresentou resultados mistos, obtendo um aumento de traduções perfeitas, porém aumentando a taxa de traduções incorretas. Contudo, a *ByT5* apresentou as melhores métricas computacionais e um ganho expressivo em conjuntos de avaliação específicos, como o conjunto de contexto e direcionalidade. No entanto, ela possui um alto custo computacional, embora o mesmo possa ser otimizado, especialmente depois de passar por um processo automático de otimização e simplificação usando a biblioteca

FastT5[17]. Esse processo é factível, porém pode resultar em uma maior complexidade de integração. A despeito da maior complexidade para integração, a *ByT5* também foi considerada uma opção viável para um sistema como o VLibras.

A arquitetura *ByT5* também se mostrou vantajosa quando avaliada sobre um conjunto de dados que compreende conteúdo de páginas da Internet de caráter institucional e/ou de serviços públicos (ver Tabela 4). Mesmo sem a introdução de exemplos de tais dados no processo de treinamento do *ByT5*, houve um ganho de 7,41 pontos percentuais na métrica *BLEU*, indicando que esse modelo tem uma melhor capacidade de generalização do que o modelo atualmente em uso pelo VLibras.

Tabela 4: Comparativo da Métrica *BLEU* entre o modelo de referência (*LightConv*) e o melhor modelo prospectado (*ByT5*)

Sentenças	LightConv	ByT5	Variação
Frases Básicas	46,55	58,09	+11,54
Cardinais	72,51	71,69	-0,82
Contexto	54,40	50,50	-3,9
Direcionalidade	19,49	26,45	+6,96
Famosos	38,31	48,75	+10,44
Intensidade	45,13	48,78	+3,65
Lugares	47,46	56,09	+8,63
Negação	57,37	58,78	+1,41
Romanos	69,52	73,27	+3,75
Genéricas (Sites)	25,38	32,79	+7,41
Média	**47,61**	**52,52**	**+4,91**

Considerando que o processo de seleção de modelos candidatos teve um espaço de busca mais amplo e que modelo melhor classificado, *ByT5*, já é pré treinado e não apresenta espaço para otimização, será realizado um estudo mais aprofundado para tentar otimizar o segundo modelo melhor classificado, *Transformer Básico*, através da busca e varredura de hiperparâmetros relevantes. Essa segunda fase de experimentação será apresentada no próximo capítulo.

[17]https://github.com/Ki6an/fastT5

6 REFINAMENTO DO MODELO SELECIONADO: *Transformer Básico*

O processo de otimização de hiperparâmetros de um modelo neural envolve encontrar a melhor combinação de valores para os parâmetros que não são aprendidos durante o treinamento do modelo. Esses parâmetros, conhecidos como hiperparâmetros, podem incluir a taxa de aprendizado, o tamanho do lote, o número de camadas e neurônios, entre outros. A escolha adequada desses hiperparâmetros pode ter um grande impacto no desempenho do modelo.

O processo de otimização do modelo *Transformer Básico* (ou *Vanilla Transformer*), para TA de Português Brasileiro para glosa, será realizada em dois passos, os quais serão descritos nas próximas seções:

- Busca Automatizada de Hiperparâmetros
- Varredura dos Hiperparâmetros Selecionados

6.1 Busca Automatizada de Hiperparâmetros

A busca automatizada de hiperparâmetros, ou *automated hyperparameter search* [Bergstra e Bengio 2012], é uma técnica usada para refinar os hiperparâmetros[18] de um modelo de aprendizado de máquina de forma automática.

Durante o processo de busca automatizada de hiperparâmetros, o desempenho do modelo é avaliado com base em uma métrica específica. No contexto em pauta, de TA de LS, continuará sendo utilizada a métrica BLEU. O objetivo é encontrar a combinação de hiperparâmetros que resulte no melhor desempenho possível do modelo *Transformer Básico*, de acordo com essa métrica.

É importante ressaltar que a busca automatizada de hiperparâmetros pode ser computacionalmente intensiva, pois envolve a execução repetida do treinamento do modelo. No entanto, seu uso pode levar a uma melhor seleção de hiperparâmetros e, consequentemente, a um desempenho aprimorado do modelo de *Aprendizagem de Máquina* (AM).

6.1.1 Seleção no Método de Busca

O primeiro passo no processo de otimização de hiperparâmetros é definir um **espaço de busca**, que consiste em uma lista de valores para cada hiperparâmetro. Em

[18]Hiperparâmetros são parâmetros que não são aprendidos pelo modelo, mas que afetam seu desempenho e comportamento.

seguida, é necessário escolher uma estratégia de busca que seja eficiente em encontrar a melhor combinação de hiperparâmetros. Existem várias estratégias disponíveis, incluindo a busca aleatória, a busca em grade e a busca bayesiana.

Dentre as várias abordagens para realizar uma busca automatizada de hiperparâmetros, como *grid search*, *randomized grid search*, *Bayesian search* e *random search*, foi adotada a última para esta etapa da pesquisa. Na *random search* (ou **busca aleatória**) é definida uma distribuição de probabilidade para cada hiperparâmetro candidato. O algoritmo então seleciona aleatoriamente valores para os hiperparâmetros, de acordo com essas distribuições e avalia o desempenho do modelo.

A principal motivação para a escolha da busca aleatória se deve ao fato da mesma ser uma abordagem simples e eficaz para a busca automatizada de hiperparâmetros em determinados cenários, sobretudo quando restrições de tempo e capacidade de processamento estão presentes, como é o nosso caso. Dentre as vantagens da busca aleatória, podemos destacar [Bergstra e Bengio 2012]:

- **Exploração eficiente**: Permite uma exploração ampla e eficiente do espaço de hiperparâmetros. Ao selecionar aleatoriamente valores de hiperparâmetros, ela abrange uma ampla gama de combinações possíveis, explorando diferentes regiões do espaço de busca. Isso é especialmente útil quando não há informações prévias sobre quais hiperparâmetros são mais relevantes, ou quais valores são mais promissores;

- **Redução do viés de seleção**: Em comparação com outras técnicas, como a *busca em grade* (*grid search*), a *busca aleatória* (*random search*) reduz o risco de ficar preso em combinações específicas de hiperparâmetros que podem não ser ideais. A busca em grade pode ser restritiva, pois examina apenas valores específicos pré-definidos. A *busca aleatória*, por sua vez, tem a capacidade de encontrar combinações inesperadas que podem ser vantajosas para o desempenho do modelo.

- **Eficiência computacional**: A *busca aleatória* pode ser computacionalmente mais eficiente do que outras abordagens mais sofisticadas, como a busca Bayesiana ou a otimização baseada em algoritmos genéticos. Essas abordagens geralmente exigem uma maior quantidade de iterações e avaliações de modelos para chegar a um resultado satisfatório. A *busca aleatória*, por outro lado, é mais direta e pode fornecer bons resultados com menos iterações, tornando-a uma escolha prática em cenários onde o poder computacional é limitado.

- **Facilidade de implementação**: A *busca aleatória* é uma técnica simples de implementar. Não requer algoritmos complexos ou ajustes finos de parâmetros. Com apenas algumas linhas de código, é possível realizar uma busca aleatória de hiperparâmetros em qualquer modelo de AM. Isso torna a *busca aleatória* uma escolha

conveniente para prototipagem rápida, experimentação e iterações iniciais durante o processo de desenvolvimento do modelo.

A estratégia desta abordagem é selecionar aleatoriamente um número fixo de combinações de valores de hiperparâmetros dentro do espaço de busca definido. Em seguida, o modelo é treinado usando cada combinação dos hiperparâmetros candidatos e a métrica BLEU será calculada para cada uma delas. A combinação de hiperparâmetros que produzir o melhor desempenho será então selecionada como a melhor combinação de hiperparâmetros para a fase seguinte de otimização do modelo selecionado.

6.1.2 Definição do Espaço de Busca

O espaço de busca refere-se ao conjunto de todos os valores possíveis que os hiperparâmetros de um modelo de AM podem assumir. Cada hiperparâmetro tem um intervalo ou conjunto de valores que pode ser considerado durante a busca de hiperparâmetros [Andonie 2019].

O espaço de busca pode variar dependendo do tipo de hiperparâmetro. Por exemplo, um hiperparâmetro numérico contínuo, como a taxa de aprendizado, pode ser definido por limites inferiores e superiores para o valor do hiperparâmetro e um valor de incremento para varrer o intervalo definido pelos limites estabelecidos. Já um hiperparâmetro categórico, pode ter um espaço de busca representado por um conjunto fixo de opções.

O espaço de busca pode ser unidimensional, quando há apenas um hiperparâmetro envolvido, ou multidimensional, quando envolve vários hiperparâmetros simultaneamente. Em um espaço de busca multidimensional, cada combinação de valores de hiperparâmetros representa um ponto no espaço de busca.

No caso específico do estudo em pauta, foi selecionado o seguinte conjunto de hiperparametros candidatos:

- **Taxa de Aprendizado** (*learning rate*): A taxa de aprendizado é um hiperparâmetro que determina a magnitude do ajuste dos pesos do modelo durante o processo de treinamento. É um valor escalar que controla o tamanho dos passos dados ao atualizar os parâmetros do modelo, com base no gradiente calculado durante a retropropagação. Uma taxa de aprendizado alta pode fazer com que o modelo se ajuste rapidamente, mas pode resultar em oscilações indesejadas. Por outro lado, uma taxa de aprendizado baixa pode tornar o treinamento lento e pode fazer com que o modelo fique preso em mínimos locais. Encontrar um valor adequado para a taxa de aprendizado é crucial para garantir a convergência e o bom desempenho do modelo;
- ***Dropout***: é uma técnica de regularização amplamente utilizada em redes neurais durante o treinamento. É um hiperparâmetro que define a probabilidade de desligar

aleatoriamente um determinado neurônio durante o treinamento, forçando a rede a aprender características robustas e reduzindo o *overfitting*. O *dropout* é aplicado em diferentes camadas da rede neural e ajuda a evitar a dependência excessiva de neurônios específicos. Valores típicos para o *dropout* variam de **0,1** a **0,5**, onde **0,1** indica que 10% dos neurônios são desligados a cada passo de treinamento;

- ***ReLU Dropout***: O *dropout ReLU* (do inglês *Rectified Linear Unit*) é uma variação do *dropout* aplicada aos neurônios com ativação *ReLU* em uma rede neural. Em vez de desligar o neurônio, o *dropout ReLU* substitui o valor de saída do neurônio por zero, com uma probabilidade definida pelo hiperparâmetro *dropout*. Essa técnica ajuda a evitar o *overfitting* e a melhorar a generalização do modelo, especialmente em redes profundas com ativações *ReLU*;

- ***Attention Dropout***: O ***dropout* de atenção** (do inglês *attention dropout*) é uma técnica de regularização aplicada a modelos de atenção, como o *Transformer*. Ela envolve a aplicação de *dropout* às pontuações de atenção durante o cálculo das representações ponderadas. Este hiperparâmetro controla a probabilidade de desligar as conexões de atenção entre as palavras em um modelo de linguagem, por exemplo. Isso ajuda a evitar que o modelo se torne excessivamente dependente de conexões de atenção específicas, promovendo uma representação mais robusta e generalizável;

- **Atualizações de Aquecimento** (*warmup updates*): As atualizações de aquecimento são um hiperparâmetro relacionado à otimização baseada em taxa de aprendizado variável, como a programação de taxa de aprendizado inversa (do inglês *inverse learning rate scheduling*). Este hiperparâmetro define o número de atualizações iniciais em que a taxa de aprendizado é gradualmente aumentada até atingir seu valor máximo. Isso permite que o modelo comece com uma taxa de aprendizado baixa, evitando grandes oscilações nos primeiros passos do treinamento, e, em seguida, aumente gradualmente para acelerar a convergência;

- ***Tokens BPE*** (*byte pair encoding*): *Tokens* BPE é um hiperparâmetro relacionado à codificação de subpalavras. Esse hiperparâmetro define o número máximo de subpalavras geradas pelo processo de codificação BPE[19]. Com um valor maior para o hiperparâmetro de tokens BPE, é possível capturar mais detalhes na estrutura das palavras, mas isso pode aumentar a complexidade do modelo e exigir mais recursos computacionais;

- ***Augmentation***: *Augmentation* é um hiperparâmetro relacionado com *data augmentation*, que envolve a aplicação de transformações artificiais nos dados de trei-

[19]O BPE é um método de tokenização que divide as palavras em subpalavras com base na frequência de ocorrência em um corpus de treinamento.

namento, para aumentar a diversidade e melhorar a capacidade de generalização do modelo. O hiperparâmetro *augmentation* controla a intensidade, ou a quantidade de transformações aplicadas, como rotações, deslocamentos, espelhamentos, cortes etc. Um valor maior do hiperparâmetro *augmentation* resulta em uma maior variedade de dados gerados, mas também pode aumentar o tempo de treinamento e a complexidade do modelo.

