Insper

Programa Institucional de Bolsas de Iniciação Científica (PIBIC)

Ciência da Computação

LanGaze: Digitação por Olhar Potencializada por Modelos de Linguagem Generativos

Nome completo: Gustavo Victor Valente Braga e Souza

Orientador: Dr. Andrew Toshiaki Nakayama Kurauchi

São Paulo

2025

Programa Institucional de Bolsas de Iniciação Cientifica

Gustavo Victor Valente Braga e Souza
Ciência da Computação

LanGaze: Digitação por Olhar Potencializada por Modelos de Linguagem Generativos

Relatório final apresentado ao Programa Institucional de Bolsas de Iniciação Cientifica (PIBIC)

Orientador: Dr. Andrew Toshiaki Nakayama Kurauchi

Resumo

A digitação por olhar (eye typing) é uma tecnologia que permite o usuário digitar num teclado virtual usando apenas os olhos por meio de rastreio ocular (eye tracking). Para tornar essa interação mais fluida, sistemas baseados em decodificar trajetórias do olhar em palavras foram propostos. Neste trabalho, introduzimos o LanGaze, um método de decodificação de trajetórias oculares que integra um componente linguístico baseado em um modelo de linguagem à decodificação por tries invertidas, com o intuito de melhorar a usabilidade de sistemas de digitação com o olhar.

O sistema introduz duas inovações principais. A primeira é o uso de uma trie invertida, que indexa palavras a partir de seus sufixos. Essa estratégia está alinhada ao processo natural da interação: em gestos oculares, o término é explicitamente indicado pelo usuário, enquanto o início tende a conter ruído ou movimentos não intencionais. Assim, iniciar a busca pelo final do traçado, a parte mais confiável do gesto, reduz erros. O alinhamento entre a trajetória observada e as sequências de letras é realizado incrementalmente via Dynamic Time Warping (DTW).

A segunda inovação é a incorporação de um modelo de linguagem (GPT-2) como componente de modelagem linguística. Durante a decodificação, o conjunto de palavras candidatas gerado pela trie é filtrado pelo LLM, que utiliza o contexto das palavras já confirmadas para calcular as probabilidades condicionais de cada candidato. Em experimentos com um vocabulário de 60 mil palavras e 100 sessões de testes reais, o LanGaze alcançou acurácia Top-5 de até 96%. A importância do componente de linguagem foi validada por um estudo de ablação, que quantificou o impacto do LLM na decodificação e revelou um ganho de 6 pontos percentuais na precisão do sistema. Ao nosso conhecimento, esse é o primeiro sistema a combinar a decodificação por trajetórias de olhar com probabilidades de LLMs

Palavras-chave: digitação por olhar; entrada de texto; gesto de digitação; LLMs; modelos generativos

Abstract

Eye typing is a technology that allows the user to type on a virtual keyboard using only their eyes through eye tracking. To make this interaction more fluid, systems based on decoding gaze trajectories into words have been proposed. In this work, we introduce **LanGaze**, a method for decoding gaze trajectories that integrates a linguistic component based on a language model into inverted-trie decoding, with the aim of improving the usability of eye-typing systems.

The system introduces two main innovations. The first is the use of an inverted trie, which indexes words from their suffixes. This strategy is aligned with the natural process of interaction: in gaze gestures, the ending is explicitly indicated by the user, while the beginning tends to contain noise or unintentional movements. Thus, starting the search from the end of the trace—the most reliable part of the gesture—reduces errors. The alignment between the observed trajectory and the sequences of letters is performed incrementally via Dynamic Time Warping (DTW).

The second innovation is the incorporation of a language model (GPT-2) as a linguistic modeling component. During decoding, the set of candidate words generated by the trie is filtered by the LLM, which uses the context of the already confirmed words to calculate the conditional probabilities of each candidate. In experiments with a vocabulary of 60,000 words and 100 real test sessions, LanGaze achieved Top-5 accuracy of up to 96%. The importance of the language component was validated by an ablation study, which quantified the impact of the LLM on decoding and revealed a 6-percentage-point gain in system accuracy. To our knowledge, this is the first system to combine gaze-trajectory decoding with LLM probabilities.

Keyword: eye typing; text input; gesture input; word gesture; LLMs; generative models

Lista de Siglas

LLM – Large Language Model

DTW – Dynamic Time Warping

WPM – Words Per Minute

Sumário

1. Introdução 11

2. Digitação Por Olhar 12

2.1. Dispositivos de Rastreio Ocular 12

2.1.1. Calibração do Rastreio Ocular 12

2.2. Movimentos do Olhar 13

2.3. Sistemas de Digitação por Olhar 14

2.3.1. Sistemas dwell-based 14

2.3.2. Sistemas dwell-free 15

3. Metodologia 16

3.1. Escopo do trabalho 16

3.2. Gerador de dados sintéticos 17

3.2.1. Geração de Dados Sintéticos 17

3.2.2 Implementação da Geração de Dados Sintéticos 17

3.2.3. Avaliação dos Dados Gerados 19

3.3. Coleta de Dados 20

3.3.1. Materiais e Configuração Experimental 20

3.3.2. Procedimento Experimental 20

3.4. Avaliação da Qualidade dos Dados Gerados 21

3.5. Desenvolvimento do Classificador 22

3.5.1. Arquitetura do Modelo 23

3.5.2. Tokenização do Vocabulário 23

3.5.3. Acurácias top-1 e top-4 23

3.5.4 Experimentos e Resultados 23

3.5.5. Limitações da Abordagem de Classificação e Nova Direção 26

3.6. LanGaze 27

3.6.1. Limitações de Sistemas Clássicos 27

3.6.2. Representação por Sufixos: A Trie Invertida 28

3.6.3. Decodificação com Alinhamento Temporal Reverso 30

3.6.4. Incorporação de Contexto com Modelo de Linguagem 31

3.6.5. Combinação Ponderada e Balanceamento de Escala 32

4. Resultados e Discussões 34

5. Considerações Finais 38

Referências **39**
Apêndice A **41**

1. Introdução

A digitação por olhar (eye typing) é uma modalidade de digitação, em que o usuário insere texto com o movimento do olhar. O eye typing é uma modalidade importante para casos em que o usuário precisa digitar, porém não tem a disponibilidade de um teclado físico, como por exemplo, em ambientes de realidade aumentada. Outro caso de uso para a digitação com o olhar é o para indivíduos com deficiências motoras severas, que não são capazes de digitar em teclados físicos.

Ao longo dos anos, alguns métodos foram desenvolvidos para resolver o problema de: como digitar com o movimento do olhar. Uma das categorias de métodos mais estudados são os que dependem de um dwell time. Nesse tipo de sistema, o usuário deve olhar para cada tecla que deseja digitar por um período curto de tempo (600ms) para confirmar a digitação da mesma, e vai formando palavras por meio de sucessivas confirmações de teclas. Esse método de digitação traz por sua natureza, um tempo de digitação muito alto, já que o tempo de digitação aumenta proporcionalmente com o tamanho da entrada de texto – cada letra terá um tempo de confirmação -.

Buscando mitigar esse efeito, outra categoria de sistemas de digitação por olhar estudada é a chamada de dwell-free, que tem como objetivo eliminar totalmente ou de forma parcial os tempos de confirmação para a digitação das teclas. Uma das abordagens é a de adaptar a digitação por gestos, já conhecida em teclados de sistemas de smartphones touchscreen. A abordagem de digitação por gestos consiste no usuário desenhar uma forma geométrica arrastando ao longo do teclado que passe por cima das teclas que ele queira digitar para formar a palavra. O EyeSwipe (Kurauchi et al., 2016) adaptou esse método de digitação por gestos para o eye typing, permitindo que os usuários digitem as palavras formando os gestos com as trajetórias do olhar. O GlanceWriter (Cui et al., 2023) também implementou um sistema de digitação por gestos com trajetórias do olhar, com uma abordagem de decodificação diferente da utilizada no EyeSwipe.

Nesse contexto, o uso de métodos baseados unicamente em trajetórias visuais pode não ser suficiente para garantir precisão e fluidez ideais na digitação por olhar. Para resolver esse problema, torna-se necessária a integração de informações adicionais que complementem as trajetórias visuais e reduzam as ambiguidades presentes nas predições feitas exclusivamente com base nas trajetórias gestuais.

Neste trabalho, propomos uma nova arquitetura híbrida para sistemas de digitação por olhar. Nossa arquitetura integra um decodificador baseado nas trajetórias do olhar com um modelo de linguagem pré-treinado (LLM), capaz de capturar o contexto linguístico anterior e assim reduzir significativamente as ambiguidades associadas à interpretação das trajetórias visuais. A combinação das probabilidades produzidas pelo decodificador do gesto com aquelas produzidas pelo modelo de linguagem é feita por meio de uma estratégia que combina matematicamente a informação vindo das fontes do decodificador do gesto e do modelo de linguagem. Além disso, avaliamos a influência dos parâmetros da combinação entre os componentes de decodificação do gesto, e o linguístico, visando identificar uma configuração ótima da combinação.

2. Digitação Por Olhar

A digitação por olhar (eye typing) é uma modalidade de digitação na qual o usuário insere texto por meio do movimento dos olhos. Essa modalidade é especialmente relevante em situações em que o usuário não pode utilizar um teclado físico, como em ambientes de realidade aumentada, ou em casos de indivíduos com deficiências motoras severas que impossibilitam o uso de teclados tradicionais.

