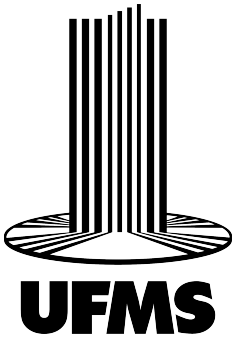


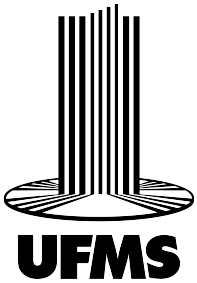
UNIVERSIDADE FEDERAL DE MATO GROSSO DO SUL
CAMPUS DE PONTA PORÃ
CURSO DE GRADUAÇÂO EM SISTEMAS DE INFORMAÇÃO

Análise e comparação de técnicas de processamento de linguagem natural aplicadas à tradução automática de texto

João José da Costa Júnior

Ponta Porã - MS

2025



UNIVERSIDADE FEDERAL DE MATO GROSSO DO SUL
CAMPUS DE PONTA PORÃ
CURSO DE GRADUAÇÂO EM SISTEMAS DE INFORMAÇÃO

Análise e comparação de técnicas de processamento de linguagem natural aplicadas à tradução automática de texto

João José da Costa Júnior

Trabalho de Conclusão de Curso apresentado como exigência para obtenção do grau de Bacharel em Sistemas de Informação da Universidade Federal de Mato Grosso do Sul – UFMS.

Orientador: Prof. Dr. Fabricio Augusto Rodrigues

Ponta Porã - MS
2025

Análise e comparação de técnicas de processamento de linguagem natural aplicadas à tradução automática de texto

Trabalho de Conclusão de Curso apresentado como exigência para obtenção do grau de Bacharel em Sistemas de Informação da Universidade Federal de Mato Grosso do Sul – UFMS.

Banca Examinadora:

Prof. Dr. Fabricio Augusto Rodrigues

Prof. Dr. Leonardo Souza Silva

Prof. Dr. Robson Soares Silva

Ponta Porã - MS

2025

Agradecimentos

Agradeço, primeiramente, a Deus, pela sabedoria, saúde e perseverança concedidas ao longo desta jornada acadêmica. Expresso minha sincera gratidão aos professores do curso, pela dedicação, competência e compromisso na transmissão do conhecimento, contribuindo de forma significativa para a minha formação profissional e pessoal.

Aos colegas de curso, agradeço pela colaboração, pelo compartilhamento de experiências e pelo espírito de companheirismo demonstrado durante todo o percurso. Dirijo, ainda, meu mais profundo agradecimento à minha família, pelo apoio incondicional, incentivo constante e compreensão nos momentos de ausência e desafios.

Por fim, estendo meus agradecimentos a todos que, direta ou indiretamente, contribuíram para a concretização deste trabalho e para o êxito desta etapa tão importante da minha trajetória acadêmica.

Resumo

Este trabalho apresenta uma análise comparativa entre duas arquiteturas proeminentes de Tradução Automática Neural (NMT): Redes Neurais Recorrentes (RNN) com unidades GRU e a arquitetura Transformer. O estudo parte da hipótese de que os Transformers, devido ao uso de mecanismos de atenção, superam as abordagens recorrentes em desempenho e capacidade de generalização. Para testar esta hipótese, foram implementados dois protótipos: um modelo RNN-GRU treinado do zero para a tradução de inglês para francês, que alcançou 93,89% de acurácia de validação, mas demonstrou limitações com vocabulário desconhecido e complexidade morfológica; e um modelo Transformer pré-treinado (mBART-50) para a mesma tarefa. Os resultados confirmaram a superioridade do Transformer, que obteve um desempenho superior em métricas como BLEU (0.8687) e METEOR (0.9329), gerando traduções fluentes e precisas. A análise conclui que a arquitetura Transformer, amplamente reconhecida na literatura como o estado da arte em tradução automática (VASWANI et al., 2017), oferecendo uma solução mais robusta e eficiente para capturar as nuances sintáticas e semânticas dos idiomas.

Palavras-chaves: Processamento de Linguagem Natural. Tradução Automática. Redes Neurais Recorrentes. Transformers. Aprendizado de Máquina.

Abstract

This work presents a comparative analysis between two prominent Neural Machine Translation (NMT) architectures: Recurrent Neural Networks (RNN) with GRU units and the Transformer architecture. The study starts from the hypothesis that Transformers, due to their use of attention mechanisms, outperform recurrent approaches in terms of performance and generalization capacity. To test this hypothesis, two prototypes were implemented: an RNN-GRU model trained from scratch for English-to-French translation, which achieved 93.89% validation accuracy but showed limitations with unknown vocabulary and morphological complexity; and a pre-trained Transformer model (mBART-50) for the same task. The results confirmed the superiority of the Transformer, which achieved higher performance in metrics such as BLEU (0.8687) and METEOR (0.9329), generating fluent and accurate translations. The analysis concludes that the Transformer architecture—widely recognized in the literature as the state of the art in machine translation (VASWANI et al., 2017)—offers a more robust and efficient solution for capturing the syntactic and semantic nuances of languages.

Keywords: Natural Language Processing. Machine Translation. Recurrent Neural Networks. Transformers. Machine Learning.

Lista de ilustrações

Lista de tabelas

Sumário

1 Introdução

1.1 Contextualização

O Processamento de Linguagem Natural (PLN) estabeleceu-se como um campo multidisciplinar de grande relevância na ciência da computação, buscando capacitar as máquinas a compreender e processar a linguagem humana. Dentre suas diversas aplicações, a Tradução Automática (TA) destaca-se como uma das tarefas mais desafiadoras e de maior impacto, evoluindo de abordagens baseadas em regras e estatística para modelos neurais complexos. A transição para a Tradução Automática Neural (NMT) marcou um avanço significativo, com arquiteturas como as Redes Neurais Recorrentes (RNNs) e, mais recentemente, os Transformers, redefinindo os padrões de qualidade e fluência.

A motivação para este trabalho surge da necessidade de compreender as vantagens e desvantagens práticas dessas duas arquiteturas dominantes no cenário atual da NMT. Enquanto as RNNs foram fundamentais para o avanço da área, sua natureza sequencial apresenta limitações inerentes, como a dificuldade em capturar dependências de longo alcance. Em contrapartida, a arquitetura Transformer, com seu mecanismo de atenção, revolucionou o campo ao permitir o processamento paralelo e uma compreensão contextual das sentenças mais robusta. Diante disso, uma análise comparativa direta entre um modelo RNN treinado do zero e um modelo Transformer pré-treinado torna-se essencial para avaliar empiricamente suas capacidades e limitações em uma tarefa de tradução.

1.2 Objetivos

1.2.1 Objetivo Geral

O objetivo geral deste trabalho é analisar e comparar o desempenho de duas arquiteturas de Tradução Automática Neural — uma baseada em Redes Neurais Recorrentes (GRU) e outra baseada em Transformers — na tarefa de tradução de texto, a fim de validar a superioridade da arquitetura Transformer.

1.2.2 Objetivos Específicos

Para alcançar o objetivo geral, foram definidos os seguintes objetivos específicos:

- Desenvolver e treinar um modelo de tradução automática do tipo sequence-to-sequence utilizando uma Rede Neural Recorrente com unidades GRU.

- Implementar uma solução de tradução utilizando um modelo Transformer pré-treinado (mBART-50) para realizar a mesma tarefa.
- Avaliar e comparar a qualidade das traduções geradas por ambas as abordagens por meio de métricas automáticas consolidadas, como BLEU, METEOR, TER e chrF.
- Analisar criticamente as limitações e os pontos fortes de cada arquitetura com base nos resultados quantitativos e qualitativos obtidos, contextualizando-os com a fundamentação teórica.

1.3 Hipótese

Diante dos avanços recentes em modelos de tradução automática e também dos desafios de analisar idiomas com estrutura gramatical complexa, como o português, parte-se da hipótese de que o modelo Transformer supera abordagens tradicionais baseadas em RNNs e técnicas estatísticas, tanto em termos de desempenho quanto em capacidade de generalização, pois utiliza mecanismos de atenção que são mais indicados para captar nuances semânticas e sintáticas do idioma.

1.4 Organização do Trabalho

Este trabalho está estruturado em quatro capítulos principais, além da conclusão e referências.

- **Capítulo 1:** Apresenta a introdução, contextualizando o tema da Tradução Automática Neural, a justificativa para a pesquisa, os objetivos gerais e específicos, a hipótese que norteia o estudo e a organização do documento.
- **Capítulo 2:** Descreve a Fundamentação Teórica, abordando os conceitos essenciais de Processamento de Linguagem Natural, as diferentes abordagens de Tradução Automática, os princípios de Redes Neurais Artificiais e, em detalhe, a arquitetura Transformer e seu mecanismo de atenção.
- **Capítulo 3:** Detalha a Implementação dos dois protótipos de tradução. Descreve as etapas de pré-processamento dos dados, a arquitetura do modelo RNN com GRU, o processo de treinamento e a configuração do modelo Transformer pré-treinado (mBART).
- **Capítulo 4:** Apresenta os Resultados da análise comparativa. Expõe as traduções geradas pelo modelo Transformer, as pontuações obtidas nas métricas de avaliação e discute as limitações da arquitetura RNN que impediram uma comparação direta em vocabulário aberto.

- **Conclusão:** Sintetiza os resultados obtidos, retoma a hipótese de pesquisa, confirma as conclusões do estudo sobre a superioridade da arquitetura Transformer e propõe direções para trabalhos futuros.

2 Fundamentação Teórica

Neste capítulo serão abordados alguns conceitos e fundamentos necessários para a compreensão das técnicas e algoritmos estudados neste trabalho e aplicados ao problema de tradução automática.

2.1 Processamento de Linguagem Natural

Processamento de Linguagem Natural (PLN), também conhecido como Linguística Computacional (SANTOS, 2001), é um campo da Ciência da Computação e da Linguística, que busca, como um dos principais objetivos, resolver problemas computacionais que requerem o processamento de determinada língua humana (português, inglês, espanhol, etc.), seja escrita ou falada.

Na verdade, existem divergências sobre em qual área, especificamente, o PLN se enquadra. Para (BOLSHAKOV; GELBUKH, 2004), PLN é uma área "mais linguística que computacional" e, dessa forma, uma subárea da Linguística Aplicada. Já para (NUGUES, 2006), é uma interseção entre a Linguística e a Ciência da Computação. Mas, como bem salienta (SILVA, 1996), os pesquisadores de PLN, tanto linguistas quanto cientistas da computação, buscam fundamentos em diversas outras áreas: Filosofia da Linguagem, Psicologia, Lógica, Inteligência Artificial (IA), Matemática, Ciência da Computação, Linguística Computacional (LC) e Linguística. Portanto, podemos considerar o PLN uma área multidisciplinar.

Para o computador conseguir "entender" a linguagem humana, são desenvolvidas diversas metodologias de pré-processamento da linguagem — ou seja, técnicas e algoritmos computacionais que transformam o texto (escrito ou falado) em um formato que a máquina possa interpretar e aprender. Essas metodologias de pré-processamento têm como objetivo adaptar os dados brutos da linguagem natural para uma forma estruturada, facilitando o aprendizado das características do idioma pelos modelos computacionais.

O PLN está presente em diversos sistemas e aplicações, com diferentes graus de desempenho, que pesquisadores desenvolveram ao longo do tempo. Alguns exemplos são:

- Sistemas de recuperação de informação, que buscam ou encontram textos (ou parte de textos) relevantes a uma dada "consulta" em uma coleção de textos ou documentos (TZOUKERMAN; KLAVANS; STRZALKOWSKI, 2004);
- Sistemas de extração de informação, que buscam encontrar certa informação, uma resposta, a dada pergunta de entrada em um ou mais documentos (GRISHMAN,

2004);

- Sistemas de tradução automática, que partem de um texto-fonte, escrito em uma linguagem natural X, e produzem um texto-alvo, uma versão do texto-fonte em uma linguagem natural Y (SLOCUM, 1985; NIRENBURG, 1989; HUTCHINS, 2004; SOMERS, 2004);

- Sistemas de sumarização automática (SA): esses sistemas caracterizam-se por gerar resumos de um ou mais textos de acordo, por exemplo, com uma determinada taxa de compressão (HOVY, 2004);

- Sistemas de correção ortográfica: processam um texto em uma língua natural para sugerir alternativas prováveis e ortograficamente corretas a cada erro identificado;

- Sistemas de diálogos: são os sistemas de interpretação de diálogos e os sistemas que participam de um diálogo. Um exemplo são os *chatbots*, sistemas cada vez mais populares atualmente.

2.1.1 Linguística e Computação

Uma das principais características que faz com que o processamento de linguagem natural seja um desafio para os sistemas computacionais, e também é uma propriedade das linguagens humanas, é que as linguagens naturais são ambíguas. Ao contrário de linguagens formais, como C, Java ou PROLOG, que são construídas para que não haja ambiguidade (COPPIN, 2010), porque se um programa tivesse mais de uma interpretação, o computador ia ter que decidir qual delas escolher.

Dependendo de qual linguagem um sistema de PLN será projetado para lidar, ele não poderá trabalhar com outra linguagem. De acordo com (COPPIN, 2010) a maioria das pesquisa nessa área são realizadas em inglês, mas existem muitas línguas que são completamente diferentes do inglês, como o Chinês e o Navajo.

Quando um sistema computacional precisa processar alguma linguagem natural, ele deve lidar e manipular essa linguagem em diversos níveis diferentes. Como:

- Fonologia - é o campo que analisa os sons que formam as palavras, sendo utilizada para reconhecer palavras por meio desses sons. Só é necessário se o sistema computacional precisar trabalhar com linguagem falada.

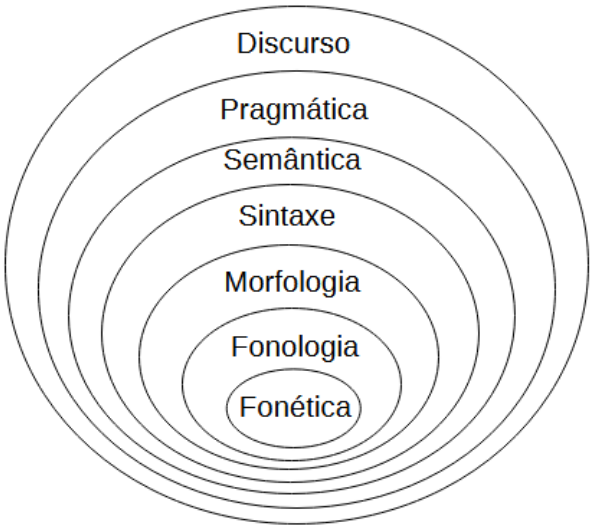
- Morfologia - é a primeira análise que é aplicada às palavras. Estuda como os morfemas (componentes das palavras) se organizam para formar palavras.

- Sintaxe - estudo de como as palavras se organizam em estruturas para formar sentenças e orações.

- Semântica - estuda o significado de palavras e frases. É possível que uma frase esteja correta, ponto de vista sintático, e semanticamente sem sentido.
- Pragmática - estuda frases para obter significados que não estão claros a partir da semântica.