O espaço de busca multidimensional completo, com hiperparâmetros e valores, pode ser visto na Tabela 5:

Tabela 5: Espaço de Busca Completo

Hiperparâmetro Candidato	**Intervalor/Conjunto Valores**
learning rate	[0,001; 0,005; 0,01]
dropout	[0,1; 0,2; 0,3]
relu dropout	[0,1; 0,2; 0,3]
attention dropout	[0,1; 0,2; 0,3]
warmup updates	[3000; 4000; 5000]
bpe tokens	[1000; 2000; 4000; 6000; 8000; 10000]
augmentation	[0; 20; 30; 50]

A adequação dos valores do hiperparâmetro depende do contexto específico do problema, do conjunto de dados, da arquitetura do modelo e do algoritmo de otimização escolhido, dentre outros. Para a atribuição dos valores de partida do espaço de busca, foi usada como referência os intervalos e conjuntos mais comuns usados para os hiperparâmetros selecionados em outros contextos. Por exemplo, valores típicos para o hiperparâmetro *learning rate* geralmente variam de **0,1** a **0,0001**. Além disso, a seleção dos valores do espaço de busca envolveu uma etapa prévia de ciclos exploratórios de experimentação e ajuste fino [Bergstra e Bengio 2012].

6.1.3 Planejamento de Experimentos

Para a realização dos experimentos associados, com a busca automatizada de hiperparâmetros do modelo *Transformer Básico*, foram planejadas e realizadas as seguintes etapas:

- Prospecção na literatura e seleção de hiperparâmetros potenciais aplicáveis ao modelo e contexto em pauta;
- Pesquisa na literatura para definição de intervalos, ou conjuntos de valores compatíveis com os hiperparâmetros candidatos;

- Pesquisa na literatura por *frameworks* e/ou plataformas que suportem a estratégia de busca selecionada (*random search*);
- Definição do espaço de busca final com 7 hiperparâmetros candidatos (vide Tabela 5);
- Preparação e separação dos dados em três conjuntos: treinamento, validação e teste;
- Definição e integração da métrica de avaliação (BLEU) no mecanismo de busca automatizada;
- Definição dos ciclos de experimentos necessários;
- Preparação e configuração do ambiente experimental;
- Execução dos experimentos;
- Consolidação e análise dos resultados.

6.1.4 Configuração do Ambiente

Para esta etapa de otimização do modelo *Transformer Básico* foi utilizada a plataforma **Weights & Biases** (W&B)[20]. A W&B é uma plataforma de gerenciamento de experimentos e monitoramento de modelos neurais que ajuda na otimização de modelos de AM. Ele permite o monitoramento e a visualização do desempenho do modelo, incluindo perda e acurácia, bem como facilita na identificação dos melhores hiperparâmetros.

Dentre os critérios para escolha do W&B, está a oferta de ferramentas de análise de treinamento. Dentre elas, se destacam a visualizações de dados de entrada e saída e o monitoramento do processo de treinamento em tempo real. Além disso, há a facilidade de integração com as bibliotecas de AM utilizadas no contexto de avaliação, o que permite o registro automático de experimentos e métricas durante o treinamento do modelo.

Também foi utilizada a biblioteca **SacredBLEU**[21] para o cálculo da métrica *BLEU 4-gram*[22], a qual foi integrada ao *workflow* do *framework* W&B para geração automatizada dos resultados de cada experimento.

O ambiente de experimentação foi configurado com os agentes locais do *framework* W&B, bibliotecas e conjuntos de dados selecionados em um servidor com as seguintes

[20] https://wandb.ai/site

[21] A biblioteca SacreBLEU fornece cálculo simplificado de métricas *BLEU* compartilháveis, comparáveis e reprodutíveis (https://huggingface.co/spaces/evaluate-metric/sacrebleu).

[22] No campo da linguística computacional, um *n-gram* é uma sequência contígua de **n** itens de uma determinada amostra de texto ou fala. Os itens podem ser fonemas, sílabas, letras, palavras ou pares de bases de acordo com a aplicação (https://en.wikipedia.org/wiki/N-gram). Nesta pesquisa, a métrica *BLEU* foi sempre calculada usando sequências *4-gram*.

características: processador AMD Ryzen 2700X, memória RAM de 32GB e placa de vídeo NVIDIA RTX 4080.

6.1.5 Execução dos Experimentos

A execução da busca automatizada foi realizada no período de abril a maio de 2023 e consistiu nas seguintes atividades para cada configuração de hiperparâmetros prevista no espaço de busca:

- Alimentação do modelo com a configuração da vez;
- Treinamento do modelo no conjunto de dados separado para treinamento;
- Avaliação do desempenho do modelo no conjunto de dados de validação;
- Avaliação do desempenho do modelo no conjunto de teste;
- Registro da configuração usada e dos resultados obtidos.

Foi realizado um total de 60 experimentos com duração média de 3 horas. Algumas combinações de hiperparâmetros dentro dos intervalos pré-definidos não resultaram em treinamentos completos e foram descartados pela otimização do próprio *framework* utilizado.

Ao final, um conjunto de 11 treinamentos foram considerados pelo *framework* como relevantes, os quais estão ilustrados na Tabela 6.

Tabela 6: Configuração dos Experimentos Realizados

Exp	***learning rate***	***dropout***	***relu dropout***	***attention dropout***	***warmup ipdates***	***bpe tokens***	***augmentation***
E01	0,001	0,1	0,2	0,1	4000	4000	20
E02	0,001	0,1	0,2	0,2	5000	6000	50
E03	0,001	0,1	0,2	0,3	3000	10000	0
E04	0,001	0,2	0,3	0,1	5000	6000	0
E05	0,001	0,3	0,1	0,2	5000	1000	0
E06	0,001	0,3	0,3	0,1	4000	1000	30
E07	0,001	0,3	0,3	0,2	5000	6000	50
E08	0,005	0,1	0,1	0,1	5000	10000	50
E09	0,005	0,1	0,1	0,3	4000	6000	0
E10	0,005	0,1	0,2	0,3	5000	10000	0
E11	0,005	0,2	0,3	0,3	5000	4000	30

Os resultados obtidos nos experimentos listados na Tabela 6 foram consolidados e serão apresentados e discutidos na próxima seção.

6.1.6 Análise de Resultados

Os resultados dos experimentos, que foram realizados para a identificação dos hiperparâmetros mais relevantes no modelo *Transformer Básico*, para o contexto de TA de LS, estão sumarizados na Tabela 7. Os resultados estão sumarizados através da métrica *BLEU*, a qual foi calculada considerando sequências *4-gram*.

Tabela 7: Resultados obtidos para a métrica *BLEU*

Experimento	***BLEU 4-gram***
E01	57,59
E02	55,83
E03	54,52
E04	56,82
E05	57,95
E06	**58,16**
E07	57,70
E08	55,31
E09	55,65
E10	55,31
E11	55,79
Média	56,06

Como pode ser observado, a média dos resultados obtidos para a métrica *BLEU 4-gram* usando o modelo *Transformer Básico*, **56,06**, foram superiores a média observada para o modelo *LightConv*, **47,61** (Tabela 4) e também superiores ao modelo *ByT5*, **52,52** (Tabela 4), o que indica um bom potencial da estratégia de refinamento da configuração de hiperparâmetros para o contexto em estudo.

A Figura 23 ilustra quais hiperparâmetros foram os melhores preditores e melhor correlacionados com os valores obtidos para a métrica *BLEU*. Esta figura reproduz um "painel de importância" produzido pelo *framework* W&B para apoiar a identificação dos hiperparâmetros que são mais relevantes em termos de previsão de desempenho do modelo.

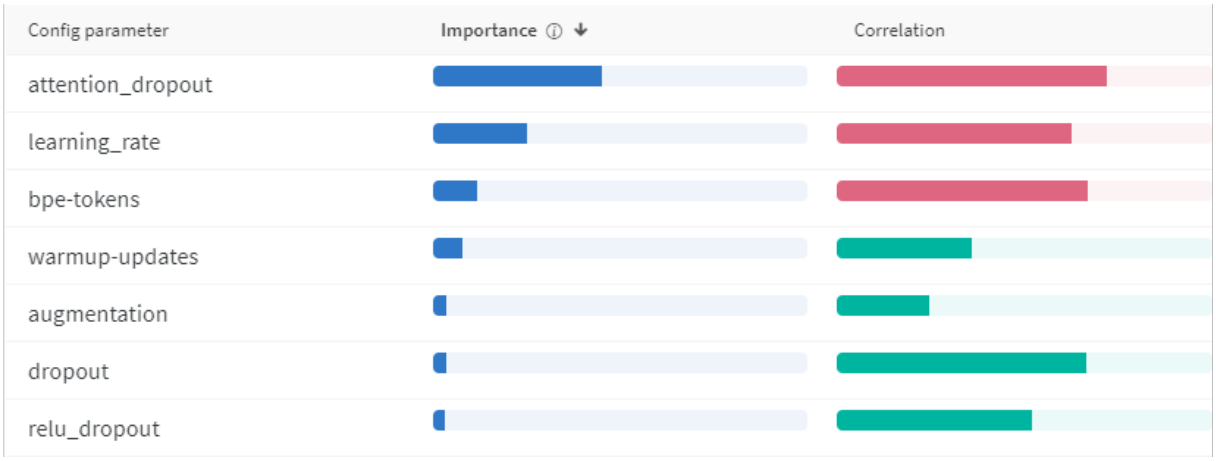

Figura 23: Relevância e Correlação Observada no Espaço de Hiperparâmetros
Fonte: Autor/W&B

O painel de importância contido na Figura 23 traz a coluna *correlation*, a qual informa a correlação linear observada entre o hiperparâmetro e a métrica considerada. Uma correlação alta significa que, quando o hiperparâmetro tem um valor mais alto, a métrica também tem valores mais altos e vice-versa. No caso específico da métrica BLEU, valores mais altos são o desejável.

