Universidade Federal da Paraíba
Centro de Informática
Programa de Pós-Graduação em Informática

Uma Arquitetura Multifluxo Baseada em Aprendizagem Profunda para Reconhecimento de Sinais em Libras no Contexto de Saúde

Diego Ramon Bezerra Da Silva

João Pessoa
Dezembro - 2020

Universidade Federal da Paraíba

Centro de Informática

Programa de Pós-Graduação em Informática

Uma Arquitetura Multifluxo Baseada em Aprendizagem Profunda para Reconhecimento de Sinais em Libras no Contexto de Saúde

Diego Ramon Bezerra da Silva

Dissertação submetida à Coordenação do Curso de Pós-Graduação em Informática da Universidade Federal da Paraíba - Campus I como parte dos requisitos necessários para obtenção do grau de Mestre em Informática.

Área de Concentração: Sistemas de Computação
Linha de Pesquisa: Computação Distribuída

Tiago Maritan Ugulino de Araujo
(Orientador)
Thaís Gaudêncio do Rêgo
(Co-orientadora)

João Pessoa, Paraíba, Brasil

Catalogação na publicação
Seção de Catalogação e Classificação

S586a Silva, Diego Ramon Bezerra da.
Uma arquitetura multifluxo baseada em aprendizagem profunda para reconhecimento de sinais em libras no contexto de saúde / Diego Ramon Bezerra da Silva. - João Pessoa, 2021.
82 f. : il.

Orientação: Tiago Maritan Ugulino de Araujo.
Coorientação: Thaís Gaudêncio do Rêgo.
Dissertação (Mestrado) - UFPB/Informática.

1. Informática. 2. Acessibilidade. 3. Libras. 4. Visão computacional. 5. Redes Neurais Convolucionais. 6. Aprendizagem profundo. I. Araujo, Tiago Maritan Ugulino de. II. Rêgo, Thaís Gaudêncio do. III. Título.

UFPB/BC CDU 004-056.263

Elaborado por MARILIA RIANNY PEREIRA COSMOS - CRB-15/862

Resumo

Os surdos são uma parte considerável da população mundial. No entanto, embora muitos países adotem sua língua de sinais como língua oficial, existem barreiras linguísticas de acesso aos direitos fundamentais, especialmente o acesso aos serviços de saúde. Essa situação tem sido o foco de algumas políticas governamentais que obrigam os prestadores de serviços essenciais a fornecer intérpretes de língua de sinais para ajudar as pessoas surdas. No entanto, esse tipo de solução possui altos custos operacionais, principalmente para atender toda a comunidade surda em todos os ambientes. Esses contratempos motivam a investigação de metodologias e ferramentas automatizadas para apoiar esse tipo de problema. Assim, neste trabalho, é proposto um modelo de várias correntes para o reconhecimento de sinais em Língua Brasileira de Sinais (Libras). A solução proposta não utiliza nenhum sensor ou hardware de captura adicional, baseando-se inteiramente em imagens ou sequências de imagens (vídeos). Os resultados obtidos com uma arquitetura de três fluxos mostram que a melhor acurácia para o conjunto de testes foi de 99,80%, considerando um cenário em que o intérprete usado no conjunto de testes não foi usado no conjunto de treinamento. Além disso, também foi criado um novo conjunto de dados na Língua Brasileira de Sinais (Libras) contendo 5000 vídeos de 50 sinais no contexto da saúde, o que pode auxiliar no desenvolvimento e na pesquisa de outras soluções.

Palavras-chaves: Acessibilidade, Libras, Visão Computacional, Redes Neurais Convolucionais, Aprendizagem profunda, Multimodal.

Abstract

Deaf people are a considerable part of the world population. However, although many countries adopt their sign language as an official language, there are linguistics barriers to accessing fundamental rights, especially access to health services. This situation has been the focus of some government policies that oblige essential service providers to provide sign language interpreters to assist deaf people. However, this type of solution has high operating costs, mainly to serve the entire deaf community in all environments. These setbacks motivate the investigation of methodologies and automated tools to support this type of problem. Thus, in this paper, we proposed a two-stream model for the recognition of the Brazilian Sign Language (Libras). The proposed solution does not use any additional capture sensor or hardware, being entirely base on images or sequences of images (videos). The results show that the best accuracy for the test set was 99.80%, considering a scenario where the interpreter used in the test set was not used in the training set. Besides, we also created a new dataset in the Brazilian sign language (Libras) containing 5000 videos of 50 signs in the health context, which may assist the development and research of other solutions.

Keywords: Accessibility, Libras, Computer Vision, CNN, Deep Learning, Multimodal.

Agradecimentos

Ao meu orientador, Tiago Maritan Ugulino de Araújo, que desde sempre me incentivou, confiou na minha capacidade, me deu total liberdade para desenvolver a dissertação e foi muito participativo ao longo de todo o processo. Muito obrigado por todos os ensinamentos, toda ajuda e todo o apoio dado durante a execução desse trabalho.

À professora Thaís Gaudêncio do Rêgo, minha eterna orientadora de Monografia e co-orientadora nesse projeto. Sempre prestativa e atenciosa. Muito obrigada pelos ensinamentos e pela paciência.

Às bancas da qualificação e da defesa final por todas as sugestões, muito obrigado. Todas as sugestões foram por mim recebidas com a certeza de que elas tinham o objetivo de melhorar a qualidade da dissertação.

À toda equipe do Laboratório de Aplicações de Vídeo Digital (LAVID), em especial a equipe responsável pela concepção da base de dados usada nessa pesquisa, muito obrigado.

À CAPES que financiou essa pesquisa através de bolsa.

Agradeço à minha família, e principalmente à minha tia, Maria Bezerra Campos, que me ensinou a priorizar os estudos desde pequeno, por todo suporte emocional, carinho e pela compreensão das minhas ausências.

A todos aqueles que direta ou indiretamente contribuíram para a elaboração deste trabalho, torceram pela minha conquista e me apoiaram nos momentos mais difíceis. Muito obrigado!

Sumário

Lista de Símbolos

CNN: Rede Neural Convolucional, do inglês *Convolutional Neural Network*
CUDA: Arquitetura de Dispositivo de Computação Unificada, do inglês *Compute Unified Device Architecture*
CNN-2D: Rede Neural Convolucional 2D, do inglês *2D Convolutional Neural Network*
CNN-3D: Rede Neural Convolucional 3D, do inglês *3D Convolutional Neural Network*
DL: Aprendizado Profundo, do inglês *Deep learning*
GPU: Unidade de Processamento Gráfico, do inglês *Graphics Processing Unit*
I3D: Rede 3D Inflada, do inglês *Inflated 3D Network*
LS: Língua de Sinais
LSA: Língua de Sinais Argentina
LIBRAS: Língua Brasileira de Sinais
LSTM: Memória de Longo Prazo, do inglês *Long short-term memory*
LRCN: Rede Convolucional Recorrente de Longo Prazo, do inglês *Long-term Recurrent Convolutional Networks*
OHE: Codificação fictícia, do inglês *One-Hot-Encode*
PLN: Processamento de Linguagem Natural
RNN: Redes Neurais Recorrentes, do inglês *Recurrent Neural Network*
SVM: Máquina de Vetor de Suporte, do inglês *Support Vector Machine*

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

1.1 Introdução

As pessoas com deficiência auditiva têm sido tema relevante de discussões, na tentativa de levar equidade social, educacional e de saúde a esta comunidade, uma vez que representam um quantitativo significativo da população [31]. No Mundo, de acordo com o Relatório da Mortalidade e Carga de Doenças (do inglês, *Mortality and Burden of Diseases*) de 2012 da Organização Mundial da Saúde (do inglês, *World Health Organization* - WHO), existem aproximadamente 360 milhões de pessoas com algum nível de deficiência auditiva, o que representa cerca de 5,3% de toda a população [68]. No Brasil, de acordo com o censo de 2010 do Instituto Brasileiro de Pesquisas Geográficas (IBGE), existem aproximadamente 9,7 milhões de brasileiros com algum tipo de perda auditiva, representando 5,1% da população [20].

Por ser uma comunidade minoritária linguística e culturalmente, os surdos enfrentam algumas barreiras de comunicação, acesso à informação e serviços, especialmente os serviços de saúde [31]. Uma das razões para essa dificuldade é que as pessoas surdas comunicam-se naturalmente através de línguas gestuais-visuais, denominadas línguas de sinais, e as línguas orais representam apenas uma "segunda língua".

A problemática de integração da comunidade surda também ganhou bastante atenção do legislativo brasileiro. Projetos de leis como PLS 465/2017, ainda em tramitação no Congresso, altera a Lei nº 10.048, de 8 de novembro de 2000, obrigando a oferta de intérpretes de Libras em instituições públicas e concessionárias de serviços públicos de assistência à saúde [7]. Além disso, a PLS 155/2017, obriga repartições públicas, empresas concessionárias de serviços públicos e instituições financeiras, os bancos, a contar com intérpretes de Libras [6].

Com a atuação do poder público, essa demanda social é convertida em uma demanda de mercado, pois o não cumprimento dessas leis podem resultar em multas expressivas e impactos negativos para a marca ou empresa em questão. Porém, o cumprimento dessas leis está sujeito a problemas de logística, tendo em vista que isso demandaria uma grande quantidade de intérpretes humanos e geraria um custo considerável para manter essa política de acessibilidade operacional e, com isso, se faz necessário um processo de pesquisa e desenvolvimento de novas ferramentas e tecnologias que possam suprir essa demanda de forma economicamente e logisticamente viáveis.

Por se tratar essencialmente de um problema de automatização, uma alternativa viável e prática é a utilização de técnicas e metodologias baseadas em Inteligência Artificial (IA) para auxiliar na resolução deste tipo de problema. Neste contexto, alguns trabalhos vem sendo desenvolvidos na literatura científica voltados para a tradução automática de conteúdos entre línguas orais e línguas de sinais [27, 29, 63].

Na literatura científica, existem vários trabalhos que estão abordando a tradução automática a partir de texto ou áudio em linguagens faladas em animações (ou vídeos) em língua de sinais [12, 28, 43, 44, 58, 59, 64, 73]. Outros trabalhos envolvem o reconhecimento de conteúdos em linguagens de sinais (por exemplo, vídeos ou imagens) em linguagens orais [9, 15, 21, 26, 62, 70]. No entanto, algumas dessas soluções geralmente requerem alguns sensores ou hardware adicionais (por exemplo, luvas, braçadeiras, entre outros), o que dificulta o uso em um cenário real [13, 23, 39, 46, 48, 50, 84]. Além disso, não foram encontradas soluções ou conjuntos de dados para o reconhecimento de sinais na Língua Brasileira de Sinais (Libras). Também não foram encontrados estudos que abordem esse tipo de solução no contexto da saúde, dificultando a avaliação se esse tipo de solução funciona com sinais nesse contexto.

Sendo assim, nesse trabalho, com o objetivo de possibilitar uma forma de comunicação dinâmica, bidirecional, considerando que a tradução de linguagem natural já foi amplamente pesquisada e com várias soluções para usuários finais disponíveis [2, 4, 5] e, consequentemente, melhor integração dessa parcela da população no contexto da saúde, é proposta uma arquitetura baseada em Aprendizado Profundo para reconhecimento de sinais de Libras.

Um dos desafios deste tipo de trabalho é que os sinais consistem de três partes principais: Atributos manuais envolvendo gestos feitos com as mãos, atributos não manuais, tais como expressões faciais ou postura corporal, que podem fazer parte de um sinal ou modificar o seu significado, e um alfabeto manual, onde palavras são escritas na linguagem verbal local. Naturalmente, essa é uma simplificação excessiva, a

língua de sinais é tão complexa quanto qualquer língua falada, e cada língua de sinais possui dezenas de milhares de sinais, diferindo por pequenas alterações de mão, forma, movimento, posição, recurso ou contexto não manual [24].

Para abordar esse problema e ajudar na inclusão e integração dos usuários surdos brasileiros no contexto da saúde, neste trabalho, propomos uma solução para o reconhecimento de sinais em Libras. Nossa proposta é que a solução possa auxiliar, por exemplo, na comunicação de um paciente surdo com seu médico. A solução pode ajudar um médico que não conhece Libras a entender alguns dos sintomas do paciente, que podem ser importantes em consultas médicas remotas (por exemplo, telessaúde ou telemedicina) e até mesmo em consultas pessoais, especialmente no contexto da nova pandemia do COVID-19.

A solução proposta combina Redes Neurais Convolucionais e Redes Recorrentes e contém três fluxos. O primeiro fluxo contém o fluxo ótico, que permite capturar informações sobre o "movimento" do sinal, um dos principais fonemas das línguas de sinais. O segundo fluxo contém as imagens RGB brutas e, portanto, é capaz de extrair informações gerais e completas sobre o sinal, não capturadas no primeiro fluxo. Por fim, o terceiro fluxo usa *keypoints*, uma representação cartesiana que codifica a postura corporal humana extraídos com o auxílio da biblioteca OpenPose, permitindo assim capturar informações sobre "movimento", "ponto de articulação" e "orientação", que são outros importantes fonemas presentes em Libras. Essa abordagem permite adicionar mais recursos espaço-temporais que discretizam as classes durante o estágio de treinamento, sem aumentar o tamanho do conjunto de dados.

Como não foram encontrados bancos de dados de sinais de saúde na Língua Brasileira de Sinais, também foi criado um novo conjunto de dados para o reconhecimento de sinais de Libras no contexto da saúde, como uma contribuição adicional ao trabalho. Esse conjunto de dados consiste em 5000 vídeos de 50 sinais, extraídos das situações cotidianas no domínio da saúde e executados em um ambiente controlado 10 vezes por 10 intérpretes de Libras. A partir dessa motivação, selecionamos os sinais para compor o conjunto de dados com base em uma investigação que mapeou quais sinais os surdos usam com mais frequência em um ambiente hospitalar [11]. Além disso, o conjunto de dados resultantes possui várias amostras e classes semelhantes a outros conjuntos de dados encontrados na literatura usados neste tipo de problema, como os conjuntos de dados apresentados em [25, 67, 69, 71]. No entanto, esses últimos foram desenvolvidos para propósitos gerais e não específicos, conforme proposto neste trabalho.

1.2 Motivação

1.2.1 Premissas e Hipóteses

Nesse trabalho, parte-se da hipótese que é possível, dada a natureza da formação dos sinais das línguas de sinais, de serem formadas e terem seu significado modificado por diferentes componentes ou por suas interações, de ser possível conceber uma arquitetura de aprendizado profundo de múltiplos fluxos capaz de reconhecer sinais de Libras no contexto de saúde de maneira eficaz. A premissa é que os componentes dos sinais são tratados de forma isolada e especializada, e que a solução é capaz de reconhecer esses sinais usando apenas imagens de uma pessoa interpretando os sinais, i.e., sem a necessidade de luvas, sensores especiais ou algum hardware adicional.

1.3 Objetivos

1.3.1 Objetivo Geral

Realizar um estudo acerca da viabilidade de reconhecer, de forma automatizada, sinais em Libras no contexto da saúde, a partir de uma sequência de imagens digitais, utilizando uma arquitetura multifluxo baseada em aprendizado profundo.

1.3.2 Objetivos Específicos

1. Especificar e conceber um conjunto de dados, vídeos da execução de sinais por usuários de Libras (intérpretes ou usuários surdos), no domínio da saúde, para ser usado no contexto de aprendizado profundo.

2. Avaliar a acurácia, precisão e capacidade de generalização da solução proposta no reconhecimento dos sinais de saúde treinados.

1.3.3 Estrutura da Dissertação

Este trabalho está estruturado da seguinte maneira:

- Capítulo 2: Encontra-se o embasamento, a fundamentação necessária para o desenvolvimento deste estudo.

- Capítulo 3: Apresenta uma seleção de trabalhos que relacionados ao presente trabalho.
- Capítulo 4: Descreve-se o método utilizado para realização do trabalho.
- Capítulo 5: Neste capítulo, todos os resultados preliminares obtidos são exibidos e discutidos.
- Capítulo 6: Dá lugar às considerações finais, os problemas encontrados, bem como as limitações do trabalho.

Capítulo 2

Fundamentação Teórica

2.1 Línguas de Sinais

As Línguas de Sinais (LS) são as línguas naturais de comunicação das pessoas surdas, configurando-se como a língua principal e, às vezes, como a única língua que essa parcela da população usa para comunicação no dia a dia. Elas são expressas através da combinação de movimentos manuais, expressões e movimentos corporais e faciais, sendo que a variação de um único componente pode alterar completamente o significado do que se deseja comunicar.

Diferente do que muitos imaginam, as Línguas de Sinais não são simplesmente mímicas e gestos soltos, utilizados pelos surdos para facilitar a comunicação, qualificação que só foi desconstruída recentemente a partir nos anos 60, inicialmente através do trabalho de [77], que argumentou que as línguas de sinais têm status de língua, pois elas são compostas pelos níveis linguísticos: o fonológico, o morfológico, o sintático e o semântico.

Além disso, os sinais de uma língua de sinais são compostos por diferentes fonemas, que são unidades básicas ou componentes no qual um sinal pode ser decomposto. Para [17], cada sinal é unicamente identificado pelos seguintes cinco fonemas:

1. **Configuração de mão**: esse parâmetro representa a posição e forma das mãos e dedos durante a reprodução de um sinal. Para [38], na Libras existem 64 configurações de mão. Cada configuração pode ser executada pela mão dominante ou usando as duas mãos, dependendo do sinal.

2. **Ponto de Articulação**: representa a parte do corpo onde os sinais são realizados, sendo delimitada pela extensão máxima dos braços do emissor, podendo

estar localizado em alguma parte do corpo ou em um espaço neutro vertical (do meio do corpo até à cabeça) e horizontal (à frente do emissor).

3. **Movimento**: representa a forma em que a mão é movimentada durante a execução do sinal, embora existam sinais estáticos em algum local e que não necessitam de movimento para serem caracterizados.

4. **Orientação**: representa a orientação ou direção do movimento, ou seja, é o plano em que a palma da mão está orientada.

5. **Expressões não manuais**: são expressões faciais e corporais que podem ser usadas como um traço diferenciador.

Dessa forma, um sinal de Libras é produzido pela combinação desses 4 ou 5 fonemas, sendo é importante ressaltar que sinais de uma língua de sinais, como em Libras, com significados distintos, podem ter mais de um componente em comum e sendo diferenciado por um único fonema.

A língua de sinais tem como meio propagador o campo gesto-visual, o que a diferencia da língua oral, que utiliza o canal oral-auditivo. Além dessa diferença, também apresenta antagonismos quanto às regras constitutivas. No entanto, a língua de sinais deve ser respeitada como língua, pois assume a mesma função da língua oral, a comunicação [33].

Assim como as línguas orais, cada país tem sua língua de sinais própria, formada e constantemente se adequando aos seus contextos socioculturais, sendo essa forma padrão podendo ainda ser modificada por regionalismos e, com isso, agregando outra característica das línguas naturais, a sua não estaticidade.

A Libras foi reconhecida através da Lei 10.436, de 24 de Abril de 2002, como sendo um meio legal de comunicação e expressão para a comunidade de Surdos brasileira. Sendo definida como o sistema linguístico de natureza visual-motora, com estrutura gramatical própria, constituem um sistema linguístico de transmissão de ideias e fatos, oriundos de comunidades de pessoas surdas do Brasil.

2.2 Redes Neurais Convolucionais

2.2.1 Introdução

As Redes Neurais Convolucionais (do inglês, *Convolutional Neural Network* - ConvNets) são atualmente a arquitetura de Aprendizado Profundo (do inglês, *Deep Learning* - DL) mais conhecida e aplicada na resolução de problemas reais, tendo aplicações nas áreas de visão computacional, processamento de linguagem natural, reconhecimento, rastreamento de objetos e muitas outras áreas. Para [54], as ConvNets são projetados para processar dados que vêm na forma de várias matrizes, por exemplo, uma imagem colorida composta por três matrizes contendo intensidades de pixel nos três canais de cores. Muitas modalidades de dados estão na forma de múltiplas matrizes: 1D para sinais e sequências, incluindo idioma; 2D para imagens ou espectrogramas de áudio; e 3D para imagens de vídeo ou volumétricas.

A principal característica que diferencia as ConvNets é o uso da operação de convolução ao invés da multiplicação de matriz. Com isso, a entrada é processada utilizando campos receptivos locais e aproveitando três ideais importantes que podem melhorar um sistema de aprendizado de máquina: interações esparsas, pesos compartilhados e representações equivarientes. Além disso, uma arquitetura convolucional usual (Ver Figura 2.1) ainda apresenta etapas de subamostragem espacial e normalização, que tem como objetivo reduzir a dimensionalidade espacial das representações e aumentar a capacidade de aprendizado de cada camada individual da rede, respectivamente [54].

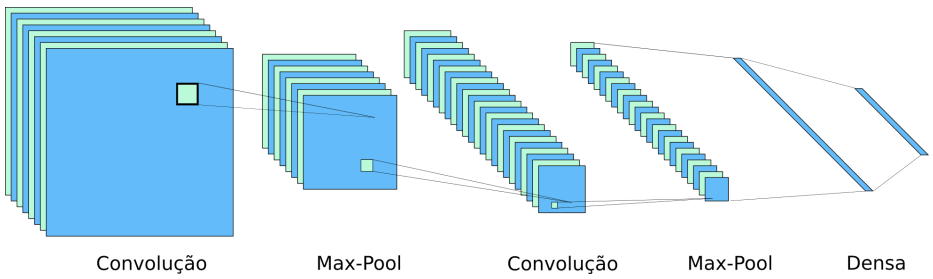

Figura 2.1: Arquitetura geral de uma Rede Neural Convolucional (CNN) incluindo camada de entrada, convolucionais, max-pooling e camadas completamente conectadas. Fonte: Autor

2.2.2 Convolução

Na sua forma mais geral, a convolução é uma das mais básicas operações realizadas no processamento de imagens e sinais. No processamento de imagens, muitos

filtros realizam o processo de convolução com alguma máscara ou *kernel* para realizar tarefas, desde uma simples redução de ruídos, até tarefas mais complexas como detecção de bordas.