A figura 1 mostra esses diferentes níveis, que são subáreas que estudam a linguística.

Figura 1 – Representação das subáreas de estudo da linguagem.



Fonte: o autor. Adaptado de (CASELI; NUNES, 2024)

Como é possível observar, há uma interdependência entre as subáreas, de forma que o acesso aos níveis mais internos pressupõe o domínio prévio dos níveis mais externos. A fonética também estuda os sons, assim como a fonologia, e o discurso tem um foco no texto como um todo, analisando como as frases se relacionam entre si.

A seguir as análises que estão envolvidas no processamento de linguagem natural serão apresentadas com um pouco mais de detalhes. Serão observadas as análises que envolvem o processamento de texto, assim, ficando de fora as análises: fonética e fonologia.

2.1.1.1 Análise Morfológica

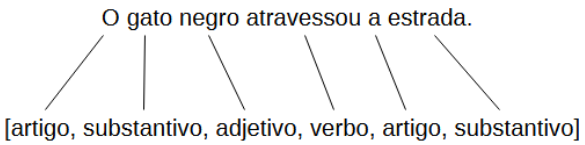
Como dito anteriormente, a morfologia trata do conhecimento sobre a estrutura das palavras. Algumas palavras, como *árvore*, não podem ser quebradas em unidades menores, mas outras palavras podem, como *árvores* ou *arvorezinhas*. Ou também *impossível* e *sobremesa*. De acordo com (VIEIRA; LIMA, 2001): "As unidades constituintes das palavras são denominadas morfemas, e tais constituintes podem ser independentes, como em *árvore* ou dependentes como no caso dos sufixos (*s* em *árvores*) e prefixos (*im* em *impossível*)".

Além da estrutura das palavras, em morfologia também se estuda como as palavras são classificadas em diferentes categorias, ou como essa área é conhecida, as palavras são classificadas em partes do discurso (do inglês, Part-of-Speech, ou POS). Alguns exemplos de categorias são: substantivos, verbos, adjetivos, preposições e advérbios.

Reconhecer essas categorias das palavras é um problema básico em linguística computacional. Algumas aplicações são desenvolvidas, inicialmente, com base nessa informação. Por exemplo, para fazer a análise da estrutura das frases, é necessário que primeiro se faça o reconhecimento das categorias (VIEIRA; LIMA, 2001). Os sistemas que fazem essa tarefa são chamados de etiquetadores de categorias gramaticais (*POS taggers*, em inglês). Esses *POS taggers* recebem um texto, e esse texto é devolvido acrescentado, a cada palavra, uma etiqueta com informação a respeito de sua categoria gramatical.

A figura 2 mostra o exemplo de uma frase que um *POS tagger* recebeu (no caso "O gato negro atravessou a estrada.") e como foi relacionada a respectiva categoria para cada palavra.

Figura 2 – Exemplo de *POS tagger*.



Fonte: o autor.

2.1.1.2 Análise Sintática

Depois de reconhecer as categorias das palavras, a próxima etapa da análise da linguagem natural é verificar se a estrutura das frases faz sentido, além de reconhecer a partes que compõem a frase. De acordo com (VIEIRA; LIMA, 2001), esse conhecimento, referente à organização das palavras de uma frase em uma determinada ordem, pode ser caracterizado por uma gramática, que consiste em um conjunto finito de regras e princípios capazes de descrever a forma como as palavras se combinam para formar sentenças gramaticalmente corretas em um determinado idioma.

Uma questão importante relacionada à estrutura da frase é a interpretação. Como existem diferentes combinações possíveis entre palavras de uma mesma frase, essa frase pode ter diferentes interpretações. Esse fenômeno é conhecido como ambiguidade. Um exemplo de ambiguidade é a frase "O homem viu o menino com o telescópio". As interpretações possíveis (o menino estava com o telescópio ou o homem utilizou o telescópio para ver) não são por causa das palavras, mas pela presença de ambiguidade na estrutura.

Como é possível observar, para fazer a análise sintática é preciso das informações morfológicas, e a análise sintática acarretará em consequências para a interpretação de

uma frase (consequências no campo da semântica).

Os sistemas que fazem a análise da estrutura das frases e suas palavras são chamados de analisadores sintáticos (conhecidos como *parsers*, do inglês). Esses sistemas sabem que a estrutura é válida com base no conjunto de palavras de um idioma e nas regras gramaticais que definem como esse idioma funciona.

2.1.1.3 Análise Semântica

A semântica é o ramo que estuda a atribuição de significado às expressões, tanto em linguagens naturais quanto formais, como a lógica e a programação. A semântica pode ser dividida em dois tipos: semântica lexical, que é centralizada no significado das palavras, e semântica lógica, que está ligada ao valor verdade de uma proposição.

Na semântica lexical, são consideradas as propriedades de cada uma das palavras de determinada língua. Um dos problemas que devem ser considerados é o fato de que algumas palavras apresentam vários sentidos. Ambiguidade lexical é quando uma palavra possui mais de um sentido. Por exemplo, a palavra "banco" pode se referir à instituição financeira ou também ao objeto feito para as pessoas sentarem (VIEIRA; LIMA, 2001).

Segundo (VIEIRA; LIMA, 2001): "a semântica lógica trata o significado através de uma especificação do domínio de conhecimento, de acordo com a teoria dos conjuntos. Para expressar o significado de expressões da linguagem natural em lógica, é preciso traduzir as expressões para uma linguagem lógica". Mas para isso, é preciso recorrer às lógicas não clássicas, que incorporam noções mais complexas, como, por exemplo, o tempo.

Como é possível perceber, a área de semântica é uma área de estudo mais complexa do que a sintaxe, pois apresenta questões que são difíceis de tratar de maneira completa e exata. O significado está mais ligado ao conhecimento de mundo. A área de estudo que integra outros fatores, como contexto e falantes, é a pragmática.

2.1.1.4 Análise Pragmática

(VIEIRA; LIMA, 2001) dão uma ideia de como funciona a análise pragmática:

> A pergunta "Sobrou um pouco de café?", por exemplo, pode ser interpretada pelo ouvinte como uma solicitação do emissor para receber uma xícara de café tendo, assim, um significado de sentença diferenciado do significado de enunciação. Situações como estas ilustram a diferença entre o significado literal da linguagem e o significado da linguagem em uso, que é o objeto de estudo da pragmática.

Na ciência da computação, em específico a inteligência artificial distribuída, o interesse é nos mecanismos interativos para modelar agentes e sociedade de agentes. Para (AUSTIN, 1962) e (SEARLE, 1969), autores da teoria dos atos de fala, que fundamenta

o estudo de comunicação entre agentes inteligentes, "diferentes tipos de enunciados têm diferentes efeitos nos estados perceptivos dos agentes e nos estados do mundo representados". De acordo com a teoria, esses enunciados fazem diferentes tipos de ação, conforme a classificação a seguir:

- Representativos: o falante comunica que acredita na verdade da expressão (por exemplo, através de asserção ou conclusão).
- Diretivos: o falante tem por intenção provocar o ouvinte a realizar uma ação (por exemplo, requisição, pergunta, ordem, proibição, permissão).
- Comissivos: o falante se compromete com a realização de uma ação no futuro (por exemplo, promessa, ameaça).
- Expressivos: o falante expressa um estado psicológico (por exemplo, agradecimento, pedido de desculpas).
- Declarações: têm como efeito imediato uma mudança de estado (por exemplo, uma declaração de guerra, a confirmação do batismo).

Essas classificações são usadas como base para a construção de protocolos de comunicação entre os agentes.

2.1.2 Paradigmas de PLN

Com o avanço da computação, a inteligência artificial (IA) sofreu mudanças em seus métodos de abordagem, o que também fez com que o processamento de linguagem natural seguisse esses modelos, conhecidos como paradigmas. Esses paradigmas do PLN, que acompanharam a evolução da IA, também seguiam a evolução dos computadores: quanto mais um computador ganhava memória e processamento, mais algoritmos eram propostos, aproveitando dessa melhoria computacional. Esses algoritmos foram enquadrados em três paradigmas principais: Paradigma Simbólico, Paradigma Estatístico e Paradigma Neural.

- Paradigma Simbólico - o PLN se baseava nesse paradigma até a década de 1980. Aqui todo o conhecimento que uma língua possui é expresso em formalismo como regras, dicionários, etc., formas que são entendíveis ao ser humano (CASELI; NUNES, 2024). Exemplo: é possível estabelecer regras que indiquem que há concordância entre o gênero de um substantivo e um adjetivo. Assim, "morango maduro" é considerada um frase correta enquanto "morango madura" é considerada errada.
- Paradigma Estatístico - dentro do PLN, a tradução automática foi a aplicação que mais deu popularidade a esse paradigma, que teve início nos anos 1990 e foi o

mais utilizado até a década de 2010. Aqui o *corpus*, que são grandes conjuntos de textos, eram utilizados como fonte para "ensinar" os computadores. Exemplo: a concordância entre substantivo e adjetivo, mencionada no paradigma simbólico, passou a ser aprendida a partir de exemplos que estão presentes no *corpus* como: "mamão estragado", "pêssego maduro". Assim, a língua é representada em modelos probabilísticos adquiridos através da frequência de ocorrências.

- Paradigma Neural - utiliza redes neurais profundas (estruturas de dados complexas que processam grande quantidade de dados) e é o paradigma mais adotado, atualmente, para atividades de PLN. Igual ao paradigma estatístico, as redes neurais também utilizam grandes conjuntos de dados para aprender, mas com a diferença que, aqui, envolve várias camadas de processamento para o aprendizado computacional. Por serem mais complexas, nas redes neurais o conhecimento é dado por valores numéricos, não por símbolos ou regras, fazendo com que não seja possível saber como o modelo foi aprendido e, assim, impossibilitando o entendimento humano do sistema.

Mas, como uma única abordagem não é suficiente, cada vez mais é utilizado o Paradigma Híbrido, que combina, principalmente o paradigma simbólico com algum dos outros dois, fazendo com que assim haja um entendimento dos passos seguidos pelos algoritmos.

Além de ser utilizado na inteligência artificial, o processamento de linguagem natural tem sido aplicado em diversos campos de pesquisa e em áreas como mineração de textos, recuperação de informação e ciência de dados. Atualmente, em praticamente todos os sistemas computacionais que processam texto de alguma forma, é possível aplicar, em maior ou menor grau, os paradigmas do PLN.

2.1.3 Principais Desafios do PLN

Os sistemas computacionais que utilizam o processamento de linguagem natural estão aumentando com o passar do tempo. Com esse aumento na utilização do PLN, também começam a aparecer ou a ficar evidente alguns problemas ou desafios que estão ligados a essa área, já que a linguagem é uma estrutura complexa e também possui diferentes variações dependendo da língua. Alguns dos principais problemas são: a linguagem não padrão, as expressões idiomáticas e o conhecimento sobre o mundo (ANDRADE; BARROS; SANTOS, 2023).

- A linguagem não padrão se refere à utilização de símbolos, erros gramaticais e, também, abreviações que dificultam a interpretação semântica da frase, por parte do computador. Exemplos: "pq" (porque), "vc" (você), "blz" (beleza), etc.

- Já as expressões idiomáticas dizem respeito às variações que a língua sofre dependendo da região ou de seu contexto cultural. Exemplo: falar que alguém "pisou na bola", quando fez algo que não devia, pode ser um desafio de semântica para o computador.

- O conhecimento sobre o mundo ocorre quando uma palavra possui duplo sentido (ambiguidade) e só é possível saber o verdadeiro sentido quando o contexto é analisado. Exemplo: a palavra "manga", dependendo do contexto, pode se referir à parte da camisa ou à fruta.

2.1.4 Principais Técnicas de PLN

Como dito anteriormente, cada vez mais os sistemas e aplicações computacionais estão incorporando em si o processamento de linguagem natural. Mas mesmo que o uso de linguagem natural faça com que o sistema seja mais acessível, é uma tarefa complexa para os computadores dominarem (DESHPANDE; DEVALE, 2012).

No presente momento, existe uma grande quantidade de algoritmos e técnicas de PLN, mas segundo (SANYAL; SHUKLA; AGRAWAL, 2021), o PLN depende muito de *Machine Learning* ou *Deep Learning* (subáreas da Inteligência Artificial) para ter sucesso na implementação da conexão entre humanos e máquinas. Algumas das técnicas/metodologias utilizadas nessa área são indicadas por esse mesmo autor, (SANYAL; SHUKLA; AGRAWAL, 2021), estão listadas a seguir.

- Reconhecimento de Entidade Nomeada (do inglês *Name Entity Recognition* ou NER): Mais utilizada em análise semântica. Identifica e classifica as expressões linguísticas, chamadas de Entidades Nomeadas (EN), que podem referenciar nomes próprios, expressões temporais e espécies biológicas (MOTA; SANTOS; RANCHHOD, 2007; NADEAU, 2007). Ou seja, aqui, os algoritmos recebem textos ou frases, como entrada, e identificam os nomes presentes neles.

  Algumas categorias que são mais utilizadas incluem as entidades que referenciam Pessoas Singulares (antropônimos), Coletivas (organizações e empresas) e Lugares (topônimos) (MOTA; SANTOS; RANCHHOD, 2007).

  Por exemplo, na frase "Renata Pereira e Maria Lima palestraram na Universidade Federal de Mato Grosso do Sul", temos as seguintes entidades nomeadas: "Renata Pereira" e "Maria Lima", que correspondem à categoria de Pessoa e "Universidade Federal de Mato Grosso do Sul" que corresponde à categoria de Organização.

  Também existem outras categorias de entidades nomeadas, como Obras ("Código Da Vinci", por exemplo), Acontecimentos ("Festa de Santo Antônio", por exemplo), Tempo ("meia-noite", por exemplo), Coisa ("carro"), entre outras. O reco-

nhecimento de entidade nomeada é muito utilizado em sistemas de perguntas e respostas, pois as perguntas, frequentemente, se referem à informações sobre essas entidades (SOCHER et al., 2012; ZELENKO; AONE; RICHARDELLA, 2003).

- Tokenização: Consiste em dividir um texto ou uma frase em um número diferente de *tokens*. *Tokens* são unidades menores que podem ser identificadas, podem ser palavras, frases pequenas, números, pontuações, etc.

  Na língua portuguesa isso é feito partindo da separação das palavras através de delimitadores, que marcam os limites das palavras e, podem ser espaços em branco ou símbolos de pontuação como “,”, “:”, “;”, “-” e “.”. Mas em casos específicos, como o caso de “,”, “-” e “.”, não devem ser separados dos outros caracteres que vêm antes ou depois. Por exemplo, a frase “Além disso, a produção será descontinuada em 8,3%.” possui 11 *tokens*, que são: “Além”, “disso”, “,”, “a”, “produção”, “será”, “descontinuada”, “em”, “8,3”, “%” e “.”, tendo em vista que 8,3 deve ser um único *token* e não três *tokens* diferentes.