No painel de importância também é informado o **indicador de importância** (*importance*) do hiperparâmetro, para o qual é treinada uma floresta aleatória (*random forest*) com os hiperparâmetros como entradas e a métrica como saída e são obtidos os valores de relevância do recurso no espaço de hiperparâmetros considerado [23].

A Tabela 8 consolida o percentual de cada hiperparâmetro considerado no estudo.

Tabela 8: Grau de Importância dos Hiperparâmetros na Predição da Métrica BLEU

Hiperparâmetro	Relevância
attention dropout	**44,9%**
learning rate	**25,1%**
bpe tokens	**11,7%**
warmup updates	**7,8%**
augmentation	3,7%
dropout	3,6%
relu dropout	3,2%

Como pode ser observado, o hiperparâmetro que mais contribuiu na predição da métrica *BLEU* foi o *attention dropout*, com 44,9% de contribuição para o *BLEU 4-gram*.

[23] https://wandb.ai/site/articles/exploring-deep-learning-hyperparameters-with-random-forests

É um resultado significativo o fato de apenas um dos hiperparâmetros selecionados estar relacionado com quase metade da variância observada.

Os outros três hiperparâmetros com uma relevância destacada, *learning rate*, *bpe tokens* e *warmup updates*, ajudam a explicar, juntos, cerca de mais 44% da variação, o que aponta para um espaço de hiperparâmetros com boa utilidade na predição da métrica em pauta.

A Figura 24 traz uma representação gráfica do relacionamento dos hiperparâmetros na composição dos resultados para métrica *BLEU 4-gram* e ajudam a visualizar a correlação dos valores adotados na configuração de cada parâmetro com o resultado obtido.

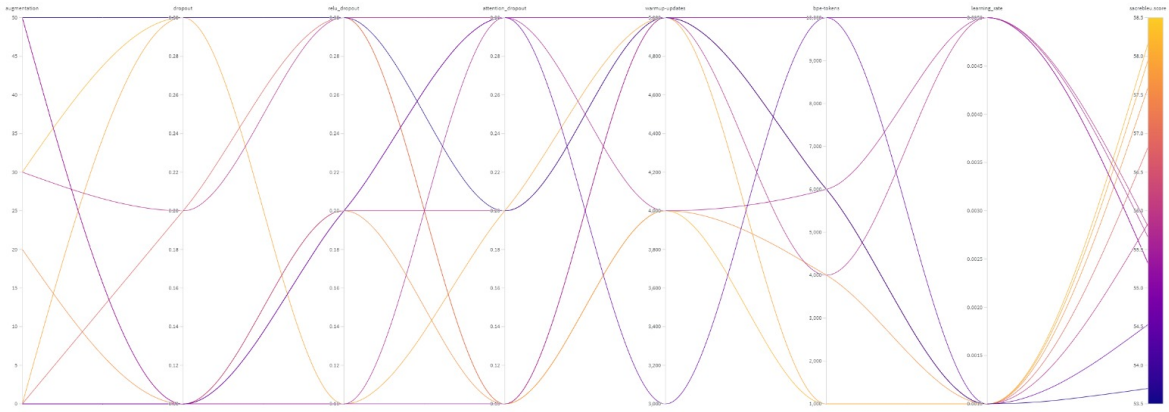

Figura 24: Relacionamento entre os hiperparâmetros para métrica *BLEU*
Fonte: Autor/W&B

Do ponto de vista da correlação, o hiperparâmetro *attention dropout* produziu um valor melhor de *BLEU*, quando configurado com valores próximos de **0,10** e o hiperparâmetro *learning rate* teve melhor desempenho, quando configurado com valores próximos de **0,0010**, ambos com correlação negativa. Por sua vez, o hiperparâmetro *bpe token*, também com correlação negativa, produziu um valor melhor de *BLEU*, quando configurado com valores entre **1.000** e **4.000**, enquanto que o hiperparâmetro *warmup updates*, já com correlação positiva, foi mais contributivo, quando configurado com valores entre **4.000** e **5.000**.

Saindo dos quatro hiperparâmetros mais relevantes, os três restantes tiveram correlação positiva e contribuição discreta. O hiperparâmetro *augmentation* contribuiu melhor com uma configuração entre os valores **20** e **30**, enquanto que o hiperparâmetro *dropout* teve uma melhor contribuição quando configurado com valores próximos de **0,30**. O último hiperparâmetro considerado, *relu dropout*, teve uma melhor contribuição quando configurado com valores próximos de **0,10** e próximos de **0,20**. A configuração dos hiperparâmetros com melhor desempenho está ilustrada na Tabela 9.

Tabela 9: Melhor Configuração Observada dos Hiperparâmetros

Hiperparâmetro	Melhor Configuração
attention dropout	[próximo de 0,10]
learning rate	[próximo de 0,0010]
bpe tokens	[entre 1.000 e 4.000]
warmup updates	[entre 4.000 e 5.000]
augmentation	[entre 20 e 30]
dropout	[0,30]
relu dropout	[entre 0,10 e 0,20]

6.2 Varredura dos Hiperparâmetros Mais Relevantes

Esta etapa de experimentação teve como objetivo a realização de um novo conjunto de treinamentos para a varredura dos quatro hiperparâmetros mais relevantes identificados, considerando a correlação observada na fase experimental anterior (ver Seção 6.1): *attention dropout*, *learning rate*, *bpe tokens* e *warmup updates*.

6.2.1 Estratégia

A varredura de parâmetros (VP) (ou *parameter sweep*) é uma técnica utilizada em otimização para encontrar a melhor combinação de parâmetros em um modelo ou algoritmo. Essa abordagem envolve a exploração sistemática de um conjunto pré-definido de valores para os parâmetros, a fim de determinar qual configuração produz os melhores resultados.

A VP é particularmente útil quando a relação entre os parâmetros e o desempenho não é conhecida, *a priori*, e é necessário explorar diferentes configurações. O seu procedimento segue, geralmente, os seguintes passos:

- **Definição dos Parâmetros**: Identificação dos parâmetros que impactam o desempenho do modelo ou algoritmo;
- **Especificação do Espaço de Parâmetros**: Determinação de um conjunto de valores possíveis para cada parâmetro;
- **Geração de Combinações**: Criação de todas as combinações possíveis de valores para os parâmetros dentro do espaço especificado;
- **Avaliação do Desempenho**: Treinamento e avaliação do modelo para cada combinação de parâmetros;

- **Identificação do Melhor Conjunto de Parâmetros**: Comparação da métrica de desempenho para determinar qual conjunto de parâmetros resultou nos melhores resultados;
- **Refinamento**: Se necessário, é possível realizar iterações adicionais, ajustando o espaço de parâmetros com base nos resultados obtidos.

6.2.2 Projeto de Experimentos

A varredura de parâmetros pretendida compreende a execução de mais **144** experimentos, conforme o planejamento de experimentos contido na Tabela 10.

Tabela 10: Varredura Complementar dos Hiperparâmetros Mais Relevantes

Hiperparâmetro Relevante	Varredura
attention dropout	[0,08; 0,10; 0,12]
learning rate	[0,0008; 0,0010; 0,0012]
bpe tokens	[1500, 2500, 3500, 4500]
warmup updates	[4200, 4400, 4600, 4800]
augmentation	[25]
dropout	[0,30]
relu dropout	[0,15]

6.2.3 Configuração do Ambiente

Assim como ocorreu na fase anterior de *busca aleatória* (Ver Seção 6.1.4), também foram utilizadas a plataforma **Weights & Biases** (W&B) e a biblioteca **SacredBLEU**. O ambiente de experimentação usado foi o mesmo (processador AMD Ryzen 2700X, memória RAM de 32GB e placa de vídeo NVIDIA RTX 4080), já configurado com os agentes locais do *framework* W&B, bibliotecas e conjuntos de dados selecionados.

6.2.4 Execução dos Experimentos

A execução da varredura de parâmetros descrita na Seção 6.2.2 foi realizada no período de setembro a novembro de 2023 e consistiu nas seguintes atividades para cada configuração de hiperparâmetros prevista no projeto de experimentos:

- Alimentação do modelo com a configuração da vez;
- Treinamento do modelo no conjunto de dados separado para treinamento;
- Avaliação do desempenho do modelo no conjunto de dados de validação;

- Avaliação do desempenho do modelo no conjunto de teste;
- Registro da configuração usada e dos resultados obtidos.

Foi realizado um total de 144 experimentos com duração média de 3 horas. Apenas quatro das combinações de hiperparâmetros, dentro dos intervalos pré-definidos, não resultaram em treinamentos completos e foram descartados pela otimização do próprio *framework* utilizado.

Ao final, um conjunto de 140 treinamentos produziram valores válidos para a métrica BLEU, cuja variação pode ser vista na Figura 25.

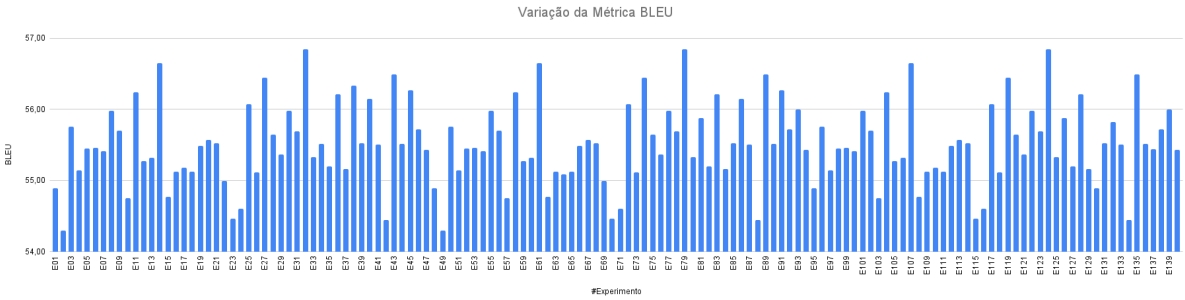

Figura 25: Variação da métrica BLEU observada na varredura de hiperparâmetros
Fonte: Autor

6.2.5 Análise de Resultados

Os resultados dos experimentos de varredura dos hiperparâmetros mais relevantes no modelo *Transformer Básico*, para o contexto de TA de LS, estão sumarizados na Figura 26.

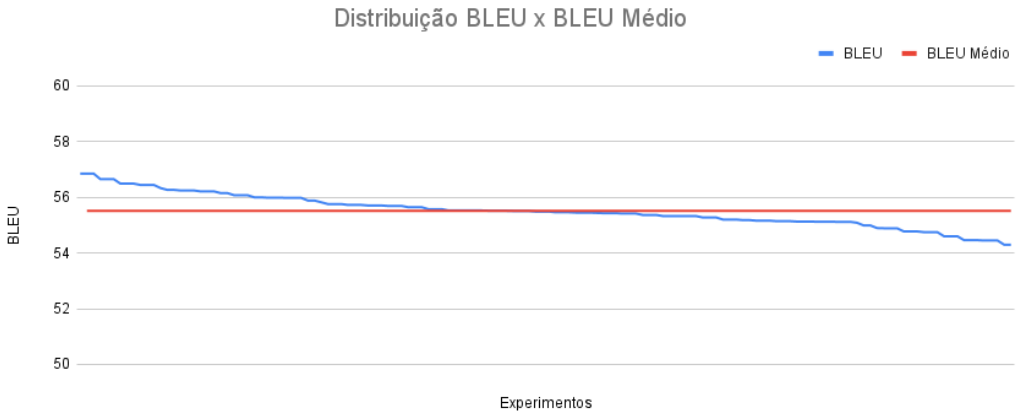

Figura 26: Distribuição decrescente do BLEU obtido em cada experimento, em comparação com o valor médio do BLEU
Fonte: Autor

Os resultados da varredura estão representados através da métrica *BLEU 4-gram* e foram organizados de forma descrescente na Figura 26 (linha azul), com a referência do BLEU médio apurado (linha vermelha). O BLEU médio dos experimentos da varredura foi **55,51%** para um BLEU máximo de **56,85%** e um BLEU mínimo de **54,20%**, o que representa uma variação de **2,65%** da métrica, no espaço de configuração dos hiperparâmetros de interesse.