2.1. Dispositivos de Rastreio Ocular

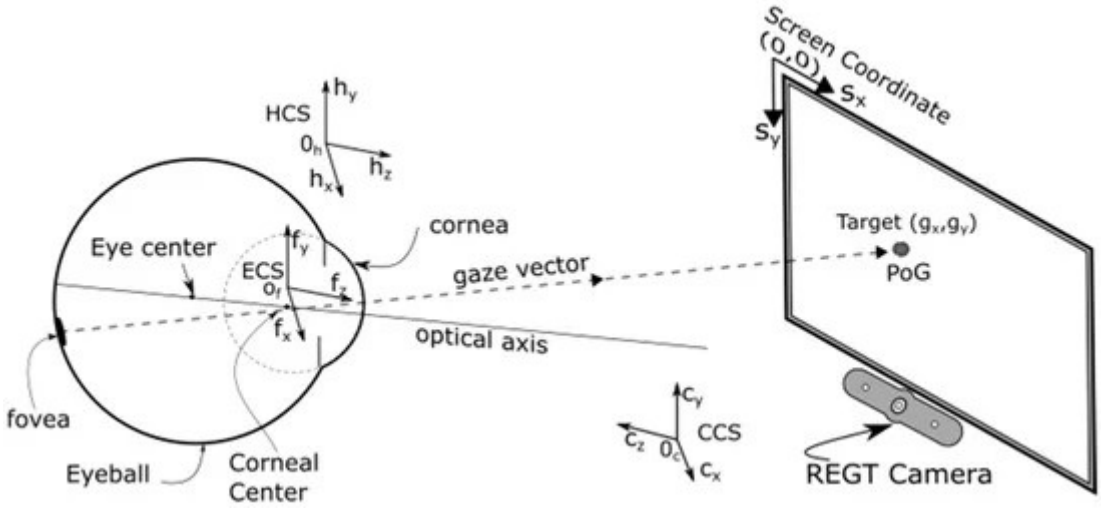
Sistemas de rastreamento ocular são tecnologias projetadas para capturar, analisar e interpretar o ponto específico da tela para onde o usuário direciona seu olhar em determinado instante. A tecnologia mais utilizada emprega iluminação infravermelha próxima ao eixo óptico da câmera, produzindo o fenômeno conhecido como Bright pupil, em que a pupila serve como referência estável para rastreamento dos olhos (Morimoto; Mimica, 2005). A precisão desse rastreamento é fortemente influenciada por características fisiológicas individuais e pela necessidade de calibração personalizada.

No entanto, o uso de rastreamento ocular em sistemas interativos enfrenta um desafio significativo conhecido como "toque de Midas" (Midas Touch). Este termo, descrito inicialmente por (Jacob, 1991), refere-se ao problema de distinguir movimentos oculares normais daqueles que representam comandos intencionais. Em outras palavras, sistemas de interação baseados em olhar podem ativar comandos indesejados devido à dificuldade em diferenciar um olhar casual de uma intenção real do usuário.

2.1.1. Calibração do Rastreio Ocular

Dado que a fisiologia ocular varia significativamente entre indivíduos, sistemas de rastreamento ocular geralmente precisam de calibração individualizada para funcionar com precisão.

Figura 1 – Representação de um sistema de rastreio de olhar.



Fonte: (Sheihu et. al., 2021)

Durante a calibração, o participante fixa o olhar em pontos específicos, e conhecidos na tela, permitindo que o sistema relacione a posição do olhar ao espaço visual exibido. Tipicamente, esse procedimento utiliza modelos polinomiais ou afins para converter as coordenadas brutas capturadas brutas capturadas pelas câmeras infravermelhas em coordenadas ajustadas que representam com maior fidelidade o ponto real de fixação (Morimoto; Mimica, 2005).

2.2. Movimentos do Olhar

O movimento ocular humano não ocorre de forma contínua ou linear, independentemente da tarefa visual realizada. Durante qualquer tipo de atividade, os olhos avançam por meio de pequenos saltos rápidos entre pontos de interesse. Esses movimentos rápidos são conhecidos como sacadas, enquanto os períodos de relativa estabilidade do olhar, durante os quais as informações visuais são capturadas e processadas, são denominados fixações.

As fixações são períodos curtos nos quais o olhar permanece relativamente estável, permitindo que o cérebro processe as informações contidas naquele ponto de fixação. Já as sacadas, são movimentos rápidos que levam o olhar rapidamente de uma posição de fixação para outra.

Durante uma sacada, o olho acelera rapidamente desde o ponto inicial de fixação até atingir uma alta velocidade intermediária, desacelerando em seguida até chegar ao próximo ponto de fixação. Esse movimento é realizado de forma balística, ou seja, após iniciado, ele não pode ser ajustado ou interrompido até que seja concluído. A velocidade máxima alcançada nesses movimentos pode ser muito alta, frequentemente atingindo centenas de graus por segundo. Durante os movimentos de sacada, o cérebro não processa as informações vistas pelo olho.

Outro fenômeno na modelagem da trajetória do olhar é o aumento dos erros de projeção nas regiões periféricas do campo visual. Isso ocorre por conta das condições que os softwares de rastreio de olhar utilizam para estimar a posição que o usuário está olhando na tela.

A origem do fenômeno de distorção está nas simplificações dos modelos de eye tracking, que frequentemente assumem que o olho rotaciona como uma esfera perfeita em torno de um eixo fixo (Morimoto; Mimica, 2005). Na realidade, o globo ocular apresenta movimento mais complexo devido a três fatores principais: o deslocamento do centro de rotação durante movimentos amplos, a forma não perfeitamente esférica do olho, e as variações na refração da córnea em diferentes ângulos. Esses erros acumulados resultam em uma incerteza total maior nas áreas extremas laterais do campo de visão, como demonstrado em (Mardanbegi; Kurauchi; Morimoto, 2018).

2.3. Sistemas de Digitação por Olhar

Um dos principais desafios no desenvolvimento de um sistema de um sistema de digitação com olhar é o toque de Midas. Especialmente na tarefa de digitação, resolver esse desafio se torna algo desafiador, por que é necessário diferenciar quando o usuário olha para as teclas procurando posições, ou lendo as letras, de quando o usuário tem a intenção de realizar a digitação.

Existem dois tipos principais de métodos para resolver esse desafio no contexto da digitação por olhar: métodos dwell-based e métodos dwell-free.

2.3.1. Sistemas dwell-based

Os sistemas dwell-based, dependem de uma curta espera para a digitação de cada tecla. O usuário deve olhar por um período fixo de tempo para uma tecla que deseja digitar, e após esse curto período, a digitação da tecla é realizada (Majaranta et. al., 2007).

Esse tempo que o usuário precisa continuar fixando para a tecla até que a digitação dela seja confirmada é o dwell time. dwell time típico é entre 400 e 1000 milisegundos.

Métodos dwell-based trazem, por natureza, um tempo maior necessário para digitação. Para tentar diminuir esse tempo necessário, o dwell time poderia ser diminuído, mas essa ação aumentaria os efeitos do toque de Midas, aumentando a incidência de digitações acidentais, fato observado em (Majaranta; Ahola; Spakov, 2009).

2.3.2. Sistemas dwell-free

Os sistemas dwell-free tentam eliminar, ou evitar o número de tempos de espera para confirmação de digitação de teclas. Uma das abordagens mais estudadas para o desenvolvimento desse tipo de sistema, é adaptar a digitação por gestos, já conhecida amplamente em dispositivos smartphones. A digitação por gestos, consiste originalmente do usuário arrastar o dedo ao longo do teclado, formando uma forma geométrica, em que essa forma o usuário faz passar por cima das teclas da palavra que ele quer digitar. A tarefa de um sistema desse tipo é a de decodificar as coordenadas desse gesto feito pelo usuário em uma palavra real.

O EyeSwipe (Kurauchi et al., 2016) adapta essa ideia de digitação por gestos para o domínio de trajetórias visuais. O principal problema, segundo o estudo, para adaptar o método de digitação por gestos diretamente para o domínio do olhar, é que quando o usuário arrasta o dedo para desenhar uma forma geométrica num teclado virtual, os pontos em que o gesto começou e terminou são bem determinados por conta da própria natureza do touchscreen do celular. Em sistemas de digitação por olhar, é difícil determinar os pontos exatos em que o usuário começou a ter a intenção de olhar ao longo das teclas para desenhar a forma geométrica com a trajetória do seu olhar, assim como também é difícil determinar em que momento exato o usuário desejou terminar o desenho daquela forma. O EyeSwipe supera esse desafio definindo dois dwell times, com uma técnica que foi nomeada de reverse-crossing, em que o usuário olha para a tecla que representa a primeira letra da palavra que ele quer digitar por um curto período de tempo, e faz um movimento de subida e descida com o olhar em uma interface interativa para indicar que está começando o gesto. O mesmo processo se repete para a última letra.

O GlanceWriter (Cui et al., 2023) traz a mesma hipótese do EyeSwipe, de adaptar a digitação por gestos para o domínio de trajetórias visuais, porém utilizando uma abordagem diferente para a decodificação da trajetória visual para palavras. Um dos objetivos do GlanceWriter é desenvolver o sistema de forma com que ele não dependa de nenhum dwell time, ou seja, eliminar o tempo de confirmação das primeiras e últimas teclas, presente no EyeSwipe.

O SkiMR (Hu; Dudley; Krinstensson, 2024), implementa um sistema de digitação dwell-free num ambiente de realidade aumentada. O SkiMR utiliza um transdutor de estados finito (Ouyang et al., 2017), Kristensson (2018) para decodificar as trajetórias visuais em palavras, atingindo uma taxa de 12 palavras por minuto (WPM).

3. Metodologia

3.1. Escopo do trabalho

O objetivo deste estudo é desenvolver um modelo de aprendizagem computacional capaz de decodificar trajetórias de olhar em palavras. Para isso, faz-se necessário um conjunto de dados que represente, de forma realista, a dinâmica do olhar durante a digitação. No entanto, um dos principais desafios é a escassez de bases públicas que registrem eventos de digitação utilizando rastreamento ocular.

Diante dessa limitação de dados, considerou-se a possibilidade de utilizar um gerador de dados sintéticos. A motivação é viabilizar o treinamento inicial e a experimentação de modelos de aprendizagem de máquina, mesmo na ausência de um grande conjunto de dados reais. Entretanto, a qualidade desses dados sintéticos precisa ser avaliada, de modo que seja possível inferir o quão bem eles se aproximam de situações reais de digitação.

No início, foi investigada a utilização do modelo WordGesture GAN (Chu et al., 2023), que mapeia palavras (strings) a trajetórias contínuas no contexto de digitação por arraste em teclados de celulares. A ideia era adaptar essa abordagem ao domínio de rastreamento ocular. Porém, foi concluído que, devido ao reduzido volume de dados disponíveis para treinar o modelo adversarial, a aplicação do WordGesture GAN se tornaria inviável neste momento.

Assim, duas frentes de trabalho foram desenvolvidas de forma paralelamente:

- Geração de dados sintéticos: Implementação de um sistema não dependente de aprendizado que simula a digitação com base em heurísticas sobre fixações, sacadas e ruídos característicos do movimento do olho humano;
- Coleta de dados reais: Desenvolvimento de um software para captura do olhar durante a digitação, a fim de criar um conjunto de dados de referência para avaliar a qualidade dos dados sintéticos.

Tendo em vista a necessidade de validar o gerador de dados sintéticos, também foi proposta uma métrica de comparação entre as trajetórias sintéticas e as trajetórias reais coletadas. Dessa forma, é possível investigar até que ponto o gerador reproduz padrões verossímeis de digitação. Finalmente, com um conjunto de dados disponível, se dá início ao desenvolvimento do modelo classificador de trajetórias, cujo propósito é decodificar as palavras escritas a partir dos registros de olhar.