  Outra tarefa que a tokenização faz é separar palavras que estejam contraídas, por exemplo, a palavra “da” é separada em dois *tokens*: “de” e “a”, e a palavra “nelas” é separada nos *tokens* “em” e “elas”. Porém, pode haver ambiguidade em algumas palavras, o que ainda é um desafio para a tokenização em português. Por exemplo, a palavra “pelo”, pode ser a junção de “por” e “o” ou o sunbstantivo “pelo”, e a palavra “consigo”, que pode ser tokenizada como “com” e “si” ou à conjugação do verbo “conseguir”.

- *Stemming* e Lematização: *Stemming*, também conhecido como radicalização, é caracterizado como um processo para reduzir palavras para sua forma base, ou radical. Isso é feito cortando os sufixos. Uma vantagem dessa tarefa é uniformizar e diminuir o vocabulário. Por exemplo, as palavras “certo”, “certidão”, “incerto”, “certamente”, “certificação”, “certeiro” e “incerteza” possuem o mesmo radical “cert” e, portanto, tornam “cert” um bom candidato a subpalavra.

  Lematização também se refere a reduzir as palavras para sua forma base, mas melhorando o vocabulário e a análise morfológica das palavras. Essa prática remove palavras desnecessárias interpretanto a parte do discurso (POS), ou o contexto da palavra. Isso é feito para remover terminações abruptas das palavras, como ocorre no *stemming*.

  Como feito de comparação, aplicar *stemming* na palavra “carreiras” retornará “carr” e aplicar lematização na mesma palavra retornará “carreira”.

- *Sentence Segmentation*: Segmentação de frases, em português, envolve a identificação de onde estão os limites das frases nos textos. Isso é feito pelo uso e ajuda de pontuações encontradas nos textos. Também é chamado de detecção de limite

de sentença, pois o problema é descobrir onde cada sentença termina (HAPKE; HOWARD; LANE, 2019).

O desafio dessa técnica não é identificar as pontuações que delimitam as frases, pois já é um conjunto conhecido (“.”, “!” e “?”, “...”). O principal desafio é separar essas ocorrências conhecidas de outras utilizações desses mesmos caracteres. Por exemplo, na sentença “Fui à clínica do Dr. Nilo.” temos dois caracteres “.”.

O primeiro “.” indica a abreviação da palavra “Doutor” e, o segundo, delimita o fim da frase. Já na frase “Fui à clínica do Dr.”, o “.” indica tanto a abreviação quanto o fim da sentença. Outro caso de ambiguidade no uso de pontuações que delimitam o fim de uma sentença é encontrado em numerais, como, por exemplo, na frase “A venda de 25.000 ações fez o índice de rentabilidade baixar para 0.5%, segundo a BOVESPA.”, ou também em definições matemáticas, como “As permutações de cinco elementos podem ser calculadas como 5!, que é o fatorial de cinco.”.

Em todos estes casos de exemplos, fica difícil detectar quando a pontuação está sendo utilizada com função de fim de sentença ou não. Atualmente três tipos de abordagens computacionais são utilizadas para resolver a segmentação de frases: abordagens baseadas em regras, abordagens baseadas em aprendizado de máquina supervisionado e abordagens baseadas em aprendizado de máquina não supervisionado.

- Remoção de *stopword*: *stopwords* são palavras que não têm valor informacional. Assim, essa técnica é caracterizada pela retirada de palavras que não possuem um significado relevante para análise, como preposições, conjunções, artigos e verbos de ligação, por exemplo. Essa retirada de palavras reduz o trabalho de processamento.

  Por exemplo, na frase “qual a capital da Grécia”, após dividir em palavras utilizando tokenização, as *stopwords* “qual”, “a” e “da” são removidas, ficando assim, apenas “capital” e “Grécia”. Por outro lado, essa remoção pode trazer uma perda de informação relevante, como na famosa expressão “ser ou não ser, eis a questão”, se retirássemos as *stopwords*, ficaria apenas “questão”, o que desfigura a expressão completamente.

2.1.5 Tradução Automática

A tradução automática (TA), também chamada tradução de máquina (do inglês, *machine translation*, (MT)), é caracterizado pela tradução de um texto por um sistema computacional de um idioma para outro, sem interferência humana. Várias abordagens foram desenvolvidas para a tradução automática, como abordagens baseadas baseadas em regras, exemplos, estatísticas e TA neural, que é mais recente (CASELI; NUNES, 2024).

Algumas aplicações que utilizam tradução automática incluem:

- Texto para texto: recebe um texto de entrada e gera uma versão traduzida, também em formato de texto;
- Texto para fala: recebe um texto de entrada e gera um áudio para a língua alvo;
- Fala para texto: recebe um áudio de entrada e gera uma versão em formato de texto;
- Fala para fala: recebe um áudio de entrada e gera uma versão traduzida, também em áudio;
- Imagem (com palavras) para texto: recebe uma imagem que contém um texto e gera uma tradução do texto.

Com a utilização da tradução automática, seu impacto pode afetar a sociedade. Por isso, de acordo com (CASELI; NUNES, 2024), "a avaliação da TA tornou-se mais importante, visando garantir a qualidade da tradução".

2.1.5.1 Abordagens

De acordo com (CASELI; NUNES, 2024), existem várias maneiras de utilizar a tradução automática, da mais simples (tradução direta), que consiste na tradução palavra-a-palavra (sequência de palavras), até a que é mais utilizada atualmente, que é a tradução baseada em redes neurais artificiais (tradução neural). Porém, entre essas duas abordagens, também existem abordagens que são intermediárias, como a baseada em regras, tradução por interlíngua e tradução estatística.

Para exemplificar essas abordagens a frase "A casa do meu avô é linda." será traduzida para o inglês.

2.1.5.2 Tradução Direta

Na tradução direta acontece a relação direta entre as palavras do texto original para as palavras da língua alvo, sem passar por outros tipos de análise. Portanto, na frase de exemplo cada palavra seria mapeada para sua palavra equivalente em inglês, usando um dicionário (também chamado de léxico) bilíngue, por exemplo.

Usando o léxico bilíngue que está disponível online[1] e a tradução palavra-a-palavra, a resposta seria como apresentada no exemplo a seguir:

> Entrada: A casa do meu avô é linda.
> Saída: _ *house* _ *my granddad* _ *beautiful.*

No texto de saída, "_" substitui palavras que não encontraram seus equivalentes no dicionário consultado.

[1] https://dl.fbaipublicfiles.com/arrival/dictionaries/pt-en.txt

Como o processo de tradução se dá pela substituição de uma palavra por outra em uma lista de pares de palavras, não há nenhum processamento das línguas envolvidas (nesse caso português e inglês). Por esse motivo uma das limitações dessa abordagem é que ela não é capaz de lidar com a estrutura (sintaxe) da língua, algo que é fundamental para o tratamento adequado da língua (CASELI; NUNES, 2024).

2.1.5.3 Tradução Automática Baseada em Regras

A tradução automática baseada em regras (do inglês, *Rule-based Machine Translation* ou RBMT), é definida por (CASELI; NUNES, 2024) como "sistemas baseados em conhecimento desenvolvidos por meio da especificação de regras linguísticas, que levam em consideração morfologia, sintaxe e semântica das línguas envolvidas", essas regras são criadas manualmente por especialista em linguagem.

Com essas regras, a RBMT também considera a sintaxe das línguas envolvidas. Desse modo, na frase de exemplo, poderia ser adicionada uma regra que inverte a posição do sujeito que possui a casa com o uso do "'s" (apóstrofo seguido de s), ficando "My grandfather's house is beautiful" ao invés de "The house of my grandfather is beautiful". A tabela 1 mostra um exemplo dessa regra.

Tabela 1 – Exemplo de regra para a tradução automática baseada em regras

<SUB> <PREP/de>+<DET/[a|o]> => 's <SUB>

Fonte: o autor. Adaptado de (CASELI; NUNES, 2024)

A regra apresentada significa que quando for encontrado, na língua original, um substantivo (representado por <SUB>) seguido da preposição (<PREP>) "de" junto (+) com os artigos (<DET>) "a" ou (|) "o", a saída deve ser o apóstrofo (') seguido de "s" e o substantivo correspondente na língua que se deseja. O símbolo "=>" separa a língua original (à esquerda) do resultado que deve ser gerado na língua que se deseja (CASELI; NUNES, 2024).

Uma grande desvantagem da tradução automática baseada em regras está na necessidade de mapear todo o conhecimento das línguas envolvidas em regras corretas, genéricas e abrangentes para que seja possível aplicar a vários exemplos. Além disso, se mudar de língua original ou língua alvo da tradução, o conjunto de regras também deve ser alterado, já que cada idioma possui suas regras específicas.

Uma vantagem desses sistemas é que não é necessário textos bilíngues para seu treinamento, assim são bons para traduções de idiomas com recursos limitados. Outra característica, é que o conhecimento é entendível pelo ser humano, facilitando a manutenção.

2.1.5.4 Tradução por Interlíngua

(CASELI; NUNES, 2024) define que "a tradução por interlíngua se propõe a usar uma língua intermediária - metalíngua - que é independente das línguas envolvidas na tradução automática e ao mesmo tempo é capaz de representar informações de qualquer outra língua". Essa metalíngua seria não ambígua, assim, mais simples de processar que as linguagens naturais. Com base nessa definição, esse processo de tradução seria composto de duas etapas: uma tradução da língua original para a metalíngua, e a outra, da metalíngua para a língua alvo.

Essa tradução foi compartilhada por vários grupos de pesquisa no passado, mas ao colocar em prática, alguns problemas foram realçados. O maior problema é que não é possível, segundo críticos dessa abordagem, criar uma linguagem que representa o significado de todas as outras, uma linguagem universal.

2.1.5.5 Tradução Automática Baseada em Exemplos

Os sistemas computacionais de tradução automática baseada em exemplos (*Example-based Machine Translation* ou EBMT, do inglês), chamados também de tradução por analogia, estão associados ao artigo publicado por (NAGAO, 1984), onde o autor apresenta um modelo que se baseia na imitação de exemplos de tradução de frases que são iguais, para efetuar a tradução a partir de exemplos existentes.

Esses sistemas de exemplos usam informações extraídas de bases de dados em que as sentenças ficam alinhadas em paralelo, de um lado a língua original e do outro a língua alvo. A tabela 2 mostra um exemplo desse tipo de sistema.

Tabela 2 – Exemplos para a tradução baseada em exemplos

A casa é muito bonita.	*The house is very beautiful.*
Meu avô foi internado ontem.	*My grandfather was hospitalized yesterday.*
Eu comprei uma jaqueta linda.	*I bought a beautiful jacket.*

Fonte: o autor. Adaptado de (CASELI; NUNES, 2024)

Com essa abordagem, esses exemplos da tabela 2 forneceriam uma base para que o sistema aprenda traduções de partes de texto, como apresentados na tabela 3.

Tabela 3 – Trechos aprendidos

a casa	*the house*
meu avô	*my grandfather*
linda	*beautiful*

Fonte: o autor. Adaptado de (CASELI; NUNES, 2024)

E com base nesses trechos aprendidos, o sistema baseado em exemplos poderia combiná-los para gerar a saída apresentada a seguir:

Entrada: A casa do meu avô é linda.
Saída: *The house _ my grandfather _ beautiful.*

De acordo com (CARL; WAY, 2003), definir exatamente os sistemas de exemplos é difícil, já que, segundo ele, esses sistemas "assumem uma posição entre os sistemas baseado em regras e os estatísticos", pois eles utilizam abordagens baseadas em regras, mas também, orientadas por dados. Mesmo assim, os sistemas baseados em exemplos e os estatísticos estão enquadrados no paradigma de tradução automática baseada em *corpus* (CASELI; NUNES, 2024).

2.1.5.6 Tradução Automática Estatística

Os sistemas de tradução automática estatísticos (do inglês, *Statistical Machine Translation* ou SMT) foram apresentados pela primeira vez por (BROWN et al., 1988). A característica principal desses sistemas é de usar modelos estatísticos para extrair pares de tradução de *corpora* bilíngues. Segundo (CASELI; NUNES, 2024), existem três abordagens principais para a tradução automática estatística:

- Tradução automática estatística baseada em palavras (*Word-based Statistical Machine Translation*): alinha as palavras do texto original a palavras do texto na língua alvo e calcula a probabilidade de tradução.
- Tradução automática estatística baseada em frases (do inglês, *Phrase-based Statistical Machine Translation ou PBSMT*): alinha fragmentos de frases e palavras do texto original ao texto na lingua alvo, comparando essas frases e suas frases vizinhas ao considerar uma tradução. Essas frases também são chamadas de n-gramas, que são sequências de n palavras (exemplo: um unigrama é uma palavra, um bigrama são duas palavras, um trigrama são três palavras, e assim por diante). Essa é a abordagem estatística mais utilizada.
- Tradução automática estatística baseada em sintaxe: traduzem unidades sintáticas usando árvores sintáticas geradas por analisadores sintáticos.

Em todas essas abordagens, a probabilidade desempenha um papel central ao determinar como um texto deve ser traduzido de um idioma para outro. A tradução é realizada com base em dois modelos computacionais (CASELI; NUNES, 2024): um modelo que especifica como mapear um texto de um idioma para outro e um modelo de língua que especifica como gerar um texto fluente na língua alvo. A tabela 4 mostra exemplos de probabilidades de tradução de frases do *corpus* FAPESP (AZIZ; SPECIA, 2011).

Tabela 4 – Exemplos de frases e suas probabilidades

id	língua original (português)	língua alvo (inglês)	probabilidade
1	a casa do	’ house	0.0207779
2	a casa do	the house from	0.0623338
3	a casa	the house	0.297619
4	a casa	the home	0.0646474
5	do meu	of my	0.0813954
6	do meu	that of my	0.191576
7	meu avô	my grandfather	0.662453
8	meu avô , meu pai , eu	my grandfather , my father , me	0.0623338
9	avô	grandfather	0.916667
10	é	is	0.611613
11	é	é	0.794943
12	linda	beautiful	0.0389678
13	linda	pretty	0.00259724

Fonte: o autor. Adaptado de (CASELI; NUNES, 2024)

Olhando para a tabela 4, algumas opções de traduções que podem ser geradas são:

The house from my grandfather is beautiful.[2]
The house that of my grandfather is pretty.[3]
The home of my grandfather é beautiful.[4]

Assim, para a escolha de qual dessas frases será a resposta, um modelo de linguagem é utilizado onde ele, segundo (CASELI; NUNES, 2024), “diz qual é a melhor sentença com base na probabilidade de ela ser encontrada na língua-alvo, ou melhor, no *corpus* de treinamento usado para gerar o modelo de língua alvo”.

Uma das vantagens dessa abordagem é que quanto mais dados são utilizados para o treinamento das aplicações, maior a qualidade e a tradução estatística sai na frente das abordagens anteriores. Uma desvantagens dessa abordagem é que é difícil criar bases de dados para idiomas com recursos limitados, e também, a tradução estatística tem dificuldade com pares de línguas com as ordens das palavras em diferentes posições.

Os sistemas de tradução estatística baseada em frases (PBSMT), igual os de (KOEHN; OCH; MARCU, 2003) eram considerados o estado da arte, até surgir a tradução neural, a partir de 2015 (CASELI; NUNES, 2024). Um exemplo é o tradutor da Google, que utilizava a PBSMT de 2006/2007 até, aproximadamente, 2016/2017. E, atualmente, a Google utilizam a tradução neural ou um sistema híbrido (estatístico e neural) (CASELI; NUNES, 2024).