O hiperparametro *Attention Dropout*, o mais relevante na variação da métrica na *random search* realizada anteriormente, obteve o melhor resultado da métrica BLEU, com todos os valores testados na *parameter sweep*. O BLEU máximo (**56,85**) foi encontrado com as três configurações de *Attention Dropout* utilizadas na varredura: {**0,08**; **0,10**; **0,12**}.

O segundo hiperparâmetro mais relevante encontrado na *random search*, *Learning Rate*, repetiu o bom desempenho anterior, quando configurado com o valor **0,0010** e também obteve o BLEU máximo da varredura. Como pode ser visto na Figura 27, quando esta configuração foi levemente aumentada para **0,0012** e diminuída para **0,0008**, houve um decréscimo no BLEU observado, o que talvez possa indicar que o valor **0,0010** seja a configuração ideal para o cenário em estudo.

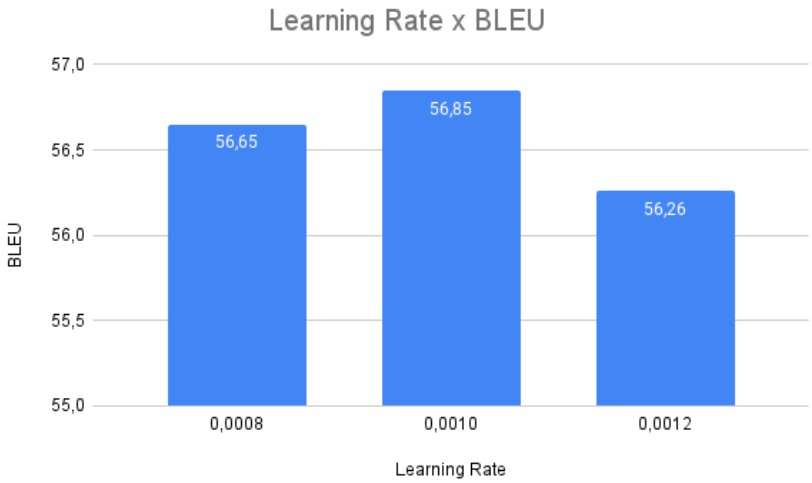

Figura 27: Máximo valor obtido para a métrica *BLEU* para cada variação do hiperparâmetro *Learning Rate*
Fonte: Autor

No caso de *BPE Tokens*, o terceiro hiperparâmetro mais relevante, apenas a configuração com o valor **3500** obteve o BLEU máximo da varredura realizada. A Figura 28 traz os valores máximos de BLEU obtidos para cada configuração deste hiperparâmetro.

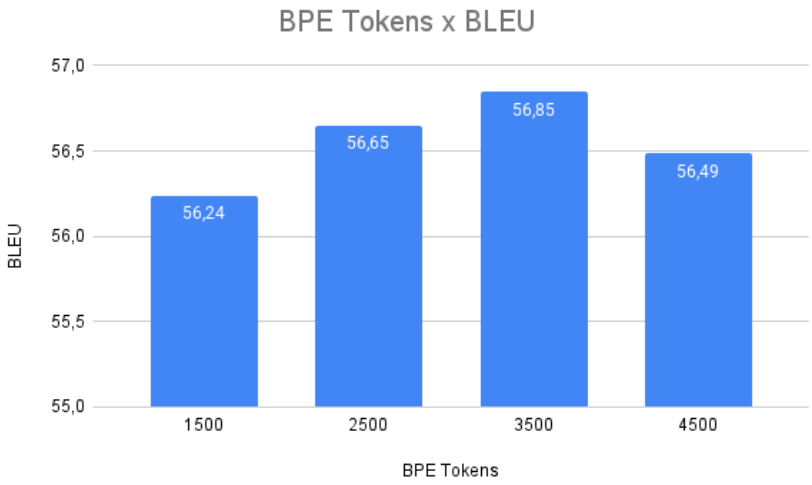

Figura 28: Máximo valor obtido para a métrica *BLEU* para cada variação do hiperparâmetro *BPE Tokens*
Fonte: Autor

Finalmente, no caso do último hiperparâmetro avaliado, *Warmup Updates*, o valor **4800** representou a melhor configuração observada e também alcançou o BLEU máximo. Como pode ser visto na Figura 29, quando este **4800** foi configurado com **4200**, **4400** e **4600**, os valores obtidos da métrica BLEU foram todos inferiores ao máximo observado na varredura.

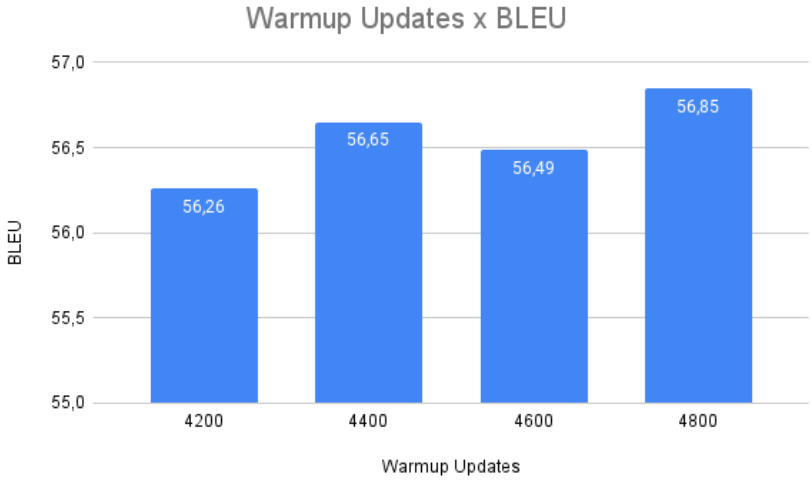

Figura 29: Máximo valor obtido para a métrica *BLEU* para cada variação do hiperparâmetro *Warmup Updates*
Fonte: Autor

A Figura 30 traz a média de todos os 140 resultados obtidos para a métrica *BLEU 4-gram*, usando o modelo *Transformer Básico* durante a *parameter sweep*, **55,51**. Como pode ser observado ela foi superior em **14,23**% à média observada para o modelo *Light-*

Conv, **47,61** (Tabela 4), e também superior em **5,39**% ao modelo *ByT5*, **52,52** (Tabela 4). Tal melhoria pode indicar um bom potencial da estratégia de refinamento da configuração de hiperparâmetros para o contexto em estudo.

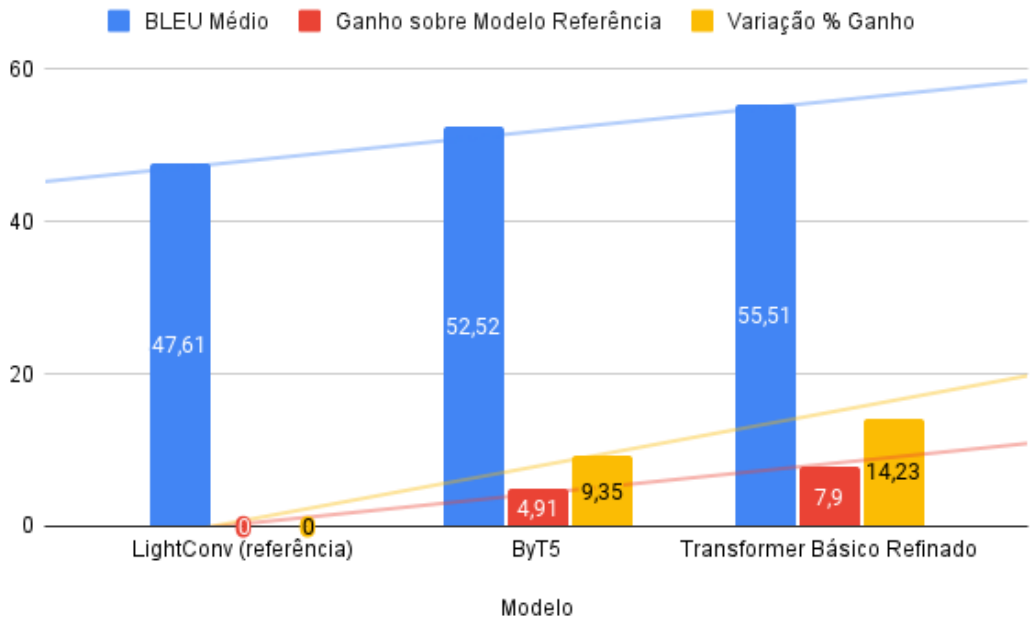

Figura 30: Comparativo do ganho da média da métrica BLEU com cada modelo avaliado em relação ao modelo de referência
Fonte: Autor

6.3 Avaliação Complementar

Foi realizada também uma avaliação complementar do modelo *Transformer Básico* refinado utilizando a técnica de *back translation*, a qual vem sendo frequentemente utilizada para melhorar a tradução produzida por modelos *Transformers* [Wang et al. 2021].

A técnica de *back translation* é um método de aprimoramento de modelos de tradução automática e consiste em traduzir uma sentença do idioma de origem para um terceiro idioma e, em seguida, traduzir a sentença resultante de volta para o idioma original. Comparando a sentença original com a versão retro traduzida (*back translated*), os modelos podem aprender a minimizar erros e melhorar a qualidade da tradução. Essa abordagem ajuda a criar conjuntos de dados adicionais (*data augmentation*) para treinamento, os quais são especialmente úteis quando dados paralelos são escassos. Ao expor o modelo a uma variedade de sentenças e contextos, durante o processo de retro tradução, ele é capaz de aprender padrões mais complexos e aprimorar seu desempenho na geração de traduções mais precisas e fluentes, o que faz *back translation* ser uma estratégia eficaz para melhorar a robustez e a generalização de modelos *transformers* em tarefas de tradução [Edunov et al. 2018].

No mesmo ambiente utilizado nos experimentos de *random search* e *parameter sweep*, realizados nas fases anteriores dessa pesquisa, foi instalada e configurada a arquitetura do modelo de tradução *AutoModelForSeq2SeqLM*. No processo de *data augmentation*, com *back translation*, foi usado o modelo pré treinado **opus-mt-ROMANCE-en**[24], para traduzir para inglês, e o modelo **opus-mt-en-ROMANCE**[25], para traduzir para português. Um script específico incorporado ao *pipeline* de treinamento do VLibras, utilizando o tokenizador **AutoTokenizer**[26], faz a tradução para inglês e depois de volta para português para, em seguida, fazer a inclusão no *corpus*. O aumento de dados obtido no processo resultou em 48.028 novas frases.

Nesta fase de experimentação foram utilizadas as configurações de hiperparâmetros que produziram os melhores valores da métrica BLEU observadas durante a *random search* e a *parameter sweep*, as quais estão detalhadas na Tabela 11.