Em suma, o trabalho se divide em quatro frentes principais:

- Geração de dados sintéticos;
- Coleta de dados reais para validação;
- Validação dos dados sintéticos, por meio de métricas de comparação entre gestos reais e simulados;
- Desenvolvimento do decodificador de trajetórias visuais

3.2. Gerador de dados sintéticos

3.2.1. Geração de Dados Sintéticos

Conforme discutido anteriormente, a utilização do modelo WordGestureGAN (Chu et. al., 2023) para a geração de trajetórias oculares se mostrou inviável devido à escassez de dados reais para o treinamento de um modelo adversarial, que contém uma natureza intrínseca de instabilidade durante o treinamento. Dessa forma, se optou por uma solução que não fosse dependente de um processo de aprendizagem baseado em dados, mas sim em heurísticas deduzidas a partir de características fisiológicas e comportamentais do olhar humano.

A ideia central consiste em simular como os usuários movimentam seus olhos ao buscar as teclas no teclado virtual, reproduzindo fixações — em que o usuário mira determinada tecla para “confirmar” visualmente o caractere — e sacadas — deslocamentos rápidos do olhar entre duas fixações. Além disso, é desejado incorporar ruídos e eventuais erros de direcionamento, a fim de tornar as trajetórias mais realistas. Uma fixação típica pode durar algumas centenas de milissegundos, enquanto as sacadas são movimentos balísticos que levam apenas poucos milissegundos. Em contextos de digitação, o usuário tende a alternar fixações breves em teclas específicas e sacadas rápidas para transitar de uma tecla à próxima.

3.2.2 Implementação da Geração de Dados Sintéticos

Diante dos desafios enfrentados na geração de dados utilizando métodos generativos, buscou-se desenvolver um método alternativo para a geração dos dados sintéticos.

A decisão para o sistema, foi a de modelar os gestos como uma série de heurísticas que pautam os movimentos do olhar, de forma que não dependa de um processo de aprendizado, porém que ainda capture variedade e confiabilidade em comparação a dados reais.

A abordagem adotada inicia-se pelo posicionamento das teclas do teclado virtual em um plano bidimensional seguindo o layout padrão QWERTY, com espaçamento horizontal fixado em 0.4 vezes o tamanho da tecla e espaçamento vertical de 1.5 vezes o tamanho das teclas. Uma tecla específica, denominada tecla de controle, desempenha papel semelhante ao botão de espaço em teclados físicos, funcionando como ponto inicial e final das trajetórias oculares, marcando claramente o início e término da digitação de cada palavra.

Durante o processo de digitação, o usuário inicia sempre a trajetória ocular fixando-se na tecla de controle. Em seguida, o olhar se move para cada tecla correspondente às letras da palavra alvo, realizando fixações estáveis intercaladas por movimentos rápidos, denominados sacadas. As fixações são modeladas utilizando uma distribuição gaussiana em torno do centro da tecla, com um desvio-

padrão (σ_{fix}) que representa as pequenas oscilações naturais do olhar. As sacadas são representadas como movimentos lineares entre as fixações, adicionando ruído gaussiano para capturar a variabilidade natural.

Adicionalmente, levando em conta que usuários reais frequentemente cometem pequenos erros ou desvios ocasionais ao procurar teclas específicas, o modelo inclui uma probabilidade (p_{erro}) de realizar fixações breves em teclas incorretas antes da fixação correta.

O método pode ser formalizado pelo Pseudocódigo 1, com os parâmetros de configuração descritos na Tabela 1.

Pseudocódigo 1 – Geração de trajetórias sintéticas.

Require: Palavra, posições das teclas, parâmetros (σ_{fix}, p_{erro}, n_{sac})
Ensure: Trajetória sintética
1: $trajetoria \leftarrow []$
2: $posicaoAnterior \leftarrow$ posição da tecla de controle
3: **for** cada letra na palavra **do**
4: $posicaoCorreta \leftarrow$ posição da tecla atual + ruído gaussiano(σ_{fix})
5: **if** $aleatorio() < p_{erro}$ **then**
6: $posicaoIncorreta \leftarrow$ posição de uma tecla incorreta aleatória + ruído
7: $trajetoria$.adiciona($posicaoIncorreta$)
8: $posicaoAnterior \leftarrow posicaoIncorreta$
9: **end if**
10: **for** $i = 1$ n_{sac} **do**
11: interpolar linearmente entre $posicaoAnterior$ e $posicaoCorreta$
12: adicionar ruído gaussiano às posições intermediárias
13: $trajetoria$.adiciona($posicaoIntermediaria$)
14: **end for**
15: $trajetoria$.adiciona($posicaoCorreta$)
16: $posicaoAnterior \leftarrow posicaoCorreta$
17: **end for**
18: Retornar à tecla de controle com sacadas intermediárias
19: $trajetoria$.adiciona($tecla\ de\ controle$) $trajetoria$

Fonte: Elaborado pelo autor.

Tabela 1 – Parâmetros do modelo de geração de trajetórias sintéticas.

Parâmetro	Descrição
σ_{fix}	Desvio-padrão do ruído gaussiano aplicado nas fixações, representando as oscilações naturais do olhar.
σ_{fix_center}	Desvio-padrão do ruído aplicado na posição inicial e final (tecla de controle).
p_{erro}	Probabilidade de ocorrer uma fixação breve em teclas incorretas antes da fixação correta.
n_{fix}	Número de pontos (fixações) gerados para cada tecla correta.
n_{sac}	Número de pontos intermediários simulados em cada movimento sacádico entre teclas.
n_{fix_erro}	Número de pontos gerados durante uma fixação incorreta.

Fonte: Elaborado pelo autor.

3.2.3. Avaliação dos Dados Gerados

Para validar o método de geração de dados, foi necessária uma métrica que permitisse a comparação de dados sintéticos e dados reais, garantindo que os dados gerados tenham uma semelhança com as amostras reais, mas ainda assim, não sejam completamente idênticas.

A avaliação se fundamenta na Distância de Wasserstein, baseada no método proposto em (Chu et. al., 2023). A Distância de Wasserstein, também conhecida como métrica de Earth Mover's Distance (EMD), quantifica o esforço mínimo necessário para transformar uma distribuição de probabilidade em outra, considerando não só a diferença entre os valores, mas também a distância geométrica entre os pontos.

O procedimento de cálculo se inicia com a definição de uma unidade de comparação. Para cada palavra P no conjunto de dados, foram estabelecidos pares de comparação entre gestos sintéticos e gestos reais. Em seguida, para cada par de gestos, foram computadas as distâncias L2 (euclidiana) entre os pontos correspondentes na sequência temporal. Formalmente, para um gesto sintético x e um gesto real y, ambos com n pontos, a distância L2 é dada por:

$$D_{L2}(x, y) = \sqrt{\sum_{i=0}^{n} \left|\left|x(i) - y(i)\right|\right|^2}$$

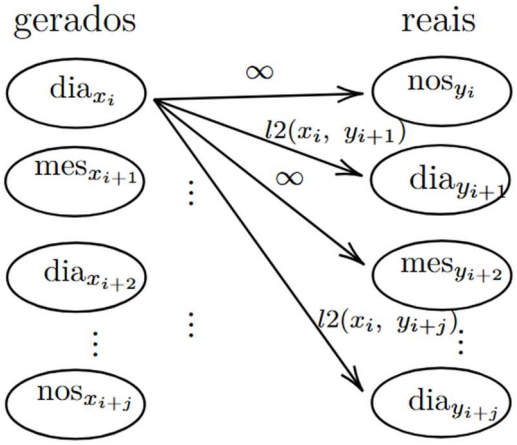
O cálculo da distância de Wasserstein é modelado como um problema de correspondência em um grafo bipartido completo. Nesse grafo, os vértices do primeiro conjunto representam gestos gerados, enquanto os do segundo conjunto representam gestos reais. As arestas possuem pesos correspondentes às distâncias L2 calculadas, e as distâncias entre gestos associados à palavras distintas são atribuídas como infinito, impondo a restrição de que a comparação só ocorra entre gestos da mesma palavra, como ilustrado na Figura 2.

Ao aplicar o algoritmo de transporte ótimo para determinar a correspondência que minimiza o custo total de transporte entre as distribuições de gestos sintéticos e reais, obtemos a Distância de Wasserstein como a média ponderada das distâncias L2 nas correspondências ótimas estabelecidas:

$$W(S, R) = \frac{1}{|M|} \sum_{(G_s, G_r) \in M} D_{L2}(G_s, G_r)$$

Em que M representa o conjunto de pares de gestos gerados e reais na correspondência ótima, e $|M|$ é o tamanho desse conjunto. A implementação do algoritmo está disponível neste arquivo. [disponível no Anexo A.1]

Figura 2 – Ilustração do algoritmo de comparação de gestos reais e sintéticos.



Fonte: Elaborado pelo autor.

3.3. Coleta de Dados

3.3.1. Materiais e Configuração Experimental

O processo experimental para a coleta dos dados reais foi realizado utilizando um sistema composto por um monitor, um rastreador ocular (eyetracker) e um software desenvolvido especificamente para essa tarefa. O eyetracker utilizado foi o modelo Tobii x2-60, escolhido devido à sua precisão, facilidade de uso e compatibilidade com o software desenvolvido. O experimento foi realizado em uma sala silenciosa e com iluminação controlada para diminuir possíveis variações relacionadas à iluminação durante a captura do olhar.

Os participantes foram acomodados em uma cadeira ajustável, posicionados a uma distância de 60 e 70cm do monitor. A altura da cadeira foi regulada de modo que a linha de visada dos participantes estivessem aproximadamente perpendiculares ao rastreador ocular. Entretanto, pequenas variações nessa posição foram permitidas intencionalmente, visando uma maior variabilidade natural no conjunto de dados coletado.

3.3.2. Procedimento Experimental

O primeiro experimento segue a seguinte estrutura: uma palavra ou frase será exibida na tela por um breve período para que o usuário a memorize. Após isso, um teclado virtual é exibido e o participante deve digitar o texto memorizado, olhando sequencialmente para as teclas correspondentes no teclado virtual. Para indicar a conclusão da digitação da palavra, o participante fixa o olhar por 2 segundos em uma área específica do teclado, que simula uma barra de espaço. Esse ciclo é repetido para cada palavra e frase do conjunto de dados.