[2] Combinando as frases 2, 7, 10 e 12.
[3] Combinando as frases 3, 6, 9, 10 e 13.
[4] Combinando as frases 4, 5, 9, 11 e 12.

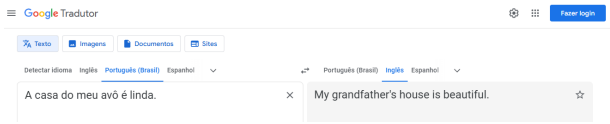
2.1.5.7 Tradução Automática Neural

Os sistemas de tradução automática neural (do inglês, *Neural Machine Translation ou NMT*), foram apresentados na década de 1990, onde alguns autores sugeriram redes neurais para a tarefa de tradução, mas segundo (KOEHN, 2020), "a complexidade computacional envolvida excedia em muito os recursos computacionais daquela época e, portanto, a ideia foi abandonada por quase duas décadas".

De acordo com (CASTILHO; WAY et al., 2017) "os modelos neurais consistem na construção de redes neurais *end-to-end* que mapeiam textos paralelos alinhados e são treinados para maximizar a probabilidade de uma sequência alvo Y, dada uma frase de origem X, sem informações linguísticas externas adicionais". Dessa forma, os sistemas neurais podem ser construídos com apenas uma rede ao invés de uma sequência de tarefas separadas, como na tradução estatística.

Uma característica da tradução neural é que na NMT a frase original é considerada no aprendizado, de uma só vez, e tanto da esquerda para a direita quanto da direita para a esquerda, dessa forma não há quebra em frases como acontecia na PBSMT, tampouco uma divisão entre modelo de tradução e modelo de língua. Assim, a tradução neural tende a ser mais fluente e natural, como mostra a figura 3:

Figura 3 – Tradução gerada pelo Google tradutor (utilizando tradução neural)



Fonte: o autor.

A tradução neural se baseia em duas tecnologias que se tornaram comuns em PLN: *embeddings* e modelo de atenção (CASELI; NUNES, 2024). *Embeddings* são formas de representar palavras, onde elas são mapeadas para vetores em um espaço de n (100, 300 ou mais) dimensões. Dessa forma, é possível mapear as características linguísticas (morfológicas, sintáticas e semânticas) nesse espaço vetorial. Por exemplo, na figura 4 é possível observar que a palavra "avô" está próxima, semanticamente, de outras palavras a partir das *word embeddings* do NILC[5] como "pai", "tio", "sobrinho", etc.

Utilizando as *word embeddings* português e inglês do MUSE[6] também é possível observar as similaridades entre línguas como mostrado na figura 5.

Dessa forma, a tradução neural se baseia em um modelo sequencial que prediz um palavra por vez, considerando toda a frase original e, também o que já foi produzido para a língua alvo. Desde o início, a modelagem sequencial neural passou por diferentes arquiteturas, desde redes neurais recorrentes (do inglês, *recurrent neural network* ou RNN)

[5] http://www.nilc.icmc.usp.br/embeddings
[6] https://github.com/facebookresearch/MUSE

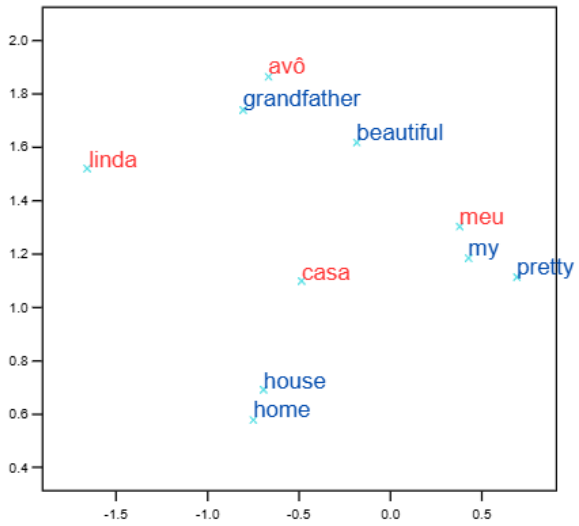
Figura 4 – Vizinhos mais próximos da palavra “avô” obtidos via consulta às *word embeddings* do NILC geradas usando o GloVe e dimensão 300.

```
[('pai', 0.7426180839538574),
 ('tio', 0.7307775020599365),
 ('sobrinho', 0.6814965009689331),
 ('irmão', 0.6784138679504395),
 ('avó', 0.6334453821182251),
 ('filho', 0.6296581029891968),
 ('paterno', 0.628169059753418),
 ('bisavô', 0.6196369528770447),
 ('sogro', 0.5976346731185913),
 ('amigo', 0.5973160862922668)]
```

Fonte: o autor. Adaptado de (CASELI; NUNES, 2024).

usadas para codificação (do inglês, *encoder*) e decodificação (do inglês, *decoder*) até os mecanismos de atenção (do inglês, *attention mechanism*) (BAHDANAU; CHO; BENGIO, 2015), que permitem ao decodificador focar em partes específicas da frase de entrada quando está em seu processo de geração da saída (CASELI; NUNES, 2024).

Figura 5 – Visualização, em duas dimensões, das palavras em português (em vermelho) das palavras da frase de exemplo e suas possíveis traduções para o inglês (em azul).



Fonte: o autor. Adaptado de (CASELI; NUNES, 2024).

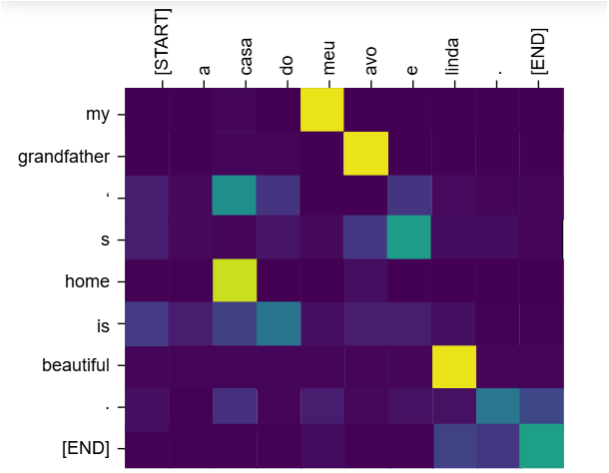
Atualmente, os *Transformers* (VASWANI et al., 2017) são o estado da arte na tradução. A figura 6 mostra a tradução da frase de exemplo, em português, para inglês usando *Transformers* (CASELI; NUNES, 2024). Nessa figura, quanto mais clara (amarelo, verde claro, azul claro, etc.) é a célula que une a linha da palavra em inglês com a coluna da palavra em português, maior a “força” da relação entre elas. Por exemplo, existe uma forte relação entre *“my”* e “meu”, *“grandfather”* e “avô”, *“home”* e “casa”, e *“beautiful”*

e “linda”.

Uma desvantagem da tradução neural é que ela é considerada uma caixa-preta (*black box*): para entender como foi feita a tradução depende da visualização do modelo de atenção, já que as previsões dos modelos neurais consistem em milhões de parâmetros. Além disso, a tradução neural também enfrenta o desempenho ruim em idiomas com recursos limitados.

Outro motivo que deve ser levado em consideração é que os sistemas neurais precisam de um *corpus* maior e com melhor qualidade do que os estatísticos, porque eles são rápidos para memorizar exemplos mal-formados (KHAYRALLAH; KOEHN, 2018). Por esse motivo, em alguns idiomas que possuem menos recursos (do inglês, *low-resourced languages*), os sistemas estatísticos ainda podem ter desempenho melhor do que alguns sistemas neurais.

Figura 6 – Visualização de um modelo de atenção usado para traduzir a frase de exemplo.



Fonte: o autor. Adaptado de (CASELI; NUNES, 2024).

2.1.5.8 Avaliação da Tradução Automática

Com a importância da tradução no mundo atualmente, surgiu o interesse na avaliação da qualidade da tradução (AQT– do inglês, *Translation Quality Assessment* ou TQA), e esse interesse tornou-se um subcampo dentro da tradução automática. Porém, segundo (CASTILHO et al., 2018), “a tradução é um processo multifacetado que envolve fatores cognitivos, linguísticos, sociais, culturais e técnicos, definir e medir a qualidade da tradução também reflete essa complexidade”.

De acordo com (CASELI; NUNES, 2024), a Avaliação da Tradução Automática (ATA) é caracterizada como a prática de analisar a saída de tradução de um sistema (ou sistemas) de TA e julgar a qualidade dessa tradução com base em critérios estabelecidos. Existem duas abordagens para a ATA: avaliação automática, que usa métricas

automáticas, e avaliação manual humana, feita por pessoas, ou às vezes combinando as duas.

2.1.5.9 Métricas Automáticas

As Métricas de Avaliação Automática (MAA) são escolhidas na tradução automática quando se busca evitar a intervenção humana direta. Essas MAAs atuam como programas computacionais, recebendo as traduções de um sistema de TA e as traduções de referência (TR) como entrada, e produzindo uma pontuação numérica que indica a similaridade entre as traduções de TA e TR (CASELI; NUNES, 2024). A seguir são apresentadas algumas métricas de avaliação automática que se destacam:

- BLEU (PAPINENI et al., 2002), METEOR (BANERJEE; LAVIE, 2005) e NIST (DODDINGTON, 2002), que são baseadas em n-grama e analisam as similaridades entre palavras;
- TER (e HTER) (SNOVER et al., 2006) e WER (SU; WU; CHANG, 1992), que calculam a distância de edição e também analisam as similaridades entre palavras;
- chrF (POPOVIć, 2015), que investiga a similaridade entre caracteres e mede a similaridade dos n-gramas;

Mais recentemente, surgiram métricas treinadas em modelos baseados em redes neurais usando a arquitetura Transformer. Dentre essas, há as métricas supervisionadas (*supervised-metrics*) e as não-supervisionadas (LEE et al., 2023), e essas categorias podem ser com *word-embeddings* e *contextual-embeddings*. Entre elas se destacam:

- MEANT (*word embedding*) (LO; WU, 2011), BERTscore (ZHANG et al., 2020), Yisi (LO, 2019; **??**) BARTscore (YUAN; NEUBIG; LIU, 2021), (*contextual-embedding*), que são não-supervisionadas;
- BEER (STANOJEVIć; SIMA'AN, 2014) e BLEND (MA et al., 2017) (ambas *word-embeddings*), BERT for MTE (SHIMANAKA; KAJIWARA; KOMACHI, 2019), BLEURT (SELLAM; DAS; PARIKH, 2020) e COMET (REI; FARINHA; MARTINS, 2020), que são supervisionadas.

Uma desvantagem das MAAs é que elas não oferecem explicações sobre erros de tradução, pontos fortes e fracos de um sistema. Elas também não dizem o que funciona no sistema ou o que precisa ser melhorado, o único objetivo é medir a semelhança com as referências. O ideal seria combinar as MAAs com as avaliações humanas e análises qualitativas, para uma melhor compreensão do desempenho dos sistemas de tradução automática.

2.1.6 Métricas Humanas

A avaliação humana oferece uma visão mais detalhada e, segundo (CASELI; NUNES, 2024), "uma análise mais ampla de fenômenos linguísticos complexos subjacentes ao desempenho dos sistemas de tradução, sendo assim imprescindível em uma compreensão mais abrangente dos sistemas de TA". Os paradigmas mais comuns são o de fluência-adequação, pós-edição, ranqueamento de segmentos e anotação de erros, e são apresentados a seguir:

- Adequação (do inglês, *"accuracy"* ou *"fidelity"*) ,também chamada de acurácia ou exatidão, foca na fidelidade semântica entre o texto original e a tradução, mostrando qual a precisão da transferência de significado, para poder determinar se a mensagem da tradução está sendo transmitida fielmente ao sentido da frase original;
- Fluência, também chamada de inteligibilidade (do inglês *"fluency"* ou *"intelligibility"*), foca diretamente na tradução e se preocupa com a naturalidade e a estrutura da tradução, determinando o grau de fluência da saída da tradução automática e como ela se adapta às normas da língua alvo;
- Ranqueamento (do inglês, *ranking*) tem como objetivo classificar e comparar duas ou mais traduções, para definir uma ordenação entre elas, permitindo identificar as diferenças de qualidade. Essas comparações podem ser feitas tanto para traduções de diferentes sistemas quanto entre traduções feitas por humanos;
- Anotação de erros se caracteriza como um método para identificar e classificar falhas presentes em textos traduzidos. Alguns tipos de erros são: palavras ausentes, ordem errada das palavras, palavras incorretas, pontuação inadequada, entre outros;
- Pós-edição (do inglês, *post-editing*) é definido por (O'BRIEN, 2011) como "a correção da saída da tradução automática bruta por um tradutor humano, de acordo com instruções e critérios de qualidade específicos". E existem três perspectivas sobre o esforço envolvido nesse processo:
  - Esforço Temporal: mede o ritmo de pós-edição, avaliando o tempo gasto por palavra ou o número de palavras pós-editadas por segundo;
  - Esforço Técnico: mede a quantidade de operações de edição realizadas, como inserções, remoções, e trocas;
  - Esforço Cognitivo: é medido por meio de diferentes abordagens, incluindo rastreamento ocular (*eye-tracking*).

Algumas considerações importantes sobre a avaliação humana incluem a necessidade de avaliar tanto a adequação quanto a fluência, além da definição do número ade-

quado de avaliadores e das competências exigidas. Mesmo com essas considerações, os avaliadores humanos possuem um papel fundamental em identificar diferenças de qualidade que as métricas automáticas não conseguem, como os aspectos culturais, ambiguidades e os detalhes da língua.

2.2 Redes Neurais Artificiais

Esta seção apresenta uma fundamentação teórica introdutória sobre *Transformers*, que são o estado da arte quando o assunto é tradução automática.

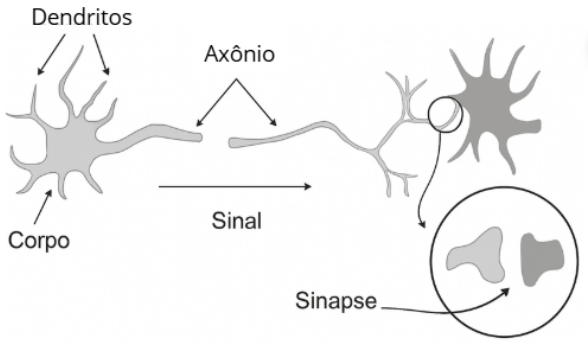
As Redes Neurais Artificiais (RNA) são representações matemáticas baseadas no funcionamento do cérebro humano. A sua função é imitar a habilidade de aprender e adquirir conhecimento que o cérebro humano possui. Desse modo, se faz necessário uma breve introdução sobre o funcionamento do sistema nervoso humano, a fim de compreender melhor seu funcionamento.

2.2.1 O Sistema Nervoso Humano

O principal elemento do sistema nervoso é o neurônio, que possui a característica que o permite responder estímulos e transmitir impulsos nervosos a outros neurônios ou a outras células do corpo humano (FACELI, K. et al.). O neurônio é composto por dendritos, axônio e corpo celular, como é possível observar na figura 7.

