

**DEEP-BASED RECURRENT APPROACHES FOR
GESTURE RECOGNITION**

IGOR LEONARDO OLIVEIRA BASTOS

**DEEP-BASED RECURRENT APPROACHES FOR
GESTURE RECOGNITION**

ADVISOR: WILLIAM ROBSON SCHWARTZ

Belo Horizonte

June 2020

Bastos, Igor Leonardo Oliveira	
B327d	Deep-based recurrent approaches for gesture recognition [manuscrito] / Igor Leonardo Oliveira Bastos. - 2020. xxvii, 88 f. il. Orientador: William Robson Schwartz Tese (Doutorado) - Universidade Federal de Minas Gerais, Instituto de Ciências Exatas, Departamento de Ciência da Computação. Referências: f.81-88 1. Computação – Teses. 2. Redes neurais recorrentes – Teses. 3. Reconhecimento de gestos – Teses. 4. Reconhecimento de padrões – Teses. I. Schwartz, William Robson. II. Universidade Federal de Minas Gerais; Instituto de Ciências Exatas, Departamento de Ciência da Computação. III. Título.
	CDU 519.6*82(043)

Ficha catalográfica elaborada pela bibliotecária Belkiz Inez Rezende Costa
CRB 6ª Região nº 1510



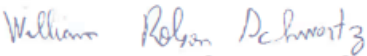
UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

FOLHA DE APROVAÇÃO

Deep-based Recurrent Approaches for Gesture Recognition

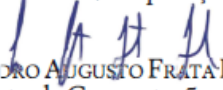
IGOR LEONARDO OLIVEIRA BASTOS


Tese defendida e aprovada pela banca examinadora constituída pelos Senhores:


PROF. WILLIAM ROBSON SCHWARTZ - Orientador
Departamento de Ciência da Computação - UFMG


PROF. ERICKSON RANGEL DO NASCIMENTO
Departamento de Ciência da Computação - UFMG


PROF. GUILLERMO CÁMARA CHÁVEZ
Departamento de Computação - UFOP


PROF. LEANDRO AUGUSTO FRATA FERNANDES
Instituto de Computação - UFF


PROF. RICARDO DA SILVA TORRES
Department of ICT and Natural Sciences - Norwegian University of Science and Technology

Belo Horizonte, 12 de Junho de 2020.

Acknowledgements

I would like to thank the National Council for Scientific and Technological Development – CNPq (Grants 438629/2018-3 and 309953/2019-7), the Minas Gerais Research Foundation – FAPEMIG (Grants APQ-00567-14 and PPM-00540-17), the Coordination for the Improvement of Higher Education Personnel – CAPES (DeepEyes Project).

“You can depend on your eyes when your imagination is out of focus”
(Mark Twain)

Resumo

O reconhecimento de gestos corresponde a uma interpretação matemática de um movimento humano por parte de uma máquina. Este movimento envolve diferentes aspectos e partes do corpo, tais como variações no posicionamento de mãos e braços, expressões faciais e corporais, posicionamento da cabeça, postura do tronco, entre outros. Por levar em consideração tanto a aparência (aparência das partes do corpo, por exemplo) quanto o movimento, o reconhecimento de gestos mostra-se relacionado a abordagens que contemplam a extração e uso de informação espaço-temporal em vídeos, tendo destaque em diferentes áreas e aplicações. Devido a esta alta aplicabilidade, diversas pesquisas têm se voltado para este tema, as quais variam em termos de características e algoritmos de aprendizado utilizados para a tarefa.

No entanto, apesar da existência de uma grande gama de trabalhos relacionados ao reconhecimento de gestos, nota-se uma lacuna no tocante a elaboração de abordagens que levem em consideração aspectos como escalabilidade (em termos do número de gestos), capacidade de incorporar novos gestos com baixo custo de tempo, além de atuação em vídeos não-segmentados, ou seja, vídeos que contemplam múltiplos gestos e não possuem informação sobre o começo e fim de cada gesto. Desta forma, este trabalho visa apresentar estratégias que preencham estas lacunas, dividindo-se em duas linhas: (i) criação de modelos escaláveis para aplicação incremental em grandes bases de dados; (ii) formulação de um modelo para realização concomitante da detecção temporal de gestos em vídeos não-segmentados e seu respectivo reconhecimento.

Para uma eficiente atuação em vídeos que representam gestos, deve-se levar em consideração a estrutura temporal bem definida destes, a qual defende a existência de uma ordem de ocorrência de sub-eventos. Devido a isso, propõe-se a formulação de modelos não somente capazes de extrair informação espaço-temporal, mas também de atender para esta estrutura temporal, ponderando a contribuição de entradas anteriores (trechos anteriores dos vídeos), para avaliar o que se apresenta a seguir. Assim, estes modelos correlacionam informação de diferentes partes dos vídeos, produzindo representações mais ricas dos gestos, as quais são usadas para um reconhecimento mais

acurado.

Por fim, de maneira a avaliar as abordagens propostas, os resultados da aplicação dos modelos descritos neste documento são apresentados. Estes foram obtidos considerando bases de dados amplamente utilizadas por trabalhos da área, assim como as métricas de avaliação empregadas para avaliar desempenho em cada uma destas bases. No *ChaLearn Isolated Gestures* (ChaLearn IsoGD) and *Sheffield Kinect Gestures* (SKIG), o método proposto neste documento alcançou valores de acurácia de 69,44% e 99,53%, respectivamente. Já no *ChaLearn Looking at People Multimodal Gesture Recognition* (ChaLearn Montalbano) e *ChaLearn Continuous Gestures* (ChaLearn ConGD), o método contemplado neste documento obteve 0,919 e 0,623 de Jaccard Score, respectivamente. Comparações com abordagens da literatura evidenciam a boa performance dos métodos propostos, os quais rivalizam com as pesquisas que são o estado da arte em todas as bases de dados avaliadas.

Palavras-chave: Reconhecimento de Gestos, Redes Recorrentes, Informação espaço Temporal, Gestos Isolados, Vídeos Não-segmentados.

Abstract

The recognition of gestures corresponds to a mathematical interpretation of a human motion by a machine. It involves different aspects and parts of human body, such as variations in the positioning of hands and arms, facial and body expressions, head positioning and trunk posture. Since gesture recognition takes into account both appearance (appearance of body parts, for example) and movement, it is related to the extraction of spatiotemporal information in videos, leading to a wide range of applications. As a consequence, many approaches focus on this topic, presenting variations in terms of employed features and learning algorithms used on the task.

However, despite the existence of a wide range of approaches related to the recognition of gestures, gaps are noticed regarding aspects such as scalability (in terms of the number of gestures), time to incorporate new gestures; and actuation over unsegmented videos, i.e., videos containing multiple gestures and no information about the start and end of these gestures. Thus, this work aims at presenting strategies that fill these gaps, addressed in two different lines: (i) creation of scalable models for incremental application in large databases; (ii) formulation of a model to detect and recognize gestures concomitantly, considering unsegmented videos.

For an efficient performance on gesture videos, it is important to take into account the well-defined temporal structure of gestures, which preaches for the existence of ordered sub-events. To handle this order of sub-events, we propose models that are capable of extracting spatio-temporal information and also weigh this temporal structure, contemplating the contribution of previous inputs (previous videos snippets) to evaluate subsequent ones. Thereby, our models correlate information from different video parts, producing richer representations of gestures that are used for a more accurate recognition.

Finally, to evaluate the proposed approach, we present the results obtained from the application of the models described in this document. These outcomes were obtained from tests on widely used databases, considering the metrics employed to evaluate performance on each of them. On ChaLearn LAP Isolated Gestures (ChaLearn

IsoGD) and Sheffield Kinect Gestures (SKIG), the method proposed in this document achieved 69.44% and 99.53% of accuracy, respectively. On ChaLearn Multimodal Gesture Recognition (ChaLearn Montalbano) and ChaLearn Continuous Gestures (ChaLearn ConGD), the method contemplated in this document obtained 0.919 and 0.623 as Jaccard Score, respectively. Comparisons with literature approaches evidence the good performance of the proposed methods, rivaling to state-of-the-art researches on all evaluated databases.

Palavras-chave: Gesture Recognition, Recurrent Neural Networks (RNNs), Spatiotemporal Information, Isolated Gestures, Unsegmented Videos.

List of Figures

1.1	Fields of application of gesture recognition approaches. (a) Virtual navigation system based on gesture recognition [Côté and Beaulieu, 2019]. (b) Italian sign language sample from Chalearn Montalbano [Escalera et al., 2014]. (c) Extraction of metrics for biometric validation based on gesture recognition [Fong et al., 2013].	1
1.2	Scenarios for the gesture recognition methods presented in this document. (a) Parts of body used to perform a gesture [Liu and Shao, 2013]. (b) Gesture performing considering half-body [Wan et al., 2016]. (c) Full body recognition of gestures associated to a sign language (Italian Sign Language) [Escalera et al., 2014].	4
2.1	Representations of a recurrent layer. Input is represented by x , outputs by y and recurrent representation (hidden state) by s . Timesteps are indexed from 1 to T [Han et al., 2018].	10
2.2	Representation of a LSTM unit. x represents the input, h the state of the unit and W represents the weights of each input regarding the gates. For each timestep, the unit receives the current input and the previous state [Chen, 2016].	11
2.3	Representation of a GRU unit. x represents the input, r represents the reset gate and z the update gate [Le et al., 2016].	12
2.4	Basic representation of an autoencoder. Encoding network is represented by blue, while decoding is represented by red [Hadji and Wildes, 2018]. . .	14
2.5	Schematic of a multi-task model for 4 different tasks [Caruana, 1997]. . . .	15
2.6	L-Softmax learned features for different classes of CIFAR-10 dataset [Krizhevsky, 2009]. The constant m is being increased, assuming values $m = 1, 2, 3, 4$ from left to right. [Liu et al., 2016]. With $m = 1$, this loss behaves as a custom softmax layer associated with a cross-entropy loss.	17

3.1	Libras hand recognition dataset [Bastos et al., 2015].	20
3.2	Architecture proposed by Duan et al. [2016]. Different networks are employed to deal with each modality. At end, a fusion is performed considering the voting response of each of them.	22
3.3	Wu et al. [2016] outcomes for a test video of ChaLearn LAP 2014 [Escalera et al., 2014]. Different strategies are presented along with their responses. Dashed lines represents no-class (silence) [Wu et al., 2016].	24
3.4	Nishida and Nakayama [2016] Multimodal Architecture. Each circle represents an input, denoted by $x^{(A)}$, $x^{(B)}$ and $x^{(C)}$. These inputs are associated to a different modality stream, composed of 3D convolutional/pooling layers. Squares represent recurrent LSTM layers. Every stream is fused on the recurrent layer on the top, represented by hidden states $h^{(*)}$. Arrows on recurrent layers (squares) expose the behavior on different timesteps, evidenced by the increasing indexes on inputs and hidden states ($x_1^{(A)}$, $x_2^{(A)}$, ..., $x_T^{(A)}$).	25
3.5	Architecture proposed by Molchanov et al. [2016]. Different timesteps of recurrent network are represented by h_t s, while softmax responses are represented by s_t s. One could notice the existence of a final layer employing CTC cost function.	26
3.6	(a) Architecture proposed by Zhang et al. [2017]. On this architecture, a 2DCNN is used to learn higher-level spatiotemporal features maps for the final gesture classification. (b) Residual block proposed by Pigou et al. [2017]	27
3.7	Architectures proposed by Zhu et al. [2018] for detection and recognition of gestures.	28
3.8	Camgoz sequence-to-sequence architecture. One could notice the existence of encoding and decoding steps. Words are outputed one-by-one.	29
4.1	MORA inputs: a) RGB video. b) Farneback Optical Flow. c) Depth.	32
4.2	MORA test phase. Index of model with lowest error represents the class.	32
4.3	Proposed autoencoder model. Layers employ ReLU (blue) and Sigmoid (purple) activations.	33
4.4	MORA autoencoder model with a skin branch (skin MORA model). A feature map is created and acts as an attention mechanism for the remaining outputs.	36
4.5	Classification model employed to enhance autoencoders with discriminative behavior. Layers employ ReLU (blue) and Sigmoid (purple) activations.	37

4.6	SALMORA pipeline with interleaved unsupervised (left and right) and supervised (middle) steps.	38
4.7	MLRRN pipeline.	40
4.8	Input of MLRRN for different timesteps.	41
4.9	Assembling of tensors on the MLRRN architecture.	42
4.10	(a) MLRRN architecture. Layers employ ReLU (blue), ELU (green), sigmoid (purple) and softmax activations (brown). (b) Spatial convolution blocks.	44
5.1	Different gesture classes of the SKIG dataset. RGB and corresponding depth videos are provided [Liu and Shao, 2013].	46
5.2	Frames from ChaLearn LAP IsoGD videos and their depth correspondences [Wan et al., 2016].	47
5.3	Frames from ChaLearn Multimodal Gesture Recognition videos and their depth, user segmentation and skeleton correspondences.	48
5.4	Frames from ChaLearn ConGD videos. Different subjects are performing the gestures, under different illumination conditions and viewpoints.	49
6.1	Model for skin detection in images. Layers employ ReLU (blue) and Sigmoid (purple) activations. On the bottom part of the Figure, the structure of <i>Encoding Layers</i> block is illustrated.	53
6.2	Images and their corresponding skin maps from Pratheepan dataset [Tan et al., 2012].	54
6.3	(a) Frame of a video of SKIG dataset. (b) MORA reconstruction without skin task. (c) MORA reconstruction with skin task.	56
6.4	(a) Disposition of features from MORA models. (b) Disposition of features from MORA models after using a discriminator with custom binary-crossentropy loss. (c) Disposition of features from MORA models after using a discriminator with Large-margin softmax loss ($m=2$).	58
6.5	Accuracy on 50 classes of Chalearn.	62
6.6	Post processing on MLRRN response. Colors indicate the label of the frame. The orange box indicates the duration, in terms of frames, of a gesture related to a specific class (orange). The dark blue lines indicate the existence of wrongly recognized frames during the recognition of the orange class.	66
6.7	MLRRN recognition outcomes considering different Jaccard scores.	68
6.8	MLRRN recognition outcomes considering different Jaccard scores on ChaLearn ConGD.	70

6.9	Detection responses of MLRRN. Dashed red line represents the threshold used to approximate responses.	71
6.10	Shortening effect on the recognition of gestures by MLRRN.	71
6.11	Frame-level confusion matrix of MLRRN on ChaLearn Montalbano. Indexes represent different dataset classes. Class-0 represents non-gesture class. . .	72
6.12	Detection responses of MLRRN on ChaLearn ConGD.	73

List of Tables

3.1	Summarization of gesture approaches on literature and the proposed methods. <i>Rec.</i> regards the usage of recurrent layers. <i>GR</i> stands for Gesture Recognition, <i>GD</i> stands for Gesture Detection, while <i>SLR</i> means Sign Language Recognition.	30
5.1	Number of videos on Training, Validation and Test subsets of ChaLearn LAP IsoGD.	47
5.2	Number of videos on Training, Validation and Testing subsets of ChaLearn Montalbano Multimodal Gesture Recognition.	48
5.3	Number of videos on Training, Validation and Testing subsets of ChaLearn LAP ConGD.	49
6.1	Hyperband search space for MORA approach. (Enc.), (Dec.), and (Rec.) Layer Factor represents an increase on the number of feature maps for layers on Encoding, Decoding and Recurrent layer of the autoencoders, respectively. Layer Growth stands for an addition or removal of a convolutional layer on encoding and decoding parts of the models.	52
6.2	Hyperband determined parameters for each model of SKIG dataset. *Despite the hyperband indication for MAE as the loss function for model 5, MSE was used to not create any discrepancy between the error computation for the reconstruction of this model and the remaining ones.	55
6.3	Accuracies of different approaches applied to the SKIG dataset.	57
6.4	Most voted set of hyperparameters with Hyperband for ChaLearn IsoGD.	60
6.5	Accuracies on the test subset of the ChaLearn IsoGD dataset.	61
6.6	SALMORA recognition accuracy considering a variable amount of selected classes associated to the classifier. These experiments considered the Large-Margin Softmax Loss [Liu and Liu, 2016] and <i>inter</i> and <i>intra</i> modalities.	61

6.7	Comparison between MORA and a discriminative model (C3D) for tests on ChaLearn IsoGD.	63
6.8	Hyperband search space for MLRRN approach. Conv. Factor represents an adjustment on the number of filters on convolutional layers, while Rec. Layer Factor stands for an adjustment of units on recurrent layer. ELU layer factor represents an adjustment on the number of units on ELU layers.	64
6.9	Ablation study performed on ChaLearn Montalbano dataset.	65
6.10	Selected search parameters obtained with Hyperband for ChaLearn Montalbano.	66
6.11	Average temporal Jaccard score on ChaLearn Montalbano dataset.	67
6.12	Selected search parameters obtained with Hyperband for ChaLearn ConGD.	68
6.13	Average temporal Jaccard score on ChaLearn ConGD dataset.	69

Acronym List

Acc	Accuracy
CNN	Convolutional Neural Network
CNNs	Convolutional Neural Networks
GR	Gesture Recognition
GRU	Gated Recurrent Unit
LSTM	Long-Short Term Memory
MLP	Multilayer Perceptron
MORA	Multi-output Recurrent Autoencoders
MLRRN	Multi-loss Recurrent Residual Network
NN	Neural Network
OF	Optical Flow
RGB	Red, Green, Blue
RNN	Recurrent Neural Network
RNNs	Recurrent Neural Networks
SVM	Support Vector Machines
VRNN	Vanilla Recurrent Neural Network
SALMORA	Skin-based Adversarial-Like Multi-output Recurrent Autoencoders