Além disso, um protocolo de calibração do rastreador de olhar é iniciado a cada 20 palavras digitadas. O experimento é controlado por um software desenvolvido em PsychoPy (disponível neste repositório. [disponível no anexo A.2]) e os dados processados são armazenados em um banco de dados MongoDB. Para proteger a privacidade dos participantes, cada registro é associado a um identificador único gerado automaticamente, sem vinculação direta a dados pessoais. Cada sessão tem duração aproximada de 30 minutos, com a possibilidade de descanso a cada nova calibração. A quantidade de palavras digitadas por cada participante depende do tempo necessário para que eles se familiarizem com o sistema e com o processo de digitação. Essa metodologia foi desenvolvida para obter um mapeamento preciso entre a trajetória e a palavra-alvo, reduzindo prováveis ruídos e otimizando a qualidade dos dados.

3.4. Avaliação da Qualidade dos Dados Gerados

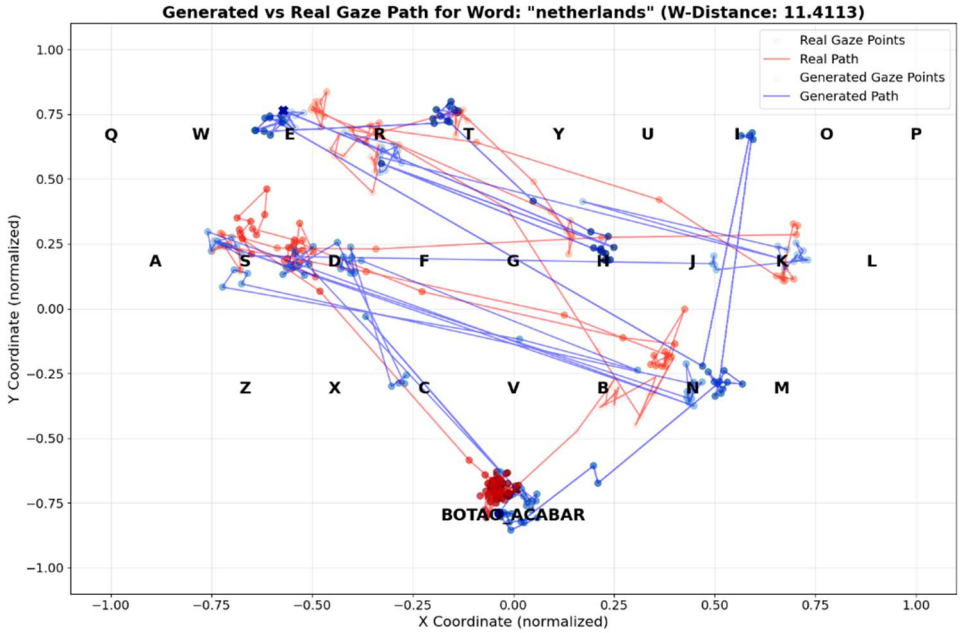
Uma coleta preliminar foi realizada entre os próprios pesquisadores para se ter uma análise do desempenho do gerador de dados sintéticos. Foram obtidas 230 amostras de palavras em inglês, seguindo o protocolo experimental descrito na sessão anterior. Ao aplicar o método descrito na sessão que tratava do gerador de dados sintéticos, obtemos o resultado de uma distância de Wasserstein média para todas as palavras do conjunto de dados de 10.40, com um desvio padrão de 1.39.

Na literatura, não estão disponíveis referências que tratem sobre comparações de dados sintéticos e reais para situações de digitação baseada em olhar, porém, para critério de comparação preliminar, usamos os resultados listados em (Chu et al., 2023). Os resultados se tratam de geradores de gestos sintéticos de arrastos em teclados virtuais de celulares.

O resultado obtido, supera o obtido no Style-Transfer GAN (Mehra et al., 2021), que atingia uma distância média de Wasserstein de 10.48, com 2.01 de desvio padrão.

Para obter uma confirmação visual, além das métricas atingirem trabalhos posteriores, na Figura 3.

Figura 3 – Exemplo de gesto gerado e real comparados.



Fonte: Elaborado pelo autor.

3.5. Desenvolvimento do Classificador

Após a confirmação preliminar da qualidade dos dados gerados, a atenção foi voltada para a tarefa principal: desenvolver um modelo para decodificar as trajetórias visuais em palavras.

O modelo é definido como um classificador, que tem como entrada features relacionadas às trajetórias visuais, e tem como objetivo prever a classe da palavra da respectiva trajetória.

Formalmente, o classificador é definido como um modelo supervisionado que recebe como entrada sequências temporais discretas que representam trajetórias visuais. Cada trajetória é composta por uma sequência de coordenadas bidimensionais, expressas como:

$$T = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}, (x_i, y_i) \in R^{\mathbb{2}}$$

Para garantir uma entrada homogênea ao modelo, todas as entradas foram ajustadas para terem um comprimento fixo de N = 170. A escolha desse valor se baseou na média da quantidade de pontos presentes nas amostras preliminares coletadas experimentalmente.

Dessa forma, o procedimento para o ajuste do comprimento das sequências segue duas estratégias:

- Caso uma trajetória possua comprimento $n < N$, a sequência é completada adicionando-se pontos $(0,0)$ ao final até que se atinja o comprimento N.

- Caso o comprimento seja $n < N$, realiza-se o truncamento dos pontos iniciais da sequência, preservando-se s últimos N pontos, que geralmente contêm informação mais relevante associada à fixação final.

  Métodos adicionais para determinar a quantidade ótima de pontos N, assim como técnicas alternativas para truncamento são temas futuros de estudo.

3.5.1. Arquitetura do Modelo

Para realizar a classificação das trajetórias foi adotada uma arquitetura baseada em redes convolucionais unidimensionais (CNN-1D). A escolha desse modelo fundamenta-se na capacidade das redes convolucionais em capturar padrões locais e temporais presentes nas trajetórias oculares. A arquitetura implementada consiste em três camadas convolucionais sequenciais, cada uma seguida por uma camada de normalização em lote (batch normalization) e uma função de ativação não-linear (ReLU). Após essas camadas convolucionais, é aplicada uma camada de pooling adaptativo, reduzindo a dimensão temporal da saída, seguida por três camadas totalmente conectadas intercaladas com camadas de dropout.

3.5.2. Tokenização do Vocabulário

O vocabulário utilizado no classificador foi tokenizado atribuindo-se um índice inteiro único para cada palavra distinta disponível, variando de 0 até $N - 1$, sendo N o tamanho total do vocabulário.

3.5.3. Acurácias top-1 e top-4

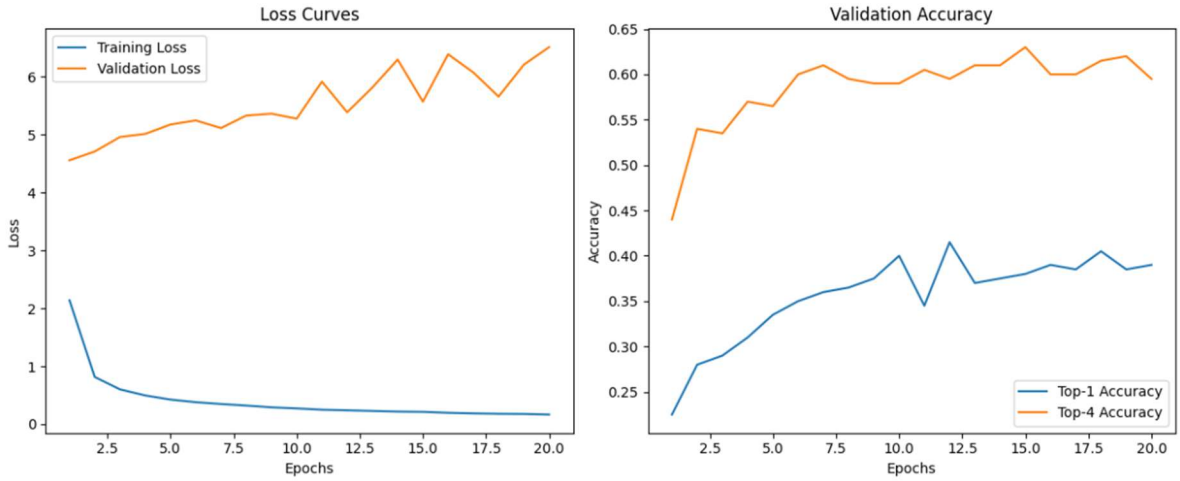
Para avaliar o desempenho do modelo, utilizam-se duas métricas de acurácia: a top-1 e top-4. A acurácia top-1 corresponde à definição padrão de acurácia, ou seja, a razão entre o número de previsões corretas e o total de previsões realizadas. Já a acurácia top-4 leva em conta as quatro classes que o modelo atribui maior probabilidade; se a classe correta estiver entre essas quatro, considera-se que essa previsão foi bem-sucedida.

3.5.4 Experimentos e Resultados

Para avaliar o modelo, primeiro foi testado o caso mais simples para treino: a entrada do modelo sendo a sequência de pontos da trajetória visual, e a saída sendo o índice do vocabulário da palavra correspondente à trajetória.

Para o treino, também foi estudada a forma com que os dados sintéticos são injetados durante o aprendizado do modelo. Uma razão ajustável entre dados sintéticos e reais é introduzida durante o processo de treino do modelo, enquanto o conjunto de dados de teste é sempre mantido apenas com dados reais. Como observado na Figura 4, a curva de validação tem uma grande diferença em relação à de treino, demonstrando um comportamento de overfitting. Analisando a curva de acurácias, foi obtida uma acurácia top-1 e de 39%, e uma acurácia top-4 de 59%.

Figura 4 – Curvas de perda e acurácia: 250 gestos sintéticos por real / 250 gestos reais.



Fonte: Elaborado pelo autor.

Um fenômeno válido de se notar é que apesar de as curvas de treino e validação demonstrarem um comportamento de overfitting, as curvas de acurácia demonstram que o modelo ainda continua a aprender padrões de forma necessária para aumentar as acurácias durante as rodadas de validação, mesmo que atinjam um ponto de saturação.

Buscando aprimorar os resultados obtidos pelo classificador, investigaram-se métodos alternativos para melhorar a representação dos dados de entrada do modelo.

Atualmente, ao utilizar os pontos exatos como entrada, uma característica importante dos movimentos oculares acaba sendo subestimada pelo modelo: em fixações relacionadas a uma tecla específica, o ponto registrado nem sempre coincidirá com o centro presumido daquela tecla, podendo apresentar pequenas variações.