Tabela 11: Configuração dos Hiperparâmetros usada com *Back Translation*

Hiperparâmetro	**Melhor Configuração *Random Search***	**Melhor Configuração *Parameter Sweep***
attention dropout	0,10	0,10
learning rate	0,001	0,001
bpe tokens	1000	3500
warmup updates	4000	4800
augmentation	30	25
dropout	0,30	0,30
relu dropout	0,30	0,15

Os resultados obtidos estão sumarizados na Figura 31. Como pode ser observado, a média dos melhores resultados obtida utilizando *back translation*, **57,68**, foi apenas **0,17** superior que a média obtida com as mesmas configurações sem usar a técnica (**57,51**).

[24]https://huggingface.co/Helsinki-NLP/opus-mt-ROMANCE-en
[25]https://huggingface.co/Helsinki-NLP/opus-mt-en-ROMANCE
[26]https://huggingface.co/docs/transformers/v4.36.1/en/model$_doc/autotransformers.AutoTokenizer$

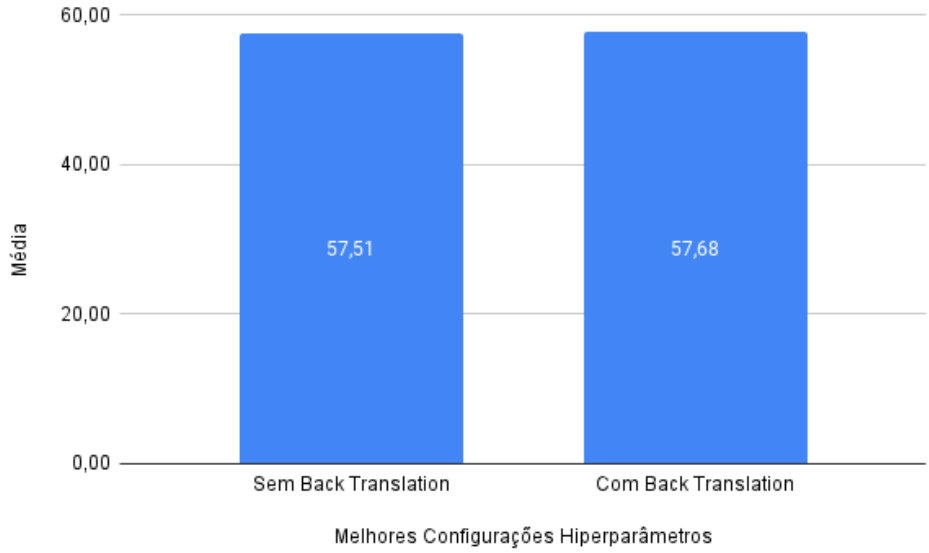

Figura 31: Média da métrica BLEU obtida com e sem *back translation* quando aplicada as melhores configurações de hiperparâmetros observada na *random search* e na *parameter sweep*
Fonte: Autor

Não houve um ganho significativo na métrica de interesse, o que talvez possa ser explicado pela dificuldades do modelo em generalizar efetivamente a diversidade de exemplos de tradução inversa, quando o conjunto de dados não é representativo o suficiente, o que pode ser o caso em um cenário de *low-resources language*. Embora a técnica de *back translation* seja uma ferramenta valiosa e tenha apresentado resultados relevantes em muitos contextos, é importante considerar suas limitações e, em alguns casos, explorar abordagens adicionais. Neste sentido e como potenciais investigações futuras, a partir deste trabalho podem ser realizadas avaliações humanas para aferir se a qualidade das traduções geradas com o uso de *back translation* são superiores, mesmo apresentando um BLEU praticamente equivalente.

6.4 Discussão

Após a prospecção e avaliação de modelos potenciais, o refinamento de hiperparâmetros, através de *random search* e *parameter sweep*, e a incorporação da técnica de *back translation*, foi obtido um aumento de *10,07* no valor médio da métrica BLEU nos mesmos conjuntos de treinamento, validação e testes utilizados atualmente na Suíte VLibras. Isso representa um acréscimo de *17,45%*, que o uso do modelo *Transformer Básico* apresentou sobre o BLEU médio obtido pelo modelo atualmente em produção no VLibras (Lightconv).

A evolução da métrica BLEU obtido em cada fase experimental está ilustrada na

Figura 32.

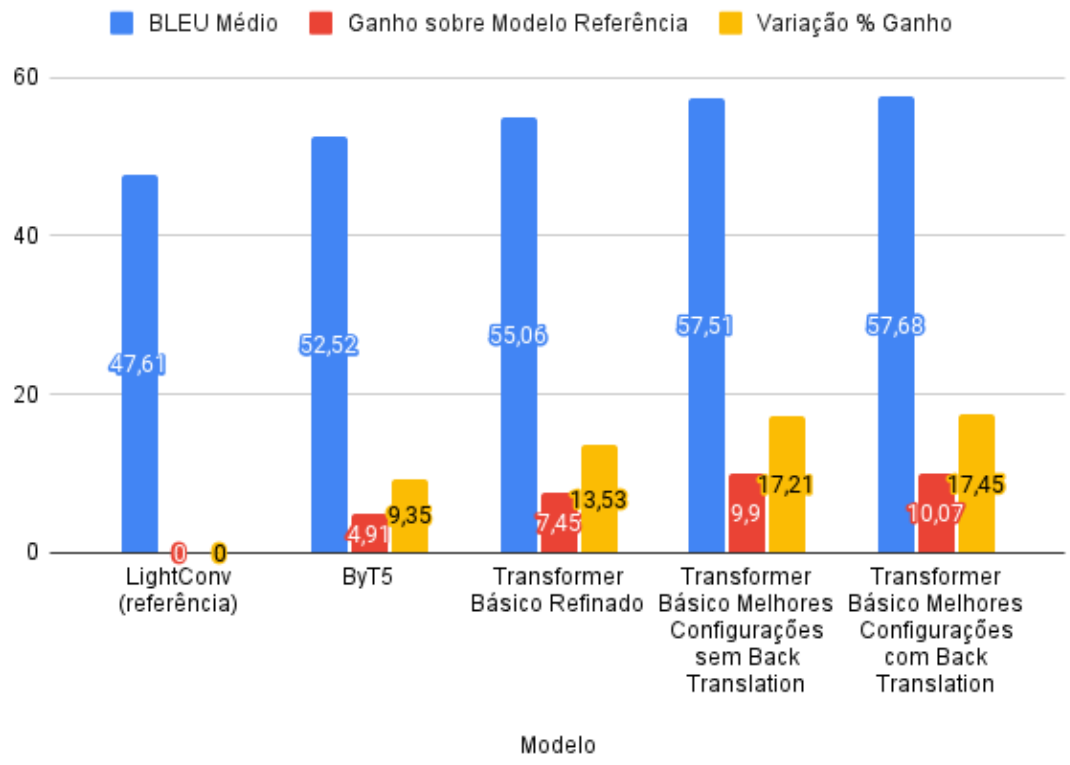

Figura 32: Evolução da métrica BLEU obtida durante as fases experimentais com relação ao modelo de referência
Fonte: Autor

O modelo *ByT5* já apresentou um ganho **9,35**% na fase de prospecção dos modelos candidatos. Entretanto, como se tratava de uma alternativa pré-treinada, sem margem para refinamento, o segundo modelo melhor classificado, *Transformer Básico*, foi o adotado para a fase seguinte de experimentação.

Através de um processo de *random search*, um conjunto de hiperparâmetros foi aplicado ao modelo *Transformer Básico* dentro de um determinado espaço de busca. Além de produzir um subconjunto dos hiperparâmetros mais relevantes, dentre os considerados, a *random search* também já forneceu uma melhoria do desempenho do modelo *Transformer Básico*, superando o modelo *ByT5* já nesta fase, e o melhor valor da métrica BLEU, obtido em todos os treinamento e testes.

Usando os hiperparâmetros relevantes indicados na fase anterior, foi realizado um processo de *parameter sweep*, para combinar o subconjunto do espaço de busca original, onde os melhores resultados da *random search* se concentraram. A melhora média da métrica BLEU obtida pelo modelo *Transformer Básico*, refinado em todos os 140 experimentos do *parameter sweep*, foi **13,53**% ao modelo de referência, o que pode indicar

uma capacidade interessante da *random search* identificação, seleção e montagem de configurações de hiperparâmetros.

Quando são consideradas apenas as configurações de hiperparâmetros que produziram os melhores resultados da métrica BLEU com o modelo *Transformer Básico* na *random search* e na *parameter sweep*, candidatas a serem usadas em um ambiente de produção, foi obtido um ganho médio de **17,21**% sobre o BLEU produzido pelo modelo *Lightconv*. A aplicação da técnica de *Back Translation* nas melhores combinações de hiperparâmetros ainda permitiu elevar o ganho para **17,45**%.

7 CONCLUSÃO

A presente pesquisa teve como objetivo realizar um estudo comparativo de modelos neurais *Transformers*, potencialmente aplicáveis na evolução do componente de tradução automática da Suíte VLibras. A plataforma em questão trata do processo de tradução do tipo texto para texto entre a língua portuguesa e a Língua Brasileira de Sinais (Libras). Para tanto, foi realizado um levantamento na literatura dos principais métodos de tradução que surgiram nos últimos anos, em especial no interstício entre 2017 e 2022. Uma breve descrição sobre a evolução dos métodos de tradução desenvolvidos ao longo deste período foi realizada.

A partir do levantamento dessa pesquisa, foi identificado um conjunto de trabalhos que endereçam o problema de TA, em especial trabalhos com foco no problema de tradução entre línguas faladas e línguas de sinais. Observamos que o uso de arquiteturas baseadas em mecanismo de atenção e, em especial, os modelos *transformers* ganharam grande relevância nos últimos anos, visto as melhorias na qualidade de tradução apresentadas por tais métodos.

Foi constatado que as soluções baseadas em arquiteturas *Transformers* são o estado da arte para praticamente todos os problemas de NLP e, até mesmo, para problemas de visão computacional através dos **Vision Transformers** [Dosovitskiy et al. 2020], sendo o novo padrão da indústria para vários problemas práticos. Nesse contexto, os experimentos foram focados em avaliar se tal adequação poderia também ser aplicada em contextos de *low-resources NLP*, que é o caso das línguas de sinais.

Já na primeira fase do estudo, foi possível constatar, através de uma série de experimentos, que a adoção de uma dessas arquiteturas viáveis (*Transformer Básico* ou *ByT5*) ajudaria a aumentar a precisão e qualidade do componente de tradução da Suíte VLibras, trazendo um aumento percentual máximo de até 12,73% nas traduções perfeitas e diminuindo as traduções incorretas em 16,21% e uma melhoria de 10,31% na métrica BLEU.

A partir de prospecção e avaliação dos modelos *Transformers*, considerando que o processo de seleção de modelos candidatos teve um espaço de busca mais amplo, foi realizado um estudo mais aprofundado para tentar otimizar o segundo modelo melhor classificado, *Transformer Básico*, pois o modelo melhor classificado, *ByT5*, já é pré treinado e não apresenta espaço para otimização.

Na segunda fase de experimentação, foi realizado um processo de identificação de hiperparâmetros relevantes mais elaborado. Foram considerados os seguintes hiperparâmetros no espaço de busca: *bpe tokens*, *warmup updates*, *relu dropout*, *attention dropout*, *dropout*, *augmentation* e *learning rate*.