Matematicamente, uma convolução é uma integral sobre duas funções reais x e w. A finalidade da operação é realizar o deslizamento da função w sobre a função x, resultando assim uma "mistura" de w e x. Para o caso contínuo, a operação de convolução pode ser expressa da seguinte integral:

$$s(t) = \int x(a)w(t-a)da \tag{2.1}$$

Embora a operação de convolução seja tipicamente denotada por um asterisco e apresentada na literatura com a seguinte representação:

$$s(t) = (x * w)(t) \tag{2.2}$$

para a nomenclatura e contexto das ConvNets, o primeiro argumento da convolução, a função x é usualmente referenciada como a entrada, e o segundo argumento, a função w, como o *kernel*. A saída é usualmente referenciada como mapa de características (do inglês, *feature map*) [54]. No contexto das ConvNets, é mais comum se trabalhar sobre valores discretizados a partir de sinais físicos e contínuos, portanto, a operação de convolução assume a seguinte representação discreta:

$$s(t) = (x * w)(t) = \sum_{a=-\infty}^{\infty} x(a)w(t-a) \tag{2.3}$$

Por fim, para o contexto do processamento digital de imagens, que podem ser vistas como sinais 2D, é necessário fazer a expansão da operação de convolução para duas dimensões (2D). Também é possível desprezar a etapa de rebatimento do *kernel*, tendo em vista que a mesma só existe para garantir a propriedade da comutatividade, que não tem relevância para o contexto, resultando, portanto, na seguinte representação:

$$S(i,j) = (K * I)(i,j) = \sum_{m}\sum_{n} I(i+m, j+n)K(m,n) \tag{2.4}$$

Esta última fórmula é a operação de convolução usualmente implementada pelas bibliotecas ou *frameworks* de Aprendizado Profundo. É pertinente observar que a operação de convolução, para ser completamente caracterizada matematicamente,

necessita do rebatimento do *kernel*. Logo, essa operação é algumas vezes referenciada na literatura como correlação cruzada (do inglês, *cross-correlation*), embora o termo convolução seja atualmente aceito na literatura da área.

2.2.3 Conexões Locais

Cada pequena região processada pelo filtro ou *kernel*, durante o processo de convolução, é denominada de campo receptivo local (do inglês, *local receptive field*) [66]. Um campo receptivo pode ser descrito por uma localização central e pelo tamanho do *kernel*. Essa é a principal característica que permite que as ConvNets lidem com grandes volumes de dados, tais como os apresentados em imagens e vídeos. Diferentemente das redes neurais clássicas, onde cada neurônio da camada de entrada é ligado a cada neurônio da camada saída, gera-se uma grande quantidade de parâmetros para serem computados. Com o uso de campos receptivos, existe uma drástica redução da quantidade de parâmetros, uma vez que cada conexão será feita sobre os campos receptivos.

Na Figura 2.2, é possível visualizar uma representação gráfica da propriedade de campo receptivo local. De acordo com a Figura 2.2, cada pixel é representado por um círculo branco. Os pixels pretos estão sob efeito do processo de convolução por um *kernel* de tamanho K, ou seja, é utilizada uma combinação linear de K^2 elementos para gerar um único pixel na camada de saída. Usualmente, os valores mais usados de K, nos sistemas e aplicações de visão computacional, são $1, 3, 5$ e 7.

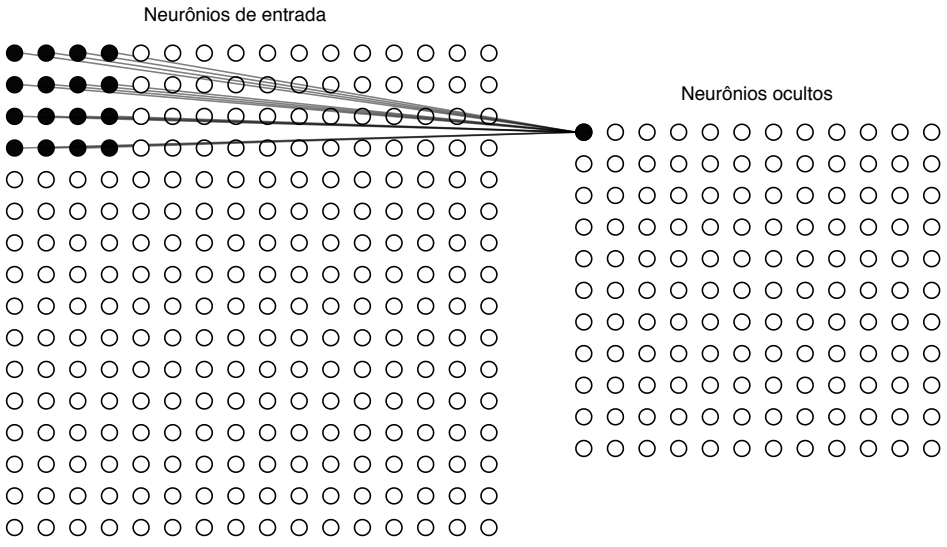

Figura 2.2: Representação gráfica do campo receptivo local. Fonte: Autor

Uma outra resultante importante desse processo é o compartilhamento de pesos

entre todos os neurônios ocultos. Essa característica implica que cada camada oculta irá detectar a mesma característica sobre a imagem de entrada. Uma visão de alto nível seria, se durante o processo de treinamento, essa camada se especializar em detectar arestas (Ver Figura 2.3), com a propriedade do compartilhamento de pesos. Nessa camada oculta será detectada somente arestas sobre toda a imagem de entrada.

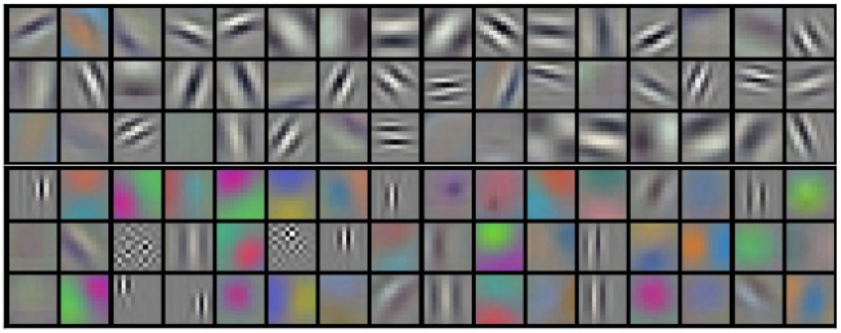

Figura 2.3: Filtros aprendidos para detecção de arestas horizontais. Fonte: [55]

Finalmente, a operação de convolução, por meio do compartilhamento de parâmetros, também apresenta a propriedade de equivariância para transação, que pode ser descrita como a capacidade de detectar a característica sobre toda a imagem de entrada. Apesar da redução da quantidade de parâmetros ocasionada pela propriedade do campo receptivo local, o volume de dados e parâmetros a serem computados ainda é muito grande. Logo, recorre-se a técnicas de subamostragem espacial para diminuir esse volume de dados e diminuir o custo computacional das ConvNets.

2.2.4 Subamostragem Espacial

Dado o grande volume de dados presente numa ConvNets, são comumente introduzidas camadas de subamostragem espacial (Ver Figura 2.4) para realizar a redução do volume de dados e diminuição da quantidade de parâmetros treináveis. Essas camadas são denominadas camadas de subamostragem (do inglês, *Pooling layers*) e, de maneira geral, realizam a simplificação e substituição de um conjunto de saídas vizinhas da rede, por uma sumarização estatística delas [54].

A forma mais comumente usada de subamostragem espacial é o *max-pooling* (Ver Figura 2.5), que consiste basicamente de, dado o tamanho da janela de amostragem K, e um passo de deslocamento denominado *stride*, usualmente com valores 2×2 e 2, respectivamente, selecionar a saída com maior valor e passar para camada seguinte, e repetir essa operação após um deslocamento definido pelo *stride*. Esse deslocamento tem como objetivo impedir que uma mesma saída seja usada em mais de uma vez

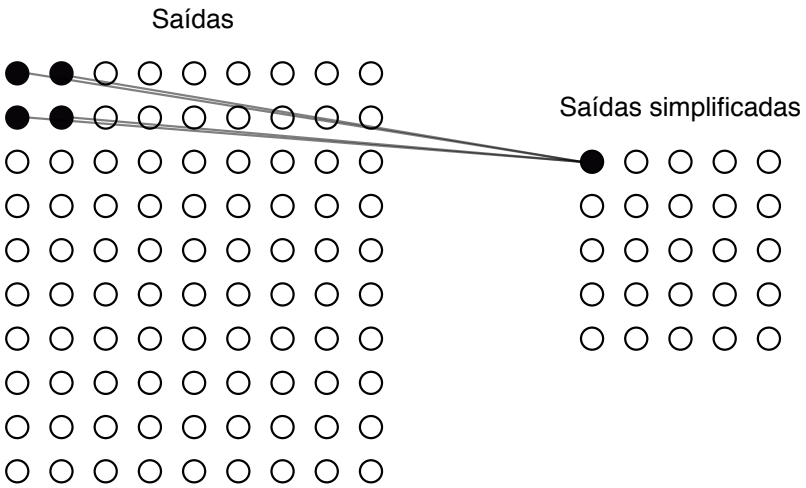

Figura 2.4: Representação gráfica do processo de subamostragem espacial. Fonte: Autor

durante a subamostragem.

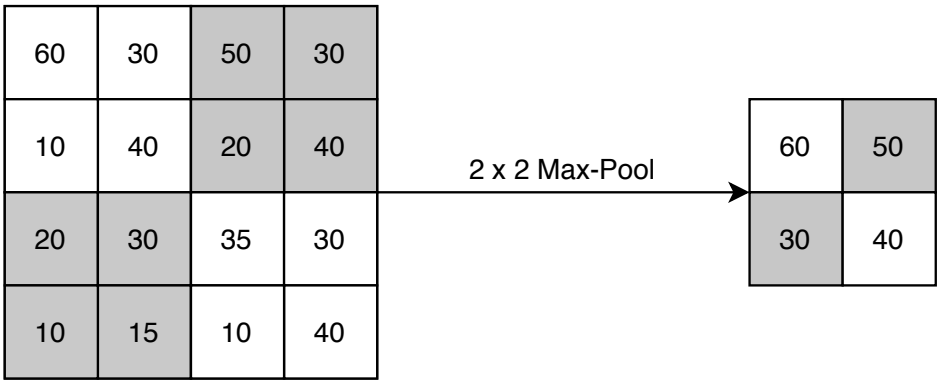

Figura 2.5: Representação gráfica do processo de max-pooling. Fonte: Autor

2.2.5 Normalização de Lotes

Todas as arquiteturas de Redes Neurais, no geral, exigem um etapa de pré-processamento visando a normalização do conjunto de dados, ou seja, forçando a média zero e variância unitária e, com isso, evitando a saturação de funções de ativação não lineares, tal como a comumente utilizada função sigmóide conforme Figura 2.6.

Esse problema é exemplificado considerando uma camada com função de ativação sigmóide $z = g(Wu + b)$, onde u é a entrada, W é a matriz de pesos e b é o vetor de vieses e $g(x) = \frac{1}{1+\exp(-x)}$. A medida que $|x|$ aumenta, $g'(x)$ tende para zero. Isso significa que, para todas as dimensões de $x = Wu + b$, exceto aquelas com pequenos valores absolutos, o gradiente que propaga para u desaparecerá e o modelo treinará lentamente. No entanto, como x é afetado por W e b e os parâmetros de todas as

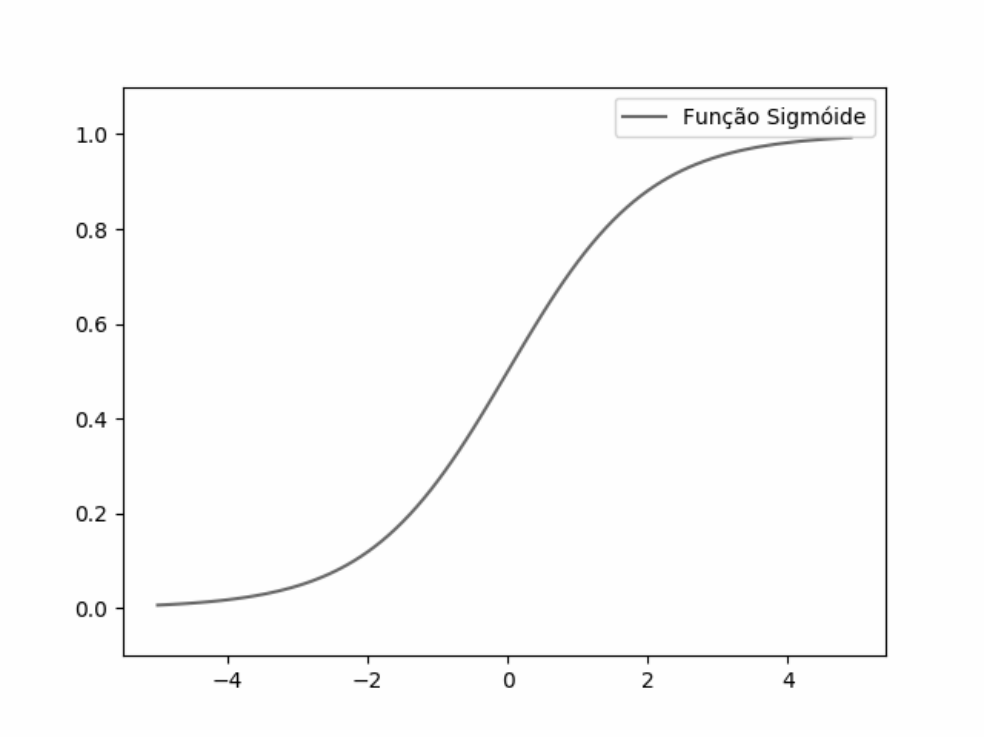

Figura 2.6: Gráfico da função sigmóide. Fonte: Autor

camadas anteriores, as alterações nesses parâmetros durante o treinamento, provavelmente moverão muitas dimensões de x para o regime saturado da não linearidade e retardarão a convergência. Este efeito é amplificado à medida que a profundidade da rede aumenta [45].

Desse modo, em camadas intermediárias, durante o processo de treinamento, a distribuição está constantemente variando, dando origem a um problema denominado deslocamento covariável interno (do inglês, *covariate shift problem*). Esse problema penaliza a velocidade de treinamento do modelo, pois cada camada deve aprender a se adaptar com a nova distribuição em cada época de treinamento.

Esse problema é contornado através da técnica de normalização de lotes (do inglês, *batch normalization*) apresentado em [45], que consiste essencialmente de forçar cada entrada de todos as camadas a terem uma distribuição aproximada e, com isso, aumentar a estabilidade da rede durante o treinamento. Matematicamente, o algoritmo de normalização de lotes primeiro realiza o cálculo da média (Equação 2.5) e variância das entradas da camada (Equação 2.6).

$$\mu_\beta = \frac{1}{m}\sum_{i=1}^{m} x_i \tag{2.5}$$

$$\sigma_B^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2 \tag{2.6}$$

Em seguida, normaliza as entradas da camada usando as estatísticas do lote previamente calculadas:

$$\bar{x_i} = \frac{x_i - \mu_b}{\sqrt{\alpha_B^2 + \epsilon}} \tag{2.7}$$

Finalmente, esse valores são escalados e deslocados para obter a saída da camada:

$$y_i = \gamma\bar{x_i} + \beta \tag{2.8}$$

A Equação (2.8) caracteriza completamente o processo de normalização de lotes. Os parâmetros γ e β são aprendidos durante o treinamento, junto com os parâmetros originais do modelo. Na prática, o uso da técnica de normalização de lotes traz as vantagens de diminuir o tempo de convergência da rede e diminuir a dependência de uma boa inicialização de pesos, ou seja, a rede ganha mais estabilidade. Em razão disso, o uso dessa técnica torna-se extremamente vantajoso, principalmente em arquiteturas profundas.

2.3 Redes Neurais Recorrentes

Existem vários tipo de problemas que não podem ser completamente caracterizados por dados pontuais ou estáticos (ver Figura 2.7), como por exemplo, dados que possuem uma natureza sequencial, ou que se comportam como séries temporais, tais como: sinais elétricos, ações na bolsa de valores ou sentenças de linguagens naturais. Esses tipos de dados precisam de informação temporal para serem caracterizados.

Esses tipos de problemas são difíceis de serem resolvidos usando redes neurais clássicas, pois tais abordagens acarretariam em dois grandes problemas. Primeiro, séries temporais, no geral, apresentam quantidade de parâmetros variáveis, isto é, a quantidade de entradas e saídas pode variar entre amostras do mesmo problema, portanto, codificar essas entradas numa rede neural clássica acarretaria em uma representação complexa, resultando em muitos parâmetros, ou seja, um modelo de difícil treinamento. Segundo, esse modelo não tem a capacidade de usar informações anteriores $0 \ldots t-1$

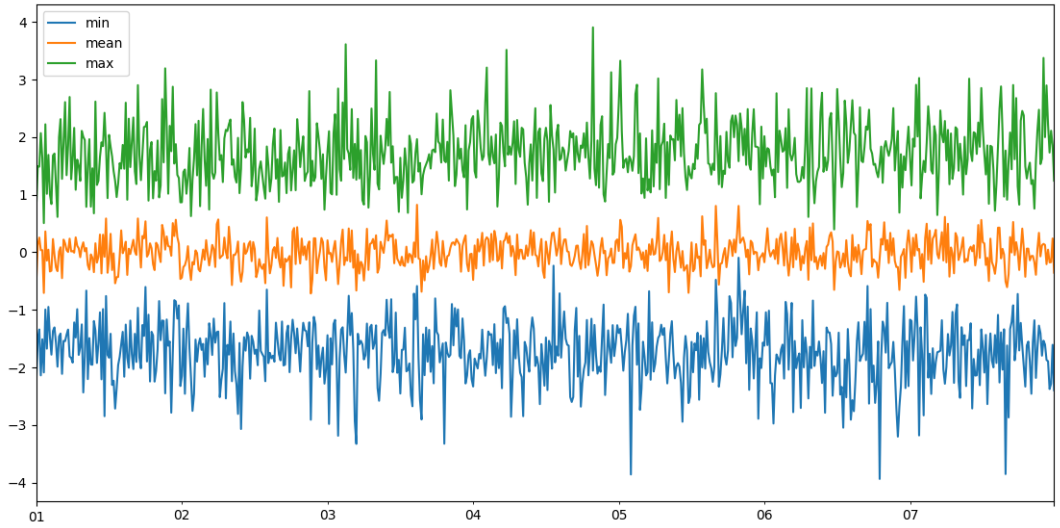

Figura 2.7: Exemplo de série temporal artificial. Fonte: Autor

para fazer a predição no tempo t, ou seja, esse modelo não apresenta memória, e dada a natureza desses problemas, o estado atual tem correlação com pontos anteriores.

Em razão disso, novos modelos foram propostos para contornar esses dois problemas: variação da quantidade de entradas e saídas e que fosse capaz de ter a propriedade de memória. Esse novo conjunto de modelos foi denominado de Redes Neurais Recorrentes (do inglês, *Recurrent Neural Network* - RNN).

Uma rede neural recorrente é um modelo de aprendizado profundo para processamento de dados sequenciais. Suas principais características são a existência da propriedade de memória e da capacidade de não requerer entradas e saídas com tamanhos fixos [72]. Uma representação genérica de uma RNN pode ser visualizada na Figura 2.8. Dado um conjunto de entradas $x^1, x^2, \ldots, x^n$, um conjunto de ativações $a^0, a^1, \ldots, a^n$ e um conjunto de saídas $y^1, y^2, \ldots, y^n$, as equações que governam uma RNN são definidas por:

$$\begin{aligned} a_t &= tanh(W_h a_{t-1} + W_x x_t + b_t) \\ y_t &= W_s a_t \end{aligned} \tag{2.9}$$

Onde W_h, W_x e W_s são as matrizes de pesos e b_t é o vetor de biases. É possível observar que o cálculo da ativação no tempo a_t também leva em consideração a ativação do tempo a_{t-1}, dando assim a propriedade de memória das RNN, pois todas as informações anteriores também pesarão na predição no tempo y_t. A ativação no tempo a_0 é inicializada com um vetor de zeros. As redes RNN possuem diferentes

arquiteturas de saídas e entradas que variam conforme a natureza do problema a ser tratado. A configuração mais básica apresentada na Figura 2.9, nada mais é do que uma rede neural clássica.

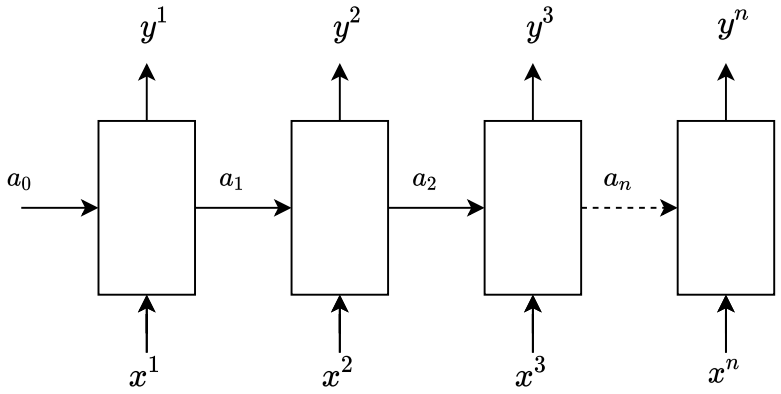

Figura 2.8: Diagrama de uma rede neural recorrente desenrolada. Fonte: Autor

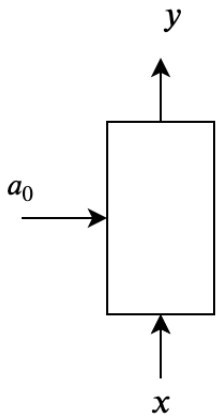

Figura 2.9: Diagrama de uma rede neural recorrente um-para-um. Fonte: Autor

Na Figura 2.10 é possível visualizar uma arquitetura de RNN com uma entrada e múltiplas saídas. Esse tipo de arquitetura pode ser aplicada, por exemplo, em geração de letras de música de forma automática. Nesse exemplo, poderia-se usar um número inteiro como entrada, representando um determinado estilo musical, e a rede produziria uma letra de uma música nesse estilo musical selecionado, como saída.