Para capturar melhor essa variabilidade natural do olhar, cada ponto agora é tratado como uma amostra de uma distribuição gaussiana centrada na posição original, com um desvio padrão igual ao dobro do tamanho de uma tecla. Essa abordagem reflete a ideia intuitiva de que, ao tentar digitar a tecla “w”, por exemplo, o usuário pode fixar seu olhar próximo a teclas vizinhas, como “q”, “a”, “s” ou “d”. Dessa

forma, o modelo passa a considerar explicitamente a probabilidade associada a essas variações, resultando em uma representação mais robusta e realista dos dados de entrada.

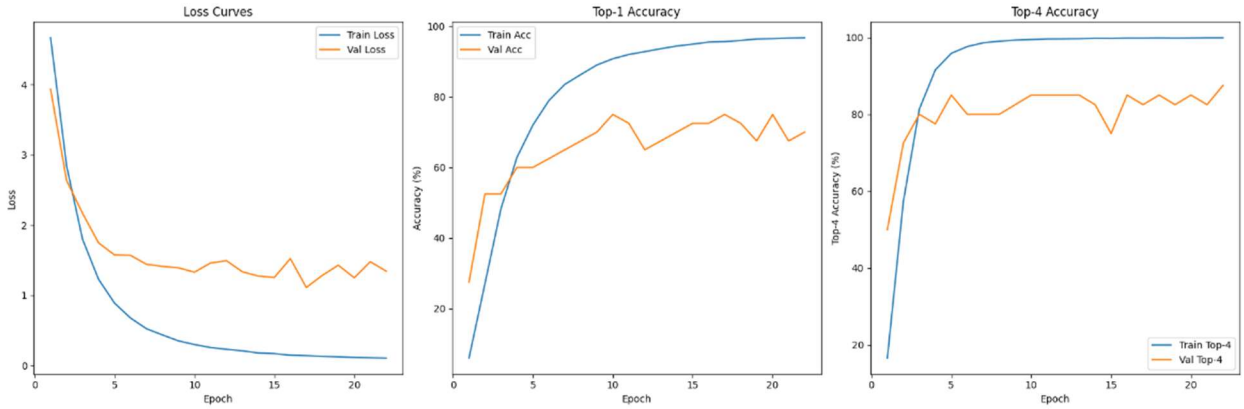
Um primeiro experimento avaliou o desempenho do modelo utilizando 150 gestos sintéticos para cada um dos 200 gestos reais, como uma divisão de 80% para treino e 20% para validação, com resultados exibidos na Figura 5. Nessa configuração, o modelo atingiu uma acurácia top-1 de 70% e uma acurácia top-3 de 87.50%.

Para investigar o impacto de diferentes proporções entre dados sintéticos e reais, foram realizados novos testes, variando a quantidade de gestos sintéticos por gesto real. Na Figura 6, o número de gestos sintéticos foi aumentado para 200 para cada gesto real (mantendo-se o número de gestos reais), o que elevou a acurácia top-1 de validação para 80%, e a acurácia top-4 de validação para 90%.

Esse resultado indica que, até certo ponto, gerar mais dados sintéticos ajuda o modelo a capturar padrões relevantes e melhorar sua capacidade de classificação, mesmo quando a validação é feita apenas com dados reais.

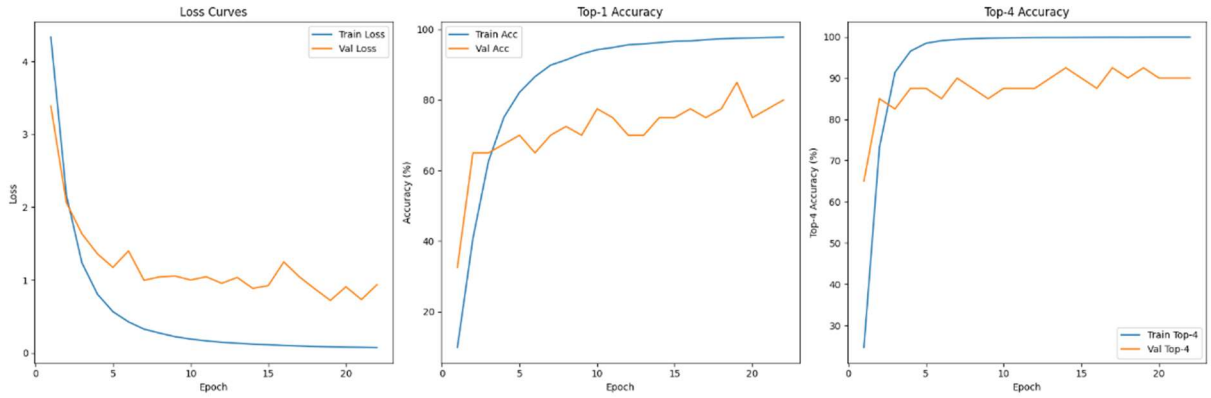
Além disso, foi realizado um experimento com 100 gestos reais e 100 gestos sintéticos para cada real, mostrado na Figura 7. Nessa configuração, a acurácia de validação atingiu 76% na acurácia top-1 e 92% no top-4. Apesar de a acurácia top-4 ter superado as configurações anteriores, deve-se considerar que, com menos gestos reais e um número reduzido de classes, o modelo tem um espaço de decisão mais restrito, facilitando a classificação correta dentro das quatro melhores previsões.

Figura 5 – Curvas de perda e acurácia: 150 gestos sintéticos por real / 200 gestos reais.



Fonte: Elaborado pelo autor.

Figura 6 – Curvas de perda e acurácia: 200 gestos sintéticos por real / 200 gestos reais.



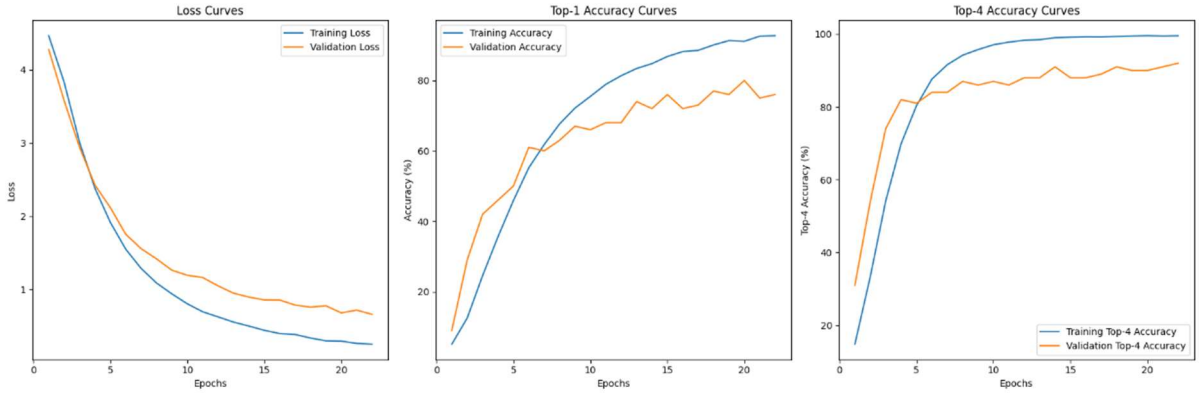
Fonte: Elaborado pelo autor.

Tabela 2 – Resultados de diferentes configurações de dados reais e sintéticos.

Configuração	Dados Reais	Acurácia Top-1	Acurácia Top-4
150 sintéticos/gesto	200	70.00%	87.50%
200 sintéticos/gesto	200	80.00%	90.00%
100 sintéticos/gesto	100	76.00%	92.00%

Fonte: Elaborado pelo autor.

Figura 7 – Curvas de perda e acurácia, 100 gestos sintéticos/100 gestos reais.



Fonte: Elaborado pelo autor.

3.5.5. Limitações da Abordagem de Classificação e Nova Direção

Os experimentos com o classificador baseados em redes convolucionais demonstraram resultados promissores, atingindo uma acurácia top-1 de 80%. Esse

resultado validou uma premissa fundamental: as trajetórias visuais contêm padrões reconhecíveis que podem ser decodificados em palavras com sucesso.

No entanto, apesar das altas taxas de acerto em um conjunto de dados controlado, a arquitetura de classificação apresenta limitações intrínsecas que dificultam sua aplicação em um sistema de digitação real. A principal barreira é a dependência de um vocabulário fixo e pré-definido durante o treinamento. O modelo é inerentemente incapaz de prever palavras que não estavam presentes em seu conjunto de treino, resultando em previsões errôneas e tornando o sistema impraticável para o uso geral. Além disso, a escalabilidade para vocabulários extensos, como o de 60 mil palavras utilizado posteriormente neste trabalho, seria computacionalmente custosa e exigiria um volume de dados de treinamento ainda maior.

Essas constatações levaram a um pivô estratégico no projeto. Em vez de classificar o gesto como uma única entidade, uma abordagem de decodificação incremental se mostrou mais adequada. Tal abordagem permitiria não apenas o uso de um vocabulário vasto de forma eficiente, mas também a integração explícita de um modelo de linguagem para resolver ambiguidades contextuais. Foi a partir dessa necessidade que se iniciou o desenvolvimento do LanGaze, um decodificador projetado desde o início para superar essas limitações, apoiando-se em métodos clássicos combinados com saídas probabilísticas de um modelo de linguagem pré-treinado.

3.6. LanGaze

O LanGaze é um sistema de decodificação projetado para traduzir trajetórias visuais em palavras de um dicionário fixo. Sua arquitetura se diferencia ao integrar duas fontes de informação: a análise geométrica do gesto e a modelagem contextual da linguagem, essa última potencializada por um LLM. O objetivo é proporcionar uma digitação mais fluída e precisa, superando limitações de sistemas clássicos.

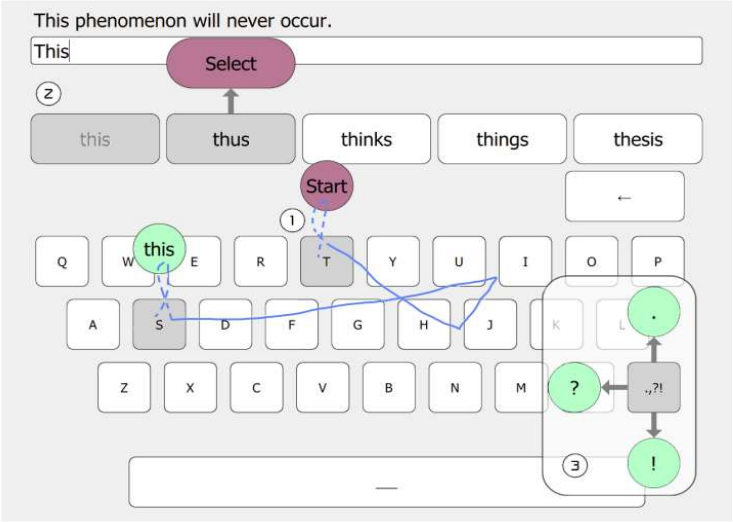
3.6.1. Limitações de Sistemas Clássicos

Sistemas como o EyeSwipe (Kurauchi et. al., 2016) e o GlanceWriter (Cui et. al., 2023) utilizam etapas explícitas para iniciar e encerrar gestos. No EyeSwipe, o usuário executa um reverse-crossing para indicar o início e outro para iniciar o fim do gesto.