Contents

Acknowledgements	ix
Resumo	xiii
Abstract	xv
List of Figures	xvii
List of Tables	xxi
Acronym List	xxiii
1 Introduction	1
1.1 Motivation	2
1.2 Challenges	3
1.3 Hypothesis	5
1.4 Contributions	6
1.5 Outline	6
2 Background Concepts	9
2.1 Recurrent Neural Networks	9
2.1.1 Long Short-Term Memory (LSTM)	10
2.1.2 Gated Recurrent Unit (GRU)	12
2.2 Autoencoders	13
2.3 Multi-Task Learning	14
2.4 Hyperparameter optimization with Hyperband	15
2.5 Large-margin softmax loss	16
3 Literature Review	19
3.1 Deep Gesture Recognition Approaches	20

3.1.1	Feedforward Deep Approaches	21
3.1.2	Recurrent Approaches	23
4	Proposed Approaches	31
4.1	Dealing with scalability issues - MORA	31
4.2	Extending MORA - SALMORA	35
4.2.1	Skin Detection as a new task	35
4.2.2	MORA with Adversarial-Like Behavior	36
4.3	Dealing with unsegmented videos - MLRRN	38
4.3.1	Proposed Model	39
5	Challenges and Benchmarks Addressed	45
5.1	SKIG dataset	45
5.2	ChaLearn Looking at People IsoGD dataset	46
5.3	ChaLearn Montalbano Multimodal Gesture Recognition	47
5.4	ChaLearn Looking at People ConGD dataset	48
6	Experimental Results	51
6.1	MORA and SALMORA Evaluation	51
6.1.1	Experimental Setup	51
6.1.2	Evaluating on the SKIG Dataset	54
6.1.3	Evaluating on ChaLearn IsoGD	59
6.1.4	Evaluating Time-to-Train Scalability on ChaLearn IsoGD	61
6.2	MLRRN evaluation	63
6.2.1	Experimental Setup	63
6.2.2	Ablation Study of MLRRN	64
6.2.3	Evaluating on ChaLearn Montalbano	65
6.2.4	Evaluating on ChaLearn ConGD	67
6.2.5	Qualitative Evaluation of MLRRN Tasks	69
7	Conclusions	75
7.1	Final considerations	75
7.2	Future Work	77
7.2.1	Extend SALMORA’s hyperparameter optimization	77
7.2.2	Improvement and better evaluation of MLRRN	77
7.2.3	Libras dataset	77
7.2.4	Alternatives for recurrent models	78
7.2.5	Adjustments on Proposed Architectures	78

7.2.6 Grouping strategy to increase scalability	78
Bibliography	81

Chapter 1

Introduction

Gestures play a relevant role on humans life, in which a non-verbal language is used on the interaction between subjects [Zhu et al., 2016]. To recognize gestures, computers need to represent and interpret human appearance and motion, involving hands, arms, face, head and/or body, in a mathematical sense [Mitra and Acharya, 2007]. Each of these parameters is important to the meaning of a gesture, being efficiently denoted by temporal and spatial information.

Being able to recognize gestures allows a wide range of applications in different contexts, such as navigation on virtual environments [Côté and Beaulieu, 2019], development of aid systems for hearing impaired [Soni et al., 2016], sign language recognition [Bastos et al., 2015], surveillance monitoring [Xu et al., 2018] and biometric validation [Fong et al., 2013]. As a consequence, gesture recognition has been investigated by several approaches, which vary in terms of features and learning algorithms employed to perform the task [Bastos et al., 2015; Molchanov et al., 2016]. Figure 1.1 depicts different applications of gesture recognition methods.

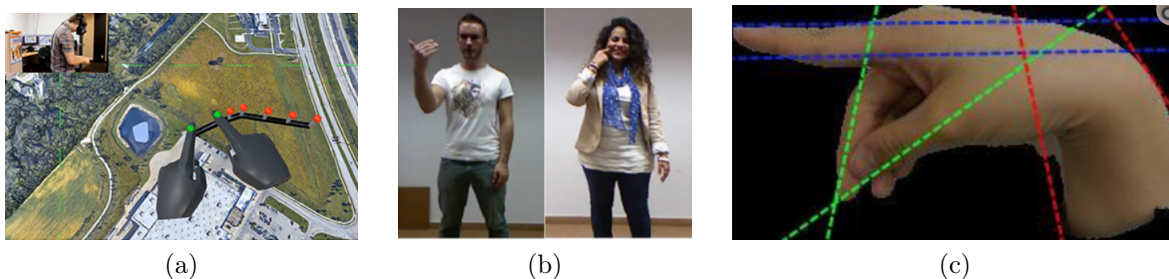


Figure 1.1: Fields of application of gesture recognition approaches. (a) Virtual navigation system based on gesture recognition [Côté and Beaulieu, 2019]. (b) Italian sign language sample from Chalearn Montalbano [Escalera et al., 2014]. (c) Extraction of metrics for biometric validation based on gesture recognition [Fong et al., 2013].

Gestures are characterized by a structured time disposition, in which the order of events (sub-actions) is significant to determine their labels [Molchanov et al., 2016]. This order indicates a dependency of previous information (i.e., sub-actions) to determine the label of the whole gesture. As a result, researchers direct their attention, before related to handcrafted-based approaches [Zhou and Ruan, 2006; Bastos et al., 2015], to deep recurrent methods, focusing on the modeling of this temporal dependency on a time-arranged input.

Despite the achievement of major advances, developing an universal model to recognize gestures is a difficult task due to problems such as illumination and acquisition conditions, inconsistent behavior among users, cultural gesture specificities, and large vocabularies [Zhang et al., 2017]. In addition, assembling gesture recognition datasets is an expensive task and most of them do not comprise many gestures [Shahroudy et al., 2016]. As a consequence, approaches usually do not invest on scalability and require to be completely retrained to handle new gestures, leading to methods that are not suitable, for instance, for sign language recognition, since these languages are extremely changeable and adhere to cultural specificities from people and places where they are employed [Souza et al., 2018]. Furthermore, most of gesture recognition approaches are not suitable to handle unsegmented video streams, since they do not perform the detection of a gesture (in a temporal sense) before its recognition, not exploring the correlation that exists between these tasks. Thus, with our study, we aim at filling these gaps on gesture recognition approaches.

1.1 Motivation

Gesture recognition approaches can be seen as computational modelings of video gesture inputs, in which the appearance and movement of a human performer are used to determine the label of the entire action. The high applicability of this topic in several research fields, with emphasis on biology, medicine, robotics and computer science [Sharma and Chawla, 2013; Mitra and Acharya, 2007], turns it attractive to be studied. However, even being tackled by many approaches, gesture recognition remains a challenging topic, mostly regarding two points: (i) scalability and (ii) ability to handle continuous unsegmented video inputs.

Scalability: Most of gesture recognition approaches are designed to deal with a fixed number of gesture classes [Molchanov et al., 2016; Zhang et al., 2017]. For some applications, this limitation does not represent an issue. In a virtual environment, it is expected a limited set of gestures to be used on the interaction between man

and machine. The same behavior can be noticed on biometric validation systems, for which a set of pre-trained gestures is sufficient to identify/validate the user. However, gestures also correspond to a non-verbal communication channel between subjects, clearly evidenced by sign languages. These languages are region-specific and extremely mutable [Souza et al., 2018], adhering to local and cultural specificities from people/places in which they are employed. Modeling sign languages through a computational approach requires the ability to deal with an increasing number of gestures, pointing to methods that concern about this problem and do not require to be completely retrained/reformulated to incorporate these new entries.

Handling unsegmented video inputs: Most of gesture recognition databases are composed of segmented video streams, i.e., each video contemplates a single gesture. As a consequence, literature approaches usually do not concern about segmenting videos before performing their classification [Molchanov et al., 2016; Zhang et al., 2017; Nishida and Nakayama, 2016]. This issue compromises the applicability of a broad range of gesture recognition approaches in uncontrolled scenarios; where the inputs are not pre-segmented (gestures temporarily detected). Tackling this issue allows the development of gesture recognition approaches able to interpret, in a human-like way, messages from gesture-performer subjects.

1.2 Challenges

The methods described in this document aim at performing gesture recognition considering scenarios where a subject performs the gesture in front of a camera. These gestures can be represented by parts or full body of the performer, as depicted in Figure 1.2. In addition, even not being designed to gather the nuances of sign languages, which involve grammar and semantics, the approaches in this document can be used to recognize the meaning of individual gestures (signs) from these languages or gestures used for any other type of communication.

Considering the aforementioned scenarios, accurate results have been achieved by gesture recognition approaches in the recent years. However, it still remains a challenging task. In this section, we discuss points that gesture recognition approaches have to handle to produce effective results.

Acquisition problems: They include factors such as occlusion, illumination, divergent viewpoint, presence of shadows, camera movements and cluttered backgrounds. All these points impact on the appearance and/or movement of the

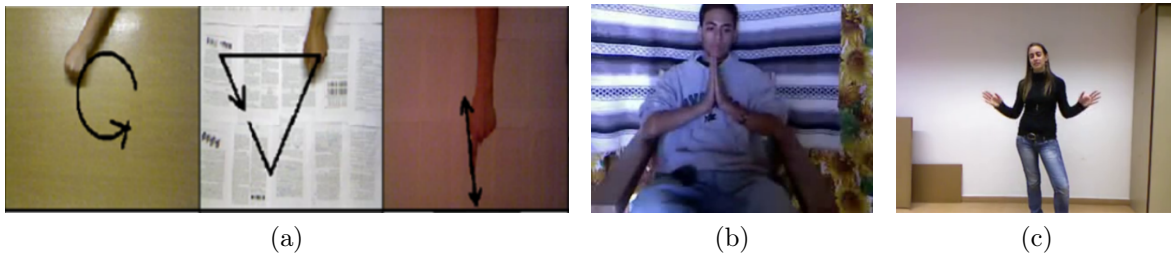


Figure 1.2: Scenarios for the gesture recognition methods presented in this document. (a) Parts of body used to perform a gesture [Liu and Shao, 2013]. (b) Gesture performing considering half-body [Wan et al., 2016]. (c) Full body recognition of gestures associated to a sign language (Italian Sign Language) [Escalera et al., 2014].

gesture performer, leading to a different spatiotemporal representation. Effective gesture recognition approaches need to be, at least in a minimal way, robust to these factors. It is common to include videos with these problems on gesture datasets, in order to challenge researchers to produce models that generalize better.

Inconsistent behavior of different performers: It is expected that different performers execute a gesture with some variation. However, it is not rare to find performers that are outliers, in the sense of movement and/or appearance. This point is more evident on the sign language recognition field, since cultural specificities can impact on the execution of a gesture. Some examples of inconsistent behavior are: divergent hand posture, hand/body movement, and facial expressions.

Large and increasing vocabularies: Many gesture recognition approaches are related to large vocabularies. With that, class similarities tend to be enforced and higher capacity/complexity models need to be employed. In addition, some applications are required to handle an increasing number of gestures. To accomplish that, studies need a mechanism to incorporate these new entries without a high time-cost to adjust the approach.

Class similarities: Different gesture classes are related to, at least, one different parameter response. Examples of these parameters are facial expressions, body movements, hand/arm movements, hand postures and inflection points. Classifying some gestures can become a challenging task due to similarities in many different parameter responses. For example, Brazilian Sign Language (Libras) gestures for letters i and j are identical in terms of appearance, body movement and facial expression. The only divergence regards the hand movement.

1.3 Hypothesis

Spatial and temporal components of videos provide important information for gesture recognition, producing a representation of appearance and motion of gesture parameters [Wu et al., 2016]. To efficiently extract spatiotemporal information from gesture videos, approaches invest on the employment of shape/temporal descriptors and deep models, mostly composed of stacked 3D convolutional layers.

The well-defined temporal structure of gestures also makes room for the application of recurrent models, being employed in most of the state-of-the-art methods for gesture recognition. Thus, a typical hypothesis found in the literature consists on the combination of these two characteristics, employing recurrence to correlate different parts of gesture video inputs and explore the well-defined temporal order of events, besides the use of 3D stacked convolutional layers to gather spatiotemporal information, with both characteristics leading to efficient representations for gesture recognition.

Concerning that hypothesis, we propose approaches that combine strategies of state-of-the-art gesture recognition methods in order to produce an effective model. In addition, we tackle two main issues of these methods, regarding scalability and aptness to deal with unsegmented gesture videos. For the former issue, we assume that it is possible to produce effective models, composed of 3D convolutional and recurrent layers, that can be scalable in terms of number of gestures. Our idea is to utilize unsupervised recurrent autoencoders to represent each gesture class in an independent way. To increase the number of gesture classes to be recognized by the approach, it is only required to train new autoencoders to represent these new classes with no impact on previously trained models. For the latter issue, we assume that temporal detection and recognition of gestures are complementary tasks. With that, a single model could perform both tasks simultaneously with improvements on their outcomes through the employment of a multi-task strategy. In a formal way, the problem statement that this dissertation deals with can be formulated as follows:

Given a gesture recognition dataset, how to produce effective models in order to extract spatiotemporal information, exploit information from gesture temporal structure and tackle the scalability issue of gesture recognition approaches? In addition, is it possible to produce a gesture recognition model that exploits information from gesture temporal structure and handles the absence of temporal segmentation?

1.4 Contributions

The main contributions of this dissertation are threefold, being described on next chapters of this document.

Multi-Output Recurrent Autoencoders (MORA): With MORA, we tackle issues that are commonly observed on gesture recognition approaches such as scalability, time/computational cost to insert a new gesture class and robustness to class imbalance. Instead of employing a custom deep classifier, as most studies do, we invest on unsupervised models that are trained to reconstruct each class. On the test phase, samples are provided to all models and the one with lowest reconstruction error indicates the gesture label.

Skin-based Adversarial-Like MORA (SALMORA): SALMORA represents an extension of MORA to incorporate discriminative behavior on autoencoders, enhancing the distance between representations of different classes. This approach is composed of an unsupervised step, similar to MORA with the training of autoencoders for each class, and a supervised step, in which a classifier is used to tune weights of these autoencoders, increasing the distance between their representations and making the reconstruction error to incorporate discriminability. In addition, SALMORA counts on an attention mechanism to prioritize the reconstruction of the gesture performer, with a lower focus on the background. This approach was submitted to *Journal of Neurocomputing* (under review).

Multi-loss Recurrent Residual Network (MLRRN): MLRRN is an approach to perform temporal detection and classification of gestures in a simultaneous way. Since these tasks are complementary, the performance of each of them can be improved by a multi-loss/multi-task model. The main point of this research is to handle unsegmented gesture videos, what increases the aptness to deal with real-life communication scenarios.

All methods described on this document have their performance compared with literature approaches, with contributions in relation to state-of-the-art techniques.

1.5 Outline

The remainder of the text is organized as follows.

This document is composed of seven chapters, following the steps that guided the development of our approaches. Chapter 2 brings theoretical concepts for a better understanding of the thesis. It introduces techniques employed on the present research and on state-of-the-art methods. In Chapter 3, we present the literature review performed during our study, with a detailed description of classical approaches for gesture recognition, clarifying the issues that we noticed on state-of-the-art methods. In turn, in Chapter 4, we introduce the gesture recognition approaches described in this document: (i) recognition based on unsupervised models with focus on scalability and its extension, Multi-Output Recurrent Autoencoders (MORA) and Skin-based Adversarial-Like Multi-output Recurrent Autoencoders (SALMORA); (ii) simultaneous recognition and detection of gestures based on a multitask model, Multi-loss Recurrent Residual Network (MLRRN). In Chapter 5, we present benchmark datasets utilized by the developed approaches, along with their characteristics and evaluation protocols. In Chapter 6, we expose the obtained outcomes for MORA, SALMORA and MLRRN approaches in comparison to state-of-the-art gesture recognition methods, with conclusions and future steps regarding these methods being presented in Chapter 7.

Chapter 2

Background Concepts

Most video gesture recognition approaches, including the ones described in this document, are related to image processing, computer vision and machine learning techniques. In this sense, the goal of this chapter is to introduce techniques to help on the understanding of the present approaches and state-of-the-art methods.

2.1 Recurrent Neural Networks

Recurrent neural networks (RNNs) have emerged as a powerful model for a broad range of machine learning problems that involve sequential data [Yang et al., 2018]. On RNNs, every input depends on previous computations, simulating a memory that captures information about what has been calculated so far [Molchanov et al., 2016; Yang et al., 2018]. In a simple manner, these networks extend conventional feedforward ones to handle variable-length sequences by accumulating the context of previous inputs in their internal state to influence proceeding outputs [Yang et al., 2018]. Figure 2.1 depicts two representations of a recurrent layer, including an unrolled version, for which multiple timesteps are presented at once.

Despite accurate outcomes regarding tasks, such as language modeling, machine translation, speech recognition and even gesture recognition, the memory of RNN networks (also called vanilla RNNs) is limited as the gradients tend to vanish or explode [Yang et al., 2018] depending on factors such as the activation function and number of timesteps. In addition, since a single set of weights is employed to model all timesteps, vanilla RNNs show to be more effective to handle inputs that do not present long-term dependency, considering multiple timesteps [Molchanov et al., 2016; Nishida and Nakayama, 2016]. Moreover, these inputs need to present a well-structured time disposition of events, what makes vanilla RNNs not recommended for tasks, such as

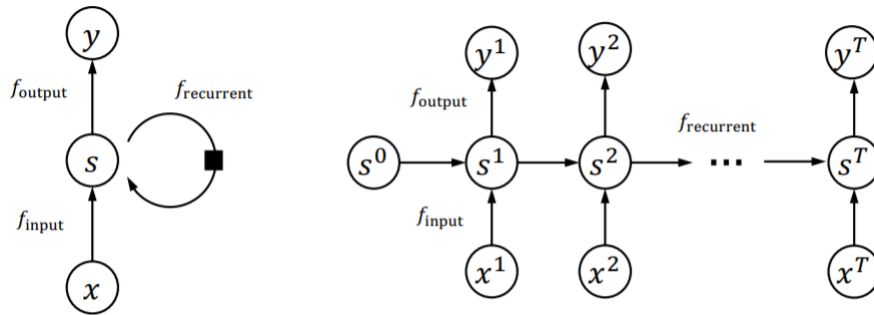


Figure 2.1: Representations of a recurrent layer. Input is represented by x , outputs by y and recurrent representation (hidden state) by s . Timesteps are indexed from 1 to T [Han et al., 2018].

activity recognition. In this context, gating mechanisms have been incorporated to recurrent networks, leading to the formulation of models, such as Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU).