A configuração apresentada na Figura 2.11, por outro lado, apresenta múltiplas entradas e múltiplas saídas. Este tipo de arquitetura pode ser usada para o problema de tradução automática de texto entre línguas distintas, uma vez que o tamanho das sentenças, mesmo com sentidos idênticos, podem variar entre diferentes idiomas. Uma outra aplicação para essa configuração de RNN é o problema de sumarização de texto, onde a rede recebe como entrada um texto qualquer e produz, como saída, uma versão resumida dele, tendo naturalmente a saída e entrada com comprimentos distintos.

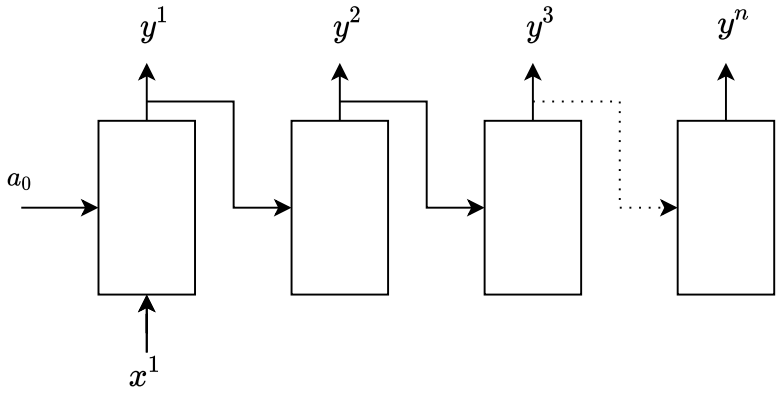

Figura 2.10: Diagrama de uma rede neural recorrente um-para-muitas. Fonte: Autor

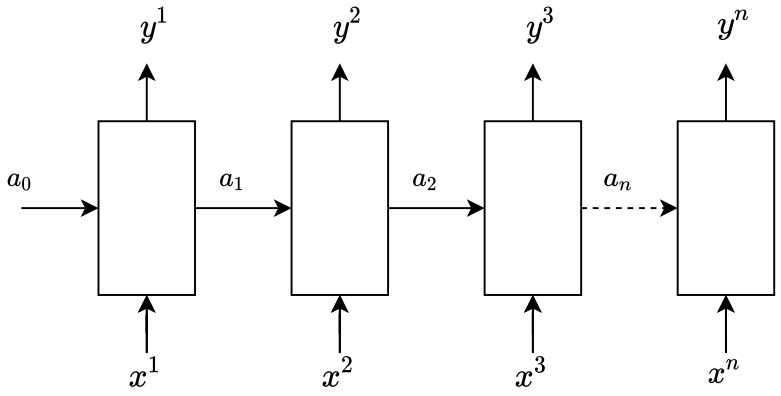

Figura 2.11: Diagrama de uma rede neural recorrente muitos-para-muitos. Fonte: Autor

Na Figura 2.12, é possível visualizar uma arquitetura de RNN com múltiplas entradas e uma saída. Uma possível aplicação para esse tipo de configuração é o problema de análise de sentimento, onde a rede recebe determinado texto, por exemplo, uma crítica de um determinado filme, e a rede tentará identificar se tal crítica é positiva ou negativa. Esse tipo de configuração também pode ser usada para aplicações de reconhecimento de ações, onde a rede tentará classificar uma ação, por exemplo, se em determinado vídeo (sequência de imagens) uma pessoa esta andando ou correndo, porém, para essa aplicação as redes RNN, devem ser usadas em conjunto com as ConvNets.

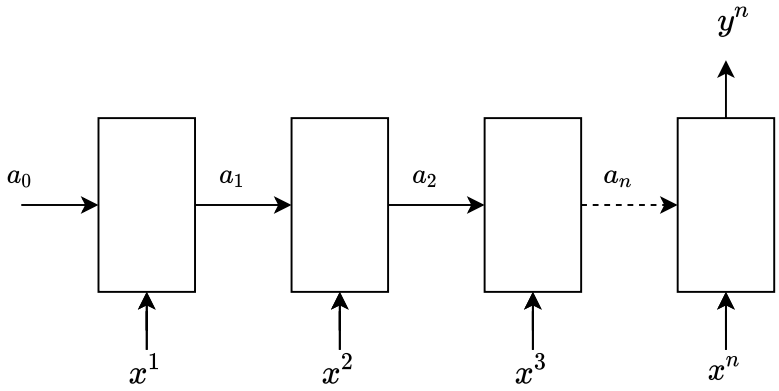

Figura 2.12: Diagrama de uma rede neural recorrente muitos-para-um. Fonte: Autor

Na teoria, as redes RNN conseguem usar informações passadas para fazer predições no presente, ou seja, possuem memória. Na prática, essa capacidade de memorização é afetada por um problema conhecido na literatura como dissipação do gradiente (do inglês, *vanishing gradient problem*). Esse problema, para fins práticos, pode ser visto como a dificuldade da rede em propagar informações, ou seja, a informação do tempo t, não necessariamente requer afirmações de estados imediatos $t-1, t-2 \ldots t-10$, mas sim de estados que foram apresentados a um tempo considerável $t-k$ e, que provavelmente, foram dissipados durante o processo de treinamento da rede.

Para contornar esse problema, foi desenvolvido um novo tipo de modelo de rede recorrente denominado redes neurais com memória de longo prazo (do inglês, *Long-short Time Memory* - LSTM). Esse tipo de modelo tem como principal característica a capacidade de lidar com longas dependências temporais, amenizando, portanto, o problema de dissipação do gradiente.

2.3.1 LSTM

As redes LSTM foram propostas primeiramente no trabalho de [41], tendo como principal característica a capacidade processar sequências mais longas e complexas. Essa capacidade é obtida com a adição de portões (do inglês, *ótico*) que permitem controlar o fluxo de informação entre as células, permitindo a rede aprender quando manter ou esquecer determinada informação [3]. Para obter uma célula de uma rede LSTM, primeiramente será representada uma rede RNN através de uma célula como visto na Figura 2.13.

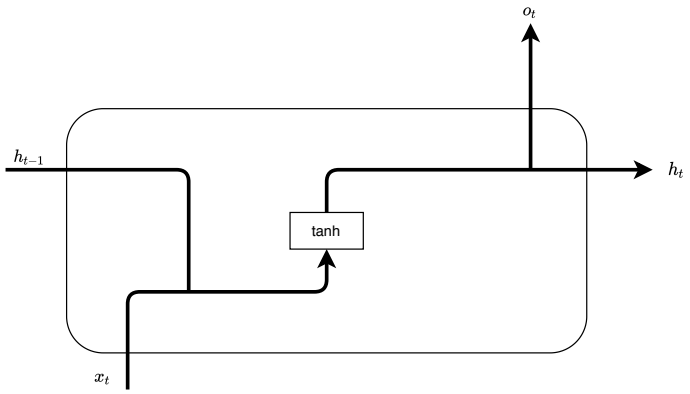

Figura 2.13: Diagrama de uma célula recorrente. Fonte: Autor

De acordo com a Figura 2.13, é possível notar que essa célula tem as seguintes entradas: a ativação do estado anterior h_{t-1} e a entrada x_t. A saída da célula é a ativação a_t, que pode ser vista como a predição dessa célula, e a ativação ou estado h_t, que alimentará a próxima célula, tendo sua saída e fluxo interno regido pela Equação (2.9).

A célula LSTM (Ver Figura 2.14), por outro lado, é construída introduzindo os conceitos de portões de atualização (do inglês, *update gate*) e esquecimento (do inglês, *forget gate*). Esse portões tem como principal objetivo regular a quantidade de informação que será mantida e passada para células posteriores.

A célula apresentada na Figura 2.14 possui uma linha principal c_{t-1}, em que os dados fluem quase sem alteração nenhuma para células posteriores, sendo denominada célula de estado. O fluxo nessa linha pode ser modificado pelo portão do esquecimento, que é obtido pela Equação (2.10), e que será responsável por decidir quais informações a célula manterá ou esquecerá. Valores próximos a 0 indicam esquecimento total e valores próximos a 1 indicam manutenção total dessa informação. Esse efeito é obtido através da aplicação da função sigmóide (Ver Figura 2.6).

$$f_t = \sigma(W_f[h_{t_i}, x_t] + b_f) \tag{2.10}$$

Em seguida, a célula LSTM realizará outra computação que selecionará quais novas informações poderão ser armazenadas na célula de estado. Nessa etapa será criado um vetor de possíveis candidatos denominado $\hat{c}$. Esse processo é descrito matematicamente pelas seguintes equações:

$$\begin{aligned} i_t &= \sigma(W_i[h_{t_i}, x_t] + b_i) \\ \hat{c}_t &= tanh(W_c[h_{t_i}, x_t] + b_c) \end{aligned} \tag{2.11}$$

Portanto, o novo valor da célula de estado c_t será calculado pelas soma das novas informações que serão adicionadas ao estado, definidas pelo produto de i_t por $\hat{c}_t$ com quais informações serão esquecidas pela célula de estado definidas pelo produto de f_t por c_{t-1}. Esse processo é regido pela equação:

$$c_t = i_t * \hat{c}_t + f_t * c_{t-1} \tag{2.12}$$

Nesse ponto, a célula LSTM sabe quais informações serão esquecidas ou propagadas e quais serão adicionadas a célula de estado c_t, ficando faltando apenas calcular qual será ativação para essa entrada o_t e a predição para esse instante de tempo h_t, que também será o estado oculto ou ativação, propagado para célula posterior. Esse processo é definido pelas seguintes equações:

$$\begin{aligned} o_t &= \sigma(W_o[h_{t_i}, x_t] + b_o \\ h_t &= tanh(c_t) * o_t \end{aligned} \tag{2.13}$$

Finalmente, uma rede LSTM pode ser formalmente e completamente descrita. Dado $x_1, x_2, \ldots, x_m$, h_{t-1} e c_{t-1}, onde m é o comprimento da sequência e $x_i \in \mathbb{R}^d$ é o vetor obtido pela concatenação de características, h_{t-1} e c_{t-1} são os estados ocultos e o estado anterior da célula LSTM (h_o e c_o são iniciados com vetores com zeros), respectivamente. O novo estado oculto e novo estado da célula são computados com as seguintes relações:

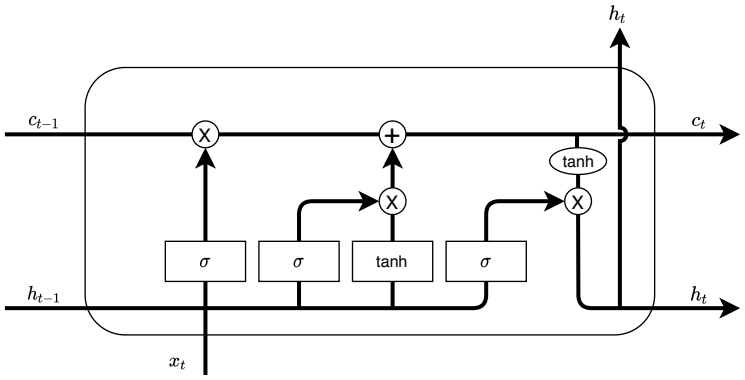

Figura 2.14: Diagrama de uma célula LSTM. Fonte: Autor

$$
\begin{aligned}
\hat{c}_t &= tanh(W_c[h_{t_i}, x_t] + b_c) \\
i_t &= \sigma(W_i[h_{t_i}, x_t] + b_i) \\
f_t &= \sigma(W_f[h_{t_i}, x_t] + b_f) \\
o_t &= \sigma(W_o[h_{t_i}, x_t] + b_o) \\
c_t &= i_t * \hat{c}_t + f_t * c_{t-1} \\
h_t &= tanh(c_t) * o_t
\end{aligned}
\tag{2.14}
$$

com σ, sendo a função de ativação sigmóide e $*$, denotando o produto de Hadamard. W_f, W_i, W_o, $W_c \in \mathbb{R}^{(N+d)\times N}$ são matrizes de pesos e b_f, b_i, b_o, $b_c \in \mathbb{R}^N$ são vetores de vieses. Matrizes de pesos e vetores de vieses são inicializados randomicamente e aprendem por uma rede neural durante a fase de treinamento. N é o tamanho da camada LSTM e d é a dimensão do vetor de características de entrada.

As redes LSTM são atualmente uma das arquiteturas de aprendizado profundo mais utilizadas em trabalhos na literatura e em aplicações comerciais. Ela é atualmente uma das principais técnicas utilizadas na área de Processamento de Linguagem Natural (PLN), e um dos principais fatores que trouxeram ganhos expressivos na qualidade de serviços de tradução automática de texto. Recentemente, ela passou também a ser utilizada em modelos de classificação de ações ou descrição de cenas a partir de vídeo, que serão apresentados na Seção 2.4.

2.4 Classificação de Vídeos

O problema de classificação de vídeos pode ser decomposto em dois sub-problemas clássicos que são: o problema da classificação de atributos espaciais (imagens) e o problema da classificação de atributos ou séries temporais, que pode ser visto como uma variação desses atributos espaciais ao longo do tempo. Cada um desses sub-problemas, classificação de imagens e classificação de séries temporais, possui vários trabalhos descritos na literatura, usualmente com metodologias baseadas em Redes Neurais Convolucionais, para resolução do problema de classificação de imagens [40, 47, 74, 82] e de Redes Recorrentes, para o problema de classificação de séries temporais [34, 60, 62, 65].

Além de modelos baseados nesse *pipeline* ou encadeamento de modelos para resolução de problemas específicos, que constitui o problema de classificação de vídeos, ainda existem modelos que conseguem classificar diretamente atributos espaço-temporais, que é o caso dos modelos baseados em convolução 3D. No entanto, esses modelos possuem um maior custo computacional e não necessariamente obtendo um resultado superior [19, 78, 79, 83].

Recentemente, novos métodos foram criados baseados na introdução de novos tipos de dados ao treinamento. Estes dados são usualmente calculados a partir do conjunto de dados original, tendo finalidades específicas que vão desde aumentar a capacidade do modelo de distinguir movimentos ou variação entre os quadros do vídeo (por exemplo, através da introdução de fluxos óticos), até o aprimoramento da capacidade do modelo de distinguir posicionamento corporal e relações entre as juntas corporais (por exemplo, através da extração de poses do atores que executam a ação em cada vídeo) [10, 37, 42, 74, 81].

Por fim, naturalmente, os modelos que fazem uso de múltiplas técnicas ou modelos, denominados de modelos multimodais, também foram apresentados na literatura. Esses modelos se baseiam no uso de vários fluxos, especializados em determinados atributos ou especificidades de cada ação, que podem ser as relações espaciais entre os objetos, determinada parte do vídeo segmentada, movimento, a pose, ou qualquer outro atributo pertinente ao domínio do problema. Estes atributos são geralmente tratados de maneira separada e depois os fluxos individuais e especializados são fundidos para formar o classificador final [47, 74, 85].

2.4.1 Redes Neurais Convolucionais 3D

As Redes Convolucionais 3D (CNN-3D) são uma extensão natural das redes convolucionais de 2 dimensões e uma abordagem natural para o problema de classificação de vídeos, pois estas conseguem modelar e classificar atributos espaço-temporais naturalmente, sem ser necessário realizar o acoplamento de diferentes modelos. As CNN-3D (Ver Figura 2.15) são obtidas através da generalização das Redes Convolucionais 2D (CNN-2D), ou seja, tanto o volume de entrada I, quanto o kernel K, têm suas dimensões expandidas para 3 dimensões. Formalmente, a operação de convolução em três dimensões pode ser definida através da seguinte equação:

$$S(i,j) = (K * I)(i,j,k) = \sum_m \sum_n \sum_p I(i+m, j+n, k+o)K(m,n,p) \tag{2.15}$$

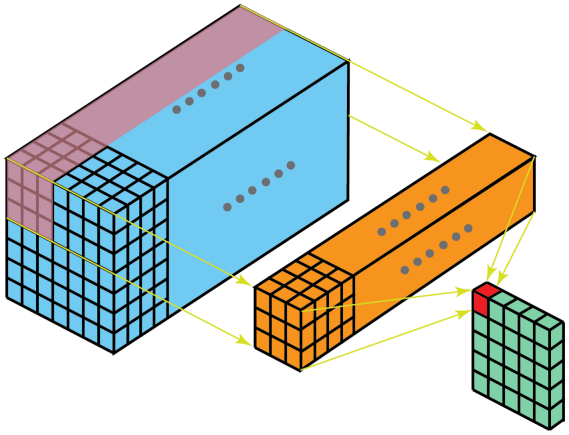

Figura 2.15: Representação gráfica da convolução 3D. Fonte: [1]

Embora as CNN-3D consigam modelar problemas com atributos espaço-temporais naturalmente, esse tipo de abordagem sofre com a sua complexidade computacional, sendo arquiteturas extremamente difíceis de ser treinadas sem a utilização de uma robusta infraestrutura. Apesar disso, algumas arquiteturas baseadas em CNN-3D [19, 32, 56, 86] têm obtido desempenhos expressivos para o problema de reconhecimento de ações em vídeos. Dentre essas arquiteturas, a que demostrou maior sucesso em termos de desempenho e reprodutibilidade foi a arquitetura I3D.

2.4.2 I3D

A arquitetura I3D foi apresentada no trabalho de [19], sendo caracterizada por ser uma rede CNN-3D profunda e com muitas ramificações (Ver Figura 2.16), sendo

treinada sobre o conjunto de dados de Kinetics [49], base de dados que foi construída visando obter uma maior diversidade e variabilidade de conjuntos de dados de ações em vídeos.

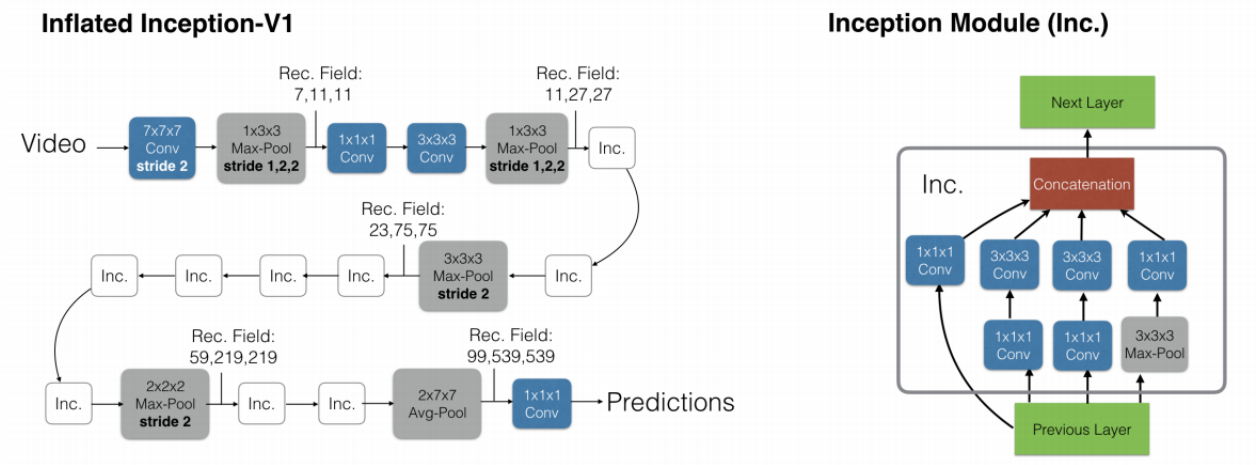

Figura 2.16: Arquitetura de uma rede I3D. Fonte: [19]

Além disso, outra característica desse modelo é aproveitar os modelos 2D pré-treinados, onde os autores repetem seus pesos 2D pré-treinados na 3ª dimensão. A entrada de fluxo espacial agora consiste em quadros empilhados na dimensão de tempo, em vez de quadros únicos, como nas arquiteturas básicas de dois fluxos.

A principal contribuição desse trabalho foi a demonstração de evidências que demostram o benefício de usar de Redes Convolucionais 2D pré-treinadas para o treinamento de CNN-3D. O conjunto de dados Kinetics e a disponibilização do código fonte [1] foram outras grandes contribuições desse artigo.

Por ser uma arquitetura robusta treinada sobre um grande conjunto de dados de ações, mais de 400 ações distintas, torna-se bastante atrativa para realização de transparência de aprendizado: técnica de aprendizado profundo onde um modelo treinado para outro propósito, a partir de um base de dados mais robusta e generalista, é reutilizado para um segundo propósito relacionado. Esse processo é realizado através de congelamento dos pesos das camadas inciais e intermediárias da rede, sendo realizado o treinamento apenas da camada final, que é construída com o intuito de especializar esse rede para um novo conjunto de dados e classes.

[1] https://github.com/deepmind/kinetics-i3d

2.4.3 CNN + LSTM

A arquitetura formada pela junção das Redes Neurais Convolucionais e Redes Recorrentes foi apresenta primeiramente no trabalho de [34], denominada de arquitetura de Redes Convolucionais Recorrentes de Longo Prazo (do inglês, *Long-term Recurrent Convolutional Networks* - LRCN). As redes LRCN têm como principal característica a divisão do problema de classificação de vídeos em dois sub-problemas: um problema de classificação de atributos espaciais e um problema de classificação de atributos temporais. Esses sub-problemas são tratados, respectivamente, por redes CNN-2D e LSTM.

Essa arquitetura pode ser decomposta em dois estágios: A CNN-2D realiza a compressão de dimensão, componente conhecido como codificador (do inglês, *encoder*) de cada *frame* do vídeo $I(t)$ em um vetor 1D $z(t)$. Essa transformação é sumarizada a seguir:

$$f_{CNN}(I(t)) = z(t) \tag{2.16}$$

Em seguida, a rede LSTM recebe uma sequência $z(t)$ codificada pela rede CNN e produz outro vetor 1D denominado $h(t)$, componente conhecido como decodificador (do inglês, *decoder*). Uma camada completamente conectada é concatenada ao fim e será responsável por produzir as predições categóricas.

A principal vantagem dessa arquitetura, quando comparada com arquiteturas baseadas em CNN-3D é o relativo baixo custo computacional. Enquanto uma rede CNN-3D apresenta parâmetros treináveis na ordem de centenas de milhões, uma rede baseada na arquitetura LRCN equivalente, geralmente, possui dezenas de milhões de parâmetros.