O reverse-crossing é uma interação em que o usuário passa um pequeno tempo olhando para a tecla que deseja começar o gesto. Após esse pequeno tempo, um botão de ação aparecerá logo acima do botão, que o usuário deve olhar para ele para iniciar o gesto. Após a finalização do gesto, um botão de ação com a palavra prevista aparecerá, também acima da última tecla, em que o usuário deve olhar para ele para confirmar a digitação, ao olhar para um dos candidatos, como ilustrado na

Figura 8, em que para escrever a palavra “This”, o usuário realiza um reverse-crossing na letra “t” para indicar o início do gesto. O reverse-crossing é indicado pela linha pontilhada azul. Após a realização do gesto, é realizado outro reverse-crossing para confirmar a digitação da palavra completa.

Figura 8 – Representação do reverse-crossing.



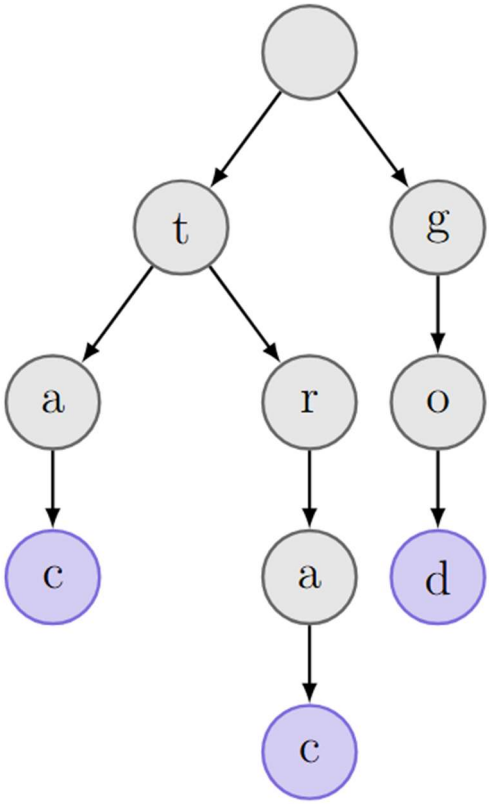
Fonte: (Kurauchi et. al., 2016)

No GlanceWriter, o início do gesto é detectado automaticamente, mas o término exige que o usuário visualize as palavras candidatas. A palavra mais provável é inserida automaticamente. Caso ocorra erro, é necessário executar duas seleções por dwell-time: uma para acionar a tecla de backspace e outra para selecionar a palavra correta na lista de candidatos.

3.6.2. Representação por Sufixos: A Trie Invertida

A decodificação requer uma estrutura de dados que permita buscas parciais e evite varrer todo o dicionário. O LanGaze utiliza uma trie para armazenar um vocabulário fixo de 60 mil palavras em inglês. Numa trie, cada aresta representa uma letra, e nós terminais indicam palavras completas. Palavras que compartilham prefixos compartilham nós, reduzindo redundâncias e acelerando buscas.

Figura 9 – Trie invertida (por sufixos) para as palavras cat, cart e dog. A busca inicia pela última letra do gesto.



Fonte: Elaborado pelo autor.

Diferentemente de uma trie convencional, o LanGaze constrói a estrutura de forma invertida, inserindo as palavras da última para a primeira letra. Por exemplo, a palavra “mover” é armazenada como “r”→”e”→”v”→”o”→”m”. Uma trie invertida está representada na Figura 9. Essa inversão parte da hipótese de que a maior parte da informação de valor sobre a forma geométrica do gesto que tem mais chance de ser o que o usuário realmente queria realizar está nos últimos instantes do gesto. Ao indexar palavras pelo sufixo, a busca inicia pela parte mais confiável do gesto, melhorando a precisão na seleção das palavras candidatas. Além disso, a trie é filtrada, deixando apenas as palavras que terminam com uma letra que esteja próxima as fixações que ocorreram nos últimos 200 milissegundos do gesto.

A implementação do algoritmo de decodificação foi realizada em Cython, buscando otimizar a latência do algoritmo, tornando-o utilizável em aplicações reais. A implementação da lógica completa está disponível neste arquivo. [disponível no anexo A.3].

3.6.3. Decodificação com Alinhamento Temporal Reverso

A decodificação do LanGaze combina uma busca em profundidade sobre a trie invertida com um alinhamento entre a sequência de letras e a trajetória do gesto, ambos no sentido reverso. A trie armazena o vocabulário por sufixos: cada palavra $\omega = \omega_1 \dots \omega_{|\omega|}$ é indexada como $\omega = \omega_{|\omega|} \dots \omega_1$. O gesto observado $G = (g_1, \dots, g_t)$ é invertido e normalizado pela escala do tamanho das teclas do teclado. A busca inicia em nós filhos do raiz compatíveis com letras finais plausíveis, estimadas a partir dos últimos 200 milissegundos do gesto, ou seja, só são consideradas possíveis palavras que possuam sufixos filtrados pela última letra próxima dos últimos pontos do gesto, diminuindo consideravelmente o espaço de busca, e se mantendo fiel à hipótese original da maior informação estar presente no final do gesto.

O gesto capturado é representado por uma sequência temporal de pontos bidimensionais $G = (g_1, g_2, \dots, g_t)$, cada um correspondendo à posição do olhar sobre o teclado. Esses pontos são invertidos para $G' = (g_t, g_{t-1}, \dots, g_1)$ de modo que o primeiro ponto da sequência seja o mais próximo do instante de confirmação.

Cada caminho percorrido na trie, partindo da letra final de uma palavra e avançando em direção à sua primeira letra, gera uma sequência de posições de teclas $C = (c_1, c_2, \dots, c_L)$, em que c_1 é o centro da tecla correspondente à última letra da palavra e c_L ao da primeira. Assim, a sequência C é derivada diretamente do caminho atual da trie e está no mesmo sentido de G'.

O alinhamento é feito entre essas duas sequências espaciais, ou seja, da sequência de referência, que são as coordenadas das letras da palavra parcial atual na trie (C), obtidas da geometria do teclado, e a sequência observada, que são os pontos do gesto invertido (G'), normalizados pelo tamanho da tecla.

O alinhamento entre C e G' é calculado de forma incremental durante a exploração em profundidade da trie, utilizando o método Dynamic Time Warping (DTW). Sendo L o comprimento atual da palavra parcial na trie, é definida uma matriz dtw de dimensão $(L_{max} + 1) \times (t + 1)$ inicializada com $dtw[0][0] = 0$ e $dtw[0][i] = +\infty$ para $i \geq 1$:

$$\mathrm{dtw} \in R^{(L_{max}+\mathbb{1})\times(T+\mathbb{1})}, \quad \mathrm{dtw}[0][0]$$
$$= 0, [0][i] = +\infty \ \text{ para } i \geq 1$$

Para cada nova letra adicionada à palavra parcial, é calculado o L correspondente segundo:

$$\mathrm{dtw}[L][i] = \mathrm{cost}(L, i) + \min\{\mathrm{dtw}[L-1][i-1],\ \mathrm{dtw}[L][i-1],\ \mathrm{dtw}[L-1][i]\},$$
$$1 \leq i \leq T$$

Em que $cost(L, i) = max\{dist(c_L, g_i' - \delta, 0\}$, sendo $dist(\cdot,\cdot)$ a distância euclidiana e δ um coeficiente de tolerância espacial, com valor de 1.2 vezes o valor do tamanho da tela.

A cada visita a um nó terminal da trie, isto é, quando a palavra parcial corresponde exatamente a uma palavra do vocabulário, o alinhamento é avaliado. São considerados múltiplos pontos de parada i na sequência G', correspondente a diferentes comprimentos de gesto, e calcula-se para cada um:

$$d_i = \frac{dtw[L][i]}{(i+1)^2}$$

O fator de normalização de $(i+1)^2$ penaliza alinhamentos que terminem muito antes do instante de confirmação, favorecendo aqueles próximos ao final do gesto. Apenas são aceitos alinhamentos cujo ponto final g_i' esteja a menos de um limiar τ do centro da última letra da palavra. O menor custo aceito d^* é convertido em uma probabilidade de gesto:

$$P_{gesto} = \frac{1}{1+d^*}$$

Esse procedimento é repetido para todos os nós terminais encontrados ao longo da busca em profundidade definida. Portanto, ao fim dessa abordagem, se obtém uma pontuação para cada palavra possível ser a do gesto.

3.6.4. Incorporação de Contexto com Modelo de Linguagem

Embora a decodificação baseada exclusivamente na trajetória do olhar produza resultados satisfatórios, o objeto central que a digitação tenta representar é a linguagem, um conceito naturalmente abstrato e difícil de ser representado e previsto. Em um cenário ideal, um teclado não deveria apenas identificar corretamente a palavra que o gesto indica, mas também considerar fatores linguísticos e contextuais para desambiguar casos de incerteza.

Considerando, por exemplo, a frase "Vou comprar a minha primeira casa". É possível que, devido à proximidade entre as teclas D e S, o gesto para a última palavra resulte em uma trajetória mais compatível com "cada" do que com "casa". Um sistema de decodificação puramente gestual cometeria o erro, enquanto a incorporação de informação linguística poderia evitar essa confusão.

Essa motivação é compatível com abordagens anteriores. No EyeSwipe (Kurauchi et al., 2016), a probabilidade final é obtida somando ao termo relacionado ao gesto um fator proporcional à frequência da palavra no corpus (ocorrências da palavra/ocorrências de todas as palavras na trie). O GlanceWriter (Cui et al., 2023) segue estratégia semelhante, e em seu trabalho é sugerido que um termo linguístico mais robusto, como um LLM, poderia melhorar o desempenho da decodificação.

LLMs são modelos generativos autorregressivos, que dado um contexto de tokens $t_0, t_1, \dots, t_{i-1}$, estimam a distribuição de probabilidades do próximo token t_i.

Essa característica se aproxima do problema enfrentado no LanGaze. Em um cenário ideal de digitação, conhecemos as $i-1$ palavras anteriores, e queremos prever a $i-esima$ palavra a partir do gesto atual.