2.1.1 Long Short-Term Memory (LSTM)

Long Short-Term Memories (LSTM) are a subset of RNNs proposed by Hochreiter and Schmidhuber [1997] to handle gradient problems. Instead of using simple nodes ruled by activation functions, each LSTM unit is composed of a cell, an input gate, an output gate and a forget gate, used to store information (memory state) over different timesteps, as illustrated in Figure 2.2. These gates weigh the contribution of past data, current data and current memory on the computation of next memory state [Hochreiter and Schmidhuber, 1997; Luo et al., 2018]. Figure 2.2 depicts the structure of a LSTM unit. One could notice the contribution of gates on information flow and memory, represented by the *cell*.

LSTM units are widely employed to model information with long-term dependencies. To efficiently actuate over these inputs, LSTMs are composed of the following elements:

Cell: The core of LSTM is the cell or memory (or memory cell), represented by c_t in Figure 2.2. The memory cell contains the same inputs (h_{t-1} and x_t) and outputs (h_t) as a normal recurrent network. However, the state of this cell is modified by the current state of the cell, along the forget and input/modulation gates.

Input gate: Usually called *save vector*, the input and input modulation gates weigh how much of input information impacts on the cell state. The difference between these two gates is that the latter is submitted to an activation function. As

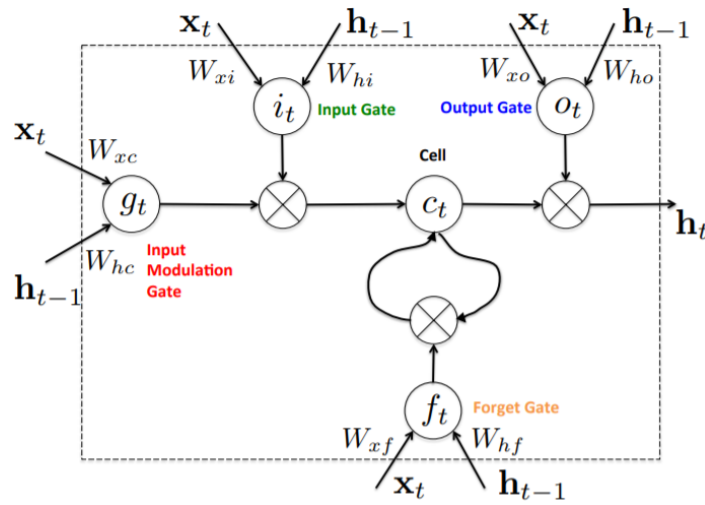


Figure 2.2: Representation of a LSTM unit. x represents the input, h the state of the unit and W represents the weights of each input regarding the gates. For each timestep, the unit receives the current input and the previous state [Chen, 2016].

vanilla RNNs, the input of LSTM units are the previous hidden state of the unit and the data for the current timestep.

Forget gate: Usually called *remember vector*, the forget gate ponders the information that should be kept in the cell state, providing low weights to those that should be forgotten.

Output gate: Usually called *focus vector*, the output gate determines what values from the cell state are going to the hidden state output (h_t).

Hidden state: Usually called *working memory*, this information is analogous to the hidden state of RNNs and Hidden Markov Models (HMM). The hidden state corresponds to the information that is going to be considered on the next timestep.

Equations 2.1, 2.2, 2.3, 2.4 and 2.5 rule the LSTM operation, representing the activations of forget, input and output gates, besides the update of cell and hidden state. In these Equations, σ represents a sigmoidal activation function, \circ represents element-wise product and b represents the bias. Previous and current timesteps are represented by $t - 1$ and t sub-indexes, respectively.

$$f_t = \sigma_g(W_f x_t + U_f h_{t-1} + b_f) \quad (2.1)$$

$$i_t = \sigma_g(W_i x_t + U_i h_{t-1} + b_i) \quad (2.2)$$

$$o_t = \sigma_g(W_o x_t + U_o h_{t-1} + b_o) \quad (2.3)$$

$$c_t = f_t \circ c_{t-1} + i_t \circ \sigma_c(W_c x_t + U_c h_{t-1} + b_c) \quad (2.4)$$

$$h_t = o_t \circ \sigma_h(c_t) \quad (2.5)$$

2.1.2 Gated Recurrent Unit (GRU)

Gated Recurrent Units (GRUs) are a variation of LSTM proposed by Cho et al. [2014]. Similarly to LSTMs, GRU units are designed to deal with gradient problems and show to be efficient on the modeling of sequential data. However, GRU units are simpler and faster to train, presenting superior performance on small datasets [Cho et al., 2014; Le et al., 2016]. Instead of input, forget and output gates, GRU cells contain two gates: an update gate and a reset gate, as illustrated in Figure 2.3. Different from LSTM, there is no persistent cell state.

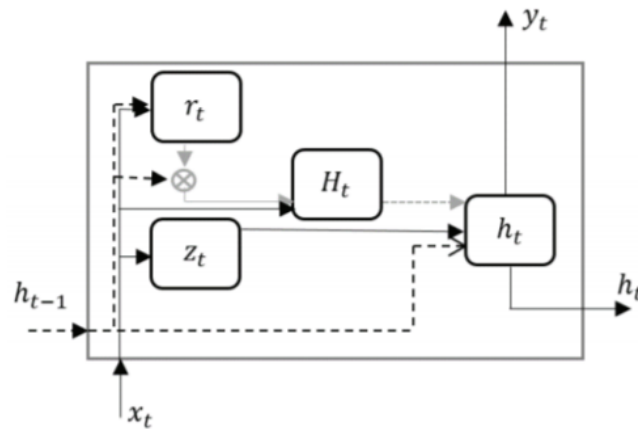


Figure 2.3: Representation of a GRU unit. x represents the input, r represents the reset gate and z the update gate [Le et al., 2016].

GRU units are composed of the following elements:

Update gate: This gate (z_t) weighs the contribution of a new input (x_t) to the hidden state of the unit (h_t).

Reset gate: This gate (r_t) controls how much of previous state is forgotten, through the combination of it (h_{t-1}) and the current input (x_t).

Hidden representation: This representation (H_t) combines information from the current input, previous hidden state and the response of reset gate, which is submitted to an activation function and is used to compute the next hidden state of the unit.

Hidden state: Analogous to the hidden state of LSTMs, the hidden state (h_t) corresponds to the information that is going to be considered on the next timestep. In the case of GRU, it takes into account the previous hidden state and the response of update gate.

Equations 2.6, 2.7, 2.8 and 2.9 rule the GRU operation, representing the activations of reset and update gates, besides the update of hidden state. In these Equations, σ represents a sigmoidal activation function and b represents the bias. Previous and current timesteps are represented by $t - 1$ and t sub-indexes, respectively.

$$r_t = \sigma_g(W_r x_t + U_r h_{t-1} + b_r) \quad (2.6)$$

$$z_t = \sigma_g(W_z x_t + U_z h_{t-1} + b_z) \quad (2.7)$$

$$H_t = \tanh(W_x x_t + U_H(r_t h_{t-1})) \quad (2.8)$$

$$h_t = (1 - z_t)h_{t-1} + z_t H_t \quad (2.9)$$

2.2 Autoencoders

Autoencoders are simple learning circuits which aim to transform inputs into outputs with the least possible amount of distortion [Baldi, 2011]. First introduced by Rumelhart et al. [1986], autoencoders try to learn a function that is regulated by back-propagation and minimizes the difference between input and output. This function (L) is commonly represented by $L(x, y) = \min \sum (x - h_{W,b}(x))^2$, where x represents the input, y the reconstructed output, $h_{W,b}$ the learning function of the autoencoder, W the weight matrix and b the bias. Since the output tends to minimize the same data provided as input with no need of labels, autoencoders are considered unsupervised models.

An autoencoder is a network that implements two transformations - encoding: $\mathbb{R}^n \rightarrow \mathbb{R}^d$ and decoding: $\mathbb{R}^d \rightarrow \mathbb{R}^n$, where n corresponds to the original dimensionality of

the data and d corresponds to the dimensionality of the encoded representation of the data [Kuchaiev and Ginsburg, 2017]. Typically employed for dimensionality reduction, autoencoders employ a d that tends to be lower than n . Figure 2.4 depicts the structure of an autoencoder. The model could be clearly divided into encoding and decoding networks and, at the middle, an encoded (compressed) representation is produced.

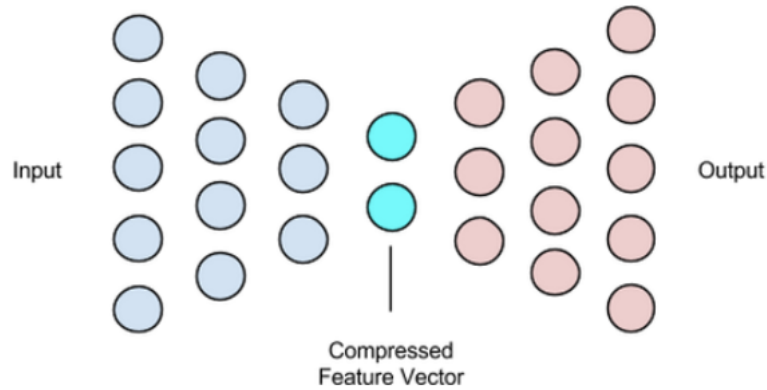


Figure 2.4: Basic representation of an autoencoder. Encoding network is represented by blue, while decoding is represented by red [Hadji and Wildes, 2018].

Besides the common employment in data compression (dimensionality reduction), autoencoders present a wide applicability in tasks, such as clustering [Guo et al., 2017], data denoising/restoration [Suganuma et al., 2018] and anomaly detection [Zhao et al., 2017], with a huge investment, on recent years, on deep autoencoder models based on convolutional operations. In addition, autoencoder variations have been formulated with successful applications in different fields, including denoising [Gondara, 2016], sparse [Dumas et al., 2016] and variational autoencoders [Inoue et al., 2018].

2.3 Multi-Task Learning

Multi-task corresponds to a paradigm in machine learning that aims at leveraging useful information contained in multiple related tasks to help to improve the generalization performance of all the tasks [Caruana, 1997; Zhang and Yang, 2017].

Since the last decade, several multi-task learning approaches have been developed with successful applications in different fields, such as natural language processing [Deng et al., 2013], speech recognition [Collobert and Weston, 2008] and computer vision [Girshick, 2015]. The employment of an auxiliary task introduces an inductive bias, producing models that prefer hypothesis that explain more than one task at once, following the idea of a biological human learning process [Zhang and Yang, 2017]. All of the tasks in a multi-task model, or at least a subset of them, are assumed to be

related to each other. In this case, it is found that learning these tasks jointly can lead to a performance improvement compared with learning them individually [Zhang and Yang, 2017].

According to Caruana [1997], multi-task learning helps models to improve their generalization accuracy, speed of learning and intelligibility through the inductive bias of different tasks. In addition, since multi-task models need to solve different tasks at once, their parameters are shared. As a consequence, these models are considered less prone to problems such as overfitting [Zhang et al., 2017]. Figure 2.5 shows a schematic of a multi-task model. One could notice the existence of several outputs. In this case, each output represents a different task.

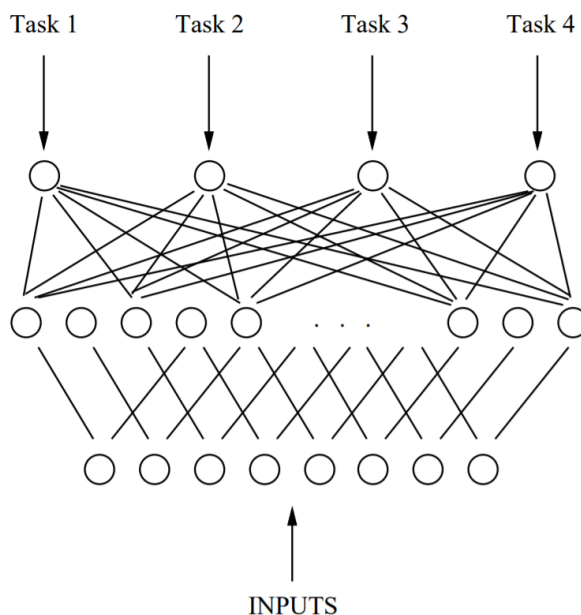


Figure 2.5: Schematic of a multi-task model for 4 different tasks [Caruana, 1997].

2.4 Hyperparameter optimization with Hyperband

The recent success of machine learning algorithms for a wide range of problems led to more and more complex architectures, composed of a growing and difficult to adjust set of hyperparameters [Li et al., 2016, 2017]. The quality of a predictive model critically depends on its hyperparameter configuration, since they present a huge impact on how the algorithm's performance generalize to unseen data [Li et al., 2017].

Hyperband is an exploration method that addresses the allocation of resources among randomly sampled hyperparameter configurations, relying on an early-stopping strategy that prioritizes promising configurations and throws out the rest, what in-

creases the performance of the method [Li et al., 2017]. The intuition behind Hyperband regards the fact that if a hyperparameter configuration is destined to be the best after a large number of iterations, it tends to perform in the top half of configurations after a small number of iterations. That is, even if the performance after a small number of iterations is very unrepresentative of the configurations absolute performance, their relative performance, compared with many alternatives trained with the same number of iterations, is roughly maintained [Li et al., 2016].

The Hyperband optimization relies on the employment of a search space, containing the possible values that each hyperparameter can assume; and two major parameters: (i) maximum number of epochs per configuration (*max - iter*) and (ii) downsampling rate (*eta*). Thereby, the configurations sampled from the defined search space are trained by a maximum number of epochs. The best of them, indicated by the amount specified on the downsampling rate, go to a next step. This process is repeated until the best configuration(s) remain(s), evidencing the best hyperparameters for the architecture.

2.5 Large-margin softmax loss

Large-margin softmax loss (L-Softmax) is an adaptation of a custom cross-entropy loss associated with a softmax layer, with the aim of encouraging intra-class compactness and inter-class separability [Liu et al., 2016]. The main point of this approach is to enhance the generalization of softmax, producing potentially larger angular margins between learned features.

L-Softmax is a special case of custom softmax, incorporating a pre-set constant m , which is taken into account to produce angular classification margins for each class. With larger margins between classes, L-Softmax presents advantages over softmax: (i) it generates more discriminative features, (ii) it partially avoids overfitting by defining a more difficult learning target, casting a different viewpoint to the overfitting problem and (iii) L-Softmax benefits not only classification, but also verification problems where ideally learned features should have the minimum inter-class distance being greater than the maximum intraclass distance [Liu et al., 2016]. The formulation of L-softmax follows the Equation 2.10, where W_{y_i} represents the weights of the y -th column of softmax layer weight matrix, associated with a sample i , with x_i being inputs of this layer. The activations of a softmax layer are the inner product between W and x , being formulated by $\|W_j\| \times \|x_i\| \times \cos(\theta_j)$, for a sample i , considering the column associated with class j and where θ_j corresponds to the angle between the vector W_j

and x_i . The consideration for the constant m is a stronger requirement to correctly classify x , producing more rigorous decision boundaries. The higher the value of m , the larger are the margins and the learning objective becomes harder to be achieved.

$$L_i = -\log\left(\frac{e^{\|W_{y_i}\| \|x_i\| \psi(\theta_{y_i})}}{e^{\|W_{y_i}\| \|x_i\| \psi(\theta_{y_i})} + \sum_{j \neq y_i} e^{\|W_{y_i}\| \|x_i\| \psi(\theta_{y_i})}}\right), \quad (2.10)$$

and

$$\psi(\theta) = \begin{cases} \cos(m\theta), & 0 \leq \theta \leq \pi/m \\ D(\theta), & \pi/m < \theta \leq \pi \end{cases}, \quad (2.11)$$

where $D(\theta)$ is required to be a monotonically decreasing function, and $D(\frac{\pi}{m})$ should equal $\cos(\frac{\pi}{m})$. Figure 2.6 depicts CNN features learned with L-Softmax. One can notice that, with an increasing m , features from different classes become more compact and distant to each other.

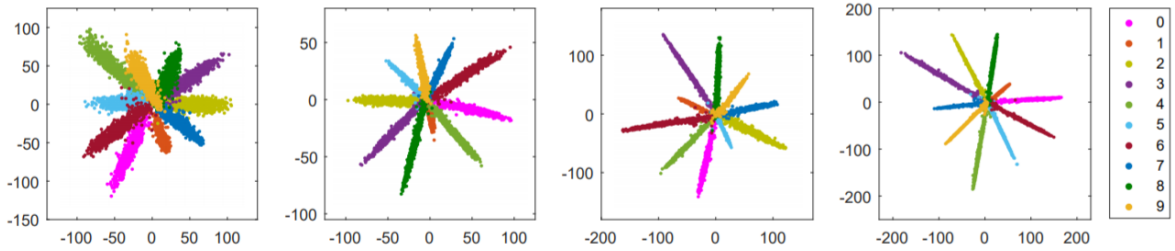


Figure 2.6: L-Softmax learned features for different classes of CIFAR-10 dataset [Krizhevsky, 2009]. The constant m is being increased, assuming values $m = 1, 2, 3, 4$ from left to right. [Liu et al., 2016]. With $m = 1$, this loss behaves as a custom softmax layer associated with a cross-entropy loss.