2.4.4 Arquiteturas com Múltiplos Fluxos

Arquiteturas com múltiplos fluxos são uma ideia natural que surgem a partir da grande quantidades de sub-problemas que o problema de classificação de vídeos pode apresentar para determinado domínio de problema. Por exemplo, no contexto do reconhecimento ou classificação de Libras, conforme visto na Seção 2.1, sabe-se que um sinal não é completamente representado apenas pela configuração das mãos, mas também pela posição da mão no corpo, pelo seu movimento, orientação, movimento

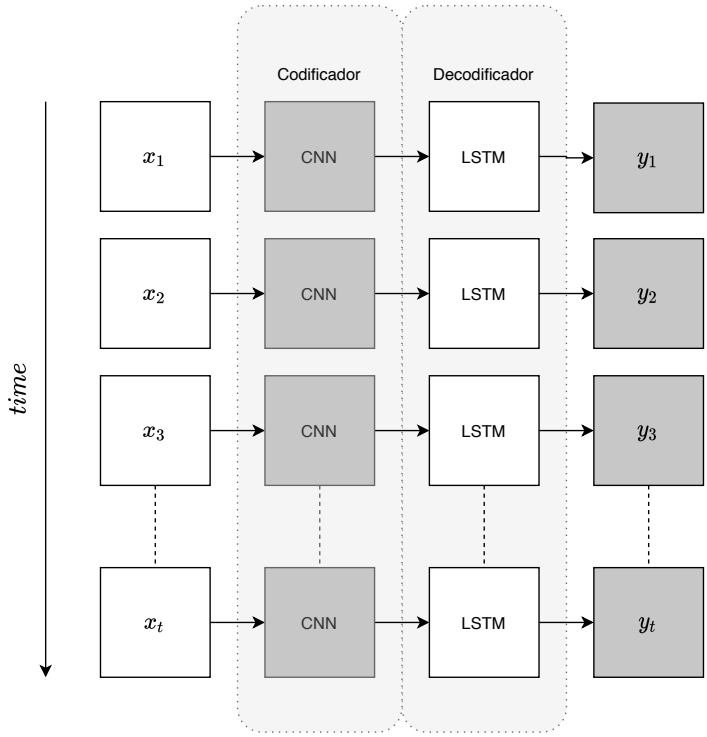

Figura 2.17: Representação gráfica de uma arquitetura de rede LRCN. Fonte: Autor.

corporal e pelas expressões faciais. A detecção e classificação dessas diferentes partes são abordadas em diversos estudos e apresentam técnicas e arquiteturas distintas na literatura. Portanto, é perfeitamente lógica a concepção de uma arquitetura especializada para um domínio específico composta de diferentes fluxos, cada uma especializada em determinado sub-problema ou atributo desse domínio.

A Figura 2.18 apresenta uma representação genérica de uma arquitetura de múltiplos fluxos. É possível observar que esse tipo de arquitetura utiliza o mesmo conjunto de dados brutos, porém, esses dados podem precisar passar por um estágio de **pré-processamento**, ou transformação, para adequá-los para o tipo de dados requeridos pelo processamento usado nesse fluxo, onde a etapa de **processamento** representa qualquer arquitetura de rede neurais disponíveis na literatura. Por fim, todos esses fluxos passam por um processo de **fusão**, para formar o classificador final, sendo esse processo extremamente crítico para o desempenho desse tipo de arquitetura, requirindo um estudo experimental em cada caso. Usualmente, esse processo de fusão pode ter caráter concatenativo, aditivo, subtrativo, multiplicativo, estatístico ou mesmo usar outros métodos de aprendizado de máquina, tais como Máquina de Vetor de Suporte (do inglês, *Support Vector Machine* - SVM) [74].

Apesar desse tipo de método trazer possíveis ganhos para o desempenho do classificador, geralmente ele possui um alto custo computacional, uma vez que o custo

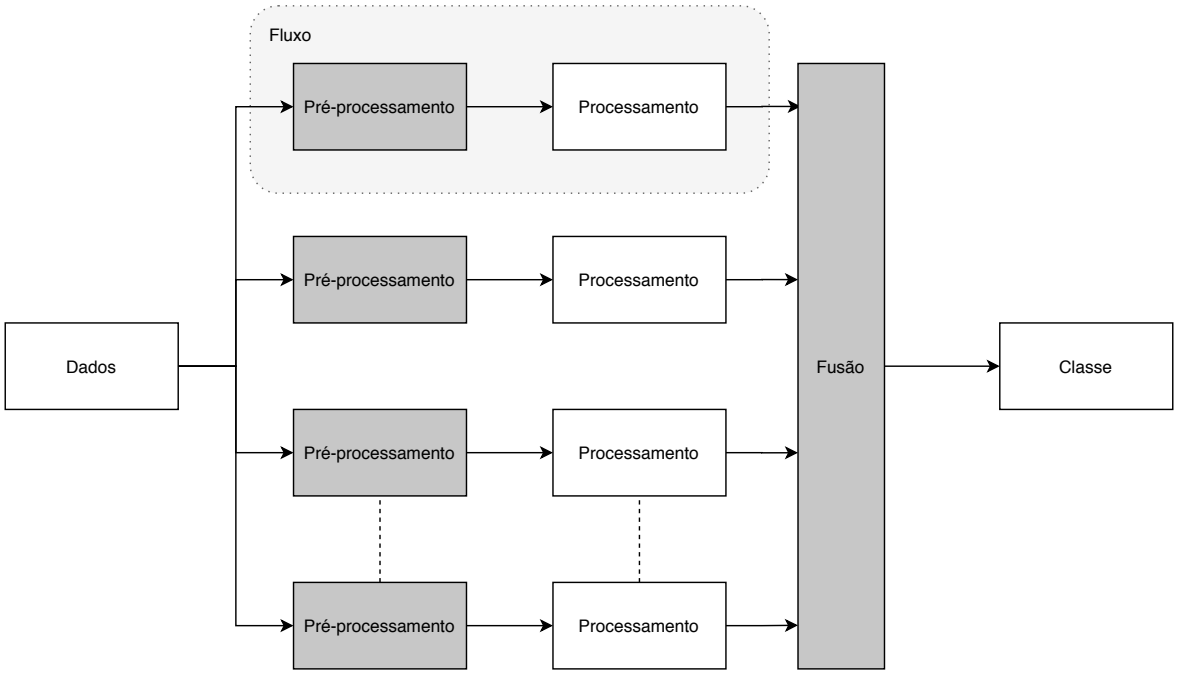

Figura 2.18: Representação gráfica de uma arquitetura com múltiplos fluxos. Fonte: Autor.

é formado pela soma dos custos individuais de cada fluxo, que são usualmente consideráveis. Esse tipo de situação dificulta colocar esses tipos de modelo em produção, uma vez que eles demandam servidores com grande poder de processamento. Isso acaba restringindo ou limitando o seu uso em ambientes ou plataformas que possuem recursos computacionais mais limitados, como, por exemplo, dispositivos móveis ou sistemas embarcados. Uma alternativa comumente usada para contornar essa limitação é expor a solução como um serviço, por exemplo, através de APIs, suportados por servidores com grande poder de processamento, e usar os dispositivos móveis ou sistemas embarcados como clientes deste tipo de serviço.

2.5 Fluxo Ótico

Uma importante técnica de estimação do movimento aparente de *pixels* em sequências de imagens, com relação a câmera ou objeto, é o chamado fluxo ótico (do inglês, *optical flow*). Esse tipo de técnica funciona como uma boa aproximação do movimento físico projetado em um plano ortogonal. Essa informação sobre a região e velocidade de movimento pode ser útil em diversas aplicações.

A técnica de fluxo ótico tem várias aplicações, sendo as mais clássicas a compressão e estabilização de vídeos, ou o mapeamento do padrões de movimentos [16]. Essa última, com a popularização do aprendizado profundo, passou a ser estendida e

incorporada em diversas arquiteturas utilizadas em problemas relacionados ao reconhecimento de movimentos em vídeos, tais como os problemas de reconhecimento de ações ou rastreamento de objetos, mas sua aplicabilidade é ainda mais extensa e também pode ser utilizado em diversos outros campos, tais como visão robótica e aplicações de vigilância [30].

2.5.1 Formalização do problema

Fluxo ótico pode ser definido como o movimento de objetos entre uma sequência de *frames* consecutivos. O problema do fluxo ótico está expresso na Figura 2.19.

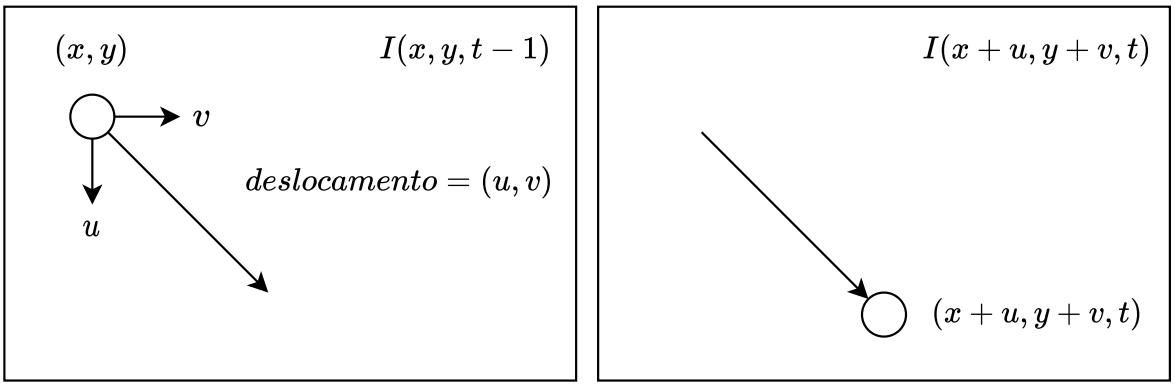

Figura 2.19: Diagrama esquemático para derivação da equação do fluxo ótico. Fonte: Autor

Para *frames* consecutivos, podemos expressar a intensidade da imagem I em função do espaço (x, y) e do tempo t. Em outras palavras, ao se tomar a primeira imagem $I(x, y, t-1)$ e mover os seus *pixels* por (u, v) sobre o tempo t, pode-se obter uma nova imagem $I(x+u, y+v, t)$.

A partir da premissa que a intensidade da imagem I é constante entre *frames* consecutivos, pode-se obter a equação da constância do brilho:

$$I(x, y, t-1) = I(x + u(x, y), y + u(x, y), t) \tag{2.17}$$

Embora essa premissa seja constantemente utilizada pelos pesquisadores, no mundo real, a natureza frequentemente viola essa condição. Como resultado, às vezes, a aproximação do fluxo ótico se torna inconsistente. A seguir, é aplicada a expansão de Taylor para linearização do lado direito da equação:

$$\begin{aligned} I(x+u, y+v, t) &\approx I(x, y, t-1) + \frac{\partial I}{\partial x}u(x,y) + \frac{\partial I}{\partial y}v(x,y) + I_t \\ I(x+u, y+v, t) - I(x, y, t-1) &= \frac{\partial I}{\partial x}u(x,y) + \frac{\partial I}{\partial y}v(x,y) + I_t \end{aligned} \tag{2.18}$$

Essa equação pode ser simplificada e manipulada para caracterizar a equação do fluxo ótico:

$$\frac{\partial I}{\partial x}u(x,y) + \frac{\partial I}{\partial y}v(x,y) + I_t \approx 0 \tag{2.19}$$

Por fim, ela é reescrita em notação matricial resultando na equação de fluxo ótico:

$$\nabla I \cdot \begin{bmatrix} u & v \end{bmatrix}^T + I_t = 0 \tag{2.20}$$

Apesar de ter o problema do fluxo ótico matematicamente formalizado através da Equação (2.20), sua resolução não é tão trivial, pois a mesma apresenta duas variáveis desconhecidas u e v e apenas uma equação. Logo, a sua resolução é dependente de métodos numéricos, sendo o método mais conhecido e amplamente utilizado, o método de Lucas-Kanade.

2.5.2 Método de Lucas-Kanade

O método de Lucas-Kanade [57] é um método não-iterativo que assume um fluxo ótico constante local. Sua principal premissa é que o fluxo ótico (u, v) é constante em pequenas janelas de tamanhos $m \times m$ e com $m > 1$, sendo o mesmo centrado nestas janelas. Cada um desses *pixels* são numerados de $1...n$. Uma representação desse processo pode ser visualizada na Figura 2.20.

Dessa forma, pode-se encontrar o seguinte conjunto de equações em sua representação matricial:

$$\begin{bmatrix} I_x(p_1) & I_y(p_1) \\ I_x(p_2) & I_y(p_2) \\ \vdots & \vdots \\ I_x(p_n) & I_y(p_n) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(p_1) \\ I_t(p_2) \\ \vdots \\ I_y(p_n) \end{bmatrix} \tag{2.21}$$

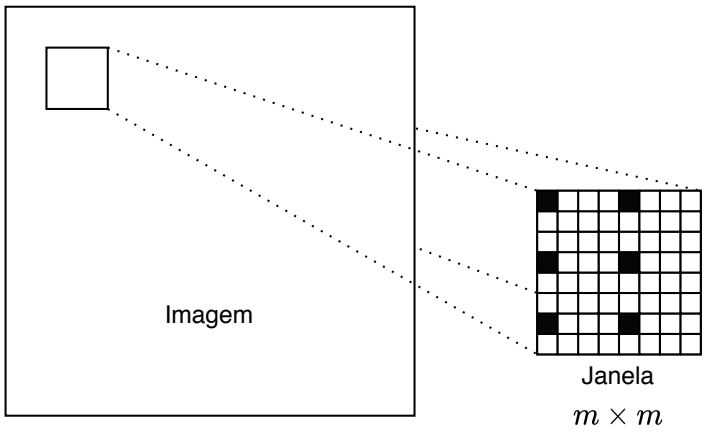

Figura 2.20: Diagrama esquemático das janelas do método de Lucas-Kanede. Fonte: Autor

Portanto, com a utilização de janelas, a quantidade de equações torna-se maior do que a quantidade de incógnitas, caracterizando o sistema como sobredeterminado. Para contornar esse problema, é aplicado o método dos mínimos quadrados, o que implica em:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \sum I_x(p_i)I_x(p_i) & \sum I_x(p_i)I_y(p_i) \\ \sum I_x(p_i)I_y(p_i) & \sum I_y(p_i)I_y(p_i) \end{bmatrix}^{-1} - \begin{bmatrix} \sum I_x(p_i)I_t(p_i) \\ \sum I_y(p_i)I_t(p_i) \end{bmatrix} \tag{2.22}$$

Onde u e v denotam o movimento de x e y sobre o tempo, respectivamente. Logo, resolver o sistema da Equação (2.22) equivale a resolver o problema de fluxo ótico. O método de Lucas-Kanade está disponível em diversas bibliotecas de visão computacional, sendo sua implementação mais amplamente usada disponível através de biblioteca OpenCV [16].

2.6 Extração de Poses

A estimação de poses é um problema da área de visão computacional que almeja realizar a estimação da posição e orientação de um objeto, usualmente de um ser humano. Esse problema pode ser resumido na detecção de um conjunto de pontos chaves (do inglês, *keypoints*), que são descritos através da tupla $(x, y, rótulo)$, onde o x e y representam a posição espacial e o rótulo representa a parte que foi detectada nessa posição, o braço direito, por exemplo. Quando esses *keypoints* são conectados, eles descrevem um esqueleto que codifica a pose ou ação que foi detectada.

Dentro desse problema, o método e ferramenta estado-de-arte é o proposto em

[18][2]. O OpenPose é uma biblioteca de detecção de articulações humanas, com licença livre para aplicações não comerciais e científicas, sendo capaz de detectar até 135 pontos que descrevam a postura corporal, as mãos e o rosto. Na Figura 2.21, é possível observar uma pose detectada através do OpenPose. De acordo com a Figura 2.21, o esqueleto é obtido através da junção do conjunto de *keypoints* detectados.

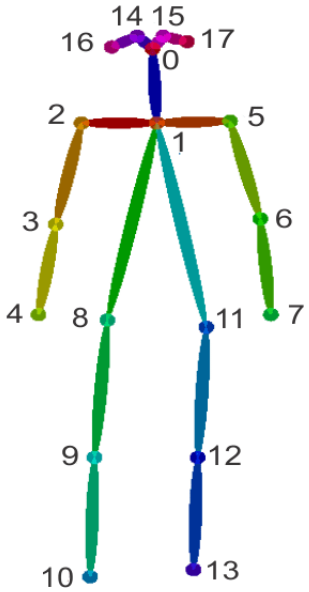

Figura 2.21: Esqueleto extraído através do OpenPose. Fonte: [18]

Uma representação geral da arquitetura usada pelo OpenPose pode ser visualizada na Figura 2.22. Seu funcionamento é dividido em 3 estágios. O primeiro estágio é responsável por realizar a extração de um mapa de características. Para isso, são utilizadas as 10 primeiras camadas da arquitetura VGG-16 [75], configurando-se assim como um estágio de extração de características.

No segundo etapa, uma CNN de dois estágios de múltiplos ramos é usada, onde o primeiro ramo prevê um conjunto de mapas de confiança 2D (S) dos locais das partes do corpo (por exemplo, cotovelo, joelho, etc.), e o segundo ramo é responsável por prever um conjunto de campos vetoriais 2D (L) de afinidades da partes corporais. Esses campos codificam o grau de associação entre as diferentes partes do corpo.

Por fim, no estágio final, os mapas de confiança e afinidade são analisados por inferência gananciosa para produzir os *keypoints* 2D para todas as pessoas na imagem. Os pontos podem ser divididos em corpo, mãos e resto, tendo cada uma dessas partes uma rede própria para predição dos *keypoints*.

O OpenPose tem ganhado grande atenção da comunidade científica, pois as poses e postural corporal detectadas podem serem utilizadas como um novo tipo de

[2]https://github.com/CMU-Perceptual-Computing-Lab/openpose

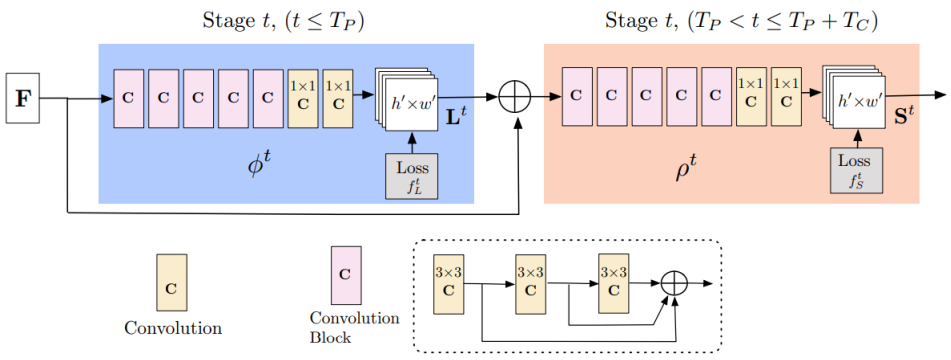

Figura 2.22: Arquitetura de rede neural usada pelo OpenPose. Fonte: [18]

dado, gerado a partir de um conjunto de dados usual (por exemplos, vídeos). Com isso, ele pode compor sistemas de detecção de ações em tempo real, aumentando a capacidade do modelo de discretizar postura corporal humana.

2.7 Considerações Finais

Neste capítulo, visou-se discorrer sobre toda a fundamentação teórica necessária, para caracterização da problemática de estudo, isto é, sobre todos os elementos que serão tratados no âmbito do presente trabalho, seja sobre a temática de estudo em si, ou dos demais elementos que fazem parte da metodologia, que será apresentada no capítulo seguinte.

Como esta dissertação tem sua metodologia baseada em técnicas ou modelos de reconhecimento de ações baseadas em aprendizado profundo, procurou-se fazer uma revisão sucinta das principais abordagens que obtiveram sucesso e podem ser aplicadas para o problema de reconhecimento de sinais de Libras.

No próximo capítulo, será apresentada uma seleção de trabalhos relacionados ao tema deste trabalho, procurando mostrar o estado da arte, as diferentes técnicas de aprendizado de máquina que já foram aplicadas na resolução desse problema, bem como trabalhos que serviram de inspiração para a metodologia, que será apresentada no Capítulo 4 desta dissertação.

Capítulo 3

Trabalhos Relacionados

O processo de reconhecimento de sinais em língua de sinais pode ser categorizado nos seguintes estágios: aquisição de dados, pré-processamento, segmentação, extração de características e classificação [21]. Porém, no contexto de aprendizado profundo (do inglês, *Deep Learning*), esse *pipeline* é simplificado e automatizado, tendo em vista que as redes neurais convolucionais englobam os estágios de segmentação, extração de características e classificação. A literatura apresenta várias técnicas para classificação de sinais, que podem ser usualmente categorizadas nos grupos de classificadores lineares, redes neurais e classificadores probabilísticos.

Akmeliawati et al. [9] aplicaram Redes Neurais Artificiais (do inglês, *artificial neural networks* - ANN) com 7392 amostras para treinar um sistema para reconhecer 13 gestos, usando uma única ANN com 45 neurônios na camada de entrada e 14 na camada de saída com duas camadas ocultas. A solução apresenta uma acurácia média de 96,02%, porém tal método exige o uso de uma luva criada pelos autores para que seja possível detecta-la, a fim de se obter as features necessárias para a ANN. Em outro trabalho, *Gesture Recognition Fuzzy Neural Network* (GRFNN), introduzido por [15] para adaptar controle fuzzy para aprendizado de parâmetros, tem a vantagem da eliminação da necessidade da pré-seleção de padrões de treinamento melhora a acurácia. No reconhecimento de 36 gestos estáticos da Língua Americana de Sinais (*American Sign Language* - ASL), a rede GRFNN obteve acurácia de 92,19%.

Em [26], Pugeault e Bowden propuseram um sistema de reconhecimento de ASL em tempo real, usando tanto imagens RGB, quanto imagens de profundidade capturadas pelo Microsoft Kinect. Os autores fizeram uso da técnica de floresta aleatória (do inglês, *Random Forest* - RF) em um conjunto de dados contendo 131548 amostras, e obtiveram uma acurácia média de 75,0%. Uma desvantagem encontrada na aplica-

ção do RF, e da maioria dos métodos de aprendizado de máquina para classificação de sinais de língua de sinais, é a características de alguns gestos de só serem completamente caracterizados através de atributos espaço-temporais, impossibilitando o uso dessas técnicas para um problema de domínio geral.