O hiperparâmetro que mais contribuiu na predição do cálculo da métrica *BLEU*, em todas as quatro variações consideradas, foi o *attention dropout*, respondendo por quase metade da variância observada (44,9%). Os outros três hiperparâmetros com uma relevância destacada, *learning rate*, *bpe tokens* e *warmup updates*, ajudam a explicar, juntos, cerca de mais de 44% da variação, o que aponta para que o espaço de hiperparâmetros observado possui boa utilidade na predição da métrica de interesse.

A média dos resultados obtidos para a métrica *BLEU* usando o modelo *Transformer Básico*, reconfigurado com as melhores combinações de hiperparâmetros, foi de **57,68**. É um superior em **17,45%** à média observada para o modelo *LightConv*, **47,61** (vide Tabela 4) e também superior em **8,95**% à média obtida pelo modelo *ByT5*, **52,52** (ver Tabela 4). Essa melhoria pode indicar um bom potencial da estratégia de refinamento da configuração de hiperparâmetros com *random search* e *parameter sweep* para o contexto em estudo.

Os resultados demonstraram que os modelos baseados na arquitetura *Transformer* são promissores e podem ser considerados para uma eventual substituição do modelo neural usado na abordagem híbrida da Suíte VLibras e, até mesmo, para uma simplificação do componente tradutor da Suíte VLibras, tornando-o puramente neural.

Algumas limitações desta pesquisa podem ser endereçadas no futuro, como a realização de uma investigação mais profunda do impacto que as técnicas de *data augmentation* e *back translation* podem trazer para a qualidade da TA no contexto considerado, sobretudo como uma estratégia para lidar com a falta de dados de treinamento adequados.

Abordar outra limitação do trabalho, relacionada com a cobertura da revisão da literatura realizada, a qual só alcançou artigos publicados até 2022, também pode ser interessante para atualização e validação dos resultados obtidos. Quando consideramos também os anos de 2023 e 2024, muitos trabalhos apresentam convergência e relação com o que foi realizado durante esta pesquisa. Por exemplo, De Martino et al. (2024) também investigam o uso de *transformers* em TA para Libras explorando a aprendizagem por transferência de modelos pré-treinados de dez pares de idiomas diferentes [Martino e Christinele 2024]. Angel et al. (2023), por sua vez, comparam modelos de tradução automática baseados em transformers para idiomas de baixos recursos da Colômbia e do México [Angel et al. 2023]. O trabalho de Vu et al. (2023) analisa a escalabilidade de técnicas de aumento de dados para tradução automática de baixo recurso entre chinês e vietnamita [Vu e Bui 2023]. Já Makwisai et al. (2023) defendem que há um impacto profundo na incorporação de informações de árvores de dependência no processo de ajuste fino de sistemas de tradução automática (MT) baseados em *Transformers* [Makwisai et al. 2023].

Ainda como exemplo de trabalhos futuros potenciais, dependendo da viabilidade

prática, também seria interessante fazer uma avaliação real da tradução obtida com os novos modelos avaliados com usuários da comunidade surda. A avaliação realizada através de métricas computacionais fornece uma visão mais quantitativa do desempenho dos modelos, sendo sempre desejável, quando possível, realizar também uma avaliação qualitativa dos modelos com usuários e especialistas de Libras. Esse processo é útil, quando viável, pois não é incomum que, no contexto de processamento de linguagem natural, modelos com melhores métricas computacionais não são, necessariamente, melhor avaliados por usuários. Além disso, a análise da aplicabilidade da técnica de ajuste fino de modelos de linguagem grandes (*LLM fine-tuning*) pode expandir o refinamento realizado e representar um caminho de investigação com potencial no problema tratado aqui [Han et al. 2024].

REFERÊNCIAS

[Abujar et al. 2021]ABUJAR, S. et al. English to bengali neural machine translation using global attention mechanism. In: SPRINGER. *Emerging Technologies in Data Mining and Information Security*. [S.l.], 2021. p. 359–369.

[Almasoud e Al-Khalifa 2011]ALMASOUD, A. M.; AL-KHALIFA, H. S. A proposed semantic machine translation system for translating arabic text to arabic sign language. In: ACM. *Proceedings of the Second Kuwait Conference on e-Services and e-Systems*. [S.l.], 2011. p. 23.

[Almeida 2013]ALMEIDA, W. G. Introdução à língua brasileira de sinais. *Letras Vernáculas*, 2013.

[Amin, Hefny e Ammar 2021]AMIN, M. R.; HEFNY, H.; AMMAR, M. Sign language gloss translation using deep learning models. *International Journal of Advanced Computer Science and Applications*, v. 12, n. 11, 2021.

[Andonie 2019]ANDONIE, R. Hyperparameter optimization in learning systems. *Journal of Membrane Computing*, Springer, v. 1, n. 4, p. 279–291, 2019.

[Angel et al. 2023]ANGEL, J. et al. Comparing transformer-based machine translation models for low-resource languages of colombia and mexico. In: SPRINGER. *Mexican International Conference on Artificial Intelligence*. [S.l.], 2023. p. 95–105.

[Angelova, Avramidis e Möller 2022]ANGELOVA, G.; AVRAMIDIS, E.; MÖLLER, S. Using neural machine translation methods for sign language translation. In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*. [S.l.: s.n.], 2022. p. 273–284.

[Antony 2013]ANTONY, P. Machine translation approaches and survey for indian languages. *International Journal of Computational Linguistics & Chinese Language Processing, Volume 18, Number 1, March 2013*, v. 18, n. 1, 2013.

[Araújo 2012]ARAÚJO, T. M. U. Uma solução para geração automática de trilhas em língua brasileira de sinais em conteúdos multimídia. Universidade Federal do Rio Grande do Norte, 2012.

[Arvanitis, Constantinopoulos e Kosmopoulos 2019]ARVANITIS, N.; CONSTANTINOPOULOS, C.; KOSMOPOULOS, D. Translation of sign language glosses to text using sequence-to-sequence attention models. In: IEEE. *2019 15th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS)*. [S.l.], 2019. p. 296–302.

[Bahdanau, Cho e Bengio 2014]BAHDANAU, D.; CHO, K.; BENGIO, Y. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[Barnes 2019]BARNES, J. *Cross-lingual sentiment analysis for under-resourced languages*. Tese (Doutorado) — Universitat Pompeu Fabra, 2019.

[Bergstra e Bengio 2012]BERGSTRA, J.; BENGIO, Y. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, v. 13, n. Feb, p. 281–305, 2012.

[Bojar et al. 2014]BOJAR, O. et al. Findings of the 2014 workshop on statistical machine translation. In: *Proceedings of the ninth workshop on statistical machine translation*. [S.l.: s.n.], 2014. p. 12–58.

[Bojar et al. 2016]BOJAR, O. et al. Findings of the 2016 conference on machine translation. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*. [S.l.], 2016. p. 131–198.

[Brour e Benabbou 2021]BROUR, M.; BENABBOU, A. Atlaslang nmt: Arabic text language into arabic sign language neural machine translation. *Journal of King Saud University-Computer and Information Sciences*, Elsevier, v. 33, n. 9, p. 1121–1131, 2021.

[Camgoz et al. 2018]CAMGOZ, N. C. et al. Neural sign language translation. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. [S.l.: s.n.], 2018. p. 7784–7793.

[Camgoz et al. 2020]CAMGOZ, N. C. et al. Sign language transformers: Joint end-to-end sign language recognition and translation. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. [S.l.: s.n.], 2020. p. 10023–10033.

[Cettolo et al. 2014]CETTOLO, M. et al. Report on the 11th iwslt evaluation campaign. In: *Proceedings of the 11th International Workshop on Spoken Language Translation: Evaluation Campaign*. [S.l.: s.n.], 2014.

[Chen et al. 2021]CHEN, C. et al. bert2bert: Towards reusable pretrained language models. *CoRR*, abs/2110.07143, 2021.

[Chéragui 2012]CHÉRAGUI, M. A. Theoretical overview of machine translation. *Proceedings ICWIT*, Citeseer, p. 160, 2012.

[Cho et al. 2014]CHO, K. et al. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.

[Cho et al. 2014]CHO, K. et al. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[Chung et al. 2014]CHUNG, J. et al. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[Coelho 2020]COELHO, F. F. Classificação orientada a objetos e redes neurais artificiais para mapeamento digital de classes de solos. 2020.

[Corrêa e Cruz 2019]CORRÊA, Y.; CRUZ, C. R. *Língua brasileira de sinais e tecnologias digitais*. [S.l.]: Penso Editora, 2019.

[Corrêa, Gomes e Cruz 2018]CORRÊA, Y.; GOMES, R. P.; CRUZ, C. R. A desambiguação de palavras homônimas em sentenças por aplicativos de tradução automática português brasileiro-libras. *Trabalhos em Linguística Aplicada*, v. 57, n. 1, p. 319–351, 2018.

[Costa e Dimuro 2001]COSTA, A. C. da R.; DIMURO, G. P. Signwriting-based sign language processing. In: SPRINGER. *International Gesture Workshop*. [S.l.], 2001. p. 202–205.

[Denil et al. 2012]DENIL, M. et al. Learning where to attend with deep architectures for image tracking. *Neural computation*, MIT Press, v. 24, n. 8, p. 2151–2184, 2012.

[Devlin et al. 2018]DEVLIN, J. et al. Bert: Pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

[Ding, Renduchintala e Duh 2019]DING, S.; RENDUCHINTALA, A.; DUH, K. A call for prudent choice of subword merge operations. *CoRR*, abs/1905.10453, 2019.

[Doell 2020]DOELL, C. *Recurrent Neural Network (RNN), Long-Short Term Memory (LSTM) Gated Recurrent Unit (GRU)*. 2020. Disponível em: <http://dprogrammer.org/rnn-lstm-gru>.

[Dosovitskiy et al. 2020]DOSOVITSKIY, A. et al. An image is worth 16x16 words: Transformers for image recognition at scale. *CoRR*, abs/2010.11929, 2020.

[Dryer e Haspelmath 2011]DRYER, M. S.; HASPELMATH, M. The world atlas of language structures (wals) online. *Max Planck Digital Library, Munich*, 2011.

[Edunov et al. 2017]EDUNOV, S. et al. Classical structured prediction losses for sequence to sequence learning. *arXiv preprint arXiv:1711.04956*, 2017.

[Edunov et al. 2018]EDUNOV, S. et al. Understanding back-translation at scale. *arXiv preprint arXiv:1808.09381*, 2018.

[Edunov et al. 2018]EDUNOV, S. et al. *Understanding Back-Translation at Scale*. 2018.

[Elbayad 2020]ELBAYAD, M. *Rethinking the Design of Sequence-to-Sequence Models for Efficient Machine Translation*. Tese (Doutorado) — Université Grenoble Alpes [2020-....], 2020.

[Elbayad, Besacier e Verbeek 2018]ELBAYAD, M.; BESACIER, L.; VERBEEK, J. Pervasive attention: 2d convolutional neural networks for sequence-to-sequence prediction. *arXiv preprint arXiv:1808.03867*, 2018.

[Fadaee, Bisazza e Monz 2017]FADAEE, M.; BISAZZA, A.; MONZ, C. Data augmentation for low-resource neural machine translation. *arXiv preprint arXiv:1705.00440*, 2017.