Chapter 3

Literature Review

Most gesture recognition approaches are based on the extraction/learning of spatiotemporal features from videos [Zhang et al., 2017]. This point is related to the importance of two main factors for the recognition of gestures: (i) appearance, which brings information about gesture parameters, such as hand configuration, body/facial expression and inflection point [Wang et al., 2017a]; and (ii) motion, which represents the movement executed by the performer [Pigou et al., 2017].

The applicability in several contexts led gesture recognition to be constantly targeted by approaches since the last decades. These methods, initially based on the employment of handcraft spatiotemporal feature descriptors [Wang et al., 2011; Song et al., 2012; Bastos et al., 2015], tend to capture shape, appearance and motion clues, mostly via image gradients and optical flow [Molchanov et al., 2016].

In the research proposed by Bastos et al. [2015], for example, it is explored the discriminability associated with hand configuration (shape) on the classification of signs of Brazilian Sign Language (Libras). On their research, Bastos et al. [2015] combined the response from Histogram of Oriented Gradients and Zernike Invariant moments, both related to image shape description. At the end, a two layer Perceptron Multilayer classifier is employed to classify Libras signs, reaching an accuracy superior to 96% in an own assembled dataset¹. Figure 3.1 illustrates hand images from the dataset proposed by Bastos et al. [2015] representing different Libras signs.

Song et al. [2012] proposed an approach to track hand and body parts to recognize gestures. Their method is able to extract appearance information from hand/body parts, which is associated with a Support Vector Machine classifier. An interesting point of the research of Song et al. [2012] is their ability to handle unsegmented

¹Libras hand recognition dataset is available: <http://sites.ecomp.uefs.br/lasic/projetos/libras-dataset>

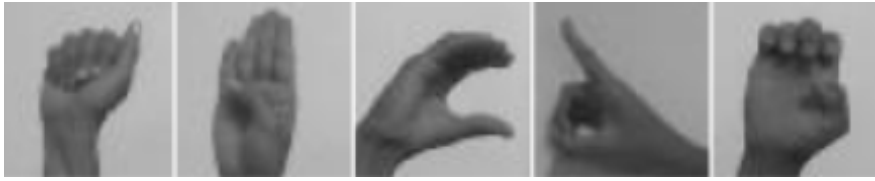


Figure 3.1: Libras hand recognition dataset [Bastos et al., 2015].

streams by the combination of local and global predictors, which take into account small portions of video inputs along with the entire videos. The approach is evaluated on NATOPS dataset [Song et al., 2011], performing both gesture detection and gesture recognition.

Still characterizing shapes, the research proposed by [Zhang and Tian, 2015] presented a descriptor to encode depth maps into a 3D shape representation named Histogram of 3D Facets. This descriptor represents both 3D shapes and structures of various depth maps, encoding information from hand structures, such as bumps and grooves. As a result, the approach achieved accuracy rates superior to 90% on gesture and activity recognition datasets, such as MSR3D gesture [Wang et al., 2012] and MSRAction3D [Li et al., 2010].

Despite accurate results of handcrafted gesture recognition methods, the advance of GPUs allowed a growing trend toward the application of deep neural networks on the task. These models are able to efficiently learn representations to characterize gestures and classify them with high accuracies [Zhang et al., 2017; Molchanov et al., 2016; Nishida and Nakayama, 2016]. Since the present study relies on the employment of these networks for the gesture recognition task, a higher emphasis is given to approaches based on deep models.

3.1 Deep Gesture Recognition Approaches

Recent studies have demonstrated that learning spatiotemporal features by deep models is a crucial point for effective gesture recognition [Nishida and Nakayama, 2016; Zhang et al., 2017]. Approaches based on deep models correspond to state-of-the-art methods in most gesture recognition datasets. They vary in terms of capacity/complexity and architecture, tackling different aspects of the task [Molchanov et al., 2016; Wang et al., 2017a].

According to the strategy employed to recognize gestures with deep models, approaches can be characterized into different groups: (i) those that employ feedforward deep classification models, with no explicit dependency between gesture timesteps [Tran

et al., 2015; Duan et al., 2016], (ii) and those that explore the time structure of gesture videos employing recurrent layers [Nishida and Nakayama, 2016; Molchanov et al., 2016; Pigou et al., 2017; Wang et al., 2017a].

3.1.1 Feedforward Deep Approaches

Feedforward deep approaches regard the assembling of models to extract spatiotemporal information mostly through the application of convolutional operations, wherein connections between nodes do not form any cycle. These models employ two basic strategies to perform the gesture recognition task: (i) extraction of spatial and temporal information isolated; and (ii) extraction of these features at the same time by the extension of convolutional operations into temporal domain.

Despite dissimilarities between activity and gesture recognition tasks, such as the background clueing and lack of a temporal sequence of sub-events on activity videos, activity recognition methods show some efficiency for the recognition of gestures, mostly due to their spatiotemporal learning behavior. Approaches, such as the ones proposed by Karpathy et al. [2014] and Simonyan and Zisserman [2014], are commonly employed for gesture recognition. These methods rely on the extraction of spatial and temporal information from inputs isolated. On the method proposed by Karpathy et al. [2014], four temporal fusion strategies are explored, and it is proposed that slow fusion can get more global information in both spatial and temporal dimensions. On the other hand, Simonyan and Zisserman [2014] proposed a two-stream convolutional architecture which incorporates spatial and temporal networks isolated, being fused together at a latter stage.

The research designed by Duan et al. [2016] is based on the idea proposed by Simonyan and Zisserman [2014]. Their method, named two-stream consensus voting network (2SCVN), deals with RGB and optical flow modalities in a separate way. In addition, a convolutional network is assembled to deal with saliency and depth inputs. Each modality generates a voting representation (considering the classes of the dataset). After that, these representations are fused to output the gesture class, as depicted in Figure 3.2. Though not employing recurrent layers or spatiotemporal operations, that research obtained recognition outcomes that rival with state-of-the-art methods on ChaLearn IsoGD dataset [Wan et al., 2016].

Narayana et al. [2018] invested in an multi-modality architecture that takes into account RGB, depth, optical flow and pose features as inputs. The main point of their approach consists on the employment of a spatial attention mechanism to focus on the hands. Besides the input modalities, Narayana et al. [2018] provide the coordinates

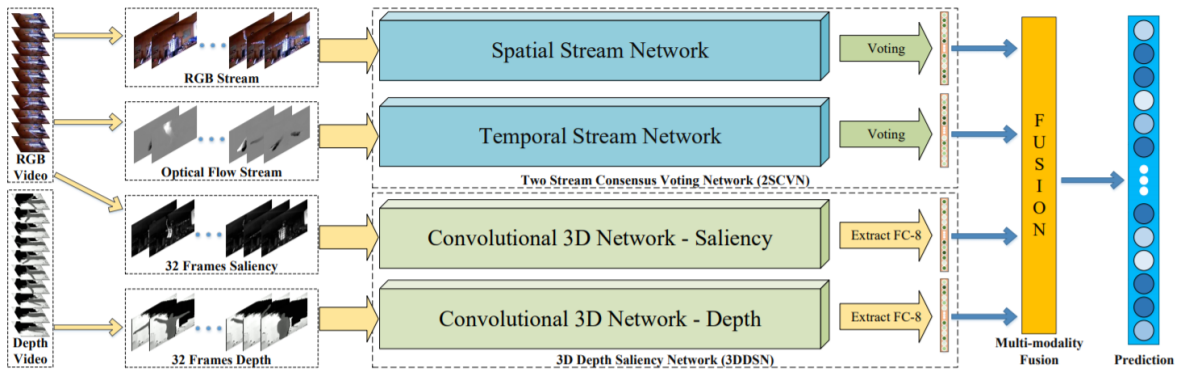


Figure 3.2: Architecture proposed by Duan et al. [2016]. Different networks are employed to deal with each modality. At end, a fusion is performed considering the voting response of each of them.

associated with each hand as inputs to their model. At the end, a sparse layer fusion is used to combine the information from each modality. Their method, associated to the pose estimation technique proposed by Cao et al. [2017] and hand detector proposed by Liu et al. [2017], achieved state-of-the-art accuracy on ChaLearn LAP Isolated Gestures [Wan et al., 2016] and NVIDIA Driving datasets [Molchanov et al., 2016].

Cardenas et al. [2019] proposed a method based on the extraction of information from RGB-D inputs, aiming at gathering texture information and describing/detecting hand movement. In addition, the hand configuration is obtained from a specific video frame, determined through the application of rank-pooling. Two convolutional neural architectures are tested on the approach, which is evaluated on LSA64 Ronchetti et al. [2016] and the proposed LIBRAS-BSL datasets Cardenas et al. [2019].

Imran and Raman [2019] proposed a methodology to deal with gestures based on a prior generation of three types of motion templates from the input videos: Motion history image (MHI), Dynamic Image (DI) and RGB Motion Image (RGBMI). These templates are computed over videos and provided to a network, based on MobileNET architecture [Howard et al., 2017], to extract feature information. Afterwards, a Kernel-based Extreme Learning Machine (KELM) classifier is used to compute the output for each template, with a late fusion being performed at the end. Imran and Raman [2019] evaluated their approach on SKIG and ChaLearn LAP Isolated Gestures datasets.

In the line of models that extract spatiotemporal information in a jointly way, Tran et al. [2015] proposed the C3D network, a straight-forward architecture based on stacked 3D convolutions, being widely used for video analysis and recognition tasks. This model is also explored by gesture recognition methods, such as the one proposed by Zhu et al. [2016], being evaluated on ChaLearn LAP Isolated Gestures [Wan et al., 2016].

The approach proposed by Zheng et al. [2019] presents a temporal segment network, composed of stacked 3D convolutional layers to extract spatiotemporal information from sampled frames of a video. Their approach considers RGB, Optical Flow and Depth as input modalities, performing a late fusion of these representations to estimate the gesture class of a video. The method achieved a competitive accuracy result, in relation to state-of-the-art methods, on ChaLearn LAP Isolated Gestures dataset [Wan et al., 2016].

3.1.2 Recurrent Approaches

Most of the recent studies employ recurrent models for the task of gesture recognition [Molchanov et al., 2016], achieving impressive state-of-the-art results for several datasets. The argument for the employment of recurrent layers stems from their ability to gather the dependency of previous events that exist on gesture videos, exploring the well-structured time disposition of sub-events on the performing of a gesture.

The structure of gesture videos is explored by Wu et al. [2016]. Their study highlights the importance of temporal information on gesture videos, pointing to the employment of recurrent layers in order to learn dynamic temporal dependencies. Wu et al. [2016] compare different approaches that handle gesture videos, ranging from purely 3-D convolutional networks to recurrent networks with different cell architectures, including vanilla RNN, GRU and LSTM. Figure 3.3 shows detection and recognition results obtained by Wu et al. [2016] considering different model architectures. The rectangular shapes on ground-truth (Targets) represent the occurrence of a gesture during a certain amount of time (or frames), while the color indicates the class this gesture belongs to. The outcomes of the different architectures try to approximate the response exposed in Targets, which evidence accurate recognition and detection of gestures. According to Figure 3.3, a better performance is achieved with a deep model involving 3-D (temporal) convolutions and LSTM units.

On the approach designed by Nishida and Nakayama [2016], a multi-stream architecture is proposed considering different modalities as inputs. Each stream, composed of 3D convolutional layers and a Long-Short Term Memory (LSTM) layer, receives frame-level information at every timestep from corresponding modalities, which include RGB, depth and optical flow. At the end, with the aim of embedding the specific modality representations in a single space, an additional LSTM layer is placed on the top of these streams. Figure 3.4 depicts the architecture proposed by Nishida and Nakayama [2016]. One could notice the existence of different streams and the application of a recurrent layer on top of them. The idea of multiple modalities as inputs

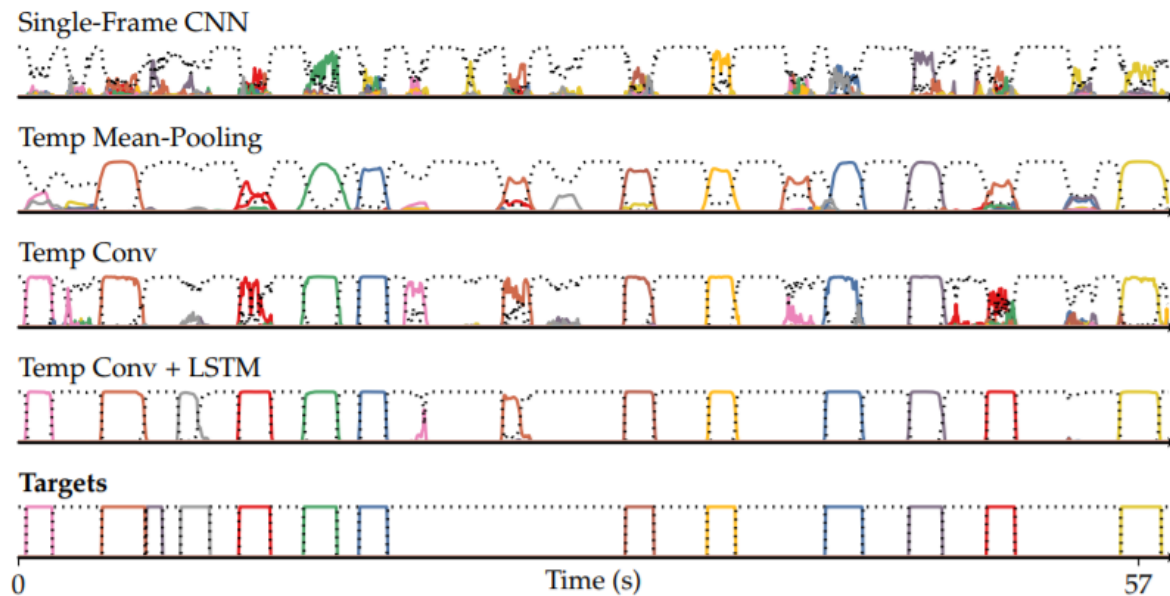


Figure 3.3: Wu et al. [2016] outcomes for a test video of ChaLearn LAP 2014 [Escalera et al., 2014]. Different strategies are presented along with their responses. Dashed lines represents no-class (silence) [Wu et al., 2016].

is explored by several gesture recognition approaches, including ours. However, differently from the method proposed by Nishida and Nakayama [2016], our methods fuse multi-modality information before providing it to the recurrent layers, performing 3D convolutional operations over multi-modality data and recurring a representation that is derived from it.

Molchanov et al. [2016] presented an approach that exploits the split of gesture streams into sub-videos (clips), taking into account modalities such RGB, depth, optical flow and infrared (IR). From each clip, spatiotemporal features are extracted through activations of C3D network [Tran et al., 2015]. After that, these features are used to feed a recurrent network, being posterly driven to a layer with connectionist temporal classification (CTC) cost function. With that, a single response can be produced for inputs with different sizes (i.e., different number of clips). In our detection/recognition approach, instead of employing CTC, we invested on a model that performs classification of one frame of the input at each timestep, producing a response for these frames with no need to split videos into clips. The architecture proposed by Molchanov et al. [2016] is showed in Figure 3.5. On their study, the contribution of each modality is evaluated for the gesture recognition task.

The accurate outcomes obtained with recurrent models led to the development of even more complex approaches, these based on the employment of bidirectional recurrent layers. Thereby, information from previous and future frames are taken into

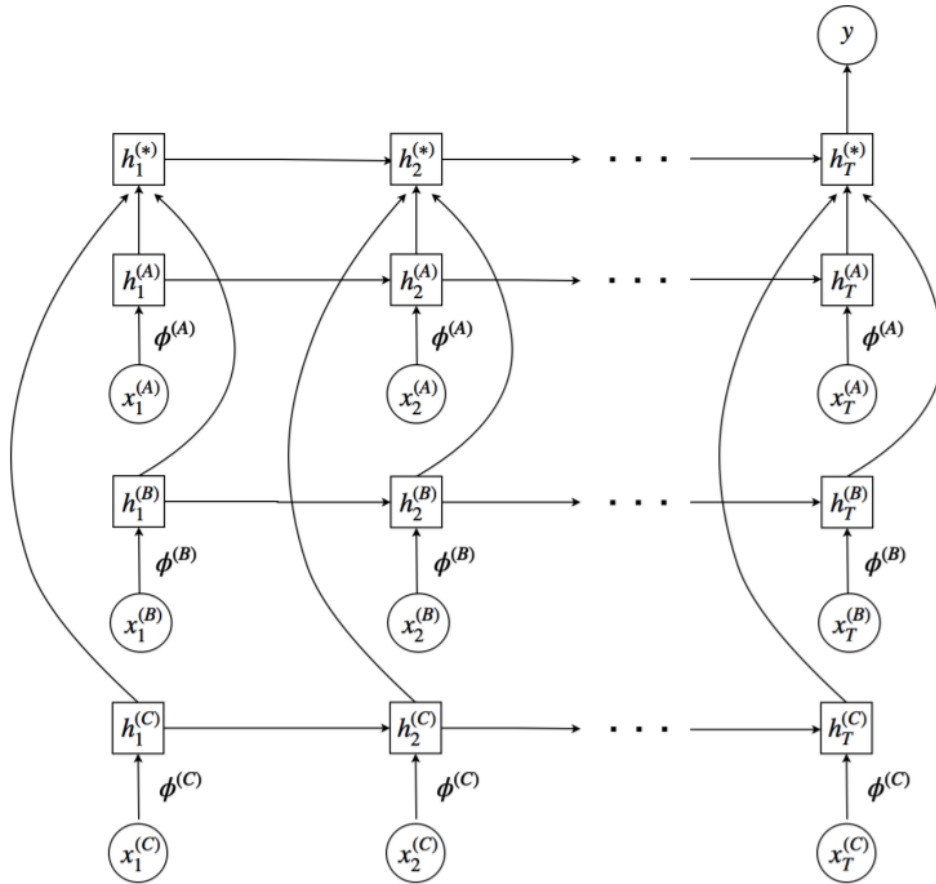


Figure 3.4: Nishida and Nakayama [2016] Multimodal Architecture. Each circle represents an input, denoted by $x^{(A)}$, $x^{(B)}$ and $x^{(C)}$. These inputs are associated to a different modality stream, composed of 3D convolutional/pooling layers. Squares represent recurrent LSTM layers. Every stream is fused on the recurrent layer on the top, represented by hidden states $h^{(*)}$. Arrows on recurrent layers (squares) expose the behavior on different timesteps, evidenced by the increasing indexes on inputs and hidden states ($x_1^{(A)}$, $x_2^{(A)}$, ..., $x_T^{(A)}$).

account on the classification process. On the method proposed by Zhang et al. [2017], for instance, a 3D convolutional model is proposed and, on top of it, bidirectional LSTM layers are placed. After gathering features from recurrent layers, a pooling strategy is employed to produce a single representation from multiple timesteps. The model proposed is illustrated in Figure 3.6(a). One could notice the application of a 2D convolutional network to act over the representation obtained with the recurrent layer.