Pigou et al. [70] aplicou CNNs em uma base de dados com 6600 amostras de 20 gestos distintos, sendo dividida em 4600 amostras para o conjunto de treinamento e 2000 amostras para o conjunto de validação. A acurácia média obtida foi de 91,7%. Uma desvantagem deste método é a necessidade de dados de profundidade da imagem, obtidos com auxílio do Microsoft Kinect, o que inviabiliza a utilização outdoor.

Bheda e Radpour [14] usaram Redes Neurais Convolucionais (CNNs) para classificação de letras e dígitos em ASL usando um conjunto de dados contendo 25 imagens de 5 pessoas distintas para cada letra e dígito do alfabeto. As amostras tinham dimensões fixas de 200x200 pixels. A acurácia média obtida nos testes deste trabalho foi de 82,5%. No entanto, a pequena quantidade de amostras presentes para treinamento, principalmente no contexto de aprendizado profundo, não permitiu que o modelo generalizasse, causando classificações incorretas decorrentes de tonalidades de peles diferentes ou pequenas variações de luz no ambiente.

Em [62], Masood, Srivastava, Thuwal e Ahmad propuseram o uso de uma arquitetura combinada de Redes Convolucionais 2D (CNN 2D) com Redes LSTM para a classificação de características espaço-temporais, arquitetura descrita primeiramente no trabalho de [35], visando realizar a classificação de gestos da Língua de Sinais Argentina (LSA). O conjunto de dados consiste de 64 classes, reproduzidas por 10 intérpretes de LSA, totalizando cerca de 3200 vídeos. Desse total, para cada cada execução, foram selecionados 8 vídeos de cada sinal para o conjunto de treinamento e 2 vídeos de cada sinal para o conjunto de testes.

Foram propostos dois modelos, sendo um deles com um número maior de camadas, e, consequentemente, uma quantidade de parâmetros mais elevada no estágio da LSTM. O modelo mais complexo apresentou uma maior acurácia, com cerca de 95,6% para o conjunto de testes, contra 91,6% do modelo com menos parâmetros. Apesar da boa acurácia, a metodologia de divisão entre conjunto de treinamento e conjunto de teste adotada nesse trabalho é susceptível a sobreajuste (do inglês, *overfitting*), situação onde o modelo se sobreajusta aos dados apresentados durante o conjunto de treinamento e perde a capacidade de generalizar para novas instâncias apresentadas ao modelo durante a fase de inferência. Para um experimento mais robusto, o recomendável seria que os intérpretes humanos que estão no conjunto de testes não estivessem no conjunto de treinamento.

No trabalho de [61], Majd e Safabakhsh introduziram uma nova arquitetura denominada *Correlational Convolutional* LSTM - C2LSTM, sendo uma evolução do modelo proposto por Donahue et al. [35]. Eles mantiveram a estrutura básica de uma arquitetura combinada de Redes Convolucionais 2D (CNN 2D) com Redes LSTM, para a classificação de características espaço-temporais, porém como uma versão modificada das Redes LSTM fazendo uso do operador de correção no processo de classificação. O modelo foi avaliado através de dois conjuntos de dados, o primeiro UFC101 [76], o qual consiste de 12.000 vídeos categorizados em 101 ações humanas. O segundo HMDB51 [36] consiste de 6766 vídeos com 30 fps categorizados em 51 ações distintas. Para esse conjunto de dados, o C2LSTM foi capaz de atingir acurácias de 92,8% e 61,3% nos conjuntos de dados, UCF101 e HMDB51, respectivamente. Em ambos os casos o estado da arte foi superado em 7%.

No trabalho de [8], foi realizado um estudo usando uma arquitetura mista de Rede Convolucional 3D e Rede Convolucional Bidirecional LSTM (ConvLSTM), mais precisamente o modelo proposto por [87]. Este modelo foi treinando para o problema de classificação de Libras usando a técnica de transferência de aprendizado sobre a base de dados ISOGD apresentada em [80], sendo caracterizada pelo uso de imagens RGB-D, isto é, imagens RGB com um canal adicional de profundidade. Os testes computacionais foram realizados sobre um conjunto de dados, que consistia em uma base de dados interna, sem acesso por parte da comunidade científica contendo 510 sinais capturados a partir de 7 intérpretes distintos com 6 repetições para cada sinal. Para esse conjunto de dados, o modelo proposto foi capaz de atingir acurácia máxima de 79,80%.

3.1 Considerações Finais

Neste capítulo, foram apresentados os principais trabalhos relacionados ao tema abordado nessa dissertação. Durante o processo de revisão sistemática, constatou-se que a temática abordada nesse trabalho é amplamente ativa, com novas publicações saindo frequentemente, demostrando a importância dessa área de pesquisa, bem como o fato de ser um problema ainda em aberto na literatura.

Uma ampla variedade de técnicas de aprendizado de máquina já foram aplicadas na resolução desse problema, porém esses trabalhos apresentam diferenças de abordagens, alguns baseados em imagens estáticas, como os trabalhos de [9, 15, 26]. Outros necessitam de hardware de sensoriamento adicional para serem viáveis [26, 70], restringindo sua reprodutibilidade em ambiente real. Os trabalhos de [61, 62] conseguem

lidar com sinais com dependência espaço-temporal, porém o trabalho [61] mais correlacionado com o presente trabalho, tem sua metodologia de experimentação imprópria, tornando difícil estimar a real capacidade de generalização desse modelo.

Até o momento em que essa revisão da literatura foi executada não foram encontrados trabalhos que abordam uma combinação dessas três características: de serem centrados no contexto de saúde dentro dos Sinais da Língua Brasileira de Sinais, usando arquiteturas multifluxos, apresentando um fluxo baseado nos *keypoints* gerados pelo OpenPose, e finalmente, com uma metodologia de experimentação mais resiliente ao problema de sobreajuste e, consequentemente, medindo a real capacidade de generalização do modelo apresentado. Também não foi encontrada nenhuma base de dados de vídeos de Libras para fins de aprendizado profundo com livre acesso e com volume de amostras e classes equiparáveis aos conjuntos de dados disponíveis na literatura internacional.

No próximo capítulo, será apresentada uma proposta de metodologia de reconhecimento de sinais de Libras baseada nos fundamentos apresentados neste capítulo. Essa metodologia é focada na característica intrínseca dos sinais Libras, que foi apresentada na Seção 2.1, de serem compostos por diferentes fonemas. Essa característica permitiu levantar a hipótese de que é possível construir uma arquitetura de múltiplos fluxos utilizando esses fonemas com técnicas específicas e, com isso, construindo um modelo de reconhecimento e classificação mais robusto.

Capítulo 4

Metodologia

Nosso sistema proposto utiliza Redes Neurais Convolucionais (CNNs), que são um tipo especializado de redes neurais para o processamento de dados que possui uma topologia conhecida, semelhante a grade, sendo usualmente usadas para processamento de imagens e vídeos, e Redes Neurais Recorrentes (RNN), arquitetura especializada em dados com dependência temporal. Essas técnicas têm obtido um enorme sucesso em várias aplicações práticas na área de reconhecimento visual e tradução automática entre linguagens.

Nosso sistema proposto usa uma arquitetura de múltiplos fluxos para reconhecer os Sinais de Libras no contexto da saúde. A principal característica dessa arquitetura é que ela é baseada em imagens RGB, fluxos óticos e *keypoints* alimentando o modelo em paralelo, com os componentes espaciais da imagem extraída das imagens RGB, as informações de velocidade de movimento codificadas, através do fluxo ótico, e as informações de postura corporal e relações entre juntas, computadas através da biblioteca OpenPose. Portanto, esperamos que a solução possa atacar múltiplos componentes distintos (ou fonemas) que formam um sinal de uma língua de sinais. As várias abordagens que consideramos são explicadas nas Seções 4.1 a 4.6.

Na Seção 4.1, será apresentada a base de dados criada para o desenvolvimento desse trabalho. Na Seção 4.2, será apresentado o pré-processamento que essa base será submetida. Na Seção 4.3, será explanado como esses dados foram divididos entre os conjuntos de treinamento, teste e validação. Na Seção 4.4, serão apresentadas as diferentes arquiteturas que foram usadas nesse trabalho, primeiro de forma individual até a apresentação da arquitetura final com múltiplos fluxos. Na Seção 4.5, serão apresentadas as condições de treinamento do modelo, desde dos hiperparâmetros utilizados até a infraestrutura utilizada para os experimentos computacionais.

4.1 Especificação da base de dados

Inicialmente será descrita o processo de aquisição e criação da base de dados utilizada neste trabalho. Para a construção da solução, tendo em vista a ausência de bases abertas disponíveis na literatura, foi especificada e criada uma base de dados própria com o apoio de usuários de Libras (intérpretes de Libras e usuários surdos). Selecionamos esses 50 sinais (Ver Tabela 4.1) com base em ocorrências reais no ambiente médico extraídas de [11], que realizaram um mapeamento da validade dos sinais e doenças expressas em Libras e sua aplicabilidade para uso na anamnese clínica de enfermagem em consultas para pessoas surdas.

A base de dados contém 50 sinais selecionados no domínio de saúde, que foram sinalizados por 10 usuários de Libras com 10 repetições. Com isso, têm-se 100 amostras (vídeos) para cada sinal, totalizando 5000 (cinco mil) vídeos na base de dados.

Tabela 4.1: Sinais capturados para compor a base de dados

Número	**Sinal**
1	Aferir pressão arterial
2	Aferir temperatura
3	Alergia
4	Alteração na fertilidade
5	Alteração sexual
6	Anemia
7	Asma
8	Azia
9	Banheiro
10	Banheiro acessível
11	Barriga flácida
12	Biópsia
13	Bronquite
14	Bruxismo
15	Bulimia
16	Cálculo renal
17	Calvície
18	Câncer
19	Capela

Continua na próxima página

Tabela 4.1 – Continuação da página anterior

Número	Sinal
20	Cardiopatia
21	Cárie
22	Catapora
23	Caxumba
24	Cego
25	Cirurgia
26	Cirurgia plástica
27	Colonoscopia
28	Consultório
29	Deficiência física
30	Deficiência auditiva
31	Deficiência intelectual
32	Deficiência múltipla
33	Deficiência visual
34	Dengue
35	Depressão
36	Derrame
37	Descontrole intestinal
38	Descontrole urinário
39	Deslocamento
40	Diabetes
41	Diarreia
42	Dificuldade em evacuar
43	Dificuldade em respirar
44	Dor
45	Dor de barriga
46	Dor de dente
47	Dor de cabeça
48	Dor de coluna
49	Dor de estômago
50	Dor no peito

A seleção dos intérpretes foi feita visando representar a variabilidade étnica

brasileira, apresentando distintas tonalidades de peles, distintos perfis corporais e com variadas faixas de idade. Essas amostras foram capturadas a partir de uma câmera de celular na qualidade HD (720p) a 30 fps, com configurações de ambientes semelhantes, contendo ao todo 5000 vídeos. As configurações do ambiente de gravação foram especificadas e fixadas, conforme ilustrado na Figura 4.1. O fundo era fixado na cor branca e o intérprete está localizado a 40 cm dele. A câmera fica posicionada a uma altura de 1,5 metros do chão e a distância entre o intérprete e a câmera foi especificada em 2,5 metros.

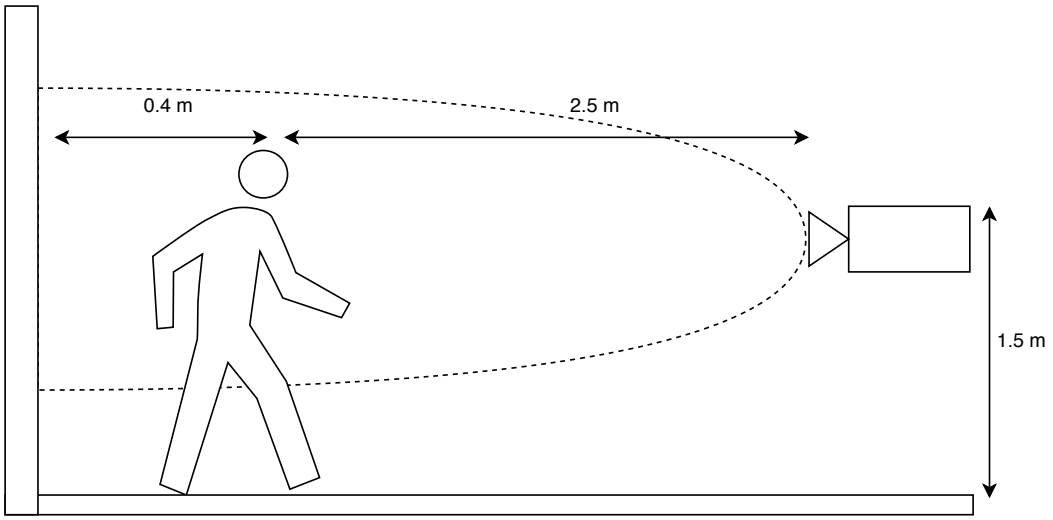

Figura 4.1: Configurações de captura de vídeos. Fonte: Autor

A Tabela 4.2 apresenta algumas estatísticas gerais sobre o conjunto de dados. É possível observar que os vídeos apresentam uma alta variabilidade na quantidade de *frames* por vídeo e, portanto, como a maioria dos métodos de aprendizado profundo exigem uma dimensão de entrada fixa, se faz necessário aplicar alguma técnica de amostragem para padronizar a quantidade de *frames*. Também foi detectado que alguns vídeos se encontram corrompidos e foram excluídos durante a fase de pré-processamento dos dados.

Tabela 4.2: Informações gerais sobre o conjunto de dados

Atributo	Máximo	Mínimo	Média
Quantidade de *frames*	237	26	50
Número de frames por segundo (fps)	30	29.3	29.9
Duração dos vídeos (s)	7.6	1.75	1

Essa base de vídeos foi usada para geração de mais duas bases específicas. A primeira delas, é uma base de poses (esqueletos), descritas através de *keypoints*. Para

isso, foi utilizada a biblioteca OpenPose, apresentada em [18]. A segunda, foi uma base contendo fluxos óticos calculadas através do algoritmo TV-L1 da biblioteca [16]. Uma visualização de uma imagem de cada base pode ser visto na Figura 4.2.

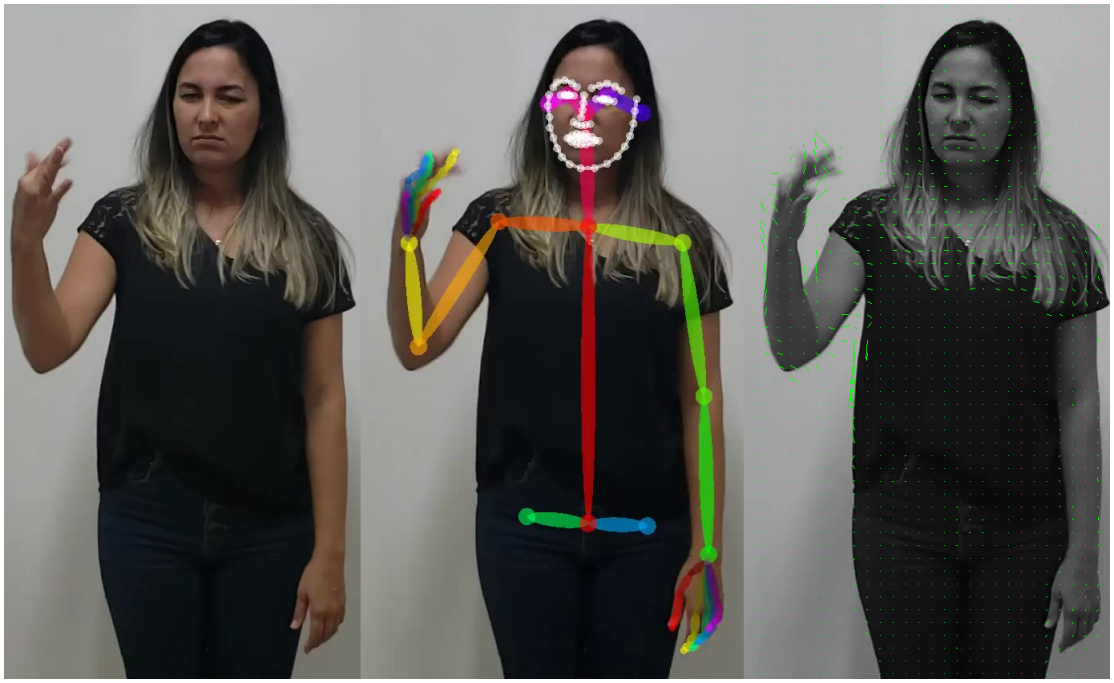

Figura 4.2: Visualização de um frame das três bases geradas. Imagem RGB (esquerda), poses (meio) e fluxo ótico (direita) Fonte: Autor.

Esses dois novos conjuntos de dados visam melhorar a detecção de movimento, aumentando a quantidade de características que podem ser usadas para distinguir os gestos, sem aumentar a quantidade de vídeos para cada classe no conjunto de dados bruto.

Além disso, o conjunto de dados resultante possui quantidade de amostras e classes semelhantes a outros conjuntos de dados encontrados na literatura, usados neste tipo de problema, como os apresentados em [25, 67, 69, 71]. No entanto, esses últimos conjuntos de dados foram desenvolvidos para propósitos gerais e não específicos, conforme proposto neste trabalho.

Após a especificação e gravação dessa base de dados, é necessário realizar uma análise exploratória sobre as características dessa base e, com isso, definir uma série de pré-processamentos que deverá ser aplicada para transforma-lá em dados úteis para o processo de Aprendizado profundo, em especial para o problema de reconhecimento de ações em vídeos.

4.2 Pré-processamento de dados

Dando continuidade ao trabalho, o conjunto de vídeos foi transformado em um conjunto de *frames*. Nesta etapa, o número de *frames* extraídos de cada vídeo foi fixado em 30, ou seja, são utilizados 30 quadros para cada amostra. Em um problema de classificação de vídeos real, a duração de cada vídeo ou ação executada pode variar em duração, o que se traduz em uma quantidade de *frames* diferente para cada vídeo, situação que implica na necessidade de fixar a quantidade *frames*, que será usada pelo modelo (Ver Figura 4.3). Usualmente, essa padronização pode ser feita de duas formas: através de uma subamostragem dos *frames* de cada vídeo, ou inserindo *frames* artificiais para completar vídeos, que sejam inferiores ao vídeo de duração máxima do conjunto de treinamento.

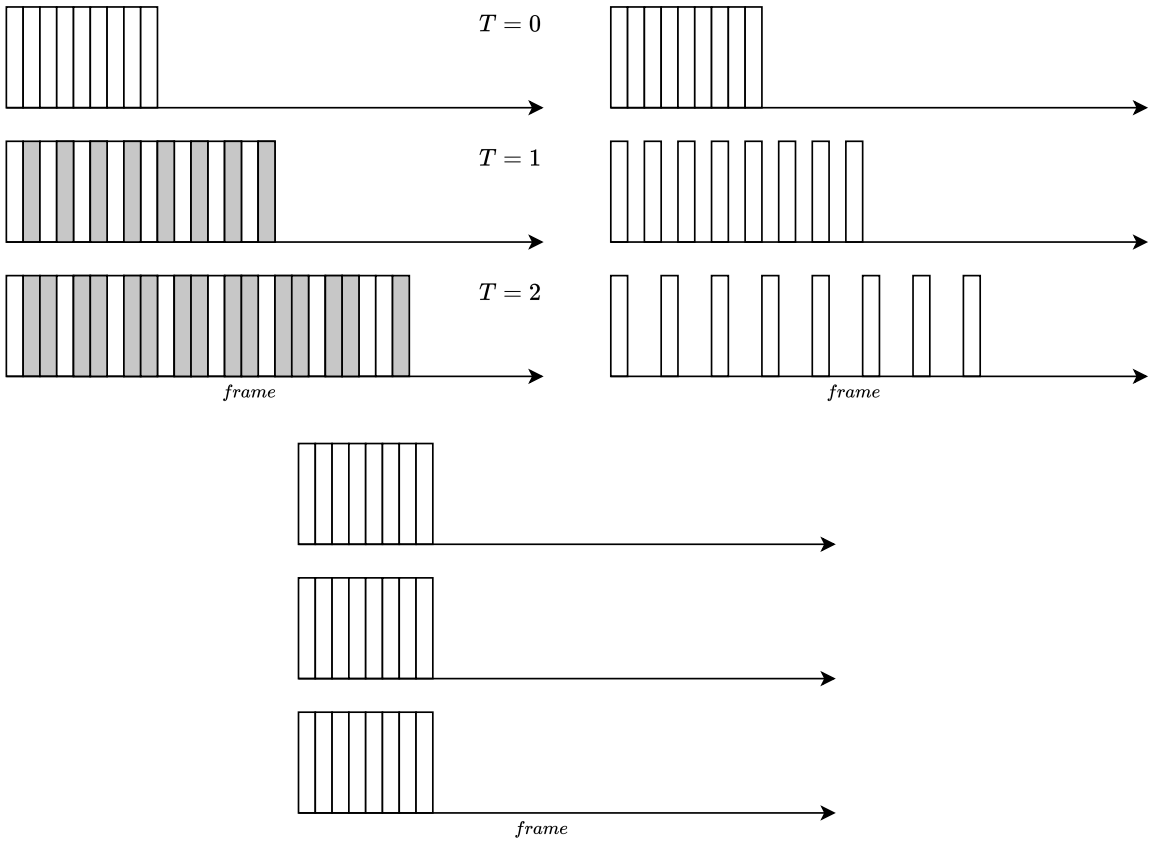

Figura 4.3: Representação gráfica do processo de sub-amostragem de *frames*. Os *frames* brancos são selecionados e os cinzas desprezados a partir de parâmetro T, que indica o período de amostragem . Fonte: Autor.

Para realizar esse processo de sub-amostragem, é preciso definir a quantidade de *frames* que será utilizada pelo modelo N, que naturalmente deve ser maior ou igual a quantidade de *frames* mínima presente no conjunto de dados. Para isso, deve-se

calcular o parâmetro T, que pode ser interpretado como o período de amostragem e é definido como a razão entre a quantidade de *frames* do vídeo $|A|$ e N.

$$T = \frac{|A|}{N} \tag{4.1}$$

Em seguida, em cada *frame* selecionado, a região de interesse, um retângulo ao redor do intérprete, foi recortado visando diminuir o volume de dados e suas dimensões foram padronizadas em 224x224 pixels. Portanto, a dimensão de entrada da rede, para cada fluxo, foi de $(batch\_size, N, 224, 224)$.