[Farooq et al. 2021]FAROOQ, U. et al. Advances in machine translation for sign language: approaches, limitations, and challenges. *Neural Computing and Applications*, Springer, v. 33, n. 21, p. 14357–14399, 2021.

[Filho et al. 2018]FILHO, J. W. et al. The brwac corpus: A new open resource for brazilian portuguese. In: EUROPEAN LANGUAGE RESOURCES ASSOCIATION (ELRA). *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. [S.l.], 2018.

[Freitas, Rocha e Bick 2008]FREITAS, C.; ROCHA, P.; BICK, E. Floresta sintá (c) tica: bigger, thicker and easier. In: SPRINGER. *International Conference on Computational Processing of the Portuguese Language*. [S.l.], 2008. p. 216–219.

[Gage 1994]GAGE, P. A new algorithm for data compression. *C Users Journal*, v. 12, n. 2, p. 23–38, 1994.

[Gago et al. 2019]GAGO, J. J. et al. Sequence-to-sequence natural language to humanoid robot sign language. *arXiv preprint arXiv:1907.04198*, 2019.

[Gao et al. 2022]GAO, P. et al. Bi-simcut: A simple strategy for boosting neural machine translation. *arXiv preprint arXiv:2206.02368*, 2022.

[Gehring et al. 2017]GEHRING, J. et al. Convolutional sequence to sequence learning. In: JMLR. ORG. *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. [S.l.], 2017. p. 1243–1252.

[Gómez, McGill e Saggion 2021]GÓMEZ, S. E.; MCGILL, E.; SAGGION, H. Syntax-aware transformers for neural machine translation: The case of text to sign gloss translation. In: *Proceedings of the 14th Workshop on Building and Using Comparable Corpora (BUCC 2021)*. [S.l.: s.n.], 2021. p. 18–27.

[Gong et al. 2018]GONG, C. et al. Frage: Frequency-agnostic word representation. In: *Advances in Neural Information Processing Systems*. [S.l.: s.n.], 2018. v. 31.

[Guo et al. 2018]GUO, D. et al. Hierarchical lstm for sign language translation. In: *Thirty-Second AAAI Conference on Artificial Intelligence*. [S.l.: s.n.], 2018.

[Ha et al. 2015]HA, T. L. et al. The kit translation systems for iwslt 2015. In: *Proceedings of the 12th International Workshop on Spoken Language Translation: Evaluation Campaign*. [S.l.: s.n.], 2015.

[Hamed, Helmy e Mohammed 2022]HAMED, H.; HELMY, A.; MOHAMMED, A. Holy quran-italian seq2seq machine translation with attention mechanism. In: IEEE. *2022 2nd International Mobile, Intelligent, and Ubiquitous Computing Conference (MIUCC)*. [S.l.], 2022. p. 11–20.

[Han et al. 2024]HAN, Z. et al. Parameter-efficient fine-tuning for large models: A comprehensive survey. *arXiv preprint arXiv:2403.14608*, 2024.

[Hanke 2004]HANKE, T. Hamnosys-representing sign language data in language resources and language processing contexts. In: *LREC*. [S.l.: s.n.], 2004. v. 4, p. 1–6.

[Hanke 2010]HANKE, T. Hamnosys–hamburg notation system for sign languages. *Institute of German Sign Language, Accessed in*, v. 7, 2010.

[Haque, Liu e Way 2021]HAQUE, R.; LIU, C.-H.; WAY, A. Recent advances of low-resource neural machine translation. *Machine Translation*, Springer, p. 1–24, 2021.

[Hirschberg e Manning 2015]HIRSCHBERG, J.; MANNING, C. D. Advances in natural language processing. *Science*, American Association for the Advancement of Science, v. 349, n. 6245, p. 261–266, 2015.

[Hochreiter 1998]HOCHREITER, S. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, World Scientific, v. 6, n. 02, p. 107–116, 1998.

[Hochreiter e Schmidhuber 1997]HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. *Neural computation*, MIT Press, v. 9, n. 8, p. 1735–1780, 1997.

[Junczys-Dowmunt et al. 2018]JUNCZYS-DOWMUNT, M. et al. Marian: Fast neural machine translation in c++. *arXiv preprint arXiv:1804.00344*, 2018.

[Kahlon e Singh 2021]KAHLON, N. K.; SINGH, W. Machine translation from text to sign language: a systematic review. *Universal Access in the Information Society*, p. 1–35, 2021.

[Kaiser, Gomez e Chollet 2017]KAISER, L.; GOMEZ, A. N.; CHOLLET, F. Depthwise separable convolutions for neural machine translation. *arXiv preprint arXiv:1706.03059*, 2017.

[Kalchbrenner e Blunsom 2013]KALCHBRENNER, N.; BLUNSOM, P. Recurrent continuous translation models. In: *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. [S.l.: s.n.], 2013. p. 1700–1709.

[Kim et al. 2021]KIM, K. et al. Self-knowledge distillation with progressive refinement of targets. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. [S.l.: s.n.], 2021. p. 6567–6576.

[Kituku, Muchemi e Nganga 2016]KITUKU, B.; MUCHEMI, L.; NGANGA, W. A review on machine translation approaches. *Indonesian Journal of Electrical Engineering and Computer Science*, v. 1, n. 1, p. 182–190, 2016.

[Koehn e Knowles 2017]KOEHN, P.; KNOWLES, R. Six challenges for neural machine translation. *arXiv preprint arXiv:1706.03872*, 2017.

[Kong, Zhang e Hovy 2020]KONG, X.; ZHANG, Z.; HOVY, E. Incorporating a local translation mechanism into non-autoregressive translation. *arXiv preprint arXiv:2011.06132*, 2020.

[Kudo e Richardson 2018]KUDO, T.; RICHARDSON, J. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *CoRR*, abs/1808.06226, 2018.

[Laki e Yang 2022]LAKI, L. J.; YANG, Z. G. Neural machine translation for hungarian. *Acta Linguistica Academica*, v. 69, n. 4, p. 501–520, 2022.

[Lee, Mansimov e Cho 2018]LEE, J.; MANSIMOV, E.; CHO, K. Deterministic non-autoregressive neural sequence modeling by iterative refinement. In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Brussels, Belgium: Association for Computational Linguistics, 2018. p. 1173–1182. Disponível em: <https://aclanthology.org/D18-1149>.

[Levenshtein 1966]LEVENSHTEIN, V. Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady*, v. 10, p. 707, 1966.

[Lewis et al. 2019]LEWIS, M. et al. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.

[Lewis 2014]LEWIS, M. P. Ethnologue: Languages of the world. *(No Title)*, 2014.

[Li et al. 2019]LI, Q. et al. Implementing neural machine translation with bi-directional gru and attention mechanism on fpgas using hls. In: IEEE. *Proceedings of the 24th Asia and South Pacific Design Automation Conference*. [S.l.], 2019. p. 693–698.

[Li e Liang 2021]LI, X. L.; LIANG, P. Prefix-tuning: Optimizing continuous prompts for generation. In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Online: Association for Computational Linguistics, 2021. p. 4582–4597. Disponível em: <https://aclanthology.org/2021.acl-long.353>.

[Lima, Araújo e Oliveira 2015]LIMA, M. A.; ARAúJO, T. M. d.; OLIVEIRA, E. S. d. Incorporation of syntactic-semantic aspects in a libras machine translation service to multimedia platforms. In: *Proceedings of the 21st Brazilian Symposium on Multimedia and the Web*. New York, NY, USA: ACM, 2015. (WebMedia '15), p. 133–140. ISBN 978-1-4503-3959-9. Disponível em: <http://doi.acm.org/10.1145/2820426.2820434>.

[Lima et al. 2015]LIMA, M. A. C. B. et al. Tradução automática com adequação sintático-semântica para libras. Universidade Federal da Paraíba, 2015.

[Lin et al. 2020]LIN, Z. et al. Pre-training multilingual neural machine translation by leveraging alignment information. *arXiv preprint arXiv:2010.03142*, 2020.

[Liu et al. 2020]LIU, X. et al. Very deep transformers for neural machine translation. *arXiv preprint arXiv:2008.07772*, 2020.

[Liu et al. 2020]LIU, Y. et al. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, v. 8, p. 726–742, 2020.

[Liu, Winata e Fung 2021]LIU, Z.; WINATA, G. I.; FUNG, P. . Continual mixed-language pre-training for extremely low-resource neural machine translation. *arXiv preprint arXiv:2105.03953*, 2021.

[Lohrenz, Möller B. e Fingscheidt 2022]LOHRENZ, T.; MÖLLER B., L. Z.; FINGSCHEIDT, T. . Relaxed attention for transformer models. *arXiv preprint arXiv:2209.09735*, 2022.

[Luong, Pham e Manning 2015]LUONG, M.-T.; PHAM, H.; MANNING, C. D. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

[Ma et al. 2022]MA, X. et al. Mega: moving average equipped gated attention. *arXiv preprint arXiv:2209.10655*, 2022.

[Macháček e Bojar 2014]MACHÁČEK, M.; BOJAR, O. Results of the wmt14 metrics shared task. In: *Proceedings of the Ninth Workshop on Statistical Machine Translation*. [S.l.: s.n.], 2014. p. 293–301.

[Magueresse, Carles e Heetderks 2020]MAGUERESSE, A.; CARLES, V.; HEETDERKS, E. Low-resource languages: A review of past work and future challenges. *arXiv preprint arXiv:2006.07264*, 2020.

[Makwisai et al. 2023]MAKWISAI, J. et al. Fine-tuning transformer-based mt using syntactic guides. In: IEEE. *2023 18th International Joint Symposium on Artificial Intelligence and Natural Language Processing (iSAI-NLP)*. [S.l.], 2023. p. 1–6.

[Martino e Christinele 2024]MARTINO, J. M. D.; CHRISTINELE, D. S. Exploring pretrained transformers for translating portuguese text to brazilian sign language. In: *Proceedings of the 16th International Conference on Computational Processing of Portuguese*. [S.l.: s.n.], 2024. p. 67–75.

[McCleary e Viotti 2007]MCCLEARY, L.; VIOTTI, E. Transcrição de dados de uma língua sinalizada: um estudo piloto da transcrição de narrativas na língua de sinais brasileira (lsb). *Bilinguismo e surdez. Questões linguísticas e educacionais. Goiânia: Cânone Editorial*, p. 73–96, 2007.

[Mehta et al. 2020]MEHTA, S. et al. Delight: Deep and light-weight transformer. *arXiv preprint arXiv:2008.00623*, 2020.

[Mishra, Bhattacharyya e Carl 2013]MISHRA, A.; BHATTACHARYYA, P.; CARL, M. Automatically predicting sentence translation difficulty. In: *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. [S.l.: s.n.], 2013. v. 2, p. 346–351.

[Modh e Saini 2018]MODH, J. C.; SAINI, J. R. A study of machine translation approaches for gujarati language. *International Journal of Advanced Research in Computer Science*, International Journal of Advanced Research in Computer Science, v. 9, n. 1, 2018.

[Mohamed A. & Hefny 2022]MOHAMED A. & HEFNY, H. A deep learning approach for gloss sign language translation using transformer. *Journal of Computing and Communication*, v. 1, n. 2, p. 1–8, 2022.

[Munappy et al. 2019]MUNAPPY, A. et al. Data management challenges for deep learning. In: *2019 45th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. [S.l.: s.n.], 2019. p. 140–147.