In a similar way to Zhang et al. [2017], Pigou et al. [2017] invested on bidirectional models to detect and recognize gestures on videos. On their research, residual modules are employed with the aim of conserving gradients; important point on the training of deep networks. These modules present separate layers to perform spatial and temporal

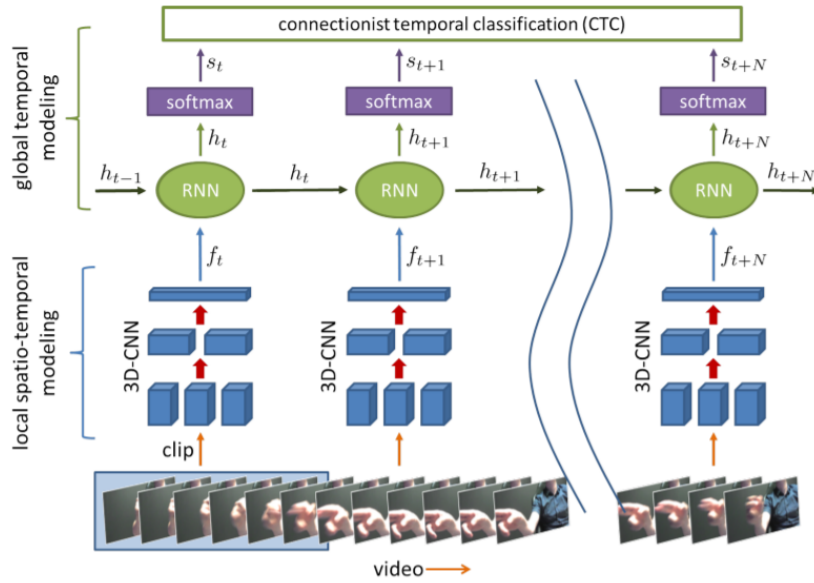


Figure 3.5: Architecture proposed by Molchanov et al. [2016]. Different timesteps of recurrent network are represented by h_t s, while softmax responses are represented by s_t s. One could notice the existence of a final layer employing CTC cost function.

operations. As a result, a lower number of parameters is employed by the network and, at end, the same operations are executed. In addition, these modules present ELU layers to incorporate a non-linear behavior. Figure 3.6(b) depicts the residual block proposed by Pigou et al. [2017]. One could notice the existence of skip connections, important factor for the gradient maintenance. These modules were incorporated by the present research in order to develop our detection/recognition approach. In addition, our model to handle unsegmented streams also employs a bi-directional recurrent layer to explore the structured order of sub-events of gesture videos. However, despite similarities between our approach and the one proposed by Pigou et al. [2017], we use different inputs and merge strategies, besides a different architecture to tackle the problem; with a higher number of spatiotemporal convolutional layers and employment of a dual-task policy, in which we gather information from the correlation that exists between temporal detection and recognition of gestures.

Zhu et al. [2018] also invested on an approach to detect and recognize gestures on unsegmented videos. However, instead of producing a single model to do both tasks, Zhu et al. [2018] employed an isolated temporal recognition network based on Res3D architecture [Tran et al., 2017], able to produce isolated videos through the recognition of boundaries (transition frames). To balance the two classes employed in this task (boundaries and non-boundaries), a balanced squared hinge loss function was applied. In addition, temporal dilations are included on the convolutional layers of this

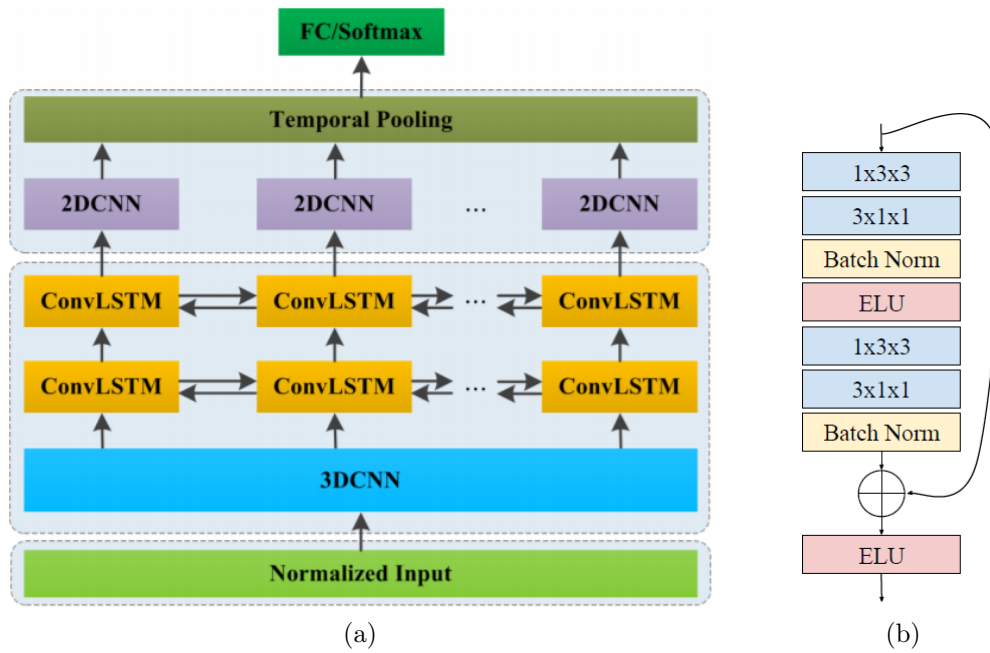


Figure 3.6: (a) Architecture proposed by Zhang et al. [2017]. On this architecture, a 2DCNN is used to learn higher-level spatiotemporal features maps for the final gesture classification. (b) Residual block proposed by Pigou et al. [2017]

network with the aim of enhancing the gathering of contextual information.

Even though the temporal segmentation being the main point of their research, Zhu et al. [2018] also proposed a network to perform the classification of gestures. This network, composed of 3D convolutional layers and convolutional LSTMs, follows a similar structure of the one proposed by Zhang et al. [2017], with the difference of using a pre-developed architecture, MobileNET [Howard et al., 2017], as a final classifier of the approach. Despite using MobileNET, the approach is end-to-end and MobileNET weights are trained along the entire network. Figure 3.7 depicts both networks used on the approach of Zhu et al. [2018]. One could notice that multiple modalities are used as inputs, such as depth, RGB and optical flow. At the end, a multimodal fusion is used to produce the recognition response.

Since gesture recognition encompasses different fields of application, approaches such as the ones proposed by Cao et al. [2018] and Camgoz et al. [2018] have been developed with expressive outcomes. The first presents an end-to-end Egocentric Gesture Recognition approach. On their research, Cao et al. [2018] elaborated a model based on the application of different spatiotemporal blocks that compensate undesired motions and estimate homography parameters to deal with warp caused by head movements. Besides that, an LSTM layer models the temporal dependencies that exist between video timesteps. Cao et al. [2018] evaluated their approach on a proposed dataset

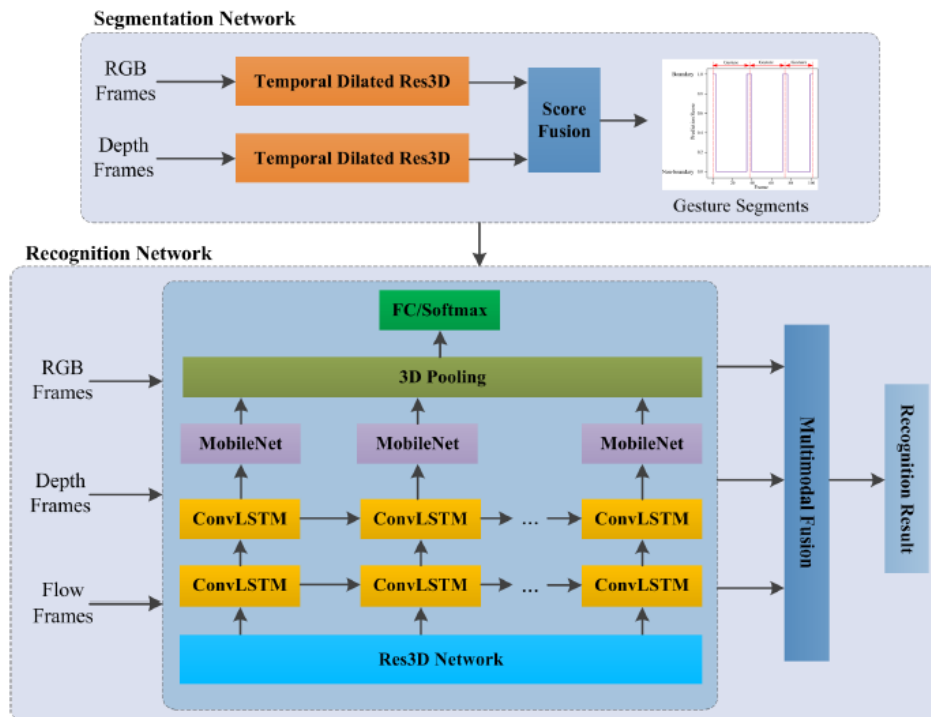


Figure 3.7: Architectures proposed by Zhu et al. [2018] for detection and recognition of gestures.

named EgoGesture, for which their method outperforms state-of-the-art approaches considering metrics such as temporal Jaccard.

Camgoz et al. [2018] presented an approach to perform sign language recognition. Despite being a field of gesture recognition, dealing with sign language recognition is a more complex task, since these approaches need to handle with linguistic aspects of the language, such as the absence of a one-to-one mapping of signs to spoken words [Bastos et al., 2015; Camgoz et al., 2018]. Thus, to incorporate the sign language behavior into the translation process, Camgoz et al. [2018] invested on a sequence-to-sequence model (seq2seq), similar to neural machine translation approaches. On that, frames are fed to a network responsible for extracting spatial information, which is projected into dense space by a fully-connected layer. A tokenization layer is employed to produce tokens from multiple inputs (several frames can be associated with a single word, for example), feeding recurrent layers. With this process, an encoded representation of an input is produced. After that, the decoding must be performed. Thus, the encoded representation is driven to an attention layer, which focuses on most important information inside it. This attention layer mitigates the problems generated by using fixed-size representation for the inputs. At the end, information is driven to a fully connected layer that outputs the sentence word by word. On their research, Camgoz

et al. [2018] performed tests considering GRU and LSTM layers. The sequence-to-sequence architecture employed in their research is illustrated in Figure 3.8.

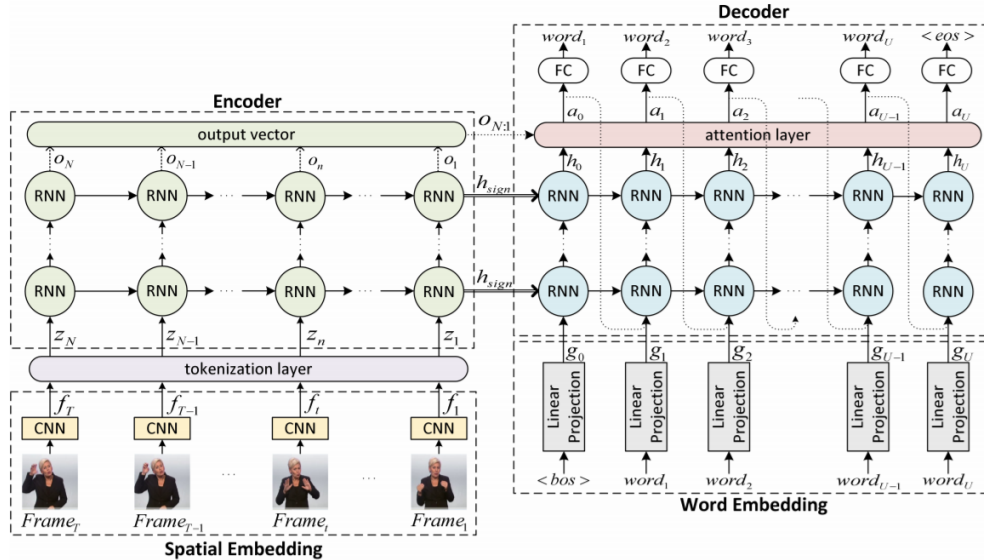


Figure 3.8: Camgoz sequence-to-sequence architecture. One could notice the existence of encoding and decoding steps. Words are outputted one-by-one.

Even presenting accurate outcomes for the gesture recognition task, we could notice gaps on literature researches, mainly regarding the lack of scalability and existence of few approaches to deal with unsegmented videos, with none of them using detection and recognition tasks with the aim of improving each other. In this sense, we propose methods to deal with these points named Multi-Output Recurrent Autoencoders (MORA) and its extension, Skin-based Adversarial Like Multi-Output Recurrent Autoencoders (SALMORA); and Multi-Loss Recurrent Residual Network (MLRRN).

Table 3.1 summarizes characteristics of the approaches encompassed in this literature review.

Table 3.1: Summarization of gesture approaches on literature and the proposed methods. *Rec.* regards the usage of recurrent layers. *GR* stands for Gesture Recognition, *GD* stands for Gesture Detection, while *SLR* means Sign Language Recognition.

Author	Approach	Rec.	Inputs	Task	Datasets
Bastos et al. [2015]	HOG Zernike	No	RGB	GR	Libras Hand Signs [Bastos et al., 2015] NTU Hand Digit [Ren et al., 2011]
Song et al. [2012]	Particle Filter HOG	No	RGB	GR GD	NATOPS [Song et al., 2011]
Zhang and Tian [2015]	Hist. of 3D Facets	No	Depth	GR	MSR 3D [Wang et al., 2012] MSRAction3D [Li et al., 2010]
Duan et al. [2016]	2-Stream Consensus Voting Network	No	RGB Optical Flow Depth Saliency	GR	Challearn IsoGD [Wan et al., 2016]
Narayana et al. [2018]	FOANet	No	RGB (Appearance) Optical Flow Depth Pose	GR	Challearn IsoGD [Wan et al., 2016] NVIDIA Driving [Molchanov et al., 2016]
Cardenas et al. [2019]	DSLRL + CNN + Texture Maps	No	RGB Depth Hand Coordinates	SLR	ISA64 [Ronchetti et al., 2016] LIBRAS-BSL [Cardenas et al., 2019]
Imran and Raman [2019]	ConvNET+KELM	No	MHI DI RGBMI	GR	Challearn IsoGD [Wan et al., 2016] SKIG [Liu and Shao, 2013]
Zhu et al. [2016]	Pyramidal 3D ConvNET	No	RGB Depth	GR	Challearn IsoGD [Wan et al., 2016]
Zheng et al. [2019]	MMFTSN	No	RGB Depth Optical Flow	GR	Challearn IsoGD [Wan et al., 2016]
Wu et al. [2016]	DDNN		LSTM Vanilla RNN GRU	GR GD	C. Montalbano [Escalera et al., 2014]
Nishida and Nakayama [2016]	MRNN		RGB Optical Flow Depth	GR	SKIG [Liu and Shao, 2013]
Molchanov et al. [2016]	R3DCNN		RGB Optical Flow Depth Infrared	GR GD	SKIG [Liu and Shao, 2013] C. Montalbano [Escalera et al., 2014] NVIDIA Driving [Molchanov et al., 2016]
Zhang et al. [2017]	3DCNN		RGB Optical Flow Depth	GR	SKIG [Liu and Shao, 2013] Challearn IsoGD [Wan et al., 2016]
Pigou et al. [2017]	SLR Network		RGB Depth	GR GD SLR	C. Montalbano [Escalera et al., 2014] Challearn ConGD [Wan et al., 2016]
Zhu et al. [2018]	3DCNN-ConvLSTM-2DCNN		RGB Depth Optical Flow	GR GD	Corpus NGT [Grasbörn et al., 2008] Corpus VGT Herweghe et al. [2015] Challearn ConGD [Wan et al., 2016]
Cao et al. [2018]	EgoGesture		RGB Depth	GR	EgoGesture [Cao et al., 2018]
Gangoz et al. [2018]	Neural SLR		RGB Depth	SLR	PHOENIX-2014 [Forster et al., 2014]
Bastos et al. [2018]	MORA		RGB Depth Optical Flow	GR	SKIG [Liu and Shao, 2013] Challearn IsoGD [Wan et al., 2016]
Bastos et al. [2019]	MLRRN		RGB Skeleton Joints	GR GD	Challearn Montalbano [Escalera et al., 2014] Challearn ConGD [Wan et al., 2016]
Bastos et al., [2020]	SALMORA		RGB Depth Optical Flow	GR	SKIG [Liu and Shao, 2013] Challearn IsoGD [Wan et al., 2016]

Chapter 4

Proposed Approaches

Gesture recognition methods invest on the collection of information from spatiotemporal nuances of gestures, gathered from relations of different parts of video clips. Despite the accurate results, most gesture recognition approaches are based on the employment of discriminative models and are adjusted to handle gesture datasets in the literature, leading to two main issues: (i) lack of scalability in terms of number of gestures; (ii) inability to handle unsegmented videos.