Os dendritos recebem estímulos nervosos e os enviam para o corpo celular, que os processa, e se for necessário, gera um novo impulso. Esse impulso é levado pelo axônio até os dendritos de outros neurônios, através de uma conexão conhecida como sinapse. As sinapses permitem que haja a comunicação entre os neurônios (FACELI, K. et al.). A figura 7 ilustra um neurônio simplificado.

Figura 7 – Estrutura simplificada de um neurônio.



Fonte: o autor. Adaptado de FACELI, K. et al.

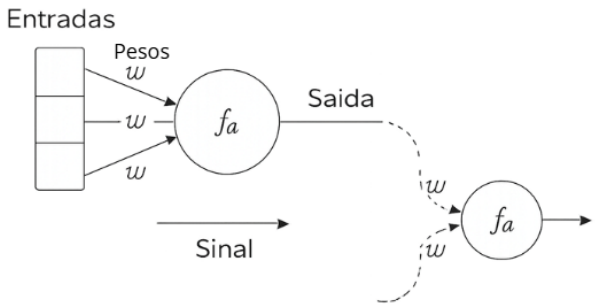
Ainda segundo FACELI, K. et al. o cérebro humano possui entre 10 e 500 bilhões de neurônios, organizados em cerca de 1000 módulos com aproximadamente 500 redes neurais

cada. Cada neurônio pode se conectar a milhares de outros, formando uma rede densa e paralela. Apesar da resposta dos neurônios ser mais lenta que a de um processador digital (cerca de 0,001 segundo), o alto grau de paralelismo torna o processamento de informações muito eficiente, permitindo ao cérebro realizar tarefas complexas mais rapidamente que um computador.

2.2.2 Definição de Redes Neurais Artificiais

Segundo FACELI, K. et al, "as Redes Neurais Artificiais são sistemas computacionais distribuídos formados por unidades de processamento simples, fortemente interconectadas". Essas unidades de processamento são chamadas de neurônios artificiais. Esses neurônios estão organizados em camadas e se comunicam por conexões unidirecionais, cada uma com um peso associado. Esses pesos modulam a informação transmitida e armazenam o conhecimento adquirido pela rede. O processo de ajuste desses pesos, chamado de aprendizado ou treinamento, tem como objetivo fazer com que a rede produza respostas adequadas. Na figura 8 temos uma representação de um neurônio artificial.

Figura 8 – Neurônio Artificial.



Fonte: o autor. Adaptado de FACELI, K. et al.

O funcionamento dessas unidades de processamento é simples. Cada entrada desse neurônio recebe um valor. Os valores recebidos são ponderados por pesos e combinados por uma função matemática (algumas funções propostas na literatura são: linear, limiar e sigmoidal). A saída da função é a resposta do neurônio dada aquela entrada.

Em uma rede neural, os neurônios podem estar em uma ou mais camadas. Quando existem mais de uma camada, a entrada de um neurônio corresponde à saída de um neurônio da camada anterior, e a saída gerada é enviada para o neurônio da camada seguinte. A figura 9 mostra uma rede neural multicamada.

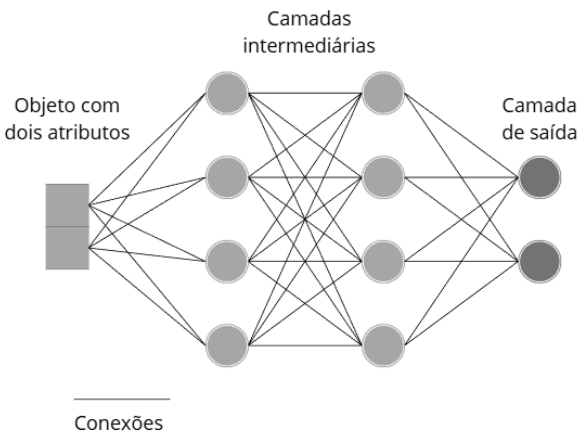
Ao longo do tempo vários modelos de redes neurais foram desenvolvidos, para diferentes aplicações. A seguir são apresentadas dois exemplos de arquiteturas de redes neurais, suas características e aplicações:

- CNN (do inglês, *Convolutional Neural Networks*) - as redes convolucionais possuem

uma arquitetura de dois estágios, que combina um classificador e um extrator de características, que precisa de pouco pré-processamento para o treinamento. Diferente de métodos tradicionais, as CNNs aprendem e reconhecem automaticamente as características dos dados, sem a necessidade da extração manual. Algumas aplicações para essa arquitetura são: detecção de objetos, reconhecimento de fala, visão computacional, classificação de imagens e bioinformática. (Shiri, Farhad et al 2024)

- RNN (*Recurrent Neural Networks*, em inglês) – as redes recorrentes possuem uma memória interna que permite capturar dependências sequenciais. As RNNs consideram a ordem temporal das entradas, fazendo com que elas sejam adequadas para tarefas que envolvem informações sequenciais, como processamento de linguagem natural, classificação de vídeo e reconhecimento de fala. Ao empregar um *loop*, as RNNs aplicam a mesma operação em cada elemento de uma série; dessa forma, a computação atual depende tanto da entrada atual quanto das computações anteriores (**??**). As RNNs possuem algumas variantes, sendo a LSTM (do inglês, *Long Short-Term Memory*) uma delas. A LSTM visa superar uma limitação da RNN simples, que é sua memória de curto prazo, restringindo sua capacidade de reter informações em sequências longas (**??**).

Figura 9 – Rede Neural multicamadas.



Fonte: o autor.

Algumas tarefas de PLN, como tradução automática e sumarização, requerem que a entrada seja um texto (uma sequência) e que a saída também seja um texto (outra sequência). Essa tarefa é chamada de *sequence-to-sequence* (sequência para sequência) ou "seq2seq" (Cho et al., 2014; Sutskever et al., 2014). Nas tarefas "seq2seq" existem dois componentes: o codificador e o decodificador.

O codificador (*encoder*, em inglês) processa a sequência de entrada, que pode ser uma sequência de letras, *tokens*, palavras, frases, e a codifica como um vetor de números, conhecido como vetor de contexto. O decodificador (*decoder*) recebe e processa o vetor de

contexto e o transforma na sequência de saída, também uma sequência de letras, *tokens*, palavras, frases (FACELI, K. et al. 2024). O codificador é uma rede neural, ou várias delas, da mesma forma que o decodificador.

2.2.3 Vantagens e desvantagens

As redes neurais são populares devido a características como a sua capacidade de generalização e sua tolerância a falhas e ruídos (BRAGA, A. P. et al. 2007) (HAYKIN, S. 1999). Isso contribui para um bom desempenho das redes neurais em diversas aplicações. Mas, em alguns casos, o desempenho não é o único aspecto relevante – também é importante compreender a lógica por trás das decisões que foram tomadas pela rede.

Por esse motivo, uma crítica recorrente ao uso de redes neurais está na dificuldade de entender como e por que elas chegaram a certas conclusões. Essa complexidade está no fato de que o conhecimento da rede é codificado em um grande número de parâmetros, que são ajustados por fórmulas matemáticas complexas. Como consequência, as redes neurais artificiais são tratadas como "caixas-pretas". Além disso, a escolha da melhor arquitetura para uma rede neural não é um processo totalmente definido.

Um problemas das redes recorrentes é que, pelo fato de serem sequenciais, o treinamento delas é bem ineficiente, o que gera outro problema: as entradas não podem ser muito grandes e também não podem exigir uma dependência de longa distância. Mesmo que as LSTMs resolvam o problema da dependência de longa distância, ainda assim, mais parâmetros precisam ser treinados, levando a ineficiência (CASELI; NUNES, 2024).

2.3 *Transformers*

Transformer foi introduzido por VASWANI et al, em 2017, para tradução automática e se tornou um modelo fundamental para processamento de linguagem natural. É uma arquitetura de rede neural que faz uso de um sistema conhecido como mecanismo de atenção. Além disso, um Transformer possui dois componentes principais, um de codificação e um de decodificação.

2.3.1 Mecanismo de atenção

O mecanismo de atenção é um conjunto adicional de parâmetros em redes neurais que permite focar nos itens mais relevantes da entrada ao gerar a saída, atribuindo a eles pesos maiores no vetor de contexto, embora outros itens também possam receber algum valor. A ideia pode ser ilustrada como procurar uma receita ideal (chave) para uma tarefa específica (consulta). O modelo avalia todas as receitas (entradas) e calcula uma pontuação para cada uma, destacando as mais relevantes (CASELI; NUNES, 2024).

Esse processo ocorre em três etapas:

1. **Pontuação de alinhamento:** Computa uma pontuação para cada estado escondido da entrada, utilizando a equação 2.1.
2. ***Softmax:*** As pontuações são normalizadas por uma função *softmax* para gerar os pesos de atenção, que representam a relevância relativa de cada item da entrada.
3. **Cálculo do vetor de contexto:** Cada estado escondido é multiplicado por seu peso de atenção. Os mais relevantes têm seu valor amplificado, e os menos relevantes, reduzido.

$$att = W_{\text{combinado}} \times \tanh\left(W_{\text{decod}} \times H_{\text{decod}} + W_{\text{codif}} \times H_{\text{codif}}\right) \tag{2.1}$$

onde W_{codif} e W_{decod} representam matrizes de pesos (parâmetros) aprendidos e H_{decod} representam estados escondidos.

Existem ainda dois tipos principais de atenção:

- **Aditiva (proposto por Bahdanau; Cho; Bengio, 2015):** Usa a equação acima e matrizes de pesos distintas para codificador e decodificador.
- **Multiplicativa (proposto por Luong et al., 2015):** Usa operações diretas de multiplicação (escalares ou com pesos) entre estados do codificador e decodificador. Pode ser: Produto direto entre estados; Produto com matriz de pesos; Soma dos estados seguida por transformações semelhantes ao mecanismo aditivo, mas com pesos compartilhados.

No final desse processo, o vetor de contexto é combinado com o estado anterior do decodificador para gerar a saída. Além disso, os *Transformers* usam um tipo especial chamado auto-atenção, que calcula relevância entre elementos da mesma sequência, geralmente na entrada.

O mecanismo de auto-atenção permite que cada palavra (ou **token**) da sequência foque nas demais, atribuindo diferentes pesos a elas. Para isso, a partir do vetor de *embedding* de cada palavra, são gerados três vetores: *query* (q), *key* (k) e *value* (v) (CASELI; NUNES, 2024). O vetor *query* representa a palavra que está sendo processada, *key* representa as demais palavras da sequência, e *value* carrega as informações associadas. Esse processo é feito de forma matricial, aproveitando o paralelismo das placas gráficas (GPUs).

O mecanismo de atenção simples pode ser insuficiente em certos casos, como em frases com ambiguidade e polissemia. Para lidar com isso, os *Transformers* utilizam o

mecanismo de atenção com múltiplas cabeças (*multi-head attention*), que aplica várias versões paralelas do mecanismo de atenção.

No artigo original de apresentação do *Transformer* (VASWANI et al., 2017), são usadas oito cabeças de atenção, cada uma com suas próprias matrizes de *query*, *key* e *value*, capazes de capturar diferentes aspectos da entrada. Os resultados dessas cabeças são concatenados e passados por uma matriz de pesos adicional, aprendida durante o treinamento, resultando em uma única matriz final que combina as diferentes perspectivas.

Esse processo aumenta a capacidade do modelo, mas também a quantidade de parâmetros treináveis, levantando preocupações sobre o impacto ambiental do treinamento de *Transformers* em larga escala.

2.3.2 Mecanismo residual e normalização

O último componente importante dos *Transformers* são as conexões residuais (He et al., 2016) dentro da subcamada de codificação, inspiradas em redes de visão computacional. Essa conexões foram incluídas porque redes neurais com muitas camadas podem esquecer uma informação importante de entrada após ela passar por muitos processamentos. Esse esquecimento ocorre por causa do problema do gradiente que vira zero após várias multiplicações de valores menores que um (Hochreiter, 1991) durante o *backpropagation* (algoritmo utilizado para aprender os pesos de uma rede neural, ele usa o método de otimização de gradiente descendentes para minimizar o erro do modelo.). A conexão residual evita uma parte dessas transformações multiplicativas, pulando algumas delas.

A saída da auto-atenção é somada à entrada original, e o mesmo ocorre com a saída da camada totalmente conectada. Isso mantém uma memória da informação original, mesmo após os processamentos. Além disso, há camadas de normalização colocadas antes das camadas de atenção e *feedforward* (conhecido como pré-normalização). Essa técnica ajuda a estabilizar o treinamento e ainda está sendo estudada quanto à sua colocação ideal. A arquitetura original possui seis subcamadas de codificadores, cada uma com 512 neurônios na camada escondida e oito cabeças de atenção.

2.3.3 Codificador e decodificador

O decodificador é semelhante ao codificador, composto por subcamadas com auto-atenção, conexões residuais, normalização e uma rede totalmente conectada. A principal diferença é a inclusão de uma camada de atenção adicional, que se conecta com o codificador. Nessa camada, as matrizes K e V vêm do codificador, enquanto Q vem da camada anterior do decodificador, permitindo que ele decida o que considerar da entrada codificada ao gerar a saída.

Outra característica é que a auto-atenção no decodificador é mascarada, ou seja,

ao prever um *token*, ele só pode acessar os *tokens* anteriores, não os futuros. Isso simula o comportamento sequencial da geração de texto, onde cada novo *token* depende apenas dos anteriores.

Por fim, a saída do decodificador passa por uma camada linear que gera um vetor de *logits* — pontuações para cada palavra do vocabulário. Esse vetor é então processado por uma função *softmax*, que converte os *logits* em probabilidades para cada possível próxima palavra.

O uso extensivo de mecanismos de atenção, combinados com outros outros componentes, faz com que os *Transformers* e suas variações sejam o estado da arte em diversas tarefas de PLN até o momento (Wolf et al., 2020). Eles são o componente principal dos modelos de linguagem em larga escala (em inglês, *large language models* ou LLMs).

As tarefas que lidam com a linguagem podem utilizar diferentes partes do *Transformer*, algumas utilizam apenas o codificador, ou apenas o decodificador. A seguir estão algumas arquiteturas que utilizam *Transformers*:

- *BERT (Bidirectional Encoder Representations for Transformers)* - considera apenas o componente codificador (Devlin et al., 2019).
- *GPT (Generative Pre-trained Transformer)* - usa blocos decodificadores para funcionar como um modelo autorregressivo de geração de texto (Radford; Narasimhan, 2018).

3 Implementação

O presente capítulo detalha o desenvolvimento e a configuração dos dois protótipos de tradução automática, que constituem o cerne da análise e comparação deste trabalho. Conforme postulado na hipótese de pesquisa (Seção 1.3), a arquitetura Transformer é testada em contraste com uma abordagem de Rede Neural Recorrente (RNN) com camada GRU (Gated Recurrent Unit). Ambos os protótipos foram concebidos como modelos de sequence-to-sequence (seq2seq) (Seção 2.2.2), adaptados para as especificidades de cada arquitetura, visando traduzir sentenças entre linguagens distintas.