[Nadkarni, Ohno-Machado e Chapman 2011]NADKARNI, P. M.; OHNO-MACHADO, L.; CHAPMAN, W. W. Natural language processing: An introduction. *Journal of the American Medical Informatics Association*, v. 18, n. 5, p. 544–551, 2011.

[Nagoudi et al. 2021]NAGOUDI, E. M. B. et al. Indt5: A text-to-text transformer for 10 indigenous languages. *arXiv preprint arXiv:2104.07483*, 2021.

[Nagoudi, Elmadany e Abdul-Mageed 2021]NAGOUDI, E. M. B.; ELMADANY, A.; ABDUL-MAGEED, M. Arat5: Text-to-text transformers for arabic language generation. *arXiv preprint arXiv:2109.12068*, 2021.

[Ngo e Trinh 2021]NGO, C.; TRINH, T. H. *Styled Augmented Translation (SAT)*. 2021. `https://github.com/vietnlp/SAT`.

[Ngo et al. 2022]NGO, C. et al. MTet: Multi-domain translation for english and vietnamese. *arXiv preprint arXiv:2210.05610*, 2022.

[Nguyen e Salazar 2019]NGUYEN, T. Q.; SALAZAR, J. Transformers without tears: Improving the normalization of self-attention. *arXiv preprint arXiv:1910.05895*, 2019.

[Okpor 2014]OKPOR, M. Machine translation approaches: issues and challenges. *International Journal of Computer Science Issues (IJCSI)*, International Journal of Computer Science Issues (IJCSI), v. 11, n. 5, p. 159, 2014.

[Oliveira et al. 2019]OLIVEIRA, C. C. M. d. et al. Analysis of rule-based machine translation and neural machine translation approaches for translating portuguese to libras. In: *Simpósio Brasileiro de Sistemas Multimídia e Web (WebMedia), 25*. Rio de Janeiro: [s.n.], 2019. p. 117–124.

[Ortega, Mamani e Cho 2020]ORTEGA, J.; MAMANI, R. C.; CHO, K. Neural machine translation with a polysynthetic low resource language. *Machine Translation*, v. 34, 12 2020.

[Ott et al. 2019]OTT, M. et al. fairseq: A fast, extensible toolkit for sequence modeling. In: *Proceedings of NAACL-HLT 2019: Demonstrations*. [S.l.: s.n.], 2019.

[Papineni et al. 2002]PAPINENI, K. et al. Bleu: a method for automatic evaluation of machine translation. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the 40th annual meeting on association for computational linguistics*. [S.l.], 2002. p. 311–318.

[Pham et al. 2014]PHAM, V. et al. Dropout improves recurrent neural networks for handwriting recognition. In: IEEE. *2014 14th International Conference on Frontiers in Handwriting Recognition*. [S.l.], 2014. p. 285–290.

[Provilkov, Emelianenko e Voita 2019]PROVILKOV, I.; EMELIANENKO, D.; VOITA, E. Bpe-dropout: Simple and effective subword regularization. *arXiv preprint arXiv:1910.13267*, 2019.

[Purohit, Yogi e Sharma 2022]PUROHIT, A.; YOGI, K. K.; SHARMA, R. A comparison of machine translation methods for natural language processing and their challenges. In: SPRINGER. *International Conference on Innovations in Computational Intelligence and Computer Vision*. [S.l.], 2022. p. 475–487.

[Quadros 2006]QUADROS, R. M. *Efeitos de modalidade de língua: as línguas de sinais*. [S.l.]: ETD, 2006.

[Quadros e Karnopp 2004]QUADROS, R. M.; KARNOPP, L. B. *Língua de sinais brasileira: estudos linguísticos*. Porto Alegre:RS: Artmed, 2004.

[Quadros, Pizzio e Rezende]QUADROS, R. M. d.; PIZZIO, A. L.; REZENDE, P. L. F. Língua brasileira de sinais i.

[Raffel et al. 2019]RAFFEL, C. et al. Exploring the limits of transfer learning with a unified text-to-text transformer. *CoRR*, abs/1910.10683, 2019.

[Raffel et al. 2020]RAFFEL, C. et al. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, v. 21, n. 140, p. 1–67, 2020.

[Rehm et al. 2018]REHM, G. et al. *Language technologies for a multilingual Europe*. [S.l.: s.n.], 2018.

[Ribeiro et al. 2020]RIBEIRO, A. H. et al. Beyond exploding and vanishing gradients: analysing rnn training using attractors and smoothness. In: PMLR. *International conference on artificial intelligence and statistics*. [S.l.], 2020. p. 2370–2380.

[Russel e Norvig 2004]RUSSEL, S. J.; NORVIG, P. *Inteligência Artificial*. Rio de Janeiro:RJ: Elsevier, 2004.

[Santos 2021]SANTOS, C. H. T. Redes neurais aplicadas à modelagem de instrumentos acústicos para síntese sonora em tempo real. *Master's thesis*, 2021.

[Santos 2019]SANTOS, R. S. D. *Portuguese-Chinese neural machine translation*. Tese (Doutorado) — University of Macau, 2019.

[Saunders, Camgoz e Bowden 2020]SAUNDERS, B.; CAMGOZ, N. C.; BOWDEN, R. Progressive transformers for end-to-end sign language production. In: SPRINGER. *European Conference on Computer Vision*. [S.l.], 2020. p. 687–705.

[Sennrich e Haddow 2016]SENNRICH, R.; HADDOW, B. Linguistic input features improve neural machine translation. *arXiv preprint arXiv:1606.02892*, 2016.

[Sennrich, Haddow e Birch 2015]SENNRICH, R.; HADDOW, B.; BIRCH, A. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*, 2015.

[Sennrich, Haddow e Birch 2016]SENNRICH, R.; HADDOW, B.; BIRCH, A. Edinburgh neural machine translation systems for wmt 16. *arXiv preprint arXiv:1606.02891*, 2016.

[Shazeer et al. 2017]SHAZEER, N. et al. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

[Souza et al. 2017]SOUZA, M. F. N. S. et al. Principais dificuldades e obstáculos enfrentados pela comunidade surda no acesso à saúde: uma revisão integrativa de literatura. *Revista Cefac*, SciELO Brasil, v. 19, p. 395–405, 2017.

[Stein 2018]STEIN, D. Machine translation: Past, present and future. *Language technologies for a multilingual Europe*, Language Science Press, v. 4, p. 5, 2018.

[Stoll et al. 2018]STOLL, S. et al. Sign language production using neural machine translation and generative adversarial networks. In: *BMVC*. [S.l.: s.n.], 2018. p. 304.

[Sutskever, Vinyals e Le 2014]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.

[Svozil, Kvasnička e Pospichal 1997]SVOZIL, D.; KVASNIČKA, V.; POSPICHAL, J. Introduction to multi-layer feed-forward neural networks. *Chemometrics and intelligent laboratory systems*, v. 39, n. 1, p. 43–62, 1997.

[Takase e Kiyono 2021]TAKASE, S.; KIYONO, S. Rethinking perturbations in encoder-decoders for fast training. *arXiv preprint arXiv:2104.01853*, 2021.

[Valli e Lucas 2000]VALLI, C.; LUCAS, C. *Linguistics of American sign language: an introduction*. [S.l.]: Gallaudet University Press, 2000.

[Vaswani et al. 2017]VASWANI, A. et al. Attention is all you need. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2017. p. 5998–6008.

[Vauquois 1968]VAUQUOIS, B. A survey of formal grammars and algorithms for recognition and transformation in mechanical translation. In: *Ifip congress (2)*. [S.l.: s.n.], 1968. v. 68, p. 1114–1122.

[Vu e Bui 2023]VU, H.; BUI, N. D. On the scalability of data augmentation techniques for low-resource machine translation between chinese and vietnamese. *Journal of Information and Telecommunication*, Taylor & Francis, v. 7, n. 2, p. 241–253, 2023.

[Wang et al. 2021]WANG, H. et al. Improving the transformer translation model with back-translation. In: SUN, X. et al. (Ed.). *Advances in Artificial Intelligence and Security*. Cham: Springer International Publishing, 2021. p. 286–294. ISBN 978-3-030-78615-1.

[Wang et al. 2022]WANG, H. et al. Progress in machine translation. *Engineering*, Elsevier, v. 18, p. 143–153, 2022.

[Wang et al. 2019]WANG, L. et al. Denoising based sequence-to-sequence pre-training for text generation. *CoRR*, abs/1908.08206, 2019.

[Wang et al. 2019]WANG, Y. et al. Multi-agent dual learning. In: *Proceedings of the International Conference on Learning Representations (ICLR)*. [S.l.: s.n.], 2019.

[Weaver 1949]WEAVER, W. The mathematics of communication. *Scientific American*, JSTOR, v. 181, n. 1, p. 11–15, 1949.

[Wei et al. 2021]WEI, J. et al. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*, 2021.

[Westin 2019]WESTIN, R. *Baixo alcance da língua de sinais leva surdos ao isolamento*. Agência Senado, 2019. Disponível em: <https://www12.senado.leg.br/noticias/especiais/especial-cidadania/baixo-alcance-da-lingua-de-sinais-leva-surdos-ao-isolamento>.

[Wu et al. 2019]WU, F. et al. Pay less attention with lightweight and dynamic convolutions. *CoRR*, abs/1901.10430, 2019.

[Wu et al. 2021]WU, L. et al. R-drop: Regularized dropout for neural networks. In: *Advances in Neural Information Processing Systems*. [S.l.: s.n.], 2021. v. 34, p. 10890–10905.

[Wu et al. 2016]WU, Y. et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

[Xu, Durme e Murray 2021]XU, H.; DURME, B. V.; MURRAY, K. Bert, mbert, or bibert? a study on contextualized embeddings for neural machine translation. *arXiv preprint arXiv:2109.04588*, 2021.

[Xu et al. 2019]XU, J. et al. Understanding and improving layer normalization. In: *Advances in Neural Information Processing Systems*. [S.l.: s.n.], 2019.

[Xue et al. 2022]XUE, L. et al. Byt5: Towards a token-free future with pre-trained byte-to-byte models. *Transactions of the Association for Computational Linguistics*, v. 10, p. 291–306, 2022.

[Xue et al. 2020]XUE, L. et al. Mt5: A massively multilingual pre-trained text-to-text transformer. *arXiv preprint arXiv:2010.11934*, 2020.

[Yin e Read 2020]YIN, K.; READ, J. Attention is all you sign: sign language translation with transformers. In: *Sign Language Recognition, Translation and Production (SLRTP) Workshop-Extended Abstracts*. [S.l.: s.n.], 2020. v. 4.

[Yin e Read 2020a]YIN, K.; READ, J. Attention is all you sign: sign language translation with transformers. In: *Sign Language Recognition, Translation and Production (SLRTP) Workshop-Extended Abstracts*. [S.l.: s.n.], 2020a. v. 4.

[Yonglan e Wenjia 2022]YONGLAN, L.; WENJIA, H. English-chinese machine translation model based on bidirectional neural network with attention mechanism. *Journal of Sensors*, 2022.

[Zhang e Duh 2021]ZHANG, X.; DUH, K. Approaching sign language gloss translation as a low-resource machine translation task. In: ASSOCIATION FOR MACHINE TRANSLATION IN THE AMERICAS. *Proceedings of the 1st International Workshop on Automatic Translation for Signed and Spoken Languages (AT4SSL)*. [S.l.], 2021. p. 60–70.