In this chapter, we introduce our proposed approaches to deal with the two main issues that were noticed in the gesture recognition literature. These approaches, named *Multi-output Recurrent Autoencoders* (MORA) and its extension, *Skin-based Adversarial Like Multi-output Recurrent Autoencoders* (SALMORA), and *Multi-Loss Recurrent Residual Network* (MLRRN), are presented in next sections.

4.1 Dealing with scalability issues - MORA

MORA is a novel strategy to recognize gestures based on the employment of multiple autoencoder models. For each class (gesture), a different model is assembled and trained. The main point that supports the employment of these multiple models is related to scalability, in the sense that the recognition of a novel gesture only requires the training of one new autoencoder model for the novel gesture, without the need of retraining for all instances of the training set, as commonly performed.

On MORA, each autoencoder model presents three inputs as well as three outputs to be reconstructed. These inputs are (i) the RGB gesture video, (ii) the Farneback optical flow [Farneback, 2003] response and (iii) the depth representation of the video, as illustrated in Figure 4.1. With these inputs, MORA contemplates factors that

play a relevant role on the recognition process, such as motion, appearance and depth (inflection point).

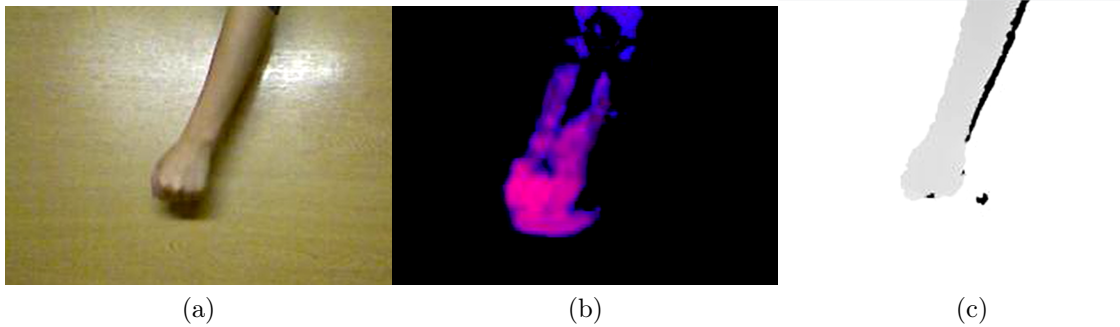


Figure 4.1: MORA inputs: a) RGB video. b) Farneback Optical Flow. c) Depth.

On the test phase, gesture videos are provided to all MORA trained models, each representing a different class. At the end, the one that presents the lowest reconstruction error indicates the class (the index of the model represents the class). This error is computed by the sum of differences between each input and its reconstructed output. The MORA recognition process is illustrated in Figure 4.2.

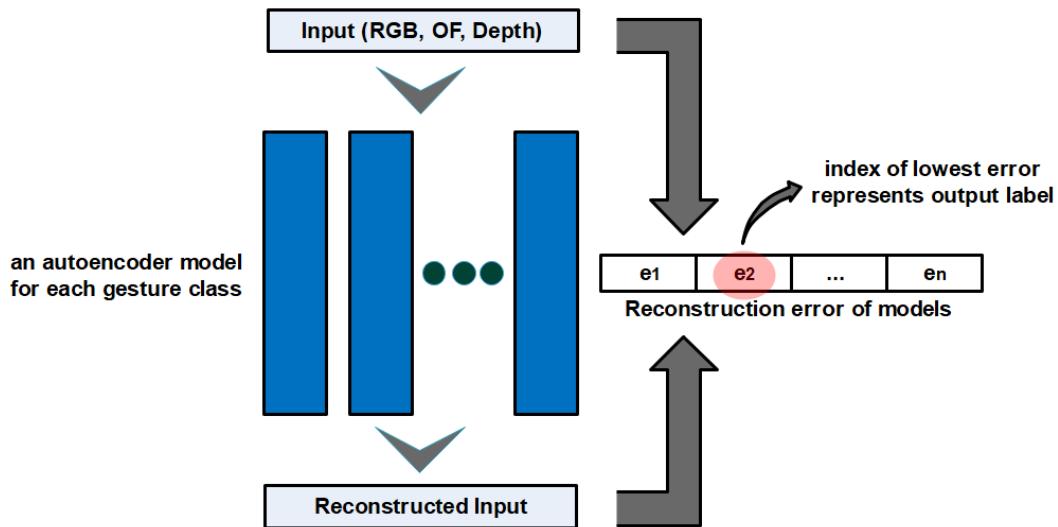


Figure 4.2: MORA test phase. Index of model with lowest error represents the class.

Since MORA is unable to deal with varied-length videos, we performed pre-processing steps before training/testing our approach. Each dataset video is resized to $112 \times 112 \times 3$ (rows, columns and channels). To standardize the length of videos and avoid redundant information, a uniform subsampling is performed, producing 32-frame videos.

As aforementioned, gestures comprise a well-defined time structure that must be properly captured for accurate recognition. With this goal in mind, we assemble our models with a Gated Recurrent Unit (GRU) layer to propagate information through different timesteps (shown in Figure 4.3). Afterwards, we adjust the input of our models accordingly to process 8-frame clips each time, resulting in four timesteps for each video, as they are sampled with 32 frames. The disposition of video inputs in different timesteps is related to the employment of a recurrent layer on MORA, which is able to correlate information from these different timesteps and model dependencies that exist on the data. The choice for four timesteps was empirically determined by tests on the validation splits of the evaluated datasets.

The autoencoder model for each gesture class learns spatiotemporal information from the three inputs, i.e., RGB, optical flow and depth, reconstructing them at the end. This reconstruction minimizes a multi-output loss function, which is based on the sum of the mean squared error between each input and their reconstructed counterpart, as showed in Equation 4.1, where $pRGB$, pOF and $pDepth$ represent MORA reconstructed outputs, while RGB , OF and $Depth$ represent the inputs. It is important to emphasize that both the state and the output of the GRU layer are used to produce a combined representation for each timestep, resulting in a vector with 784 dimensions.

$$MSELoss = \frac{1}{n} \sum_{i=1}^{i=n} (RGB_i - pRGB_i)^2 + \frac{1}{n} \sum_{i=1}^{i=n} (OF_i - pOF_i)^2 + \frac{1}{n} \sum_{i=1}^{i=n} (Depth_i - pDepth_i)^2 \quad (4.1)$$

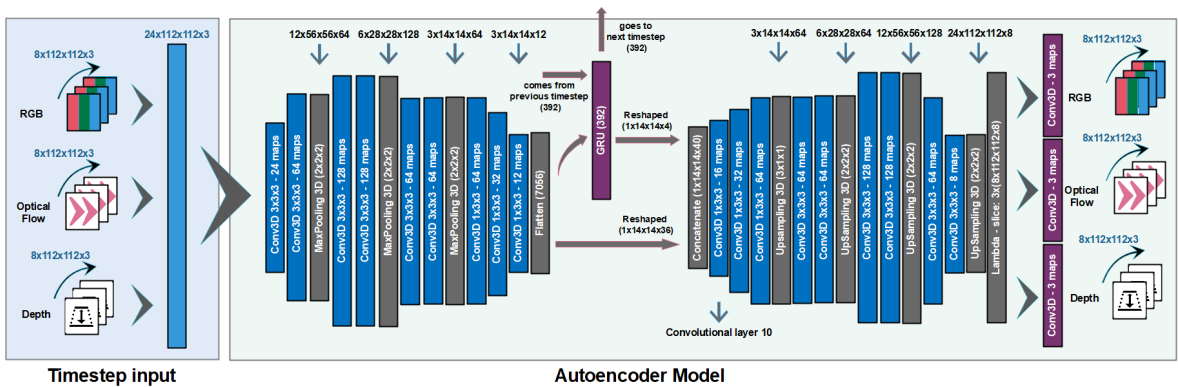


Figure 4.3: Proposed autoencoder model. Layers employ ReLU (blue) and Sigmoid (purple) activations.

As depicted in Figure 4.3, MORA inputs for each timestep are clips composed of $8 \times 112 \times 112 \times 3$ dimensions, representing the number of frames, width, height and channels, respectively. These inputs are merged, resulting in a 24-dimension input (8

frames for each input type at once) for each timestep. One can notice in Figure 4.3 that the *Depth* input is presented as a 3-channel input. To achieve this channel dimensionality, we replicated the monochromatic depth information in 3 channels for two main reasons: (i) to facilitate operations that actuate over data dimensions, such as pooling and convolutions and (ii) to keep a balance on the reconstruction loss at the end, since the model needs to reconstruct the 3 channels, similarly to the RGB input. A similar adjustment was performed for the optical flow input, since the Farneback optical flow response is a 2-channel output. To adequate the optical flow for our model, we created an empty channel (all pixels are black) and composed a 3-channel input. Without this 3-channel disposition, a higher weight would be given to the RGB input, leading to a dominance of RGB reconstruction over depth and optical flow.

After creating this 24-dimension input, the data is driven to consecutive 3D-convolutional and pooling layers, producing encoded information that goes to the GRU layer. In turn, this layer produces a recurrent representation, which is concatenated to the encoded information for each timestep, as depicted in Figure 4.3. The idea behind this merge lies on the fact that it would not be possible to minimize the value of the multi-output reconstruction loss function with only the low-dimensional representation obtained from the GRU layer. Besides that, increasing the size of this recurrent layer would greatly increase the number of parameters and the time to train the models, requiring more training data as a consequence. It is important to notice that the models consider a 4-timestep tensor as input and, after that, the state of the GRU layer is reset.

Besides scalability, the employment of a specific model for each class on MORA method leads to the other positive aspects, listed as follows.

Lower complexity: Since intra-class variation tends to be lower than inter-class variation, it is possible to generate autoencoder models with less parameters, resulting in models with faster training, easier convergence, less prone to overfitting and with a lower training data requirement.

Specificity: Classes with a higher complexity requirement can be associated with higher capacity models without requiring any retraining of previous models. In addition, since MORA employs a multi-output loss function, it is possible to assign specific loss weights to each class, considering the most relevant aspects for each of them.

Open-set applicability: Since models tend to minimize the reconstruction error of trained instances, it is expected to obtain a high error for an instance of an unknown class. Therefore, it is possible to associate high error values with non-trained classes, allowing the employment of MORA in open-set applications. Furthermore, MORA

presents high similarity with incremental learning methods and could be easily adapted to consider new data without requiring any change on previously trained models.

Robustness to unbalanced classes: Each model of MORA aims to minimize the reconstruction error for a specific gesture class. Therefore, the only impact MORA suffers from unbalanced datasets is that classes with more samples tend to require models with higher capacity.

4.2 Extending MORA - SALMORA

The trained autoencoders of MORA method are specific for each gesture class, aiming at reconstructing the inputs at the end. With this strategy, it is expected that models, trained for a particular class, present lower reconstruction error for video instances of this class. However, since MORA models are independent unsupervised autoencoders, these networks are not trained to enhance the distance/discriminability between different classes, what could lead to models, associated with different classes, containing weights tuned with certain level of similarity. Consequently, the reconstruction error becomes a less accurate metric and misclassifications can be obtained. Thus, to insert discriminability on the training process of MORA models, we enhanced our approach to encompass new elements: (i) a skin detection task and (ii) an adversarial-like behavior, contemplating interleaved steps of unsupervised autoencoders and a discriminator network. These improvements led to *Skin-based Adversarial-Like Multi-Output Recurrent Autoencoders*, or *SALMORA*.

4.2.1 Skin Detection as a new task

MORA reconstruction takes into account the input frames in an uniform way, attributing the same weight to regions related to the subject performing the gesture and the scene background. Consequently, gestures with similar backgrounds can present akin representations, leading the approach to misclassifications, even if the motion and appearance, mostly denoted by the performer, present huge differences. Thus, we incorporated a new task to MORA: Skin detection.

On the architecture depicted in Figure 4.3, a new branch is created with a new output, as showed in Figure 4.4, composed of extra convolutional layers, responsible for the skin map of the corresponding input. Before predicting the skin map, we create a feature representation with the same spatial dimensionality of the remaining outputs of the model, i.e., RGB, optical flow and depth. This feature representation results from a layer that employs logistic sigmoidal activation. In addition, this layer is adjacent to

the new skin output of the model, leading the representation to be imbued with skin information. As a result, the values that compose each frame of the representation can be associated with the other outputs, containing a probability score of each pixel to contain a skin zone. At the end, we perform an element-wise multiplication of this skin feature map by every channel of the remaining outputs of the model, resembling an attention mechanism. Equations 4.2, 4.3 and 4.4 represent this element-wise multiplication for the RGB, Optical Flow and Depth outputs, respectively. Every channel of an output for a sample s is multiplied by the skin feature map ($SFM(s)$).

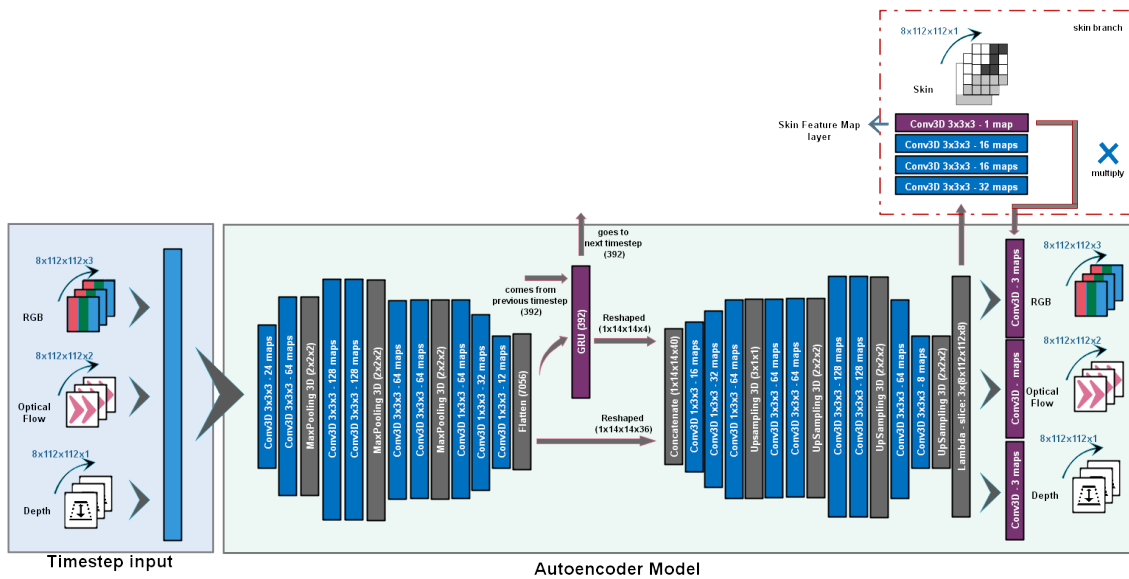


Figure 4.4: MORA autoencoder model with a skin branch (skin MORA model). A feature map is created and acts as an attention mechanism for the remaining outputs.

$$RGB(s) = [RGB_r(s) \odot SFM(s), RGB_g(s) \odot SFM(s), RGB_b(s) \odot SFM(s)] \quad (4.2)$$

$$OF(s) = [OF_x(s) \odot SFM(s), OF_y(s) \odot SFM(s)] \quad (4.3)$$

$$Depth(s) = [Depth(s) \odot SFM(s)] \quad (4.4)$$

4.2.2 MORA with Adversarial-Like Behavior

As a final step to enhance MORA results, we developed a strategy to increase the distance between the error response of the models representing each class. To accomplish that, we incorporate a discriminative step into the MORA pipeline.

The starting point of this adversarial approach is the training of unsupervised skin MORA models for each gesture class, as described in Section 4.2.1. After that, we freeze the weights of the autoencoders, except for the *Convolutional layer 10*, shown in Figure 4.3. The outputs of this layer are connected to the discriminator network depicted in Figure 4.5. This discriminator employs the binary Large-Margin Softmax Loss (L-Softmax) [Liu et al., 2016], which separates the class associated with the model from the remaining gesture classes with the highest possible margin, following a *one-against-all* strategy [Wu et al., 2019]. For example, if we consider a 5-class dataset, for an autoencoder representing the gesture class 1, the binary classifier to be attached to this autoencoder aims at separating class 1 from classes 2, 3, 4 and 5.

The large-margin softmax loss, showed in Equation 2.10, is designed to encourage intraclass compactness and interclass separability, tuning the weights of the *Convolutional layer 10* to incorporate a discriminative behavior with emphasis on the maximum separation between classes. As a final step of SALMORA, the discriminator network is removed from the pipeline, the weights of *Convolutional layer 10* are frozen and the remaining weights of each autoencoder model are unfrozen, being tuned again, in an unsupervised manner, to minimize the reconstruction error for the associated class. As a result, the autoencoder models intend to minimize the reconstruction error considering the discriminative weights of *Convolutional layer 10*, making the representations produced by the layers of the models to be more distant to each other. The full pipeline is depicted in Figure 4.6.