As imagens RGB também passaram por uma etapa de aumento de dados (do inglês, *data augmentation*), onde ruídos artificiais foram introduzidos visando simular variação de luz, perda de foco, distorções espaciais e outras distorções presentes numa gravação de celular não controlada. Para rotulação das instâncias foi aplicada a codificação *one-hot-encode* (OHE), necessária para classificação do modelo CNN utilizando *softmax*.

4.3 Divisão da base de dados

O processo de divisão da base de dados para o processo de treinamento e testes procurou ser o mais condizente com o método científico e, principalmente, resiliente ao problema de sobreajuste, endereçando assim que os resultados sinalizem se o modelo possui boa capacidade de generalização.

Portanto, para todos os experimentos computacionais que serão realizados, 70% dos vídeos que compõem a base (cerca de 3500 vídeos) serão utilizados como conjunto de treinamento dos modelos de aprendizado profundo, 20% serão utilizados para a validação do modelo e 10%, para uma etapa final de testes. Nessa subdivisão, o intérprete que foi utilizado no conjunto de testes não foi visto pelo modelo em nenhum momento durante o processo de treinamento.

4.4 Arquiteturas

A arquitetura proposta nesse trabalho é baseada na característica intrínseca dos Sinais da Língua de Sinais Brasileira de serem formados por 5 fonemas distintos que são: configuração de mão, ponto de articulação, movimento, orientação e expressões

não manuais. Essa característica levou à hipótese de que um modelo de aprendizado profundo que conseguisse considerar diferentes fonemas em paralelo conseguiria atingir melhor desempenho do que uma arquitetura mais generalista para classificação e reconhecimento de ações em vídeos.

Para esse fim, já existia na literatura o conceito de arquiteturas multifluxos, que são arquiteturas que apresentam múltiplos canais com dados e processamento distintos, que ao final são fundidos através de técnicas de caráter concatenativo, aditivo, subtrativo, multiplicativo, estatístico ou mesmo usam outros métodos de aprendizado de máquina, tais como SVM. Baseado nesse conceito, foram estudado dois modelos duas arquiteturas de multifluxos que serão apresentadas a seguir.

A primeira arquitetura estudada foi uma arquitetura baseada em Redes Convolucionais 2D e redes LSTM. Uma representação gráfica desse modelo pode ser visualizada na Figura 4.4, e é caracterizada pelo acoplamento de CNN-2D e LSTM, sendo usualmente conhecida como arquitetura LRCN e, com isso, sendo capaz de classificar atributos espaço-temporais. Essa arquitetura pode ser decomposta em dois estágios: a CNN realiza a compressão de dimensão, componente conhecido como codificador (do inglês, *encoder*) de cada frame do vídeo $I(t)$ em um vetor 1D $z(t)$, transformação sumarizada a seguir:

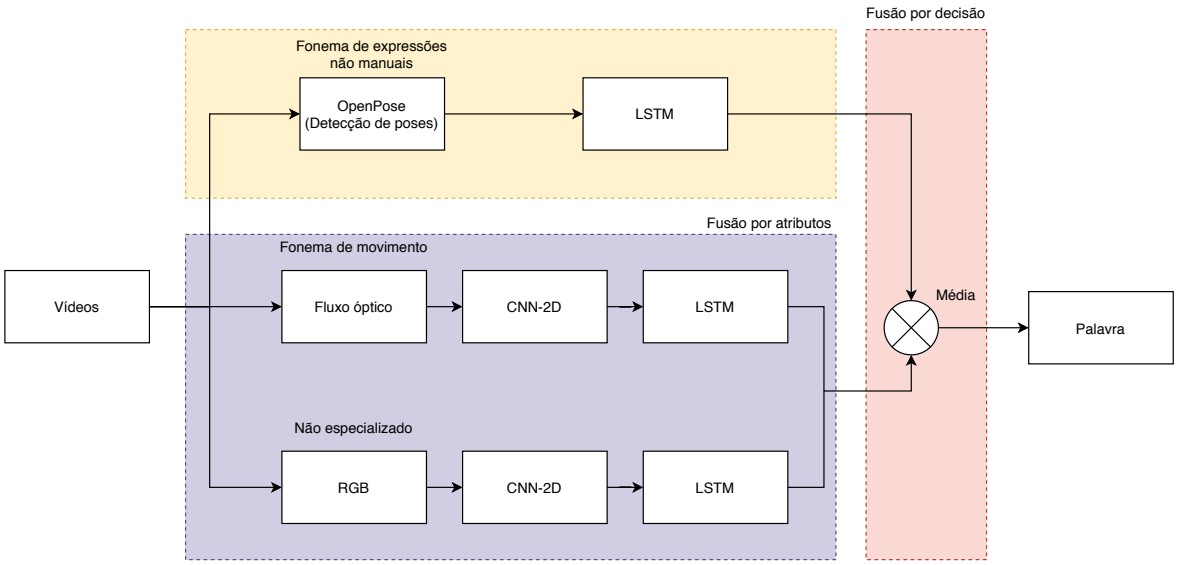

Figura 4.4: Arquitetura com 3 fluxos baseada na rede LRCN. Fonte: Autor

$$f_{CNN}(I(t)) = z(t) \tag{4.2}$$

Em seguida, a rede LSTM recebe uma sequência $z(t)$ codificada pela rede CNN e produz outro vetor 1D denominado $h(t)$, componente conhecido como decodificador (do inglês, *decoder*).

A arquitetura CNN apresenta uma configuração usual, contendo camadas convolucionais responsáveis por extrair características de imagens com função de ativação ReLU, seguida por uma normalização de *batch* (do inglês, *batch normalization*) com o objetivo de aumentar a estabilidade da rede, para fechar o bloco convolucional ($[CONV] \rightarrow [BATCH] \rightarrow [CONV] \rightarrow [BATCH] \rightarrow [POOL]$). Camadas de *pooling* foram adicionadas para realizar sub-amostragem desses mapas de características produzidos por cada bloco convolucional e, com isso, reduzindo a quantidade de parâmetros para computar. Esse bloco é repetido 5 vezes, com o número de filtros variando de 64 para 512. A maioria das camadas convolucionais realizam convolução com filtros 3x3, com exceção das duas primeiras camadas, que realizam a convolução com filtros 7x7 e 5x5, respectivamente.

Esse bloco é usado tanto no fluxo de imagens RGB, quanto no fluxo contento fluxos óticos. Em adição a esses fluxos, almejando melhorar o reconhecimento de gestos e usar aumento de dados sobre os movimentos do corpo, foi adicionado um novo fluxo responsável por classificar as poses extraídas de cada gesto. Esse fluxo é composto de apenas duas redes LSTM, onde cada rede terá 256 unidades. Esses fluxos são concatenados juntos para formar um vetor de características, ou seja, esse modelo realiza uma fusão de atributos (tensores), que por fim, alimentam uma camada totalmente conectada responsável por produzir as predições categóricas.

A segunda arquitetura estudada foi baseada na rede I3D. Uma representação gráfica desse modelo pode ser visualizada na Figura 4.5. A principal diferença desse modelo é que como a rede I3D é uma arquitetura de Rede Convolucional 3D, naturalmente capaz de modelar atributos espaço-temporais.

Esse bloco baseado na arquitetura I3D, usado tanto no fluxo de imagens RGB, quanto no fluxo de fluxos óticos. Em adição a esses fluxos, foi novamente adicionado um novo fluxo responsável por classificar as poses extraídas de cada gesto. Esse fluxo não sofrerá alterações, sendo composto de apenas duas redes LSTM, onde cada rede terá 256 unidades. Esses fluxos são concatenados juntos para formar um vetor de características, ou seja, esse modelo realiza uma fusão de atributos (tensores), que por fim, alimentam uma camada totalmente conectada responsável por produzir as predições categóricas.

Com a utilização da rede I3D, essa arquitetura também ganha uma nova característica que a diferencia da primeira arquitetura proposta, que é o fato que será

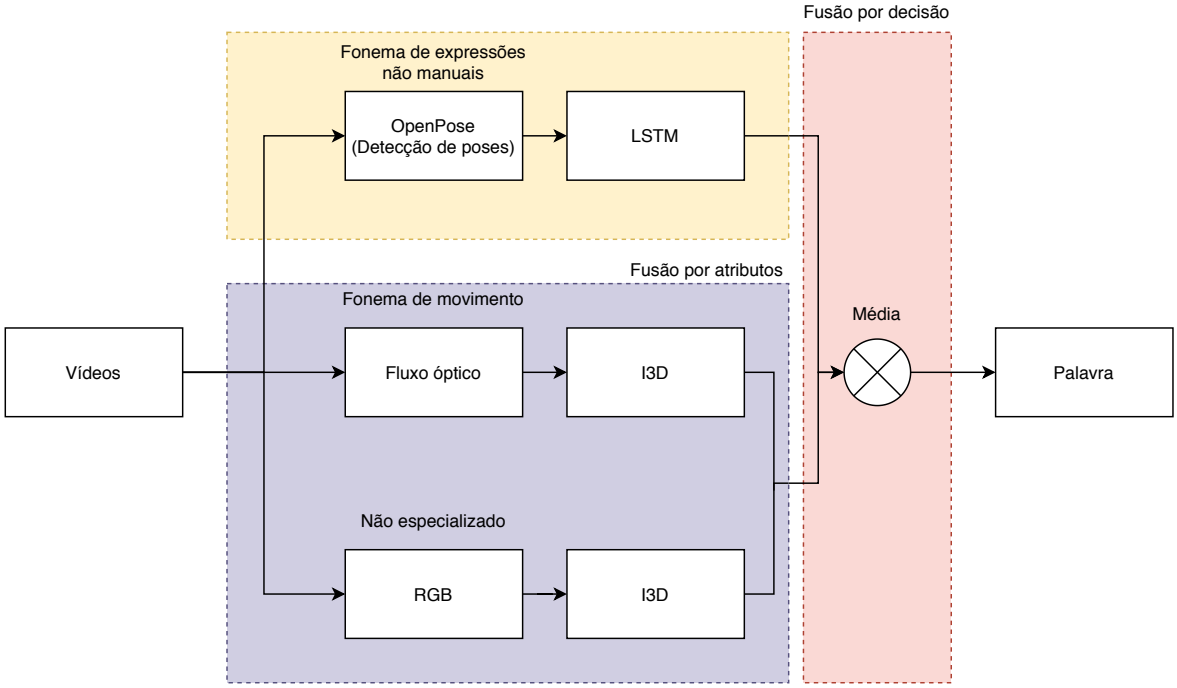

Figura 4.5: Arquitetura com 3 fluxos baseada na rede I3D. Fonte: Autor

realizada transferência de aprendizado, portanto, a rede I3D, para o fluxo baseado em imagens RGB e fluxos óticos, terá seus pesos congelados durante a fase de treinamento. Somente as camadas finais da rede serão treinadas e especializadas para reconhecer os sinais de Libras no contexto de saúde. Esse processo só será possível, porque a arquitetura I3D também é uma arquitetura de reconhecimento de ações em vídeos, logo, a transparência de aprendizado é natural e direta.

4.5 Treinamento

Para o treinamento do modelo, foi utilizado o aprendizado supervisionado, com os seguintes hiperparâmetros (Ver Tabela 4.3), a taxa de aprendizado (do inglês, *learning rate*) foi inicializada em 0.001 e o decaimento dos pesos (do inglês, *weight decay*) em $1e^{-8}$ usando o otimizador Adam [51] e a biblioteca Keras [22] [1]. A rede CNN é treinada com um *batch size* de 4, com um total de 200 iterações. Do total de amostras presentes no conjunto de dados, 70% (3500 instâncias) foram utilizadas para o treinamento da rede, 20% (1000 instâncias) para validação durante treinamento e 10% (500 instâncias) para teste com modelo treinado. Ressalta-se que o intérprete do conjunto de teste é o único que não participa do treinamento, ou seja, amostras desconhecidas

[1] https://keras.io/

pelo modelo. Este conjunto é utilizado para validar o modelo já treinado.

Tabela 4.3: Especificação dos Hiperparâmetros

Hiperparâmetro	Valor
Taxa de aprendizado	10e-4
Decaimento dos pesos	10e-6
Otimizador	Adam
Frames por vídeo	24
Dimensão da imagem	(224,224,3)
Batch size	4
Dropout	0,2

A Tabela 4.4 mostra as especificações de hardware da máquina em que todos os experimentos computacionais foram executados. Todos os testes foram realizados no ambiente Linux através da distro Ubuntu 18.04 LTS, com a Versão 10.1 do CUDA, plataforma de computação paralela da NVIDIA e a Versão 7.6 do cuDNN, biblioteca de primitivas para processamento otimizado de Redes Neurais Convolucionais em placas de vídeo da NVIDIA.

Tabela 4.4: Especificações do Hardware

Especificações	CPU	GPU
Fabricante	Intel	Nvidia
Número de Cores	4	3840
Frequência de Clock (GHz)	2,40	1,25
Memória (GB)	32	24

4.6 Considerações Finais

Durante a realização da pesquisa do referencial teórico não foram encontrados nenhum trabalho que propõe uma solução com a seguinte combinação de características: baseado em uma arquitetura de múltiplos fluxos, com a utilização de um fluxo baseado nos *keypoints* gerados pelo OpenPose e dentro do domínio de saúde. Pondera-se ainda que a Língua de Sinais Brasileira apresenta 5 fonemas básicos, portanto, esse trabalho ainda pode ser expandido para lidar com os demais fonemas, principalmente no que tange ao fonema de expressões não manuais.

Capítulo 5

Resultados

Como a arquitetura final proposta nesse trabalho usará múltiplos fluxos para para realizar a classificação e reconhecimento de gestos da Língua de Sinais Brasileira e cada arquitetura tem suas características únicas, foi necessário realizar um estudo individual em cada uma delas para analisar o comportamento acerca de viabilidade, bem como fazer o ajuste de hiperparâmetros desses modelos. Esse estudo individual também foi necessário por dois fatores: Primeiro, em função do custo computacional do modelo, pois seria inviável realizar essa sintonia de hiperparâmetros em cima de um modelo custoso, como é o modelo de múltiplos fluxos. Segundo, para analisar os impactos e as limitações de forma individual.

Dito isso, a seção de resultados será organizada da seguinte forma. Na Seção 5.1, serão apresentados os resultados para os experimentos computacionais baseados em uma arquitetura CNN + LSTM, usando como entrada imagens RGB e fluxos óticos. Na Seção 5.2, serão apresentados os resultados para os experimentos computacionais baseados na arquitetura I3D usando como entrada imagens RGB e fluxos óticos. Na Seção 5.3, serão apresentados os resultados para a arquitetura LSTM com *keypoints* gerados pelo OpenPose como entrada. Na Seção 5.4, serão apresentados os resultados para a arquitetura multifluxo. Por fim, na Seção 5.6, serão feitas considerações finais sobre esses resultados.

5.1 CNN + LSTM

Os resultados são sumarizados na Tabela 5.1, onde podemos observar que a solução proposta obteve uma acurácia média de 79,36%. No entanto, apresenta grande dificuldade em classificar sinais que possuem fonemas similares, tais como “asma” ou

"bronquite", e os mais variados sinais relacionados a condição de deficiência, por exemplo, deficiência física, intelectual ou motora, e outros sinais que produzem atributos espaço-temporais semelhantes. Essa situação pode sugerir que os dados usados, provindos das imagens e dos fluxos óticos, não fornecem atributos suficientes para distinção de classes com componentes espaço-temporais semelhantes, porém, como a acurácia média durante o treinamento atingiu 99,0%, levanta-se outra hipótese que é a da ocorrência de superajuste do modelo ao conjunto de dados utilizados durante treinamento.

Tabela 5.1: Acurácia média dos modelos estudados

Modelo	Entrada	Top 1	Top 5
LRCN	RGB	73,63%	92,88 %
LRCN	Flow	74,31%	92,88%
2-Stream LRCN	RGB+Flow	79,36%	95,01%

Para verificar o impacto do uso da regularização nos resultados do modelo, também fizemos alguns testes usando algumas técnicas de *dropout*. Em relação ao *dropout*, usamos três técnicas diferentes: (i) *dropout* espacial, que é aplicado sobre as camadas convolucionais pela desativação de pixels, (ii) *dropout* recorrente, que é aplicado na rede LSTM e (iii) *dropout* nas camadas completamente conectadas, que tiveram os melhores resultados, reduzindo o sobreajuste do modelo.

De acordo com a Tabela 5.2, o cenário que obteve o melhor resultado foi um *dropout* único de 0,2, entre as camadas finais totalmente conectadas e um *dropout* de 0,5, na rede LSTM. Entretanto, apesar da melhora da acurácia do modelo, esta ainda se encontra relativamente distante das acurácias obtidas para o conjunto de treinamento e validação, e com isso, fortalecendo a hipótese que talvez o modelo precise de mais dados (atributos) para aprender a discriminar melhor esses sinais.

Tabela 5.2: Acurácia para diferentes métodos de *dropout*

Dropout	Espacial	Recorrente	Acc
0,2	×	0,5	79,36%
0,5	×	×	76,44%
0,5	0,2	×	77,26%

Também calculamos a precisão, revocação (do inglês, *recall*) e medida-F1 (do inglês, *F1-measure*) para o modelo LRCN e para o modelo proposto (considerando os dois fluxos). Os resultados são apresentados na Tabela 5.3. Dado que não é possível

otimizar as métricas de revocação e precisão ao mesmo tempo, e como a medida f1 é definida como a média harmônica entre essas duas métricas, e possível concluir que o número de falsos positivos e falsos negativos é minimizado quando o modelo de dois fluxos é utilizado.

Tabela 5.3: Métricas para comparação entre classificadores

Modelo	Precision	Recall	f1-score
Two-stream	0,81	0,79	0,77
LRCN	0,73	0,74	0,71

Além disso, para diminuir o tempo de treinamento e iniciar o treinamento com bons pesos, isto é, pesos de uma rede previamente treinada sobre um conjunto de dados robusto, também realizamos alguns testes com alguns modelos pré-treinados. No entanto, de acordo com a Tabela 5.4, podemos observar que o modelo treinado a partir do zero teve melhores resultados.

Tabela 5.4: Acurácia para modelos pré-treinados

Modelo	Pré-treinado	Acurácia
Two-stream	×	79,36%
Two-stream	InceptionV3	64,36%
Two-stream	MobileNetV2	34,01%

Também estudamos maneiras diferentes de fundir os dois fluxos. No entanto, de acordo com a Tabela 5.5, o método de fusão que obteve melhores resultados foi o método de adição, onde os tensores produzidos por cada fluxo são adicionados e alimentam uma camada final totalmente conectada.

Tabela 5.5: Acurácia para diferentes métodos de fusão

Modelo	Tipo de fusão	Acurácia
Two-stream	Adição	79,36%
Two-stream	Concatenação	77,02%
Two-stream	Média	74,01%

5.2 I3D

Os resultados para os experimentos baseados na arquitetura I3D são sumarizados na Tabela 5.6. Podemos observar que houve uma melhora na média de quase 16% na acurácia média para todos os cenários estudados. Esse ganho de acurácia pode ser justificado principalmente por dois fatores. Primeiro, está sendo feita a transferência de aprendizado de uma arquitetura treinada para o problema de reconhecimento de ações, para um problema de reconhecimento de ações (classificação e reconhecimento de sinais em Libras). Segundo, com a diminuição da complexidade do modelo, isto é, a quantidade de parâmetros treináveis, tendo em vista que somente as camadas finais do modelo são treinadas e o resto são congeladas durante a fase de treinamento, foi possível aumentar a quantidade de *frames* amostrados de cada vídeo, de 24 para 40 *frames*, e, com isso, o modelo é alimentado com mais informações e com uma descontinuidade menor entre os *frames*.

Tabela 5.6: Acurácia média dos modelos estudados baseados na arquitetura I3D

Modelo	Entrada	Top 1	Top 5
I3D	RGB	95,32 %	100%
I3D	Flow	95,31 %	100%
2-Stream I3D	RGB+Flow	96,12 %	100%

Na Figura 5.1 é possível visualizar a matriz de confusão para o experimento com imagens RGB como dados de entrada de rede I3D. Embora tenha-se conseguido obter quase que uma diagonal perfeita, isto é, o classificador está acertando quase todas as classes, no entanto, o erro desse classificador está praticamente concentrado em uma única classe, a classe “cego”, que está sendo classificada incorretamente em 100% das vezes como a classe “bulimia”. Apesar do significado ser completamente diferente, uma análise visual sobre esses sinais mostra que os mesmos apresentam fonemas semelhantes. Portanto, existem indícios de que apenas imagens RGB não fornecem ao modelo dados suficientes para realizar a distinção de todos os fonemas presentes no conjunto de dados estudado.