Todo o processo de implementação, treinamento e avaliação dos modelos foi conduzido no ambiente do *Google Colab*, utilizando sua versão gratuita. Essa escolha foi motivada pela praticidade e pela disponibilidade de recursos computacionais em nuvem, como GPUs. No entanto, a versão gratuita impôs limitações de processamento que impactaram o estudo. Em determinados momentos, o limite de uso de recursos (como tempo de GPU alocado e memória RAM disponível) foi atingido. Isso resultou em desconexões da sessão, restringindo o número total de épocas de treinamento[1] viáveis e a possibilidade de testar configurações de arquitetura mais complexas e computacionalmente exigentes.

3.1 Implementação com Rede Neural Recorrente (GRU)

A primeira abordagem de implementação seguiu o paradigma clássico de tradução automática neural (Neural Machine Translation - NMT) de arquitetura Codificador-Decodificador com unidades recorrentes. Inicialmente, tentou-se treinar um modelo para a tradução do Português para o Inglês. No entanto, o modelo não apresentou convergência satisfatória, atingindo uma acurácia de apenas 53%, indicando uma dificuldade intrínseca em aprender as complexidades da tradução entre esses dois idiomas com a arquitetura proposta.

Diante disso, a implementação foi ajustada para a tradução de sentenças do Inglês para o Francês, que se mostrou mais viável para a capacidade do modelo. A escolha por este par, em detrimento do Português-Inglês, baseia-se em duas hipóteses principais:

1. A Complexidade Morfológica do Português: A hipótese central para a dificuldade no par Português-Inglês (que atingiu apenas 53% de acurácia) é a maior complexidade morfológica do português. Embora o português e o francês sejam ambas línguas

[1] Uma 'época' (epoch) refere-se a um ciclo completo em que o algoritmo de aprendizado processa todo o conjunto de dados de treinamento uma vez. Assim como um estudante lê um livro didático várias vezes para aprender, o modelo processa os dados por múltiplas épocas para ajustar seus parâmetros e minimizar erros (GOODFELLOW et al., 2016).

românicas, o português preservou um sistema de conjugação verbal muito mais complexo e sintético. Por exemplo, o português possui tempos verbais como o futuro do subjuntivo e o pretérito mais-que-perfeito em sua forma simples ('eu fora'), que são raros ou inexistentes no francês moderno, onde se prefere o uso de tempos compostos (*"j'avais été"*). Essa riqueza morfológica na língua de origem gera um vocabulário muito mais vasto e esparso, um desafio conhecido para modelos de Tradução Automática Neural. Modelos que operam no nível de palavras, como a RNN implementada, são especialmente suscetíveis a esse problema de esparsidade.

2. A Viabilidade e Semelhança do Par Inglês-Francês: O par Inglês-Francês apresenta um cenário mais favorável por duas razões. Primeiramente, o inglês (língua de origem) possui uma morfologia verbal e nominal muito mais simples que a do português, reduzindo drasticamente o problema da esparsidade do vocabulário de entrada. Em segundo lugar, embora o francês (língua de destino) seja uma língua românica, ele é morfologicamente menos complexo que o português, como visto anteriormente. Além disso, o inglês e o francês partilham uma extensa sobreposição lexical, o que facilita a tarefa de alinhamento de palavras. Consequentemente, a maior regularidade morfológica de ambos os idiomas (comparado ao português) e a semelhança lexical tornam o aprendizado para o modelo de tradução uma tarefa mais direta.

Portanto, o objetivo deste protótipo passou a ser a construção de um modelo de aprendizado supervisionado treinado do zero para a tradução de sentenças do Inglês para o Francês. Especificamente, a Rede Neural Recorrente (RNN) foi construída utilizando a camada GRU (Gated Recurrent Unit), uma variante que visa mitigar o problema de dependência de longo prazo (memória de curto prazo) inerente às RNNs tradicionais.

3.1.1 Pré-processamento e Preparação de Dados

A etapa de pré-processamento é crucial para converter o texto em um formato interpretável pelo modelo (Seção 2.1). As etapas executadas foram:

- Coleta do Corpus: O conjunto de dados é formado por sentenças paralelas alinhadas em Inglês e Francês. Os dados foram obtidos de um repositório público e aberto[2], disponibilizado gratuitamente para fins de pesquisa e aprendizado. Este formato é um padrão para o treinamento de sistemas NMT (Seção 2.1.5.6).

- Tokenização: O texto em cada língua foi convertido em sequências de identificadores (IDs) de palavras, utilizando a função *Tokenizer* do Keras. A tokenização é o

[2] Arquivos de dados disponíveis em: https://raw.githubusercontent.com/projjall/datasets/master/small_vocab_en.txt (Inglês) e https://raw.githubusercontent.com/projjall/datasets/master/small_vocab_fr.txt (Francês)

processo que divide um texto em unidades menores (*tokens*), geralmente palavras, e é uma das técnicas fundamentais do PLN (Seção 2.1.4).

- Vocabulário e *Embedding*: Foram determinados o tamanho do vocabulário do Inglês (199) e do Francês (344) excluindo o *token* de *padding*. O modelo utiliza vetores de *Embedding* para mapear palavras para um espaço vetorial denso (Seção 2.1.5.7), capturando suas características linguísticas.

- *Padding*: Para uniformizar o comprimento das sequências, conforme exigido para o treinamento em *batch* (Seção 2.1.4), a função pad_sequences foi aplicada, preenchendo as sequências menores com um *token* especial (<PAD>). O comprimento máximo de sequência (*maxlen*) da língua alvo (Francês) foi de 21 *tokens*. A saída do pré-processamento foi um *array NumPy* com a *label* do Francês na forma tridimensional, conforme requerido para a função de perda sparse_categorical_crossentropy do Keras.

3.1.2 Arquitetura do Modelo GRU

O modelo embed_model foi implementado como uma rede *Sequential* do Keras e estruturado com as seguintes camadas:

- *Embedding*: Camada de entrada que converte as sequências de IDs (Inglês) em vetores densos (256 dimensões).

- GRU: Uma camada recorrente com 256 unidades que funciona como o Codificador e o Decodificador em uma arquitetura unificada. A opção return_sequences=True garante que a saída em cada passo de tempo (sequência) seja passada para a camada seguinte. *TimeDistributed(Dense)*: Uma camada densa de 1024 neurônios com ativação ReLU, envolta pelo *wrapper TimeDistributed*. Esse *wrapper* (invólucro) permite que a camada densa seja aplicada de forma independente a cada vetor de saída da GRU (ou seja, a cada passo de tempo), garantindo que o modelo mantenha a correspondência entre as dimensões temporais da entrada e da saída.

- *Dropout*: Uma camada de regularização com taxa de 0.5 para prevenir *overfitting*.

- *Time Distributed(Dense)* (Saída): A camada final, também *TimeDistributed*, utiliza uma função de ativação *Softmax* (Seção 2.2.2). O número de neurônios é igual ao tamanho do vocabulário do Francês (345, incluindo o *token* <PAD>), gerando a distribuição de probabilidade de cada palavra do vocabulário ser a próxima palavra na sequência traduzida.

O modelo foi compilado com a função de perda sparse_categorical_crossentropy e o otimizador Adam com uma taxa de aprendizado de 0.005.

3.1.3 Treinamento e Saída

O treinamento foi realizado por 20 épocas com um batch size de 1024. A acurácia de validação do modelo, calculada sobre o conjunto de validação, atingiu 0.9389 na última época. A função logits_to_text foi implementada para converter os logits (saídas da camada final) para a sequência de palavras em Francês. Essa função mapeia o índice do logit de maior valor (usando np.argmax) de volta para a palavra correspondente no dicionário do tokenizer Francês, um processo inverso ao da tokenização.

3.2 Implementação com Arquitetura *Transformer*

A segunda abordagem utiliza a arquitetura *Transformer*, que representa o estado da arte na tradução automática neural. A principal distinção reside na utilização de mecanismos de auto-atenção (*self-attention*) e múltiplas cabeças de atenção (*multi-head attention*) (Seção 2.3.1), que permitem capturar dependências de longo alcance e diferentes nuances sintáticas e semânticas de forma mais eficiente e paralela , superando as limitações sequenciais de treinamento das RNNs (Seção 2.2.3).

Para esta implementação, foi utilizado um modelo pré-treinado, o mBART-50 (*Multilingual Denoising Pre-training for Neural Machine Translation*). O mBART-50 é um modelo Codificador-Decodificador baseado em *Transformer* (Seção 2.3.3) que foi treinado em 50 idiomas. O uso de um modelo pré-treinado permite a tradução zero-shot, ou seja, tradução de alta qualidade sem a necessidade de treinamento no corpus específico do usuário.

3.2.1 Configuração e Preparação

A implementação foi realizada utilizando a biblioteca *Hugging Face Transformers* e o *framework PyTorch*.

- Carregamento dos Modelos: Foram carregados os modelos *MBartForConditionalGeneration* e *MBart50TokenizerFast*, utilizando o identificador *facebook/mbart-large-50-many-to-one-mmt*.
- Configuração de Idioma de Origem: O tokenizer foi inicializado especificando o idioma de origem (src_lang) como inglês (en_XX).
- Aceleração de Hardware: O modelo foi movido para um dispositivo GPU (cuda), caso disponível, para explorar o paralelismo (Seção 2.3.1) inerente à arquitetura *Transformer*.

3.2.2 Processo de Tradução

A tradução da frase de teste ("My grandfather's house is beautiful") é um processo direto de inferência:

- Tokenização de Entrada: A frase de origem em inglês é tokenizada (transformada em tensores de entrada) e enviada para o dispositivo (GPU/CPU).
- Geração Condicional: A função model.generate() é invocada, desempenhando o papel do Decodificador (Seção 2.3.3) para gerar a sequência alvo. O argumento forced_bos_token_id (Begin-of-Sequence) é crucial para instruir o modelo a começar a gerar na língua alvo, que é o francês (fr_XX). Este processo faz com que a rede mapeie as representações internas do inglês para o francês.
- Decodificação de Saída: Os tokens de saída (IDs) gerados pelo modelo são convertidos de volta para texto legível em Francês (Seção 2.1.4), utilizando tokenizer.batch_decode().

O resultado esperado para a frase de exemplo é uma tradução de alta fluência e adequação, como "La maison de mon grand-père est magnifique".

3.3 Comparação das Abordagens

A Tabela 5 resume as principais distinções metodológicas entre os dois protótipos, estabelecendo a base para a discussão dos resultados (Capítulo 4).

Tabela 5 – Comparativo Metodológico das Abordagens de Tradução Automática

Critério	**RNN (GRU)**	**Transformer (mBART)**
Arquitetura	Codificador-Decodificador Recorrente com GRU	Codificador-Decodificador com Autoatenção e Multi-Head Attention
Treinamento	Treinado do zero, alto custo computacional e de tempo	Pré-treinado, sem necessidade de re-treinamento
Paralelismo	Limitado pela natureza sequencial	Elevado, devido à arquitetura de atenção paralela
Dependências	Dificuldade em capturar dependências longas	Alta capacidade para lidar com dependências de longo alcance
Suporte Multilíngue	Restrito ao par de línguas do *corpus*	Suporte embutido para múltiplos idiomas (até 50)

3.4 Conclusão Preliminar da Implementação

A implementação demonstrou a viabilidade de ambas as abordagens para a tarefa de tradução automática. O modelo RNN (GRU) validou o paradigma clássico NMT, atingindo uma boa acurácia de 93.89 em um corpus restrito. Por outro lado, a implementação do Transformer (mBART-50), sem a necessidade de treinamento específico, estabelece um ponto de referência de qualidade superior e escalabilidade devido à sua robustez e vocabulário vasto, suportando múltiplos idiomas e tradução de frases complexas. Este capítulo serviu como a fundação empírica para a análise comparativa de desempenho que será detalhada no próximo capítulo.

4 Resultados

A Tradução Automática Neural (NMT) evoluiu significativamente, com duas arquiteturas principais dominando o campo: Redes Neurais Recorrentes (RNNs) e Transformers. Este capítulo analisa e compara a eficácia dessas duas arquiteturas, contextualizando os resultados de um modelo Transformer (mBART) na tradução do inglês para o francês e explicando as limitações encontradas em um modelo RNN.

A tabela 6 mostra as frases originais, as traduções de referência e as traduções geradas pelo modelo Transformer, seguidas por uma tabela com as métricas de avaliação.

Tabela 6 – Comparativo da Tradução

Frase Original (Inglês)	**Tradução de Referência (Francês)**	**Tradução do Transformer**
we plan to visit china in march	nous prévoyons une visite en chine en mars	nous prévoyons de visiter la chine en mars
india is sometimes snowy during september , but it is never warm in winter	l' inde est parfois enneigée en septembre , mais il est jamais chaud en hiver	l'inde est parfois enneigée en septembre, mais il n'est jamais chaud en hiver
france is never busy during march , and it is sometimes pleasant in september	la france est jamais occupée en mars , et il est parfois agréable en septembre	la france n'est jamais occupée en mars, et il est parfois agréable en septembre
india is sometimes beautiful during spring , and it is snowy in june	l' inde est parfois belle au printemps , et il est neigeux en juin	l'inde est parfois belle au printemps, et il est enneigé en juin
india is never wet during summer , but it is sometimes chilly in winter	l' inde est jamais mouillé pendant l' été , mais il est parfois froid en hiver	l'inde n'est jamais humide en été, mais il est parfois froid en hiver
france is never chilly during january , but it is never mild in october	la france est jamais froid en janvier , mais il est jamais doux en octobre	la france n'est jamais froide en janvier, mais il n'est jamais doux en octobre
the orange is her favorite fruit , but the banana is your favorite	l'orange est son fruit préféré , mais la banane est votre favori	l'orange est son fruit préféré, mais la banane est votre préférée
how do you feed your family	comment nourrissez-vous votre famille	comment nourrissez-vous votre famille
well, we're about to begin our story	eh bien, nous sommes sur le point de commencer notre histoire	bien, nous sommes sur le point de commencer notre histoire
what is light	qu'est-ce que la lumière	qu'est-ce que la lumière
who are we	où sommes-nous	qui sommes-nous
where did we come from	d'où venons-nous	d'où venons-nous
are we alone	sommes-nous seuls	sommes-nous seuls

O modelo mBART, baseado na arquitetura Transformer, demonstrou um desempenho elevado na tarefa de tradução. As métricas agregadas (presentes na tabela 7), com uma pontuação BLEU de 0.8687 e METEOR de 0.9329, indicam uma alta fidelidade e fluência nas traduções. A pontuação chrF de 91.54 reforça essa observação, mostrando que o modelo captura com precisão não apenas a semântica, mas também a morfolo-

gia e a sintaxe do idioma-alvo. A baixa taxa de erro de edição (TER de 8.78) confirma que as traduções geradas são muito próximas das referências humanas, exigindo poucas correções.