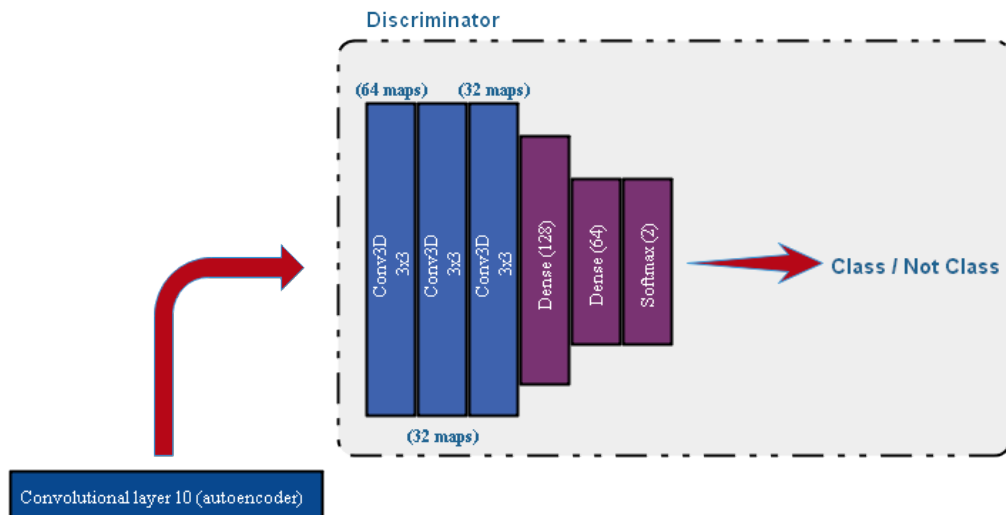


Figure 4.5: Classification model employed to enhance autoencoders with discriminative behavior. Layers employ ReLU (blue) and Sigmoid (purple) activations.

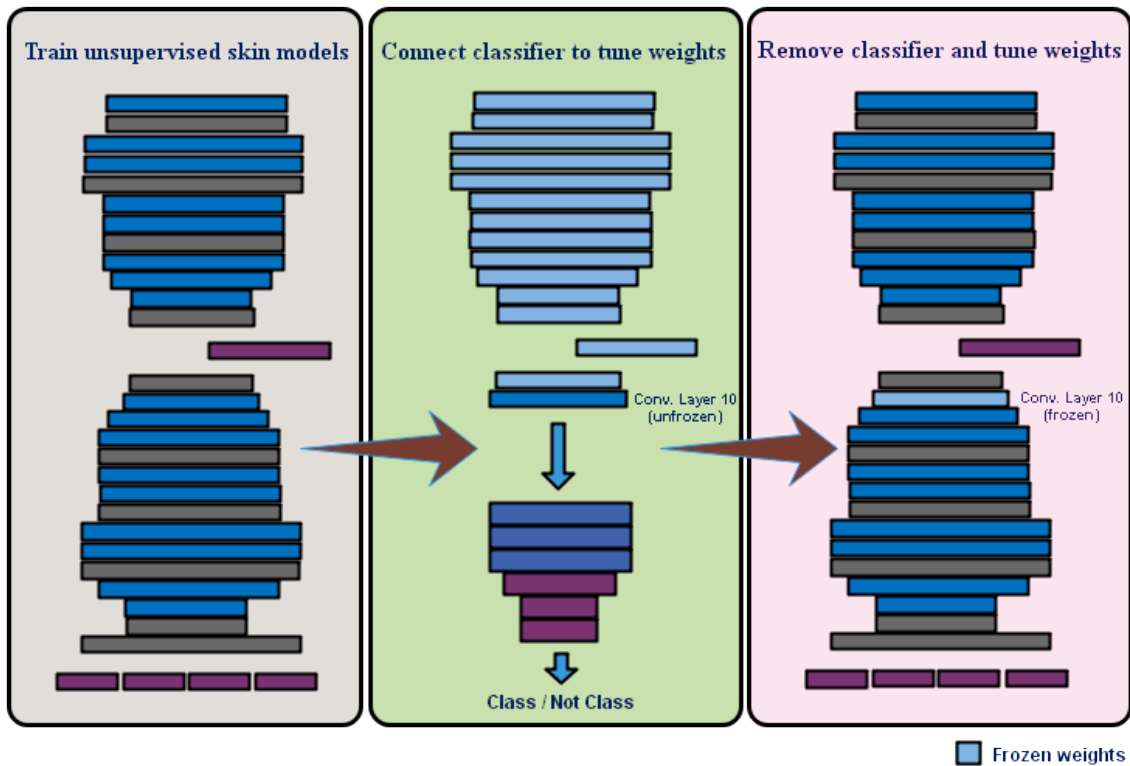


Figure 4.6: SALMORA pipeline with interleaved unsupervised (left and right) and supervised (middle) steps.

4.3 Dealing with unsegmented videos - MLRRN

Most gesture recognition approaches are designed to handle segmented videos, i.e., a single gesture is contemplated on the entire set of frames that compose the video. Despite the existence of accurate methods applied to segmented gesture recognition datasets, such as SKIG [Liu and Shao, 2013] and ChaLearn IsoGD [Wan et al., 2016], they are not suitable to perform gesture recognition on real-life situations, in which there exists a fluid conversation that is less and less dependent on any control over the communication scenario [Song et al., 2012]. In this sense, to handle unsegmented gesture videos, a different task shows to be relevant: the gesture detection.

Gesture detection or gesture temporal detection [Molchanov et al., 2016] is related to the limiting of the start and the end of a gesture in an unsegmented video, i.e., videos contemplating multiple gestures. This task corresponds to the marking of the frame interval comprised by this gesture. Despite simulating a real-life scenario (i.e., unsegmented stream of data), few approaches tackle temporal detection due to its high complexity. The assignment of the label gesture or no-gesture to each frame is a difficult task since the positive class (i.e., gesture) tends to present a very high

intra-class variation and the negative class (i.e., no-gesture) tends to suffer from the lack of standard postures, producing inconsistent behavior among users and even similarities with the gesture class [Molchanov et al., 2016; Pigou et al., 2017]. In addition, it is important to take into account the existence of three temporal phases on the gestures: preparation, nucleus and retraction. The nucleus, the central part of the gesture, is associated with motion and postures executed by the performer that characterize each gesture. In turn, preparation and retraction are transition phases that involve assuming a posture to start the gesture or going to relax postures, respectively. While the nucleus is the discriminative phase [Gavrila, 1999], the other two phases can be quite similar for different gesture classes and hence less useful or even detrimental to accurate classification, just representing transitions between no-gesture frames to gesture frames and vice-versa.

4.3.1 Proposed Model

To deal with unsegmented videos, we propose a model, named Multi-Loss Recurrent Residual Network (MLRRN), which performs gesture detection and recognition at once. This model presents three main characteristics:

Recurrent layers: As aforementioned, gestures present a well-structured time disposition of events, making room for an efficient application of recurrent models. Since state-of-the-art results are mostly achieved by these models and we intend to handle unsegmented videos, recurrent layers are extremely suitable for this task. With recurrent layers, our model is able to extract long-term dependencies and to establish relations between different frames of the input.

Frame-level input: MLRRN model was developed with a frame-level input, i.e., the input corresponds to one frame of the video per timestep. However, to provide local temporal information, for each frame we present to the model, we also provide the previous and the next four frames, producing a 9-frame tensor.

Multi-task (Detection and Recognition): Our model outputs labels for each frame of a video, performing both detection and recognition tasks at once. These tasks present a complementary behavior and when considered in a jointly way, they enhance the outcomes of the other. The detection task, for instance, gives a negative response for no-gesture frames, evidencing that these frames can not be associated with any class of recognition task. In turn, the recognition task emphasizes transition frames, revealing margins of gestures and no-gesture intervals. Thus, we developed a multi-task model to perform both tasks at once. Figure 4.7 illustrates the pipeline of MLRRN, with steps ranging from the video input providing to the network, to the class and detection

predictions. Sections 4.3.1.1 and 4.3.1.2 discuss the steps of the method.

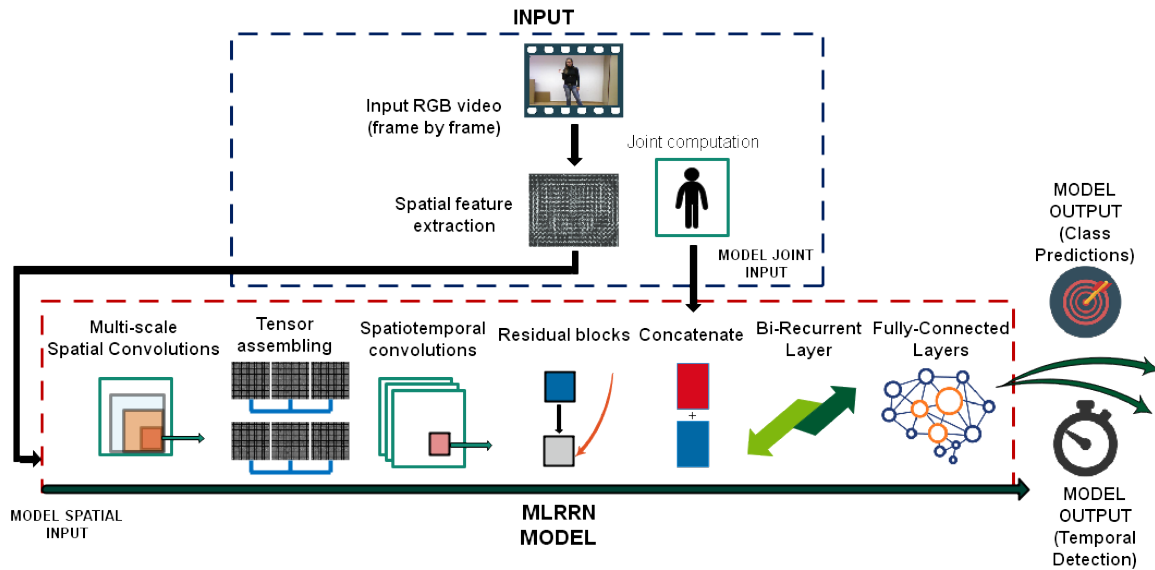


Figure 4.7: MLRRN pipeline.

4.3.1.1 Input

The main input of the MLRRN consists of individual frames of a video sequence. For each time offset (timestep), a next frame from the sequence is fed to the model. However, instead of using the raw RGB frame, we extract activations from the fully connected layer 7 ($fc7$) of VGG-16 trained on the ImageNet, with the aim of producing a spatial description of every frame. This feature contains 4,096 dimensions and was reshaped to a 64x64 representation before feeding our model (model spatial input). With this network, we achieved better outcomes than with the adding of spatial layers to the model, leading to a reduction on the number of parameters and a model that is less prone to overfitting, with lower training data requirement and easier convergence. As showed in Figure 4.7, we consider a secondary input (model joint input). This input corresponds to human body joints computed with the OpenPose technique [Cao et al., 2017], which provides coordinates of several joints of the human body. This information is used to produce a pose signature of individuals that are performing gestures on the video. Although human joints are provided by some gesture datasets, such as ChaLearn Montalbano [Escalera et al., 2014], we incorporated a pose estimation technique into our method, with the aim of making MLRRN applicable in different scenarios and datasets, even in those that do not provide joint coordinates.

4.3.1.2 MLRRN Model

The Multi-Loss Recurrent Residual Networks (MLRRN) is a multi-task architecture that performs two different tasks: gesture temporal detection and gesture classification. Since the inputs of the architecture are frames of a video, detection and classification labels are generated for each frame, considering both losses of the model, as depicted in Figure 4.7.

The first aspect to notice on MLRRN model is the *Spatial Input*. This input receives the spatial representation of VGG-16 network from every frame. However, instead of only using one frame at once, MLRRN takes into account the previous and subsequent frames, empirically determined as a 9-frame input for every frame of the video. For each new timestep, a 1-frame offset is performed and the current frame is updated, followed by the gathering of four previous and next frames in relation to this new current frame, as illustrated in Figure 4.8. According to this strategy, the first and last four frames of a video are never considered as the main input, but are used as auxiliary inputs for adjacent frames.

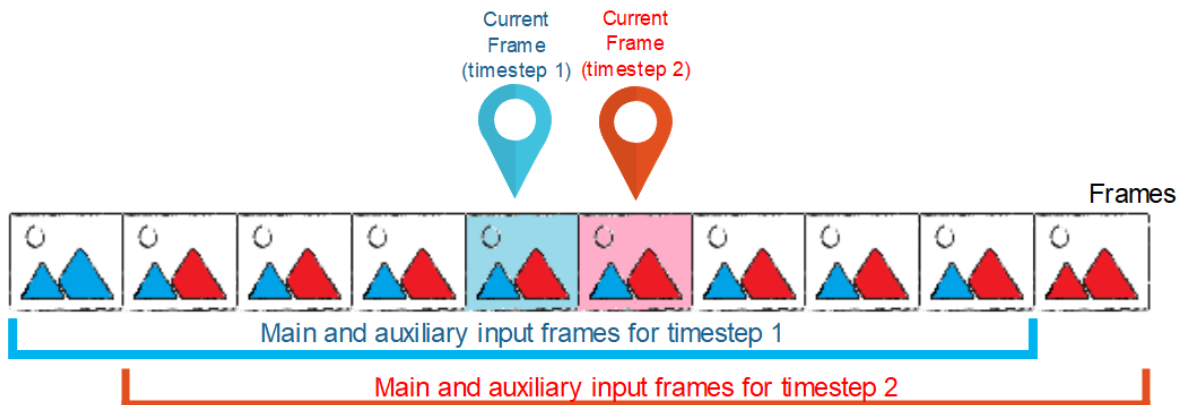


Figure 4.8: Input of MLRRN for different timesteps.

The second block of MLRRN model is called *Multi-scale Spatial Convolutions*. This block is represented by many spatial convolutional layers that consider different filter dimensions. As a consequence, information from different scales is gathered. In addition, this multi-scale block intends to mitigate the fact that MLRRN performs convolutions over fully-connected activations (reshaped to 64x64) from VGG-16. We enforce a spatial relation that is spread over the fully-connected representation and, with these convolutions considering different scales, we tend to better capture information from that. At last, it is important to mention that these convolutions are performed over each frame information (activations from current and auxiliary frames) isolated. The goal is to increase the capacity of the model at this point, produc-

ing richer representations from each frame separately, which contributes to provide wealthier information about frame appearance differences along the input frames for each timestep.

The next block is the *Tensor Assembling*. On this block, feature maps obtained from convolutional layers of individual frame responses are concatenated in different ways. Consequently, a tensor is produced by each combination of these maps, allowing the performing of spatiotemporal convolutions, since the maps from different frames represent a time variation on the input. Figure 4.9 illustrates the combinations that are performed and tensors produced from MLRRN inputs. One could notice that feature maps are produced from convolutions performed over activation responses of VGG-16. After that, 3-depth and 9-depth tensors are created, from which spatiotemporal (3D) convolutions are performed. At the end, responses of convolutions over all tensors are concatenated and propagated through the network.

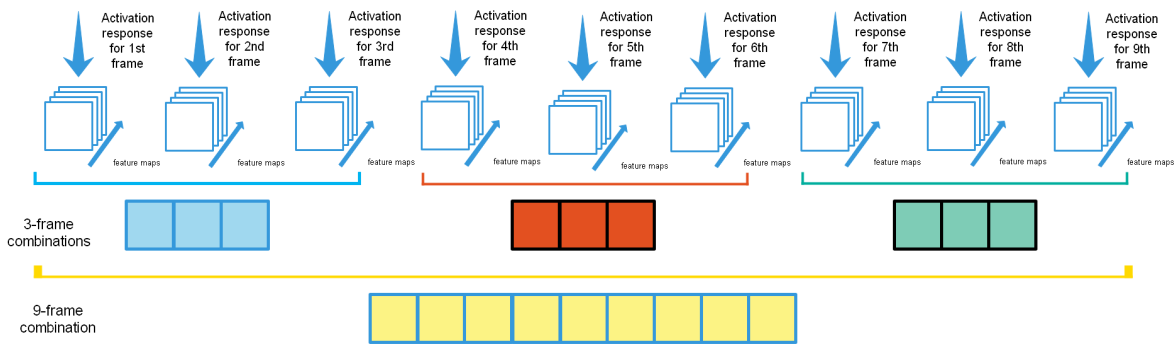


Figure 4.9: Assembling of tensors on the MLRRN architecture.