Na Figura 5.2 é possível visualizar a matriz de confusão para o experimento com fluxos óticos como dados de entrada de rede I3D. Novamente, o erro desse classificador está praticamente concentrado em uma única classe, a classe “dor de dente”, que está sendo classificada incorretamente 30% das vezes como a classe “dor de cabeça” e 60% das vezes como “cárie”. Desta vez é possível observar uma justificativa mais óbvia para

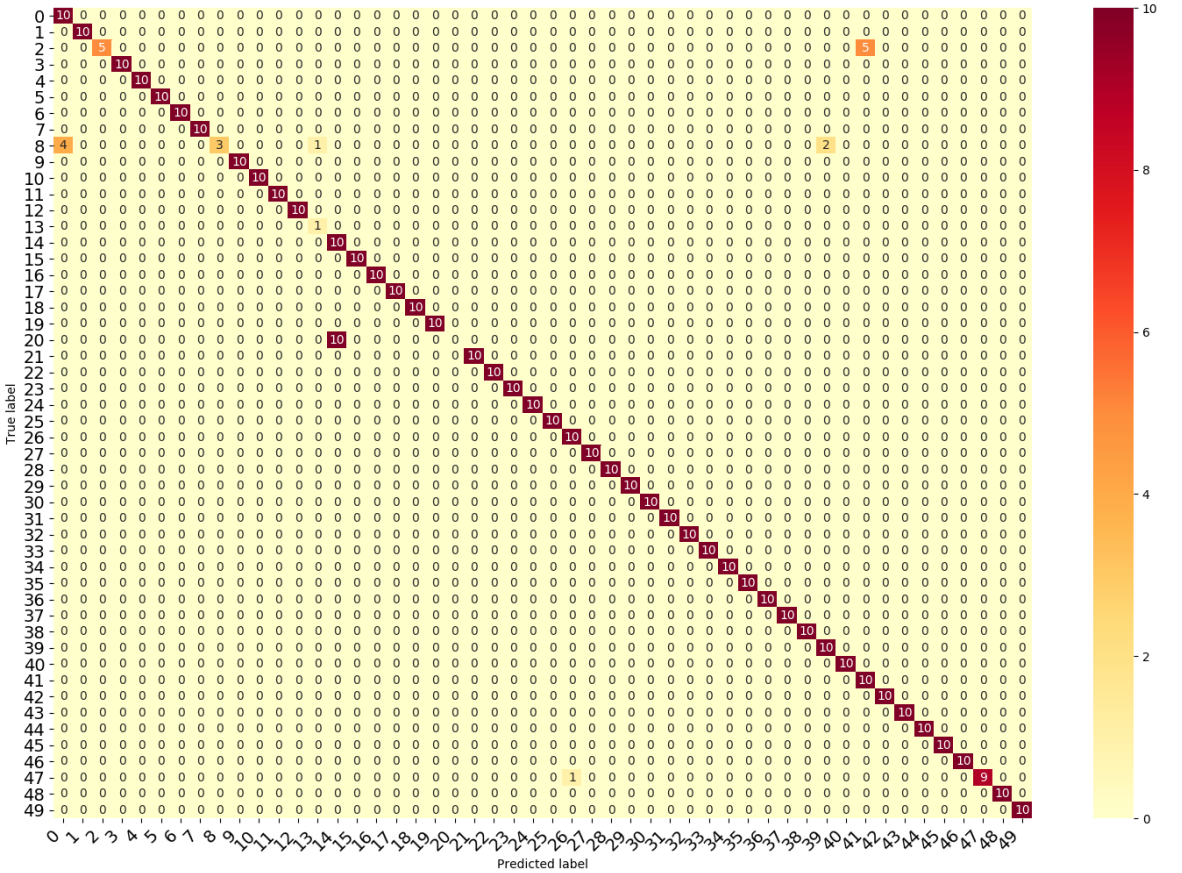

Figura 5.1: Matriz de confusão para arquitetura I3D com imagens RGB como entrada. Fonte: Autor

esse erro, já que ambos os sinais apresentam fonemas bem semelhantes, principalmente no que tange a condição de dor. Portanto, existem indícios de que, o uso de apenas fluxos óticos não fornece ao modelo dados suficientes para realizar a distinção de todos os fonemas presentes no conjunto de dados estudado.

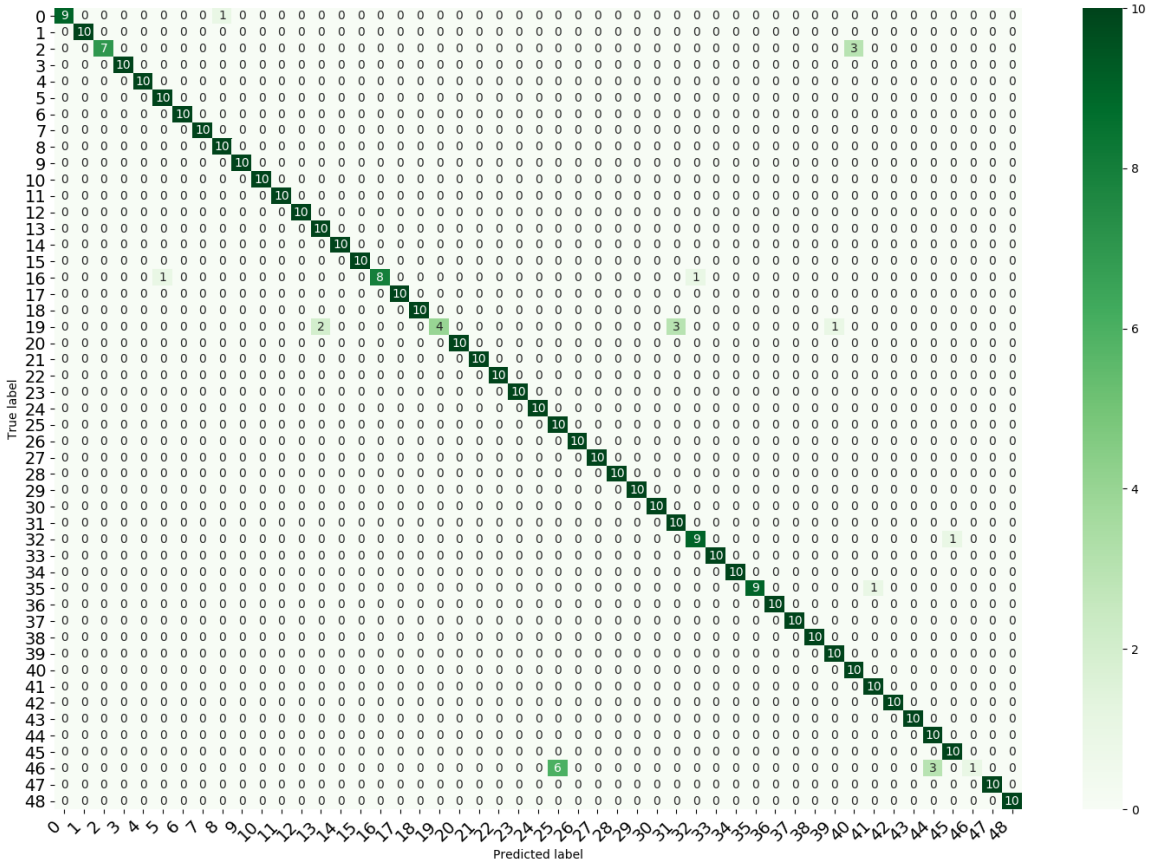

Figura 5.2: Matriz de confusão para arquitetura I3D com fluxos óticos como entrada. Fonte: Autor

Na Figura 5.3 é possível visualizar a matriz de confusão para o experimento com imagens RGB e fluxos óticos como dados de entrada da rede I3D. Apesar desse modelo ter obtido a melhor acurácia média de 96,12%, o erro desse classificador ainda está praticamente concentrado em uma única classe, a classe "dor de dente", que está sendo classificada incorretamente 100% como a classe "cárie". Uma hipótese plausível para esse resultado é que o fonema que está causando a má classificação desse sinal não é abordado nem na arquitetura de imagens RGB e nem na arquitetura de fluxos óticos. Portanto, conclui-se que o uso de apenas imagens RGB e fluxos óticos não fornece ao modelo dados suficientes para realizar a distinção de todos os fonemas presentes no conjunto de dados estudado.

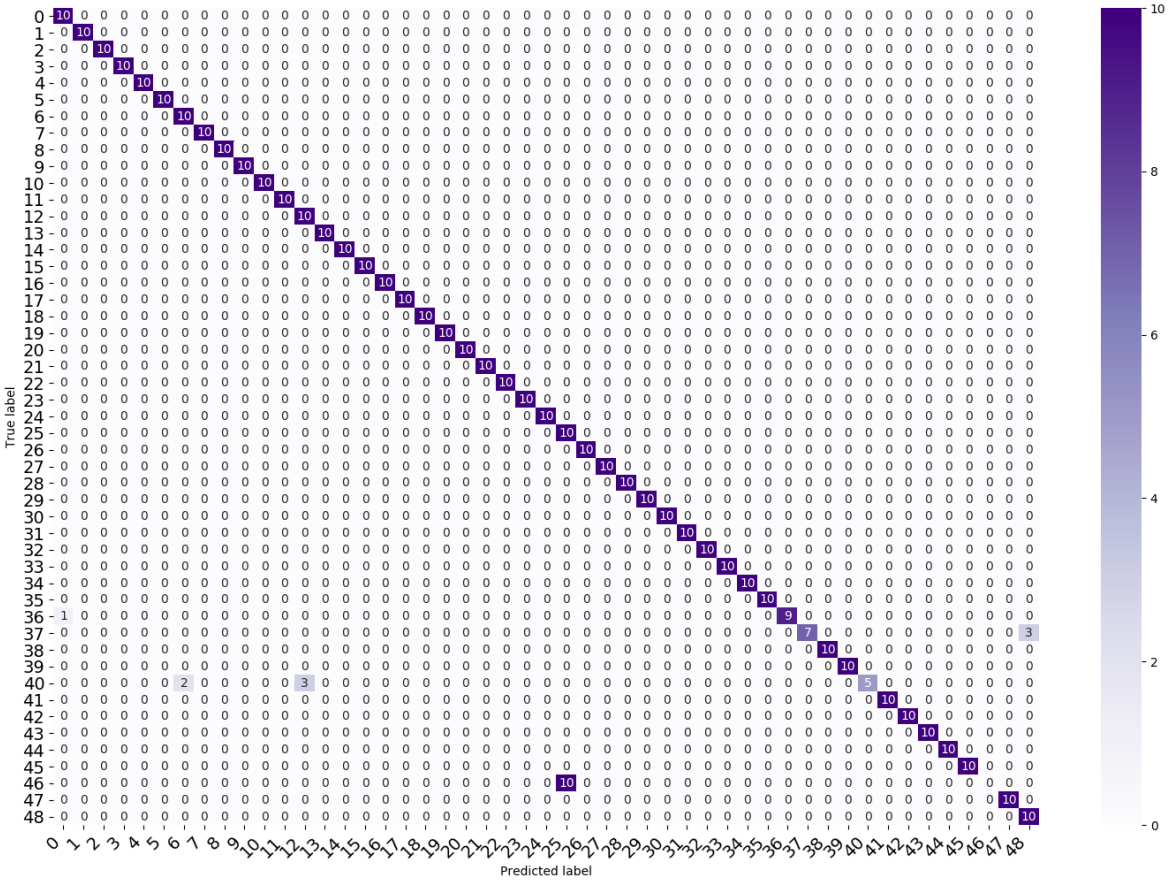

Figura 5.3: Matriz de confusão para arquitetura I3D com dois fluxos. Fonte: Autor

5.3 LSTM + *Keypoints*

Os resultados para os experimentos baseados na arquitetura LSTM com *keypoints* produzidos pelo OpenPose são sumarizados na Tabela 5.7. A acurácia foi em média 2,69% superior aos resultados obtidos pela arquitetura I3D. Além disso, foram constatados três pontos positivos ao usar apenas *keypoints* para reconhecimento e classificação de Libras. Primeiro, foi comprovada a viabilidade de se realizar a classificação de ações em vídeos usando apenas *keypoints* como entrada para um relativo grande número de classes, ou seja, a acurácia desse tipo de modelo apresenta uma performance estável, isto é, não apresenta uma queda drástica, mesmo quando o universo de classes aumenta.

Segundo, constatou-se que apesar desse dados exigirem uma série de etapas de pré-processamento para serem úteis para o processo de aprendizado profundo, o seu custo computacional e tempo de inferência são extremamente baixos. O modelo que originou os resultados apresentados na Tabela 5.7 foi treinando em apenas 10 minutos, o que contrasta muito com o tempo de treinamento exigido pela arquitetura I3D, que foi de praticamente 14 horas. Portanto, do ponto de vista computacional, introduzir esse novo fluxo não afetará muito a demanda computacional do modelo.

Tabela 5.7: Acurácia média dos modelos estudados baseados na arquitetura LSTM

Modelo	Entrada	Top 1	Top 5
LSTM	Pose	98,0%	100%

Terceiro, conforme melhor visualizado na matriz da confusão, vista na Figura 5.4, é possível observar que os erros desse classificador estão mais esparsados dentro do universo de classes do conjunto de dados, portanto, isso é um indício de que esse modelo treinado usando *keypoints* se configura um melhor classificador do que os demais e está considerando diferentes fonemas presentes nos sinais estudados. Desta forma, reforçamos a hipótese de que o uso de uma arquitetura de múltiplos fluxos considerando os diferentes fonemas da Língua de Sinais Brasileira produzirá um classificador mais robusto.

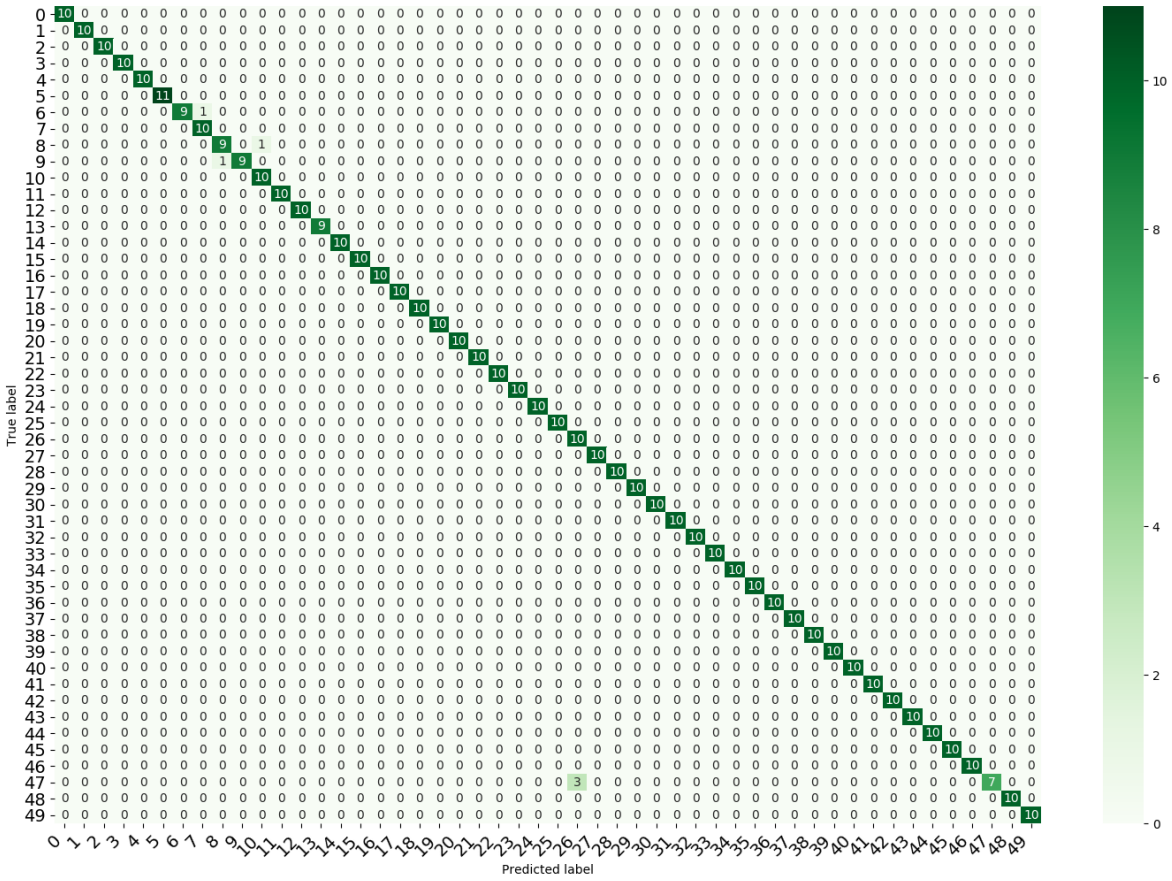

Figura 5.4: Matriz de confusão para arquitetura LSTM com keypoints como entrada. Fonte: Autor

5.4 Arquitetura Multifluxo

Finalmente, essas arquiteturas individuais são fundidas usando fusão por decisão, isto é, a predição será obtida a partir da média das predições dos fluxos individuais. Os resultados são sumarizados na Tabela 5.8, onde podemos observar que a solução proposta obteve uma acurácia média de 99,80% (Ver Figura 5.5). Essa arquitetura consegue minimizar quase que completamente os erros obtidos nos cenários de fluxos individuais, ou usando apenas dois fluxos, confirmando a hipótese de que ao se considerar diferentes fonemas de Libras, de forma especializada, consegue-se conceber uma modelo de reconhecimento de sinais de Libras muito mais robusto.

Tabela 5.8: Acurácia média dos modelos estudados baseados na arquitetura de três fluxos

Modelo	Atributos	Top 1	Top 5
3-Stream I3D	RGB+Flow+Pose (corpo)	98,80 %	100%
3-Stream I3D	RGB+Flow+Pose (mãos)	99,20 %	100%
3-Stream I3D	RGB+Flow+Pose (corpo e mãos)	99,80 %	100%

A Tabela 5.8 também mostra que a postura corporal e as relações espaciais entre as articulações, pelo menos dentro do universo de sinais do conjunto de dados, apresenta um alto poder de discriminação. Este resultado mostra que os sistemas baseados em luvas ou sensores externos são ineficazes, não só em termos de aplicabilidade em cenários reais, mas também em termos de eficácia na discriminação dos sinais de Libras.

Na Tabela 5.9, é possível observar um comparativo com trabalhos relacionados. Como a maioria desses trabalhos usa a base de dados apresentada em [71], um conjunto de dados da Língua Argentina de Sinais (LSA) contendo 64 sinais, para fins de comparação, a metodologia proposta nesse trabalho foi replicada usando esse conjunto de dados. Os resultados mostram que a metodologia proposta conseguiu uma acurácia de 100% usando esse conjunto de dados. O resultado da metodologia proposta tem uma acurácia 20% superior ao único trabalho encontrado dentro do contexto de Libras e com uma metodologia próxima.

5.5 Considerações sobre Desempenho

Por ser uma arquitetura multifluxo, o custo computacional do modelo é maior ou igual do que soma dos fluxos individuais, já que cada fluxo necessita realizar trans-

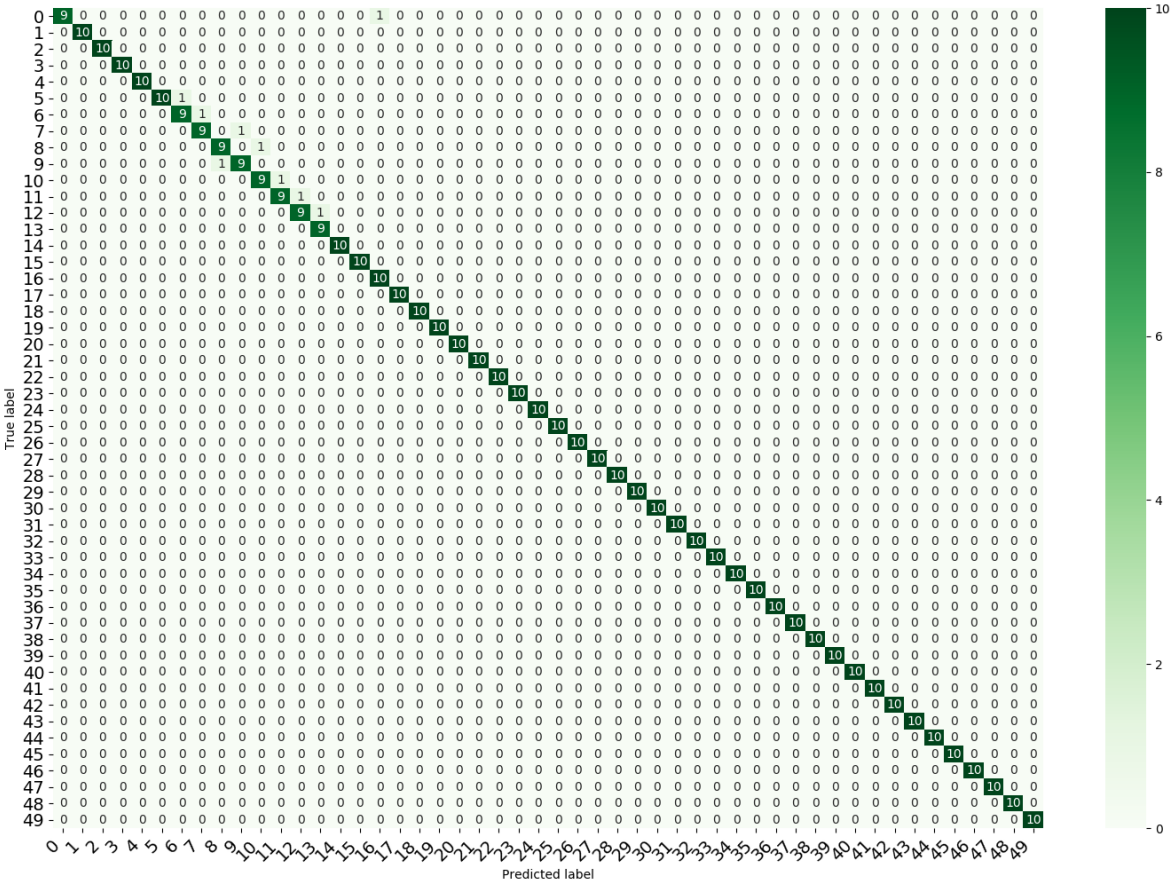

Figura 5.5: Matriz de confusão para arquitetura com três fluxos. Fonte: Autor

Tabela 5.9: Comparação com trabalhos relacionados.

Autor, Ano	Classificador usou	Base	Acurácia
Masood et al., 2018 [62]	CNN-2D + LSTM	LSA	95,60%
Konstantinidis et al., 2018 [53]	CNN-2D	LSA	98,09%
Konstantinidis et al., 2018 [52]	CNN-2D + LSTM	LSA	99,84%
Machado, M.C, 2018 [8]	CNN-3D + ConvLSTM	Libras	79,80%
Autor, 2020	CNN-3D + LSTM	Libras	**99,80%**
Autor, 2020	CNN-3D + LSTM	LSA	**100,0%**

formações específicas para adequar os dados para o método ou arquitetura empregada em cada fluxo, especialmente nos fluxos baseados em fluxo óticos e na detecção de poses. Este custo computacional, para o ambiente experimental explanado na Seção 4.5, é sumarizado na Tabela 5.10.