Tabela 7 – Métricas de avaliação

Métrica	**Pontuação Média**	**Descrição**
BLEU	8.687	Pontuação alta, indicando grande sobreposição de n-gramas com as referências.
METEOR	9.329	Pontuação muito alta, mostrando excelente correspondência de palavras e ordem.
TER	8.78	Taxa de erro de edição baixa, significando poucas edições para chegar à referência.
chrF	91.54	Pontuação excelente, indicando alta fluência e qualidade em nível de caractere.

Qualitativamente, as traduções apresentam alta correspondência com as referências humanas, o que é consistente com as métricas obtidas (BLEU de 0.8687 e METEOR de 0.9329). As variações observadas são majoritariamente lexicais e não comprometem o significado (ex.: *"visiter la Chine"* vs. *"une visite en Chine"*), indicando que o modelo gera traduções naturais e contextualmente adequadas.

Merece destaque o caso da tradução da frase *"who are we"*. Conforme a tabela 6, a tradução de referência fornecida no corpus de teste foi *"où sommes-nous"* (que significa "onde estamos"), o que representa um erro semântico. No entanto, o modelo Transformer gerou corretamente *"qui sommes-nous"* (que significa "quem somos nós"). Este exemplo é uma forte evidência do poder de generalização do Transformer: o modelo não está simplesmente replicando os dados de um corpus limitado, mas utilizando seu vasto conhecimento pré-treinado para gerar uma tradução semanticamente mais precisa do que a própria referência humana utilizada para esta avaliação.

A eficácia do mBART deriva diretamente das inovações da arquitetura Transformer:

- Mecanismo de Auto-Atenção (Self-Attention): Diferente das RNNs, que processam a informação sequencialmente, o mecanismo de atenção permite que o Transformer pese a importância de todas as palavras na sequência de entrada simultaneamente. Isso resolve o problema do "gargalo de informação" das RNNs, onde a informação do início da sequência pode se perder ao final. O modelo consegue capturar dependências de longo alcance, como a concordância de gênero e número em frases longas, de forma muito mais eficaz.

- Processamento Paralelizável: A ausência de recorrência permite que o Transformer processe todos os tokens de uma sequência em paralelo. Isso não só acelera drasticamente o treinamento em hardware moderno (GPUs/TPUs), mas também permite que o modelo tenha uma visão global da frase desde o início, melhorando a compreensão contextual.

- Codificação Posicional (Positional Encoding): Para compensar a falta de sequencialidade, o Transformer injeta informações sobre a posição dos tokens na sequência. Isso permite que o modelo entenda a ordem das palavras, um aspecto crucial para a gramática e o significado.

4.1 Limitações da Arquitetura RNN

Na implementação do modelo RNN, a tradução retornava um erro quando os dados de teste possuíam alguma palavra que não constava nos dados de treinamento. Contudo, quando o vocabulário da sentença de teste era totalmente conhecido pelo modelo, a tradução ocorria como esperado. Essa ocorrência evidencia um problema prático comum relacionado à generalização de vocabulário.

Adicionalmente, mesmo em cenários onde funciona, a arquitetura RNN possui limitações teóricas intrínsecas em comparação com os Transformers:

- Dificuldade com Dependências de Longo Alcance: O processamento sequencial das RNNs (e suas variantes como LSTM e GRU) torna difícil a manutenção de informações através de longas distâncias. O estado oculto da rede precisa carregar o contexto de toda a frase, o que leva ao problema do "desvanecimento do gradiente" (vanishing gradient), onde a influência de palavras distantes diminui.

- Processamento Sequencial Lento: A natureza recorrente da RNN impede a paralelização, tornando o treinamento e a inferência mais lentos, especialmente em frases longas.

- Gargalo de Informação: A informação da sequência de entrada é comprimida em um único vetor de contexto (o último estado oculto do codificador), que é passado para o decodificador. Esse vetor fixo representa um gargalo, limitando a quantidade de informação que pode ser transmitida.

A análise comparativa, embora limitada pela impossibilidade de execução do modelo RNN em vocabulário aberto, destaca a superioridade da arquitetura Transformer para tarefas de tradução automática. Os resultados do mBART são um testemunho do poder do mecanismo de atenção para capturar relações complexas dentro e entre os idiomas.

Enquanto as RNNs foram um passo fundamental na evolução da NMT, os Transformers representam o estado da arte atual, oferecendo traduções mais rápidas, precisas e fluentes, como evidenciado pelas altas pontuações nas métricas BLEU, METEOR e chrF. A dificuldade de generalização do vocabulário do modelo RNN também aponta para a robustez dos modelos Transformer, que são frequentemente treinados em corpora massivos e multilíngues, tornando-os mais versáteis.

Conclusão

Este trabalho se propôs a analisar e comparar o desempenho de diferentes arquiteturas de redes neurais aplicadas à tarefa de tradução automática, com foco na contraposição entre uma abordagem baseada em Redes Neurais Recorrentes (RNN) com unidades GRU e a arquitetura Transformer. A hipótese central, de que o modelo Transformer superaria as abordagens tradicionais em termos de desempenho e capacidade de generalização devido ao seu mecanismo de atenção, foi categoricamente confirmada pelos resultados obtidos.

A implementação do protótipo com RNN-GRU, embora tenha alcançado uma acurácia de validação considerável de 93,89%, em um corpus limitado de inglês para francês, expôs as fragilidades intrínsecas da arquitetura. A dificuldade inicial de convergência na tradução do par português-inglês, atribuída à maior complexidade morfológica do português, e a incapacidade de generalizar para palavras fora do vocabulário de treinamento demonstraram suas limitações práticas. Tais desafios estão alinhados com as desvantagens teóricas das RNNs, como o processamento sequencial lento e a dificuldade em capturar dependências de longo alcance, criando um "gargalo de informação".

Em contrapartida, a abordagem baseada no modelo Transformer pré-treinado (mBART-50) apresentou um desempenho elevado. As traduções geradas para o par inglês-francês alcançaram métricas de avaliação robustas, como BLEU de 0.8687, METEOR de 0.9329 e chrF de 91.54, indicando altíssima fidelidade, fluência e qualidade estrutural. O sucesso do Transformer deriva de suas inovações arquitetônicas, notadamente o mecanismo de autoatenção, que permite o processamento paralelo e uma análise global da sentença, superando eficazmente as limitações das RNNs.

É importante ressaltar que todo o desenvolvimento prático, incluindo a implementação das técnicas e o treinamento dos modelos neurais, foi conduzido no ambiente do Google Colab, conforme mencionado no Capítulo 3. Essa escolha, embora conveniente, impôs limitações significativas, principalmente no que tange à quantidade de treinamento que pôde ser executada. Em diversas ocasiões, o limite de uso de recursos computacionais da plataforma foi atingido, o que restringiu a exploração de um maior número de épocas de treinamento ou o uso de configurações mais complexas que demandariam maior poder de processamento.

Portanto, este estudo conclui que a arquitetura Transformer não apenas representa o estado da arte na Tradução Automática Neural, mas sua eficácia, especialmente quando se utiliza modelos pré-treinados, oferece uma solução mais robusta, escalável e qualitativamente superior para os desafios da tradução entre idiomas.

Com base nos aprendizados e limitações deste trabalho, diversas direções para pesquisas futuras podem ser traçadas:

- Treinamento do Transformer em Corpus Restrito: Realizar um comparativo mais direto entre as arquiteturas, treinando um modelo Transformer do zero com o mesmo conjunto de dados utilizado para o modelo RNN-GRU. Isso permitiria isolar os ganhos de desempenho da arquitetura em si, separando-os dos benefícios do pré-treinamento em larga escala.

- Retomada da Tradução com o Idioma Português: Aplicar a arquitetura Transformer, que se mostrou superior, para a tarefa original de tradução do português para o inglês. Isso permitiria validar a hipótese em um cenário de maior complexidade morfológica e sintática, que foi o desafio inicial para o modelo RNN.

- Avaliação Humana da Qualidade: Complementar as métricas automáticas de avaliação (BLEU, METEOR, etc.) com uma análise de avaliação humana. A utilização de métricas de adequação e fluência forneceria uma perspectiva mais detalhada sobre a qualidade e a naturalidade das traduções geradas por ambos os modelos.

- Aplicação de Técnicas de Subword Tokenization: Investigar a aplicação de técnicas de tokenização em nível de subpalavra (como BPE ou WordPiece) no modelo RNN-GRU. Essa abordagem poderia mitigar o problema de palavras fora do vocabulário, potencialmente melhorando sua capacidade de generalização e permitindo uma comparação mais equitativa com o Transformer.

- Exploração de Variações da Arquitetura Transformer: Analisar o desempenho de outras variações e modelos pré-treinados baseados em Transformers que foram otimizados para pares de idiomas específicos ou para tarefas de tradução com recursos limitados, aprofundando o entendimento sobre o estado da arte da área.

Referências

ANDRADE, L. M. S.; BARROS, R. C.; SANTOS, M. A. B. dos. Processamento de linguagem natural (pln): Ferramentas e desafios. In: *Anais do Congresso Nacional de Inovação e Desenvolvimento Sustentável (CONIDIS)*. Salgueiro, PE: [s.n.], 2023.

AUSTIN, J. L. *How to Do Things with Words*. Oxford: Clarendon Press, 1962.

AZIZ, W.; SPECIA, L. Fully automatic compilation of a portuguese-english parallel corpus for statistical machine translation. In: *Proceedings of the STIL 2011 – Simpósio em Tecnologia da Informação e da Linguagem Humana*. Cuiabá, MT, Brazil: [s.n.], 2011.

BAHDANAU, D.; CHO, K.; BENGIO, Y. Neural machine translation by jointly learning to align and translate. In: *3rd International Conference on Learning Representations, ICLR 2015, Conference Track Proceedings*. San Diego, California, USA: [s.n.], 2015. Disponível em: <http://arxiv.org/abs/1409.0473>http://arxiv.org/abs/1409.0473.

BANERJEE, S.; LAVIE, A. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In: GOLDSTEIN, J. et al. (Ed.). *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*. Ann Arbor, Michigan, USA: Association for Computational Linguistics, 2005. Disponível em: <https://aclanthology.org/W05-0909>https://aclanthology.org/W05-0909.

BOLSHAKOV, I.; GELBUKH, A. *Computational Linguistics: Models, Resources and Applications*. Mexico City: Centro de Investigación en Computación, Instituto Politécnico Nacional, 2004. 186 p.

BROWN, P. F. et al. A statistical approach to language translation. In: *Proceedings of the 12th Conference on Computational Linguistics (COLING)*. Budapest, Hungary: Association for Computational Linguistics, 1988. p. 71–76. Disponível em: <http://portal.acm.org/citation.cfm?doid=991635-.991651>http://portal.acm.org/citation.cfm?doid=991635.991651.

CARL, M.; WAY, A. (Ed.). *Recent Advances in Example-Based Machine Translation*. Dordrecht: Springer Netherlands, 2003.

CASELI, H. M.; NUNES, M. G. V. (Ed.). *Processamento de Linguagem Natural: Conceitos, Técnicas e Aplicações em Português*. 3. ed. BPLN, 2024. ISBN 978-65-01-20581-6. Disponível em: <https://brasileiraspln.com/livro-pln/3a-edicao-/>https://brasileiraspln.com/livro-pln/3a-edicao/.

CASTILHO, S. et al. Approaches to human and machine translation quality assessment. In: *Translation Quality Assessment: From Principles to Practice*. Cham, Switzerland: Springer International Publishing, 2018, (Machine Translation: Technologies and Applications, v. 1). p. 9–38.

CASTILHO, S.; WAY, A. et al. Is neural machine translation the new state of the art? *The Prague Bulletin of Mathematical Linguistics*, v. 108, n. 1, p. 109–120, jun. 2017.

COPPIN, B. *Inteligência Artificial*. Rio de Janeiro, RJ: LTC, 2010.

DESHPANDE, A. K.; DEVALE, P. R. Natural language query processing using probabilistic context free grammar. *International Journal of Advances in Engineering & Technology*, Citeseer, v. 3, n. 2, p. 568–573, 2012. ISSN 2231-1963.

DODDINGTON, G. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In: *Proceedings of the Second International Conference on Human Language Technology Research (HLT '02)*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2002.

GRISHMAN, R. Information extraction. In: MITKOV, R. (Ed.). *The Oxford Handbook of Computational Linguistics*. Oxford: Oxford University Press, 2004. p. 545–559.

HAPKE, H.; HOWARD, C.; LANE, H. *Natural Language Processing in Action: Understanding, Analyzing, and Generating Text with Python*. New York: Manning, 2019.

HOVY, E. Text summarization. In: MITKOV, R. (Ed.). *The Oxford Handbook of Computational Linguistics*. Oxford: Oxford University Press, 2004. p. 583–598.

HUTCHINS, W. J. Information extraction. In: MITKOV, R. (Ed.). *The Oxford Handbook of Computational Linguistics*. Oxford: Oxford University Press, 2004. p. 545–559.

KHAYRALLAH, H.; KOEHN, P. On the impact of various types of noise on neural machine translation. In: *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*. Melbourne, Australia: Association for Computational Linguistics, 2018. Disponível em: <https://aclanthology.org/W18-2709>https://aclanthology.org/W18-2709.

KOEHN, P. *Neural Machine Translation*. Cambridge, UK: Cambridge University Press, 2020.

KOEHN, P.; OCH, F. J.; MARCU, D. Statistical phrase-based translation. In: *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology (NAACL '03)*. Association for Computational Linguistics, 2003. Disponível em: <http://dx.doi.org/10.3115-/1073445.1073462>http://dx.doi.org/10.3115/1073445.1073462.

LEE, S. et al. A survey on evaluation metrics for machine translation. *Mathematics*, MDPI, v. 11, n. 4, 2023.

LO, C.-k. Yisi – a unified semantic mt quality evaluation and estimation metric for languages with different levels of available resources. In: *Proceedings of the Fourth Conference on Machine Translation (WMT 2019), Volume 2: Shared Task Papers, Day 1*. Florence, Italy: Association for Computational Linguistics, 2019. Disponível em: <https://doi.org/10.18653/v1/w19-5358>https://doi.org/10.18653/v1/w19-5358.

LO, C.-k.; WU, D. Meant: An inexpensive, high-accuracy, semi-automatic metric for evaluating translation utility based on semantic roles. In: *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Portland, Oregon, USA: [s.n.], 2011. Disponível em: <https://aclanthology-.org/P11-1023/>https://aclanthology.org/P11-1023/.

MA, Q. et al. Blend: A novel combined mt metric based on direct assessment – casict-dcu submission to wmt17 metrics task. In: *Proceedings of the Second Conference on Machine Translation (WMT 2017)*. Copenhagen, Denmark: Association for Computational Linguistics, 2017. Disponível em: <https://doi.org/10.18653/v1/w17-4768>https://doi.org/10.18653/v1/w17-4768.

MOTA, C.; SANTOS, D.; RANCHHOD, E. Avaliação de reconhecimento de entidades mencionadas: princípio de harem. In: *Avaliação conjunta: um novo paradigma no processamento computacional da língua portuguesa*. [S.l.: s.n.], 2007. p. 161–175.

NADEAU, D. *Semi-Supervised Named Entity Recognition: Learning to Recognize 100 Entity Types with Little Supervision*. Tese (Ph.D. thesis) — University of Ottawa, Ottawa, Canada, 2007.