Para explorar essa semelhança, foi desenvolvida uma estratégia de integração entre o decodificador e um LLM. Durante a busca na trie, que foi descrita nas sessões anteriores, a cada nó terminal visitado, as palavras candidatas compatíveis com o gesto até aquele ponto são registradas. Essas palavras compõem o conjunto $\mathcal{W}$ de hipóteses plausíveis para a palavra seguinte.

O modelo escolhido foi o GPT-2 padrão, ou seja, que não foi treinado para instruction-following. A escolha do GPT-2 foi feita justamente por se tratar de um modelo leve (124 Milhões de parâmetros na versão small), com custo computacional reduzido e uma capacidade adequada para a tarefa de next-token prediction. O GPT-2 recebe como entrada o contexto da frase, isto é, as palavras já confirmadas, devidamente tokenizadas, e retorna um vetor de probabilidades sobre todo o seu vocabulário de tokens.

Do vetor de probabilidades produzido pelo modelo, são selecionadas apenas aqueles associados às palavras de $\mathcal{W}$. Essa filtragem requer atenção ao processo de tokenização, pois, em modelos como o GPT-2, existem tokens distintos para uma palavra no início de uma frase (por exemplo, “Good”) e para a mesma palavra precedida por um espaço (“ good”). Foi implementado um mecanismo de mapeamento para unificar ambos os casos e atribuir a mesma probabilidade para a palavra-alvo, de forma independente de sua posição na frase.

As probabilidades linguísticas, restritas a $\mathcal{W}$, são então combinadas com as probabilidades dos gestos, calculadas pelo decodificador de gestos, como será detalhado na sessão seguinte.

3.6.5. Combinação Ponderada e Balanceamento de Escala

A combinação das pontuações provenientes de decodificador gestual e do modelo de linguagem foi projetada com o objetivo de integrar duas fontes de informação: a primeira baseada na similaridade geométrica entre as trajetórias captadas e os gestos ideais, e a segunda baseada na coerência linguística das palavras candidatas selecionadas, conforme discutido nas sessões anteriores. A formulação inicial segui a abordagem mais simples possível, por meio de uma combinação linear ponderada das probabilidades, de forma que a probabilidade combinada fosse dada pela expressão:

$$p_{comb} = \alpha_1 \cdot P_{ges} + \alpha_2 \cdot P_{lan}$$

No entanto, os resultados obtidos com essa configuração indicaram um ganho prático reduzido, já que o termo relacionado à linguagem apresentava mais influência quase nula na decisão final, mesmo quando se atribuía a ele um peso elevado.

Uma análise sobre esses resultados revelou que essa limitação é diretamente associada a um desbalanceamento de escala entre as ordens de grandeza das duas pontuações. O decodificador de gestos produz valores relativamente altos, porque sua normalização é derivada de custos geométricos entre a trajetória observada e as ideais, então essas operações sempre se mantém numa ordem de grandeza da escala do teclado normalizada. Já o modelo de linguagem, por sua vez, gera uma distribuição softmax sobre um vocabulário amplo, resultando em probabilidades frequentemente muito baixas, muitas das vezes da ordem de 10^{-3} a 10^{-5}. Nessa condição, mesmo com um peso elevado, o termo relacionado à linguagem não exerceria impacto significativo.

Para corrigir esse desbalanceamento, foi adotada uma reformulação na qual apenas o termo linguístico é elevado a um expoente x, resultando na expressão:

$$P_{comb} = \alpha_1 \cdot P_{ges} + \alpha_2 \cdot (P_{lan})^x$$

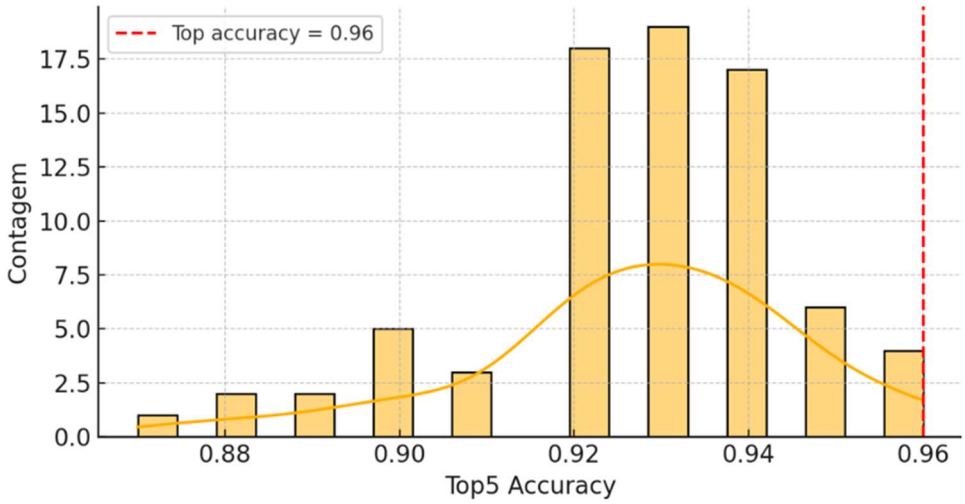
Com x sendo um expoente fracionário entre 0 e 1. O expoente x tem o efeito de expandir de forma proporcional as probabilidades pequenas, aproximando a ordem de grandeza do termo relacionado ao modelo de linguagem ao do decodificador de gestos, mas ainda assim preservando a ordenação relativa imposta pelas probabilidades originais do modelo de linguagem.

4. Resultados e Discussões

A introdução dos três coeficientes de decodificação (α_1, α_2, e x) exigiu a definição de um método para encontrar sua combinação ótima. Para essa tarefa, realizou-se um estudo paramétrico utilizando um conjunto de dados previamente coletado por (KURAUCHI et. al., 2016), que contém trajetórias de olhar e as palavras correspondentes. Esse conjunto de dados foi utilizado por conta de seu volume maior e uma maior variância nos participantes presentes. Além disso, os inícios e finais dos gestos, correspondentes aos reverse-crossings, são cortados durante o processamento dos dados, tornando-os utilizáveis para o escopo deste trabalho. No estudo, os valores dos coeficientes foram variados sistematicamente em faixas discretas, e as métricas de acurácia foram calculadas para cada combinação.

Conforme ilustrado na Figura 10, os experimentos geraram uma distribuição de acurácias top-5 que atingiram um valor máximo de 0.96. A Figura 11 detalha a relação entre os parâmetros, mostrando que essa acurácia máxima ocorre de forma consistente em um platô de desempenho. Na Figura 11, foi utilizado como expoente x o valor que obteve o melhor resultado para cada combinação de parâmetros correspondente.

Figura 10 – Distribuição de todas as acurácias top-5 obtidas durante os experimentos realizados.



Fonte: Elaborado pelo autor.

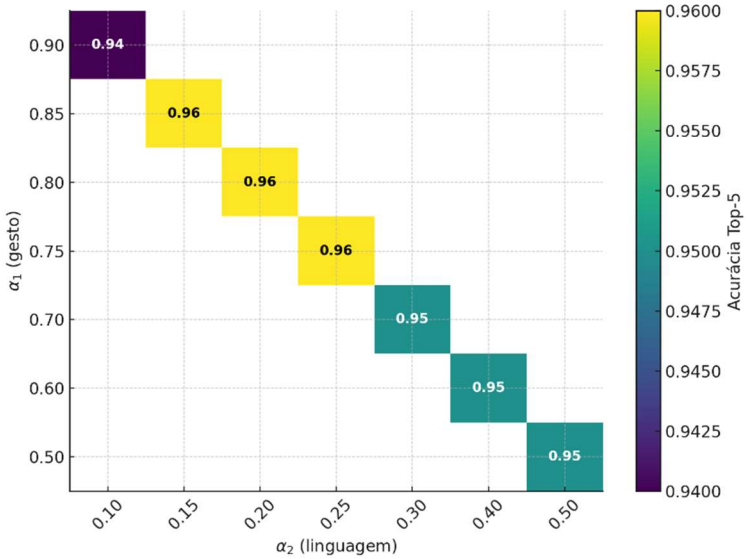
A Tabela 3 traz as combinações de parâmetros registradas que atingiram o platô de desempenho citado.

Tabela 3 – Combinações de parâmetros que atingem a acurácia top-5 máxima.

α_1	α_2	x	Top-1	Top-3	Top-5
0.75	0.25	0.25	0.24	0.84	0.96
0.80	0.20	0.15	0.23	0.84	0.96
0.85	0.15	0.10	0.23	0.83	0.96
0.80	0.20	0.10	0.22	0.83	0.96

Fonte: Elaborado pelo autor.

Figura 11 – Mapa de calor variando os parâmetros de α_1 e α_2.



Fonte: Elaborado pelo autor.

Da Tabela 3, é possível enxergar uma diferença considerável de desempenho entre os valores das acurácias top-5 e top-1. O motivo desse comportamento merece uma análise aprofundada e é um tema para estudos futuros. A hipótese dos autores para essa disparidade está diretamente ligada à arquitetura de decodificação invertida do LanGaze.

A decisão de indexar as palavras por seus sufixos em uma trie invertida e iniciar a decodificação pelo final do gesto é altamente eficaz para reduzir o espaço de busca para o decodificador. Ao priorizar a parte mais confiável do traçado do gesto, o sistema rapidamente gera uma lista de palavras candidatas que correspondem bem à parte final da trajetória. É por isso que a palavra correta muito provavelmente estará entre as 5 primeiras sugestões, resultando em uma excelente acurácia top-5.

Contudo, a informação crucial para diferenciar gestos candidatos com sufixos semelhantes (ex: “casa” vs “rasa” ou “part” vs “cart”) reside no início do gesto. No LanGaze, essa porção inicial da trajetória é analisada por último no processo de

alinhamento reverso com Dynamic Time Warping (DTW). Suspeita-se que, nesse alinhamento reverso, a influência desses pontos iniciais pode ser parcialmente suprimida ou ter um peso menor na determinação do custo final. O fator de normalização, projetado para penalizar alinhamentos curtos, pode não recompensar suficientemente uma correspondência precisa no início do gesto (o final da sequência invertida). Como resultado, o decodificador pode ter dificuldade em eleger com confiança o único candidato correto a partir de uma lista de opções geometricamente muito similares, o que explicaria a queda na acurácia top-1.