Tabela 5.10: Tempos médios de cada estágio do *pipeline*

Modelo	Pré-processamento (s)	Predição (s)	Tempo total (s)	Acurácia (%)
I3D	0,230	0,226	0,500	96,34
LSMT	3,1	0,044	3,2	98,00
Multifluxo	3,3	0,320	4,6	99,80

De forma individual, o fluxo baseado na rede I3D é o mais performático, já que exige apenas a extração, subamostragem e cálculo dos fluxos óticos, que são operações relativamente rápidas, resultando em um sistema modelo com tempo de resposta inferior a 1 segundo. Porém, com a introdução de um fluxo baseado na detecção de poses, o custo computacional tem um aumento considerável, passando, em média, de 1 para 3 segundos para realizar a predição, já que o OpenPose consegue processar em média 14 *frames* por segundo, sendo o componente mais custoso da arquitetura.

No entanto, existe uma relação linear entre a quantidade de GPUs e o tempo de processamento do OpenPose, portanto, uma implantação desse modelo é factível, porém, deve ser totalmente baseada em GPUs, preferencialmente em um ambiente multi-GPU (com múltiplas GPUs), como por exemplo, um cluster de GPUs, ou no contexto de computação em nuvem, aliado à aplicação de um *trade-off* entre acurácia e o tempo de resposta do sistema.

5.6 Considerações Finais

Os resultados demonstram a viabilidade de se reconhecer, de maneira automática, os sinais da Língua de Sinais Brasileiras no contexto de saúde, com metodologia totalmente baseada em vídeos (sequência de imagens) utilizando uma arquitetura multifluxo de aprendizado profundo.

Os resultados sugerem a veracidade da hipótese estudada, pois a arquitetura que obteve o melhor desempenho atingindo acurácia máxima de 99,80% foi uma arquitetura que utiliza múltiplos fluxos. No entanto, apesar dessa acurácia, essa arquitetura, em função do seu *pipeline* complexo, apresenta um alto custo computacional, já que é cada fluxo necessita realizar transformações específicas para adequar os dados para os

métodos ou arquiteturas empregadas em cada fluxo. Outra limitação é a alta complexidade ou mesmo impossibilidade de se fazer um processo de simplificação deste modelo visando adequá-lo para dispositivos móveis.

Isso acaba restringindo ou limitando o seu uso em ambientes ou plataformas que possuem recursos computacionais mais limitados, como, por exemplo, dispositivos móveis ou sistemas embarcados. Uma alternativa comumente usada para contornar essa limitação é expor a solução como um serviço, por exemplo, através de APIs, suportados por servidores com grande poder de processamento, e usar os dispositivos móveis ou sistemas embarcados como clientes deste tipo de serviço.

Capítulo 6

Considerações Finais e Trabalhos Futuros

6.1 Conclusão

Neste trabalho, propusemos um modelo multifluxo para o reconhecimento da Língua Brasileira de Sinais (Libras). Os resultados foram obtidos sem a necessidade de hardware ou sensor de captura adicional (por exemplo, luvas, braçadeiras, entre outros), sendo inteiramente baseados em imagens ou sequência de imagens (vídeos). Os resultados mostram que a melhor precisão para o conjunto de testes foi de 99,80%, considerando um cenário em que o intérprete usado no conjunto de testes não foi usado no conjunto de treinamento.

A metodologia exposta também é genérica o suficiente para ser usada em sinais de outros domínios, isto é, fora do contexto da saúde, já que a metodologia é inteiramente baseada em vídeos e a suas categorizações.

É importante observar que, embora a metodologia proposta tenha atingindo bons resultados, para uso prático, os experimentos demonstraram que será necessária uma infraestrutura considerável e altamente dependente de GPUs, para realizar a implantação do modelo como um serviço, isto é, ser usado em uma aplicação real.

Além disso, também criamos um novo conjunto de dados na língua brasileira de sinais (Libras), contendo 5000 vídeos de 50 sinais no contexto da saúde, que podem auxiliar no desenvolvimento de soluções para auxiliar na comunicação de surdos no contexto da saúde, bem como mais pesquisas sobre o assunto.

6.2 Trabalhos Futuros

Entre as propostas de trabalho futuro, pretendemos incluir novos fluxos para abordar outros elementos dos sinais (ou fonemas) em Libras. Como mencionado na Seção 2.1, um sinal consiste em vários elementos (movimento, forma da mão, localização, expressões não manuais, entre outros). Assim, acreditamos que possivelmente novos fluxos abordando cada um desses elementos (ou fonemas) poderiam melhorar a precisão do modelo.

Além disso, também planejamos realizar uma análise mais robusta da qualidade do conjunto de dados, quantificando isso através de uma métrica que captura a variabilidade espaço-temporal. Outra proposta é aumentar o tamanho do banco de dados, incluindo outros sinais no contexto da saúde. Também planejamos realizar estudos com o objetivo de reduzir a complexidade e, como resultado, os parâmetros do modelo, visando torná-lo mais compatível com aplicativos móveis.

Uma outra proposta de trabalho futuro imediata é fazer a adaptação da metodologia para lidar com sentenças, isso pode ser feito, por exemplo, através da adição de um componente adicional de processamento de linguagem natural, que consiga fazer a modelagem de uma sequência de sinais para uma sequência de palavras, tais como as redes baseados na arquitetura Seq2seq (do inglês, *Sequence-to-Sequence*).

Por fim, outra proposta para trabalhos futuros é realizar testes com usuários surdos para avaliar o potencial da solução em um cenário de uso em ambiente real.

Referências Bibliográficas

[1] A comprehensive introduction to different types of convolutions in deep learning | by kunlun bai | towards data science. https://cutt.ly/JhTATtS. (Acesso em: 07 de dez. 2020). vii, 23

[2] Rybená web. Disponível em: http://www.rybena.com.br/site-rybena/conheca-o-rybena/. Acesso em: 14 de nov. 2019. 2

[3] Understanding lstm networks – colah's blog. https://colah.github.io/posts/2015-08-Understanding-LSTMs/. (Acesso em: 07 de dez. 2020). 19

[4] Vlibras - acessibilidade em libras. Disponível em: https://www.handtalk.me/. Acesso em: 14 de nov. 2019. 2

[5] Vlibras - tradução de português para libras. Disponível em: https://www.vlibras.gov.br/. Acesso em: 14 de nov. 2019. 2

[6] Projeto de lei do senado n 155, de 2017. https://www25.senado.leg.br/web/atividade/materias/-/materia/129246, 2017. (Acesso em: 05 de jul. 2020). 1

[7] Projeto de lei do senado n 465, de 2017. https://www25.senado.leg.br/web/atividade/materias/-/materia/131721, 2017. (Acesso em: 05 de jul. 2020). 1

[8] Classificação automática de sinais visuais da língua brasileira de sinais representados por caracterização espaço-temporal. Master's thesis, 2018. Instituto de Computação. 35, 58

[9] Rini Akmeliawati, Melanie Po-Leen Ooi, and Ye Chow Kuang. Real-time malaysian sign language translation using colour segmentation and neural network. In *2007 IEEE Instrumentation & Measurement Technology Conference IMTC 2007*. IEEE, may 2007. 2, 33, 35

[10] F. Angelini, Z. Fu, Y. Long, L. Shao, and S. M. Naqvi. 2d pose-based real-time human action recognition with occlusion-handling. *IEEE Transactions on Multimedia*, pages 1–1, 2019. 22

[11] Jamilly da Silva Aragão, Inacia Sátiro Xavier de Francisco, Alexsandro Silva Coura, Francisco Stélio de Sousa, Joana D'arc Lyra Batista, and Isabella Medeiros de Oliveira Magalhões. A content validity study of signs, symptoms and diseases/health problems expressed in LIBRAS. *Revista Latino-Americana de Enfermagem*, 23:1014 – 1023, 12 2015. 3, 38

[12] Tiago Araujo, Felipe Ferreira, Danilo Silva, Leonardo Oliveira, Eduardo Falcão, Vandhuy Martins, Igor Portela, Yurika Nóbrega, Hozana Lima, Guido Souza Filho, Tatiana Tavares, and Alexandre Duarte. An approach to generate and embed sign language video tracks into multimedia contents. *Information Sciences*, 281:762–, 04 2014. 2

[13] S. Bessa Carneiro, E. D. F. De M. Santos, T. M. De A. Barbosa, J. O. Ferreira, S. G. Soares Alcalá, and A. F. Da Rocha. Static gestures recognition for brazilian sign language with kinect sensor. In *2016 IEEE SENSORS*, pages 1–3, 2016. 2

[14] Vivek Bheda and Dianna Radpour. Using deep convolutional networks for gesture recognition in american sign language. *CoRR*, abs/1710.06836, 2017. 34

[15] Nguyen Dang Binh and Toshiaki Ejima. Real-time malaysian sign language translation using colour segmentation and neural network. In *Proc. ICGST Int. Conf. Graph. Vision Image Process*, pages 1–6, 2005. 2, 33, 35

[16] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000. 27, 30, 41

[17] Fabio Buttussi, Luca Chittaro, and Marco Coppo. Using web3d technologies for visualization and search of signs in an international sign language dictionary. volume 2007, pages 61–70, 01 2007. 6

[18] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. OpenPose: realtime multi-person 2D pose estimation using Part Affinity Fields. In *arXiv preprint arXiv:1812.08008*, 2018. viii, 31, 32, 41

[19] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset, 2017. vii, 22, 23, 24

[20] Roberto Cavararo. *Características gerais da população, religião e pessoas com defciência.* Instituto Brasileiro de Geografa e Estatística (IBGE), 2010. 1

[21] Ming Jin Cheok, Zaid Omar, and Mohamed Hisham Jaward. A review of hand gesture and sign language recognition techniques. *International Journal of Machine Learning and Cybernetics*, aug 2017. 2, 33

[22] François Chollet et al. Keras. `https://keras.io`, 2015. Acesso em: 14 de nov. 2019. 46

[23] C. Chuan, E. Regina, and C. Guardino. American sign language recognition using leap motion sensor. In *2014 13th International Conference on Machine Learning and Applications*, pages 541–544, 2014. 2

[24] Helen Cooper, Brian Holt, and Richard Bowden. Sign language recognition. In *Visual Analysis of Humans*, pages 539–562. Springer London, 2011. 3

[25] Helen Cooper, Eng-Jon Ong, Nicolas Pugeault, and Richard Bowden. *Sign Language Recognition Using Sub-units*, pages 89–118. Springer International Publishing, Cham, 2017. 3, 41

[26] Helen Cooper, Nicolas Pugeault, and Richard Bowden. Reading the signs: A video based sign dictionary. In *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*. IEEE, nov 2011. 2, 33, 35

[27] T. Dasgupta, A. Basu, P.K. Bhowmick, and P. Mitra. A framework for the automatic generation of indian sign language. *Journal of Intelligent Systems*, 19(2), jan 2010. 2

[28] Tiago Maritan Ugulino de Araújo, Felipe Lacet S. Ferreira, Danilo Assis Nobre dos S. Silva, Felipe Hermínio Lemos, Gutenberg Pessoa Botelho Neto, Derzu Omaia, Guido Lemos de Souza Filho, and Tatiana A. Tavares. Automatic generation of brazilian sign language windows for digital tv systems. *Journal of the Brazilian Computer Society*, 19:107–125, 2012. 2

[29] Tiago Maritan Ugulino de Araújo, Felipe Lacet Silva Ferreira, Danilo Assis Nobre dos Santos Silva, Felipe Hermínio Lemos, Gutenberg Pessoa Neto, Derzu Omaia, Guido Lemos de Souza Filho, and Tatiana Aires Tavares. Automatic generation of brazilian sign language windows for digital TV systems. *Journal of the Brazilian Computer Society*, 19(2):107–125, sep 2012. 2

[30] Vantuil José de Oliveira Neto and David Menotti Gomes. Comparação de métodos para localização de fluxo óptico em sequências de imagens. 28

[31] Maria Fernanda Neves Silveira de Souza, Amanda Miranda Brito Araújo, Luiza Fernandes Fonseca Sandes, Daniel Antunes Freitas, Wellington Danilo Soares, Raquel Schwenck de Mello Vianna, and Árlen Almeida Duarte de Sousa. Principais dificuldades e obstáculos enfrentados pela comunidade surda no acesso à saúde: uma revisão integrativa de literatura. *Revista CEFAC*, 19(3):395–405, jun 2017. 1

[32] Ali Diba, Mohsen Fayyaz, Vivek Sharma, Amir Hossein Karami, Mohammad Mahdi Arzani, Rahman Yousefzadeh, and Luc Van Gool. Temporal 3d convnets: New architecture and transfer learning for video classification, 2017. 23

[33] Liliane Correia Toscano de Brito Dizeu and Sueli Aparecida Caporali. A língua de sinais constituindo o surdo como sujeito. *Educação & Sociedade*, 26:583 – 597, 08 2005. 7

[34] Jeff Donahue, Lisa Anne Hendricks, Marcus Rohrbach, Subhashini Venugopalan, Sergio Guadarrama, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description, 2014. 22, 25

[35] Jeff Donahue, Lisa Anne Hendricks, Marcus Rohrbach, Subhashini Venugopalan, Sergio Guadarrama, Kate Saenko, and Trevor Darrell. Long-term Recurrent Convolutional Networks for Visual Recognition and Description. *arXiv e-prints*, page arXiv:1411.4389, Nov 2014. 34, 35

[36] Ruo Du, Qiang Wu, Xiangjian He, and Jie Yang. Object categorization based on a supervised mean shift algorithm. In Andrea Fusiello, Vittorio Murino, and Rita Cucchiara, editors, *Computer Vision – ECCV 2012. Workshops and Demonstrations*, pages 611–614, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg. 35

[37] Alexei A Efros, Alexander C Berg, Greg Mori, and Jitendra Malik. Recognizing action at a distance. In *null*, page 726. IEEE, 2003. 22

[38] Tanya Amara Felipe. Os processos de formação de palavra na libras. *ETD - Educação Temática Digital*, 7(2):200, November 2008. 6

[39] R. Galicia, O. Carranza, E. D. Jiménez, and G. E. Rivera. Mexican sign language recognition using movement sensor. In *2015 IEEE 24th International Symposium on Industrial Electronics (ISIE)*, pages 573–578, 2015. 2

[40] Chuang Gan, Naiyan Wang, Yi Yang, Dit-Yan Yeung, and Alexander G. Hauptmann. Devnet: A deep event network for multimedia event detection and evidence recounting. In *CVPR*, pages 2568–2577. IEEE Computer Society, 2015. 22

[41] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997. 19

[42] Z. Huang, Y. Liu, Y. Fang, and B. K. P. Horn. Video-based fall detection for seniors with human pose estimation. In *2018 4th International Conference on Universal Village (UV)*, pages 1–4, Oct 2018. 22

[43] Matt Huenerfauth. A multi-path architecture for machine translation of english text into american sign language animation. 05 2004. 2

[44] Matt Huenerfauth. Generating american sign language animation: overcoming misconceptions and technical challenges. *Universal Access in the Information Society*, 6:419–434, 02 2008. 2

[45] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015. 13

[46] A. B. Jani, N. A. Kotak, and A. K. Roy. Sensor based hand gesture recognition system for english alphabets used in sign language of deaf-mute people. In *2018 IEEE SENSORS*, pages 1–4, 2018. 2

[47] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classication with convolutional neural networks. In *Proceedings of International Computer Vision and Pattern Recognition (CVPR 2014)*, 2014. 22

[48] L. Kau, W. Su, P. Yu, and S. Wei. A real-time portable sign language translation system. In *2015 IEEE 58th International Midwest Symposium on Circuits and Systems (MWSCAS)*, pages 1–4, 2015. 2

[49] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, Mustafa Suleyman, and Andrew Zisserman. The kinetics human action video dataset, 2017. 24

[50] F. Kaya, A. F. Tuncer, and Ş. K. Yildiz. Detection of the turkish sign language alphabet with strain sensor based data glove. In *2018 26th Signal Processing and Communications Applications Conference (SIU)*, pages 1–4, 2018. 2

[51] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. 46

[52] D. Konstantinidis, K. Dimitropoulos, and P. Daras. A deep learning approach for analyzing video and skeletal features in sign language recognition. In *2018 IEEE International Conference on Imaging Systems and Techniques (IST)*, pages 1–6, 2018. 58

[53] D. Konstantinidis, K. Dimitropoulos, and P. Daras. Sign language recognition based on hand and body skeletal data. In *2018 - 3DTV-Conference: The True Vision - Capture, Transmission and Display of 3D Video (3DTV-CON)*, pages 1–4, 2018. 58

[54] Yann LeCun, Y. Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521:436–44, 05 2015. 8, 9, 11

[55] Fei-Fei Li, Andrej Karpathy, and Justin Johnson. Cs231n: Convolutional neural networks for visual recognition 2016. vii, 11

[56] Kun Liu, Wu Liu, Chuang Gan, Mingkui Tan, and Huadong Ma. T-c3d: Temporal convolutional 3d network for real-time action recognition. In *AAAI*, 2018. 23

[57] Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'81, page 674–679, San Francisco, CA, USA, 1981. Morgan Kaufmann Publishers Inc. 29

[58] Verónica López-Ludeña, Carlos Morcillo, Juan Carlos López, Roberto Barra-Chicote, Ricardo Cordoba, and Ruben Hernandez. Translating bus information into sign language for deaf people. *Engineering Applications of Artificial Intelligence*, 32, 06 2014. 2

[59] Verónica López-Ludeña, Carlos Morcillo, Juan Carlos López, E. Ferreiro, Javier Ferreiros, and Ruben Hernandez. Methodology for developing an advanced communications system for the deaf in a new domain. *Knowledge-Based Systems*, 56:240–252, 01 2014. 2

[60] S. Ma, L. Sigal, and S. Sclaroff. Learning activity progression in lstms for activity detection and early detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1942–1950, June 2016. 22

[61] Mahshid Majd and Reza Safabakhsh. Correlational convolutional lstm for human action recognition. *Neurocomputing*, 04 2019. 35, 36

[62] Sarfaraz Masood, Adhyan Srivastava, Harish Thuwal, and Musheer Ahmad. *Real-Time Sign Language Gesture (Word) Recognition from Video Sequences Using CNN and RNN*, pages 623–632. 01 2018. 2, 22, 34, 35, 58

[63] Taro Miyazaki, Naoto Kato, Seiki Inoue, Shuichi Umeda, Makiko Azuma, Nobuyuki Hiruma, and Yuji Nagashima. Proper name machine translation from

japanese to japanese sign language. In *Proceedings of the EMNLP 2014 Workshop on Language Technology for Closely Related Languages and Language Variants*. Association for Computational Linguistics, 2014. 2

[64] Sara Morrissey and Andy Way. Manual labour: Tackling machine translation for sign languages. *Machine Translation*, 27, 03 2013. 2

[65] Joe Yue-Hei Ng, Matthew Hausknecht, Sudheendra Vijayanarasimhan, Oriol Vinyals, Rajat Monga, and George Toderici. Beyond short snippets: Deep networks for video classification, 2015. 22

[66] Michael A. Nielsen. Neural networks and deep learning, 2018. 10

[67] Eng-Jon Ong, Oscar Koller, Nicolas Pugeault, and Richard Bowden. Sign spotting using hierarchical sequential patterns with temporal intervals. 06 2014. 3, 41

[68] World Health Organization. Millions of people in the world have hearing loss that can be treated or prevented. WHO, mar 2013. 1

[69] M. Oszust and M. Wysocki. Polish sign language words recognition with kinect. In *2013 6th International Conference on Human System Interactions (HSI)*, pages 219–226, 2013. 3, 41

[70] Lionel Pigou, Sander Dieleman, Pieter-Jan Kindermans, and Benjamin Schrauwen. Sign language recognition using convolutional neural networks. In *Computer Vision - ECCV 2014 Workshops*, pages 572–578. Springer International Publishing, 2015. 2, 34, 35

[71] Franco Ronchetti, Facundo Quiroga, Cesar Estrebou, Laura Lanzarini, and Alejandro Rosete. Lsa64: An argentinian sign language dataset. 2016. 3, 41, 57

[72] Alex Sherstinsky. Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. *CoRR*, abs/1808.03314, 2018. 15

[73] Umar Shoaib, Nadeem Ahmad, Paolo Prinetto, and G. Tiotto. Integrating multiwordnet with italian sign language lexical resources. *Expert Systems with Applications*, 01 2013. 2

[74] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos, 2014. 22, 26

[75] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014. 31

[76] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. UCF101: A Dataset of 101 Human Actions Classes From Videos in The Wild. *arXiv e-prints*, page arXiv:1212.0402, Dec 2012. 35

[77] William Stokoe. Sign language structure: An outline of the visual communication systems of the american deaf. *Journal of deaf studies and deaf education*, 10:3–37, 02 2005. 6

[78] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions, 2014. 22

[79] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks, 2014. 22

[80] J. Wan, S. Z. Li, Y. Zhao, S. Zhou, I. Guyon, and S. Escalera. Chalearn looking at people rgb-d isolated and continuous datasets for gesture recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 761–769, 2016. 35

[81] H. Wang, A. Kläser, C. Schmid, and C. Liu. Action recognition by dense trajectories. In *CVPR 2011*, pages 3169–3176, June 2011. 22

[82] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks: Towards good practices for deep action recognition, 2016. 22

[83] Chao-Yuan Wu, Christoph Feichtenhofer, Haoqi Fan, Kaiming He, Philipp Krähenbühl, and Ross Girshick. Long-term feature banks for detailed video understanding, 2018. 22

[84] J. Wu, L. Sun, and R. Jafari. A wearable system for recognizing american sign language in real-time using imu and surface emg sensors. *IEEE Journal of Biomedical and Health Informatics*, 20(5):1281–1290, 2016. 2

[85] Zuxuan Wu, Ting Yao, Yanwei Fu, and Yu-Gang Jiang. Deep learning for video classification and captioning. *Frontiers of Multimedia Research*, page 3–29, Dec 2017. 22

[86] Li Yao, Atousa Torabi, Kyunghyun Cho, Nicolas Ballas, Christopher Pal, Hugo Larochelle, and Aaron Courville. Describing videos by exploiting temporal structure, 2015. 23

[87] L. Zhang, G. Zhu, P. Shen, J. Song, S. A. Shah, and M. Bennamoun. Learning spatiotemporal features using 3dcnn and convolutional lstm for gesture recognition. In *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pages 3120–3128, 2017. 35