NAGAO, M. A framework of a mechanical translation between japanese and english by analogy principle. In: NIRENBURG, S.; SOMERS, H. L.; WILKS, Y. A. (Ed.). *Readings in Machine Translation*. Cambridge, MA: The MIT Press, 1984.

NIRENBURG, S. Knowledge and choices in machine translation. In: NIRENBURG, S. (Ed.). *Machine Translation – Theoretical and Methodological Issues*. Cambridge: Cambridge University Press, 1989. p. 1–15.

NUGUES, P. M. *An Introduction to Language Processing with Perl and Prolog: An Outline of Theories, Implementation, and Application with Special Consideration of English, French, and German*. New York: Springer-Verlag, 2006. 514 p.

O'BRIEN, S. Towards predicting post-editing productivity. *Machine Translation*, Springer, v. 25, p. 197–215, 2011.

PAPINENI, K. et al. Bleu: A method for automatic evaluation of machine translation. In: *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics (ACL '02)*. USA: Association for Computational Linguistics, 2002. Disponível em: <https://doi.org/10.3115/1073083.1073135>https://doi.org/10.3115/1073083.1073135.

POPOVIć, M. chrf: Character n-gram f-score for automatic mt evaluation. In: *Proceedings of the Tenth Workshop on Statistical Machine Translation*. Lisbon, Portugal: Association for Computational Linguistics, 2015. Disponível em: <https://aclanthology.org/W15-3049>https://aclanthology.org/W15-3049.

REI, R.; FARINHA, A.; MARTINS, A. F. T. Comet: A neural framework for mt evaluation. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, 2020. Disponível em: <https://aclanthology.org/2020.emnlp-main-.213>https://aclanthology.org/2020.emnlp-main.213.

SANTOS, D. Introdução ao processamento de linguagem natural através das aplicações. In: RANCHHOD, E. (Ed.). *Tratamento das Línguas por Computador. Uma introdução à linguística computacional e suas aplicações*. Lisboa: Caminho, 2001. p. 229–259. Disponível em: <http://www.linguateca.pt/Diana/download/Santos2001Aplicacoes-.pdf>http://www.linguateca.pt/Diana/download/Santos2001Aplicacoes.pdf.

SANYAL, H.; SHUKLA, S.; AGRAWAL, R. Natural language processing technique for generation of sql queries dynamically. In: *2021 6th International Conference for Convergence in Technology (I2CT)*. Pune, India: IEEE, 2021. p. 1–6.

SEARLE, J. R. *Speech Acts: An Essay in the Philosophy of Language*. Cambridge: Cambridge University Press, 1969.

SELLAM, T.; DAS, D.; PARIKH, A. P. Bleurt: Learning robust metrics for text generation. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL 2020)*. Online: Association for Computational Linguistics, 2020. Disponível em: <https://doi.org/10.18653/v1/2020.acl-main-.704>https://doi.org/10.18653/v1/2020.acl-main.704.

SHIMANAKA, H.; KAJIWARA, T.; KOMACHI, M. Machine translation evaluation with bert regressor. *arXiv preprint arXiv:1907.12679*, 2019. Disponível em: <https:/-/arxiv.org/abs/1907.12679>https://arxiv.org/abs/1907.12679.

SILVA, B. C. D. da. *A Face Tecnológica dos Estudos da Linguagem: o Processamento Automático das Línguas Naturais*. 272 p. Tese (Tese de Doutorado) — Universidade Estadual Paulista, Araraquara, SP, 1996.

SLOCUM, J. A survey of machine translation: Its history, current status, and future prospects. In: SLOCUM, J. (Ed.). *Machine Translation Systems*. Cambridge: Cambridge University Press, 1985. p. 1–41.

SNOVER, M. G. et al. A study of translation edit rate with targeted human annotation. In: *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas: Technical Papers (AMTA 2006)*. Cambridge, Massachusetts, USA: [s.n.], 2006. Disponível em: <https://aclanthology.org/2006.amta-papers.25-/>https://aclanthology.org/2006.amta-papers.25/.

SOCHER, R. et al. Semantic compositionality through recursive matrix-vector spaces. In: *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*. [S.l.: s.n.], 2012. p. 1201–1211.

SOMERS, H. Machine translation: Latest developments. In: MITKOV, R. (Ed.). *The Oxford Handbook of Computational Linguistics*. Oxford: Oxford University Press, 2004. p. 512–528.

STANOJEVIć, M.; SIMA'AN, K. Beer: Better evaluation as ranking. In: *Proceedings of the Ninth Workshop on Statistical Machine Translation, WMT@ACL 2014*. Baltimore, Maryland, USA: Association for Computational Linguistics, 2014. Disponível em: <https://doi.org/10.3115/v1/w14-3354>https://doi.org/10.3115/v1/w14-3354.

SU, K.-Y.; WU, M.-W.; CHANG, J.-S. A new quantitative quality measure for machine translation systems. In: *Proceedings of the 14th Conference on Computational Linguistics*. Association for Computational Linguistics, 1992. Disponível em: <http://dx.doi.org/10.3115/992133.992137>http://dx.doi.org/10.3115/992133.992137.

TZOUKERMAN, E.; KLAVANS, J. L.; STRZALKOWSKI, T. Information retrieval. In: MITKOV, R. (Ed.). *The Oxford Handbook of Computational Linguistics*. Oxford: Oxford University Press, 2004. p. 529–544.

VASWANI, A. et al. Attention is all you need. In: *Advances in Neural Information Processing Systems 30 (NeurIPS 2017)*. Curran Associates, Inc., 2017. Disponível em: <https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html>https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

VIEIRA, R.; LIMA, V. L. S. Lingüística computacional: Princípios e aplicações. In: NEDEL, L. (Ed.). *Anais da IX Escola de Informática da SBC-Sul*. Passo Fundo, Maringá, São José: SBC-Sul, 2001. p. 27–61.

YUAN, W.; NEUBIG, G.; LIU, P. Bartscore: Evaluating generated text as text generation. In: *Advances in Neural Information Processing Systems 34 (NeurIPS 2021)*. Virtual Conference: Curran Associates, Inc., 2021. Disponível em: <https://proceedings.neurips.cc/paper/2021/hash/e4d2b6e6fdeca3e60e0f1a62fee3d9dd-Abstract.html>https://proceedings.neurips.cc/paper/2021/hash/e4d2b6e6fdeca3e60e0f1a62fee3d9dd-Abstract.html.

ZELENKO, D.; AONE, C.; RICHARDELLA, A. Kernel methods for relation extraction. *Journal of Machine Learning Research*, v. 3, n. Feb, p. 1083–1106, 2003.

ZHANG, T. et al. Bertscore: Evaluating text generation with bert. In: *8th International Conference on Learning Representations (ICLR 2020)*. Addis Ababa, Ethiopia: OpenReview.net, 2020. Disponível em: <https://openreview.net/forum?id=SkeHuCVFDr>https://openreview.net/forum?id=SkeHuCVFDr.

Apêndices

APÊNDICE A – Códigos

A.1 Apêndices - Materiais do autor

Este apêndice contém o código-fonte utilizado nas análises do trabalho.

```
import tensorflow as tf
import numpy as np
import collections
import re
import os

from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import GRU, Embedding, TimeDistributed,
    Dense, Dropout, Bidirectional
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.losses import sparse_categorical_crossentropy


english_path = 'https://raw.githubusercontent.com/projjal1/datasets/
    master/small_vocab_en.txt'
french_path = 'https://raw.githubusercontent.com/projjal1/datasets/
    master/small_vocab_fr.txt'

def load_data(path):
    """
    Carrega dados a partir de uma URL usando tf.keras.utils.get_file.
    Corre o: Usa o basename para garantir um nome de arquivo local
        nico e est vel.
    """
    file_name = os.path.basename(path)
    local_path = tf.keras.utils.get_file(file_name, origin=path, extract
        =False, cache_subdir='datasets')

    with open(local_path, "r", encoding='utf-8') as f:
        data = f.read()

    return data.split('\n')

def tokenize(x):
```

```
32      """Tokeniza senten as e retorna sequ ncias e o tokenizer."""
33      tokenizer = Tokenizer()
34      tokenizer.fit_on_texts(x)
35      return tokenizer.texts_to_sequences(x), tokenizer
36
37  def pad(x, length=None):
38      """Adiciona padding  s  sequ ncias."""
39      return pad_sequences(x, maxlen=length, padding='post')
40
41  def preprocess(x, y):
42      """Pr -processa as senten as de entrada e sa da."""
43      preprocess_x, x_tk = tokenize(x)
44      preprocess_y, y_tk = tokenize(y)
45
46      # pad com base no maior comprimento de sequ ncia (para o Encoder-
            Decoder)
47      max_len = max(len(s) for s in preprocess_y)
48      preprocess_x = pad(preprocess_x, length=max_len)
49      preprocess_y = pad(preprocess_y, length=max_len)
50
51      # Keras requer labels em 3 dimens es para
            sparse_categorical_crossentropy com TimeDistributed
52      preprocess_y = preprocess_y.reshape(*preprocess_y.shape, 1)
53
54      return preprocess_x, preprocess_y, x_tk, y_tk
55
56  def logits_to_text(logits, tokenizer):
57      """Converte logits de volta para texto."""
58      index_to_words = {id: word for word, id in tokenizer.word_index.
            items()}
59      index_to_words[0] = '<PAD>'
60      # palavra correspondente ao  ndice  com a maior probabilidade (
            argmax)
61      return ' '.join([index_to_words[prediction] for prediction in np.
            argmax(logits, 1)])
62
63  # --- 2. CARREGAMENTO E PR -PROCESSAMENTO DE DADOS ---
64
65  english_sentences = load_data(english_path)
66  french_sentences = load_data(french_path)
67
68  preproc_english_sentences, preproc_french_sentences, english_tokenizer,
        french_tokenizer = \
69      preprocess(english_sentences, french_sentences)
70
71  english_vocab_size = len(english_tokenizer.word_index) + 1
72  french_vocab_size = len(french_tokenizer.word_index) + 1
```

```
73 max_sequence_length = preproc_french_sentences.shape[1]
74
75 tmp_x = preproc_english_sentences
76 tmp_x = tmp_x.reshape((-1, max_sequence_length))
77
78 # --- 3. CONSTRU  O DO MODELO OTIMIZADO PARA ALTA ACUR CIA ---
79
80 def high_accuracy_model(input_shape, output_sequence_length,
       english_vocab_size, french_vocab_size):
81     """
82     Constr i um modelo Encoder-Decoder com Bidirectional GRU e Dropout.
83     """
84     learning_rate = 0.001
85
86     model = Sequential()
87
88     # Camada 1: Embedding
89     model.add(Embedding(english_vocab_size, 256, input_length=
           input_shape[1], input_shape=input_shape[1:]))
90
91     # Camada 2: Bidirectional GRU (ajuda a capturar contexto em ambas as
           dire   es)
92     # Aumentando unidades para 512
93     model.add(Bidirectional(GRU(512, return_sequences=True)))
94     model.add(Dropout(0.3)) # Regulariza   o
95
96     # Camada 3: GRU extra para maior profundidade e capacidade
97     model.add(GRU(512, return_sequences=True))
98     model.add(Dropout(0.3))
99
100    # Camada 4: TimeDistributed Dense (fun   o 'relu' como
           intermedi rio)
101    model.add(TimeDistributed(Dense(1024, activation='relu')))
102
103    # Camada 5: Dropout
104    model.add(Dropout(0.5))
105
106    # Camada 6: TimeDistributed Dense (Output, tamanho do vocabul rio
           franc s, 'softmax')
107    model.add(TimeDistributed(Dense(french_vocab_size, activation='
           softmax')))
108
109    # Compila   o
110    model.compile(loss=sparse_categorical_crossentropy,
111                  optimizer=Adam(learning_rate),
112                  metrics=['accuracy'])
113    return model
```

```
114
115 model = high_accuracy_model(
116     tmp_x.shape,
117     max_sequence_length,
118     english_vocab_size,
119     french_vocab_size)
120
121 model.summary()
122
123 # --- 4. TREINAMENTO DO MODELO OTIMIZADO ---
124
125 print("\n--- INICIANDO TREINAMENTO OTIMIZADO (100  POCAS ) ---")
126 # aumento de  pocas  para 100 e redu  o do batch size de 1024 para
    512.
127 history = model.fit(
128     tmp_x,
129     preproc_french_sentences,
130     batch_size=512,
131     epochs=25,
132     validation_split=0.2)
133
134 model.save('high_accuracy_model.h5')
135
136
137 # --- 5. FUN  O DE PREDI  O FINAL ---
138
139 def final_predictions(model, text):
140   """Faz a tradu  o de uma frase de entrada e imprime o resultado."""
141   clean_text = re.sub(r'[^\\w]', ' ', text.lower())
142
143   # Tokeniza  o e Padding
144   try:
145     sentence = [english_tokenizer.word_index[word] for word in
        clean_text.split() if word in english_tokenizer.word_index]
146   except KeyError as e:
147     print(f"Aviso: A palavra '{e}' foi removida ou n o est  no
        vocabul rio de treinamento.")
148     return
149
150   if not sentence:
151     print("Aviso: A frase de entrada n o cont m palavras conhecidas
        para tradu  o.")
152     return
153
154   sentence = pad_sequences([sentence], maxlen=max_sequence_length,
      padding='post')
155
```

```
156   # Predi   o
157   logits = model.predict(sentence)[0]
158
159   # Convers o de volta para texto
160   translation = logits_to_text(logits, french_tokenizer)
161
162   print(f"\nEntrada (Ingl s): {text}")
163   print(f"Tradu   o (Franc s): {translation.replace('<PAD>', '').strip
          ()}")
164
165 # --- 6. TESTE COM EXEMPLO DO USU RIO ---
166 print("\n--- TESTE DE PREDI   O ---")
167 final_predictions(model, "she liked the big green automobile")
```

Listing A.1 – Código em Python para RNN utilizado nas análises

```
1
2  # !pip install transformers sentencepiece torch -U -q
3
4  from transformers import MBartForConditionalGeneration,
       MBart50TokenizerFast
5  import torch
6
7  frase_en = input("Digite uma frase em ingl s para traduzir para o
       franc s: ")
8
9  model_name = "facebook/mbart-large-50-many-to-many-mmt"
10
11 tokenizer = MBart50TokenizerFast.from_pretrained(model_name)
12 model = MBartForConditionalGeneration.from_pretrained(model_name)
13
14 tokenizer.src_lang = "en_XX"
15
16 device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
17 model = model.to(device).half() if torch.cuda.is_available() else model.
       to(device)
18
19 inputs = tokenizer(frase_en, return_tensors="pt").to(device)
20
21 generated_tokens = model.generate(
22     **inputs,
23     forced_bos_token_id=tokenizer.lang_code_to_id["fr_XX"]
24 )
25
26 traducao_fr = tokenizer.batch_decode(generated_tokens,
       skip_special_tokens=True)[0]
27
28 print(f"\nFrase original: {frase_en}")
```

```
print(f"Tradu  o em franc s: {traducao_fr}")
```

Listing A.2 – Código em Python para Trasnformers utilizado nas análises