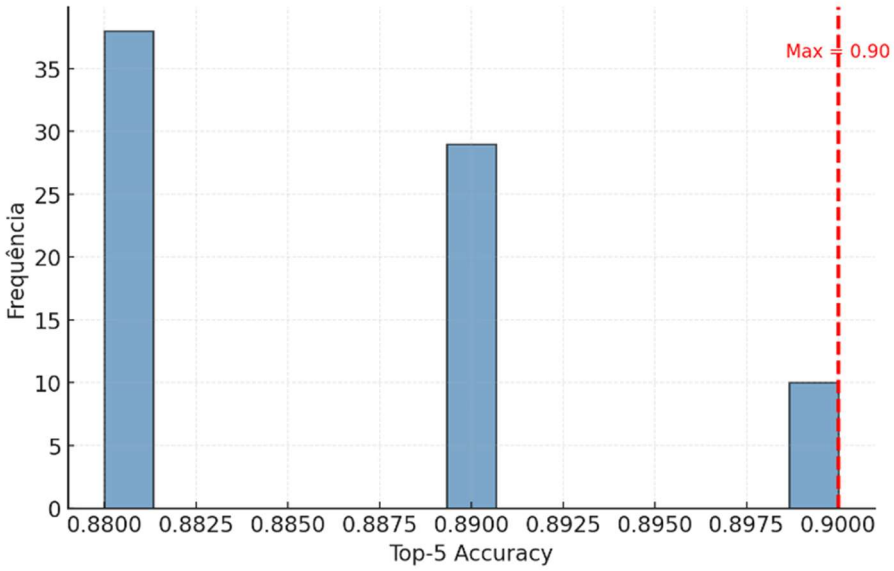
Uma possível direção para trabalhos futuros seria investigar um modelo de duas etapas: utilizar a abordagem invertida atual para gerar de forma eficiente uma lista restrita de candidatos de alta qualidade, e em seguida, aplicar um segundo algoritmo de reclassificação focado especificamente em alinhar o início do gesto com o início das palavras candidatas para refinar a escolha da melhor previsão.

A escolha final dos coeficientes adotados foi $\alpha_1 = 0.8$, $\alpha_2 = 0.2$ e $x = 0.2$. Essa combinação foi uma das que apresentou o máximo de acurácia, e nela, a pontuação relacionada ao gesto ainda possui o papel de ancoragem da decisão, enquanto o modelo de linguagem atua como um filtro para eventuais correções.

Para quantificar a contribuição do modelo de linguagem, a análise paramétrica foi repetida, desta vez anulando a influência do componente linguístico ao fixar seu peso (α_2) em zero. Os resultados apresentados na Figura 12 revelam que a acurácia top-5 máxima alcançada nessas condições foi de 90%. Esse valor é 6 pontos percentuais inferior ao pico de 96% obtido com o sistema completo, o que demonstra o ganho de desempenho proporcionado pela integração do LLM.

Adicionalmente, uma análise da distribuição dos resultados revela um segundo efeito importante: sem o componente linguístico, as acurácias se concentram em apenas três valores distintos. Isso sugere que o decodificador de gestos produz pontuações mais discretas e uniformes. Em contrapartida, a integração do LLM resulta em dez limiares de acurácia distintos (conforme a Figura 10), indicando que o modelo de linguagem introduz uma maior granularidade na pontuação final, ajustando-a com base na especificidade e no contexto linguístico de cada amostra.

Figura 12 – Distribuição das acurácias top-5 com peso de linguagem fixo em zero.



Fonte: Elaborado pelo autor.

5. Considerações Finais

Neste trabalho, apresentamos o LanGaze, um novo decodificador para sistemas de decodificação por olhar. Assim como em abordagens anteriores, como o GlanceWriter (Cui et. al., 2023) e o EyeSwipe (Kurauchi et. al., 2016), o sistema adota um framework baseado na tradução de trajetórias visuais em palavras a partir de um vocabulário estruturado em uma trie. A principal diferença está na integração de um novo componente de modelagem de linguagem implementado por meio de um modelo de linguagem pré-treinado (LLM), que atua de forma complementar ao processo de alinhamento do gesto. Essa integração permitiu combinar, pela primeira vez, a estrutura clássica de decodificação de trajetórias visuais com a capacidade preditiva de um LLM, resultando em desempenhos promissores. Os resultados dos experimentos mostraram uma acurácia top-5 de até 96%.

Outro aspecto central foi a proposta de indexação por sufixos em uma trie invertida. Essa escolha se mostrou particularmente adequada ao cenário de digitação por olhar, em que os pontos iniciais dos gestos tendem a ser mais ruidosos. Ao priorizar o final do traçado, a decodificação se torna mais robusta e abre caminho para novas formas de interação com sistemas de digitação por olhar, em que o usuário pode ter uma maior liberdade e controle, tendo que exercer menos esforço cognitivo para confirmar a digitação de uma palavra. Essa linha de investigação é especialmente promissora em ambientes de realidade aumentada (AR), nos quais a captura do olhar pode estar sujeita a maior variabilidade no início dos gestos por conta da presença maior de informações e estímulos no campo de visão do usuário. Uma extensão natural deste trabalho seria a implementação do LanGaze em interfaces de Realidade Aumentada, explorando como a indexação por meio de sufixos pode otimizar a interação dos usuários em tais cenários.

Para validar a eficácia do LanGaze em cenários práticos, trabalhos futuros envolverão uma análise comparativa com usuários. O desempenho do sistema será avaliado em relação a métodos de referência, como o EyeSwipe (Kurauchi et. al., 2016) e o GlanceWriter (Cui et. al., 2023), utilizando métricas de interação como a Taxa de Palavras por Minuto (WPM) e a taxa de erro na digitação.

Além disso, o componente de linguagem do LanGaze foi desenvolvido de maneira modular, o que implica que, à medida que modelos de linguagem mais sofisticados e leves se tornem disponíveis, eles poderão ser integrados ao decodificador sem a necessidade de alterar sua estrutura central. Dessa forma, a tendência é que o desempenho do sistema acompanhe a evolução dos LLMs.

Referências

CHU, J. et al. Wordgesture-gan: Modeling word-gesture movement with generative adversarial network. In: CHI CONFERENCE ON HUMAN FACTORS IN COMPUTING SYSTEMS, 2023, New York. Proceedings […]. New York: Association for Computing Machinery, 2023. Disponível em: https://doi.org/10.1145/3544548.3581279. Acesso em: 28 ago. 2025.

CUI, W. et al. Glancewriter: Writing text by glancing over letters with gaze. In: CHI CONFERENCE ON HUMAN FACTORS IN COMPUTING SYSTEMS, 2023, New York. Proceedings […]. New York: Association for Computing Machinery, 2023. Disponível em: https://doi.org/10.1145/3544548.3581269. Acesso em: 28 ago. 2025.

HU, J.; DUDLEY, J.; KRISTENSSON, P. Skimr: Dwell-free eye typing in mixed reality. 2024. Disponível em: https://www.repository.cam.ac.uk/handle/1810/371264. Acesso em: 28 ago. 2025.

JACOB, R. J. K. The use of eye movements in human-computer interaction techniques: what you look at is what you get. ACM Transactions on Information Systems, New York, v. 9, n. 2, p. 152–169, abr. 1991. Disponível em: https://doi.org/10.1145/123078.128728. Acesso em: 28 ago. 2025.

KRISTENSSON, P. O. Statistical language processing for text entry. In: Computational Interaction. Oxford: Oxford University Press, 2018. Cap. 43. Disponível em: https://doi.org/10.1093/oso/9780198799603.003.0003. Acesso em: 28 ago. 2025.

KURAUCHI, A. et al. Eyeswipe: Dwell-free text entry using gaze paths. In: CHI CONFERENCE ON HUMAN FACTORS IN COMPUTING SYSTEMS, 2016, New York. Proceedings […]. New York: Association for Computing Machinery, 2016. p. 1952–1956. Disponível em: https://doi.org/10.1145/2858036.2858335. Acesso em: 28 ago. 2025.

MAJARANTA, P.; AHOLA, U.-K.; SPAKOV, O. Fast gaze typing with an adjustable dwell time. In: SIGCHI CONFERENCE ON HUMAN FACTORS IN COMPUTING SYSTEMS, 2009, New York. Proceedings […]. New York: Association for Computing Machinery, 2009. p. 357–360. Disponível em: https://doi.org/10.1145/1518701.1518758. Acesso em: 28 ago. 2025.

MAJARANTA, P.; RÄIHÄ, K.-J. Text entry by gaze: Utilizing eye-tracking. In: MACKENZIE, I. S.; TANAKA-ISHII, K. (ed.). Text entry systems: Mobility, accessibility, universality. Burlington: Morgan Kaufmann, 2007. p. 175–187. Disponível em: http://www.cs.uta.fi/~curly/publications/Majaranta_and_Raiha_2007--DRAFT.pdf. Acesso em: 28 ago. 2025.

MARDANBEGI, D.; KURAUCHI, A. T. N.; MORIMOTO, C. H. An investigation of the distribution of gaze estimation errors in head mounted gaze trackers using polynomial functions. Journal of Eye Movement Research, v. 11, n. 3, p. 1-13, jun.

2018. Disponível em: https://www.mdpi.com/1995-8692/11/3/24. Acesso em: 28 ago. 2025.

MEHRA, A. et al. Implicit vs. explicit style transfer? a comparison of gan architectures for continuous path keyboard input modeling. In: EUROPEAN SIGNAL PROCESSING CONFERENCE (EUSIPCO), 29., 2021, Dublin. Proceedings […]. [S.l.]: IEEE, 2021. p. 1396–1400. Disponível em: https://eurasip.org/Proceedings/Eusipco/Eusipco2021/pdfs/0001396.pdf. Acesso em: 28 ago. 2025.

MORIMOTO, C. H.; MIMICA, M. R. M. Eye gaze tracking techniques for interactive applications. Computer Vision and Image Understanding, v. 98, n. 1, p. 41-65, 2005. Disponível em: https://latin.ime.usp.br/media/papers/pdf/EyeGazeTrackingTechniquesForInteractiveApplicationsCVIU05.pdf. Acesso em: 28 ago. 2025.

OUYANG, T. et al. Mobile Keyboard Input Decoding with Finite-State Transducers. 2017. Disponível em: https://arxiv.org/abs/1704.03987. Acesso em: 28 ago. 2025.

SHEHU, I. S.; WANG, Y.; ATHUMAN, A. M.; FU, X. Remote eye gaze tracking research: A comparative evaluation on past and recent progress. Electronics, v. 10, n. 24, 2021. Disponível em: https://www.mdpi.com/2079-9292/10/24/3165. Acesso em: 28 ago. 2025.

Apêndice A

1. https://github.com/gustavovvbs/Wgaze-experiment-API/blob/main/src/comp_metric.py
2. https://github.com/gustavovvbs/WGaze-experiments
3. https://gist.github.com/gustavovvbs/22680f41c1f01542da6b52fba06f3472