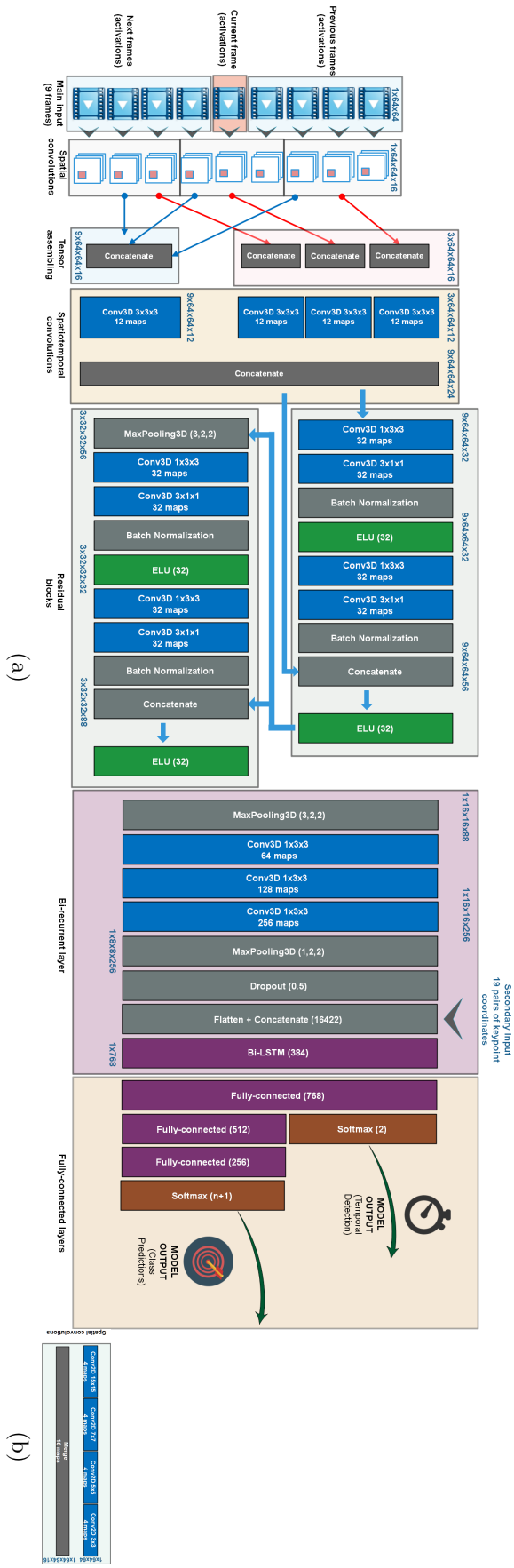
After creating the tensors and performing spatiotemporal convolutions, residual blocks are employed in MLRRN. These blocks, adapted from the research proposed by Pigou et al. [2017] and depicted in Figure 3.6(b), are important to maintain the gradient in a deep network, such as MLRNN. In addition, they allow the employment of operations, mostly convolutions, over representations with lower and higher degree of semantics.

The bi-recurrent layer is the next one on MLRRN model. This layer is crucial to explore the well-defined temporal behavior of gestures. In addition, with this layer, it is possible to obtain a response for each frame taking into account the high dependency that exists in relation to other frames. In the case of MLRRN, a bi-recurrent layer is employed, what leads to the gathering of information from previous and future frames. According to our tests, showed in Section 6.2.2, bi-LSTM layers presented better results than vanilla RNNs or GRUs, which is expected due to the long-term dependency that exists between a frame and its previous and future instances. Finally, the secondary

input of the model (human joints) is concatenated to the representation obtained from residual blocks, being both fed to this recurrent layer.

At the end, the MLRRN model presents a stack of fully-connected layers that act over the recurrent representation. These layers result in two output layers, one responsible for the class prediction (output is the label of a frame considering gesture classes) and the other responsible for the temporal detection (determines if a frame is part of a gesture or not). The complete MLRRN model is showed in Figure 4.10. One can notice that the outputs of the model are a binary and a $(n + 1)$ -class softmax layers, which apply binary and categorical cross entropy loss functions, respectively. The binary output corresponds to temporal detection, classifying frames as gesture and no-gesture. The other output has its dimensionality associated with the number of gesture classes of the dataset added by one. This addition of an extra class regards the existence of the no-gesture class also for this output. However, this class receives no-weight on the training of the model.

Figure 4.10: (a) MRRN architecture. Layers employ ReLU (blue), ELU (green), sigmoid (purple) and softmax activations (brown). (b) Spatial convolution blocks.



Chapter 5

Challenges and Benchmarks Addressed

Different benchmark datasets are used on gesture recognition and provide, along with data and their corresponding labels, evaluation protocols that aid to standardize outcomes of researches on this field.

This chapter introduces the datasets we used in this dissertation, including information such as number of classes, amount of training and testing videos and the challenges provided by each of them.

5.1 SKIG dataset

The Sheffield Kinect Gesture (SKIG) dataset [Liu and Shao, 2013] is a public video dataset commonly used by gesture recognition approaches. SKIG is composed of ten different gesture classes disposed in 2,160 videos, of which 1,080 are RGB and the 1,080 remaining are their corresponding depth sequences, all captured with Kinect sensors. The gesture classes are: *circle (clockwise)*, *triangle (anti-clockwise)*, *up-down*, *right-left*, *wave*, *Z*, *cross*, *come here*, *turn-around*, and *pat*, as depicted in Figure 5.1.

SKIG videos present a spatial resolution of 720x480 pixels. The dataset classes are balanced, with each one comprising 216 samples (108 RGB and 108 depth videos). To incorporate diversity, videos are recorded with six different subjects, under two illumination conditions (poor and strong light) and with three different backgrounds (wooden board, white plain paper and paper with characters). Since the authors did not establish a protocol on this dataset, gesture recognition approaches usually employ a 3-fold cross validation, as described by Zhang et al. [2017].

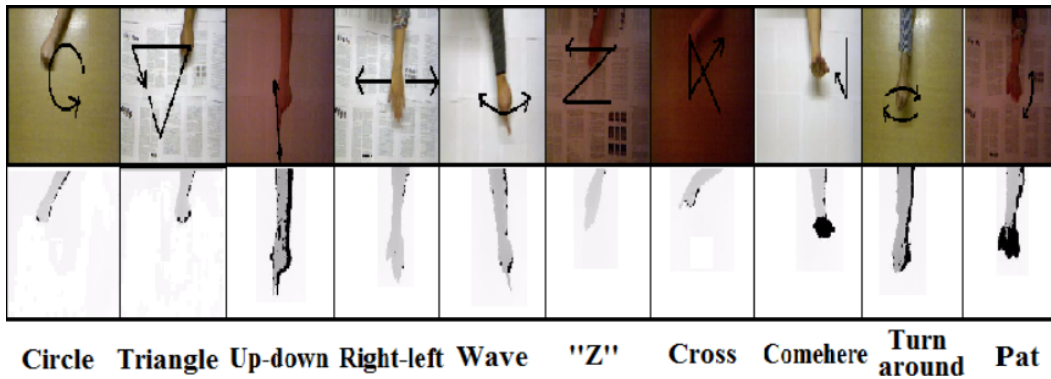


Figure 5.1: Different gesture classes of the SKIG dataset. RGB and corresponding depth videos are provided [Liu and Shao, 2013].

Although simple, SKIG includes challenges regarding the variations in terms of gesture execution of different performers, illumination variation and background cluttering. As a result, this dataset is employed by gesture recognition approaches, being widely used to estimate model hyperparameters.

5.2 ChaLearn Looking at People IsoGD dataset

ChaLearn Looking at People Isolated Gestures (ChaLearn LAP IsoGD) [Wan et al., 2016] is a public dataset composed of 47,933 RGB and the same amount of corresponding depth videos. Each video is related to a single human gesture, belonging to 249 different classes. ChaLearn videos present a varied spatial resolution and length, with long (about 100 frames) and short (about 12 frames) videos. To incorporate diversity, ChaLearn videos are performed by 21 different subjects, under different light conditions and with different backgrounds. Figure 5.2 presents frames from ChaLearn videos and their depth correspondences.

Due to the huge number of videos, classes and high variability, ChaLearn LAP IsoGD is a challenging dataset. In addition, the dataset classes are not balanced and recognition methods need to be robust to this issue to efficiently handle the dataset.

ChaLearn IsoGD presents a standard evaluation protocol, with three mutually exclusive subsets, used for training, validating and testing models. The number of videos for each subset is showed in Table 5.1.

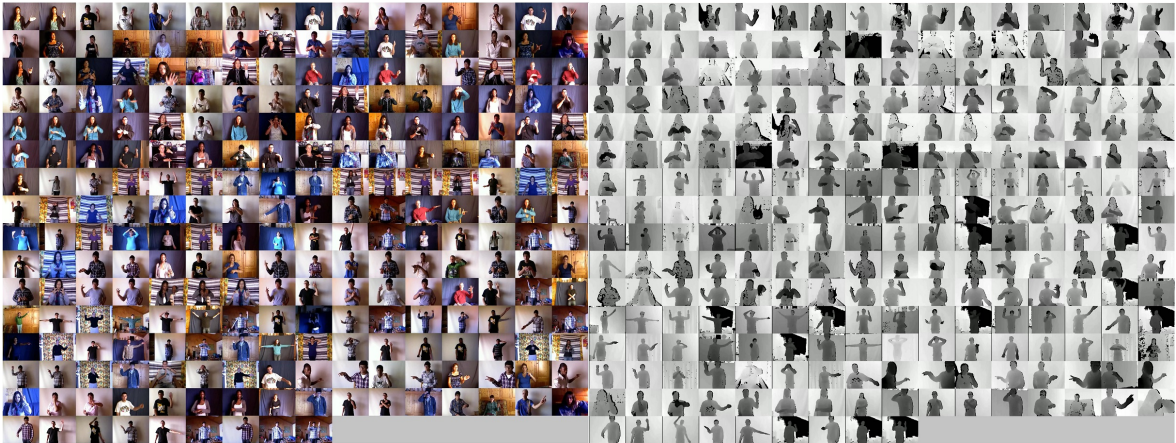


Figure 5.2: Frames from ChaLearn LAP IsoGD videos and their depth correspondences [Wan et al., 2016].

Table 5.1: Number of videos on Training, Validation and Test subsets of ChaLearn LAP IsoGD.

	Labels	Gestures	RGB videos	Depth videos	Performers
Training	249	35,878	35,878	35,878	17
Validation	249	5,784	5,784	5,784	2
Testing	249	6,271	6,271	6,271	2

5.3 ChaLearn Montalbano Multimodal Gesture Recognition

ChaLearn Montalbano Multimodal Gesture Recognition (ChaLearn Montalbano) [Escalera et al., 2014] is a public dataset composed of 940 RGB, depth and user-segmented videos. Differently from ChaLearn LAP IsoGD, this dataset simulates a continuous recognition scenario, with each video containing multiple gestures, resulting in more than 14,000 gestures from 20 Italian sign gesture categories. Figure 5.3 depicts a frame from a video of the ChaLearn Montalbano dataset, along with its depth correspondence, user-segmentation and skeleton annotation¹.

Despite not encompassing many gesture classes, ChaLearn Montalbano is a challenging dataset, since approaches need to temporally detect gestures before performing their recognition, i.e., they need to determine the frame interval in which a gesture occurs before assigning its class label. Consequently, ChaLearn Montalbano is used as an initial dataset for gesture detection/recognition approaches, for which the temporal

¹Despite stating that skeleton annotations are provided, one could notice that many annotations are missing and present wrongly annotated coordinates.

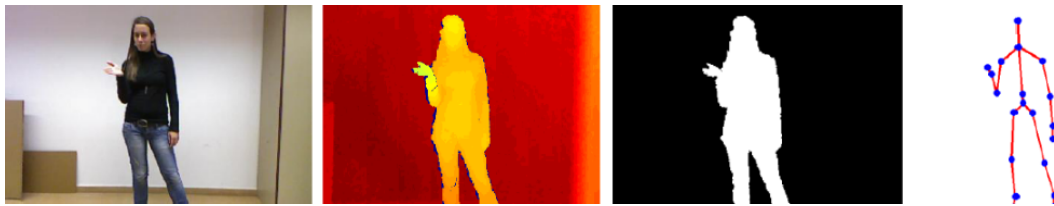


Figure 5.3: Frames from ChaLearn Multimodal Gesture Recognition videos and their depth, user segmentation and skeleton correspondences.

Jaccard is used as main evaluation metric besides the recognition accuracy. The amplitude of Temporal Jaccard is computed by $Amplitude(J_{s,n}) = \frac{A_{s,n} \cap B_{s,n}}{A_{s,n} \cup B_{s,n}}$, where $A_{s,n}$ is the ground truth of gesture n at sequence s and $B_{s,n}$ is the prediction of such gesture at sequence n .

On ChaLearn Montalbano, gestures are separated by intervals of frames in which the performers relax, being associated with none of the 20 classes of the dataset (no-class frames). In terms of evaluation, it designates a similar protocol to ChaLearn LAP IsoGD [Wan et al., 2016], with mutually exclusive training, validation and testing subsets. Table 5.2 shows the number of videos on each subset.

Table 5.2: Number of videos on Training, Validation and Testing subsets of ChaLearn Montalbano Multimodal Gesture Recognition.

	Labels	RGB videos	Depth videos	Performers
Training	20	470	470	20
Validation	20	230	230	20
Testing	20	240	240	20

5.4 ChaLearn Looking at People ConGD dataset

ChaLearn Looking at People Continuous Gestures (ChaLearn LAP ConGD) [Wan et al., 2016] is a public dataset that is analogous to the previously mentioned ChaLearn IsoGD. The main difference between these datasets is that ChaLearn ConGD videos contain multiple gestures, leading approaches to perform temporal detection before recognition.

ChaLearn CongGD regards 249 different classes, distributed in 22,535 videos. Similarly to IsoGD, this dataset presents a standard evaluation protocol with mutually exclusive training, validation and testing splits, with the amount of videos in each split showed in Table 5.3, along some dataset characteristics.

Table 5.3: Number of videos on Training, Validation and Testing subsets of ChaLearn LAP ConGD.

	Labels	Gestures	RGB videos	Depth videos	Performers
Training	249	30442	14314	14314	17
Validation	249	8889	4179	4179	2
Testing	249	8602	4042	4042	2

ChaLearn ConGD presents all challenges of its isolated version, with the addition of the necessity of temporal detection of gestures. This dataset is considered one of the most important benchmarks for continuous gesture recognition, presenting videos with high variability. Differently from ChaLearn Montalbano, ChaLearn ConGD dataset is annotated with no intervals (no-class frames) between gestures, with a 1-frame difference between gesture annotations. Figure 5.4 shows frames of ChaLearn ConGD videos.



Figure 5.4: Frames from ChaLearn ConGD videos. Different subjects are performing the gestures, under different illumination conditions and viewpoints.

Chapter 6

Experimental Results

In this chapter, we present the experimental results obtained with our proposed approaches: MORA, SALMORA and MLRRN. All approaches are evaluated in widely employed gesture recognition datasets. In the case of MORA and SALMORA, SKIG [Liu and Shao, 2013] and ChaLearn IsoGD [Wan et al., 2016] are used as benchmarks. MLRRN, however, is evaluated in ChaLearn Montalbano [Escalera et al., 2014] and ChaLearn ConGD [Wan et al., 2016], both considered challenging datasets for the gesture detection and recognition tasks.

The results will be presented in the following order: First, MORA and SALMORA outcomes will be presented along with their corresponding discussions, followed by MLRRN results. To place our approaches in literature, we carefully followed the protocols determined by each dataset guidelines, making our outcomes comparable to other methods.

6.1 MORA and SALMORA Evaluation

In this section, we describe the experimental results obtained with MORA and SALMORA for the gesture recognition problem, comparing their results to literature approaches.

6.1.1 Experimental Setup

The SKIG dataset [Liu and Shao, 2013] was used as an initial benchmark for MORA. We followed the protocol proposed by Zhang et al. [2017], consisting in a 3-fold cross validation, with specific parts for training, validating and testing the approach. SKIG validation subsets served as the basis for the setting of the initial hyperparameters of

MORA. The starting configuration of each layer, depicted in Figure 4.3, was determined through tests on these subsets.

MORA models are autoencoders used to reconstruct the input data. Consequently, some care had to be taken on the assembling of this initial architecture. It was important, for example, not to stack layers with a large difference in number of parameters [Szegedy et al., 2015] (number of feature maps, for instance). Besides, producing very low-dimensional representations could lead to high reconstruction errors, hindering the minimization of the loss function. In MORA, GRU layers are preferred in relation to LSTM. This choice is justified by the fact that this type of recurrent layer is able to extract relevant information from data that presents a well-defined temporal structure. LSTMs are also capable of such extraction, however, they have a larger number of parameters to be adjusted and tend not to behave well when a small set of input data is available [Greff et al., 2015]. This assertion was confirmed by tests considering the validation subsets of SKIG, for which the employment of a GRU layer provided better outcomes than LSTM.

As a final step, to improve the performance of the method, hyperband algorithm [Li et al., 2016, 2017] was used to adjust the hyperparameters associated with each of the 10 models trained on the dataset. For each MORA autoencoder, a set of hyperparameters was adjusted with the aim of enhancing the results. The architecture depicted in Figure 4.3 was used as a starting point, from which we executed hyperband ($max - iter = 18$ and $eta = 3$) to adjust the hyperparameters. The hyperband search space is shown in Table 6.1, for which $v1$, $v2$ and $v3$ represent the possible values a hyperparameter can assume.

Table 6.1: Hyperband search space for MORA approach. (Enc.), (Dec.), and (Rec.) Layer Factor represents an increase on the number of feature maps for layers on Encoding, Decoding and Recurrent layer of the autoencoders, respectively. Layer Growth stands for an addition or removal of a convolutional layer on encoding and decoding parts of the models.

	v1	v2	v3
Weight Initializer	normal	uniform	glorot normal
Optimizer	RMSProp	Adam	Adagrad
Enc. Factor	0.75	1.00	1.50
Dec. Factor	0.75	1.00	1.50
Batch Size	4	9	16
Rec. Layer Factor	0.75	1.00	1.50
Layer Growth	Removal	Addition	Preserve
Loss	mean absolute error	mean squared error	-

In terms of SALMORA, the extension of MORA, a skin detection task was incor-

porated to the models by a new branch created over the autoencoders, as illustrated in Figure 4.4. The hyperparameters of these models followed the arrangement obtained with the execution of hyperband for MORA models, with the addition of the layers to detect skin. To train the skin output of this new architecture, it was necessary to perform the skin rotation of the employed datasets, SKIG and ChaLearn IsoGD. To accomplish that, we assembled the skin detection model, represented in Figure 6.1 and trained on the Pratheepan dataset [Tan et al., 2012], which provides images and their corresponding skin maps, as shown in Figure 6.2. To detect skin, the model receives an image in RGB, HSV and YCbCr color models, and outputs a skin map with skin probability for each image pixel.

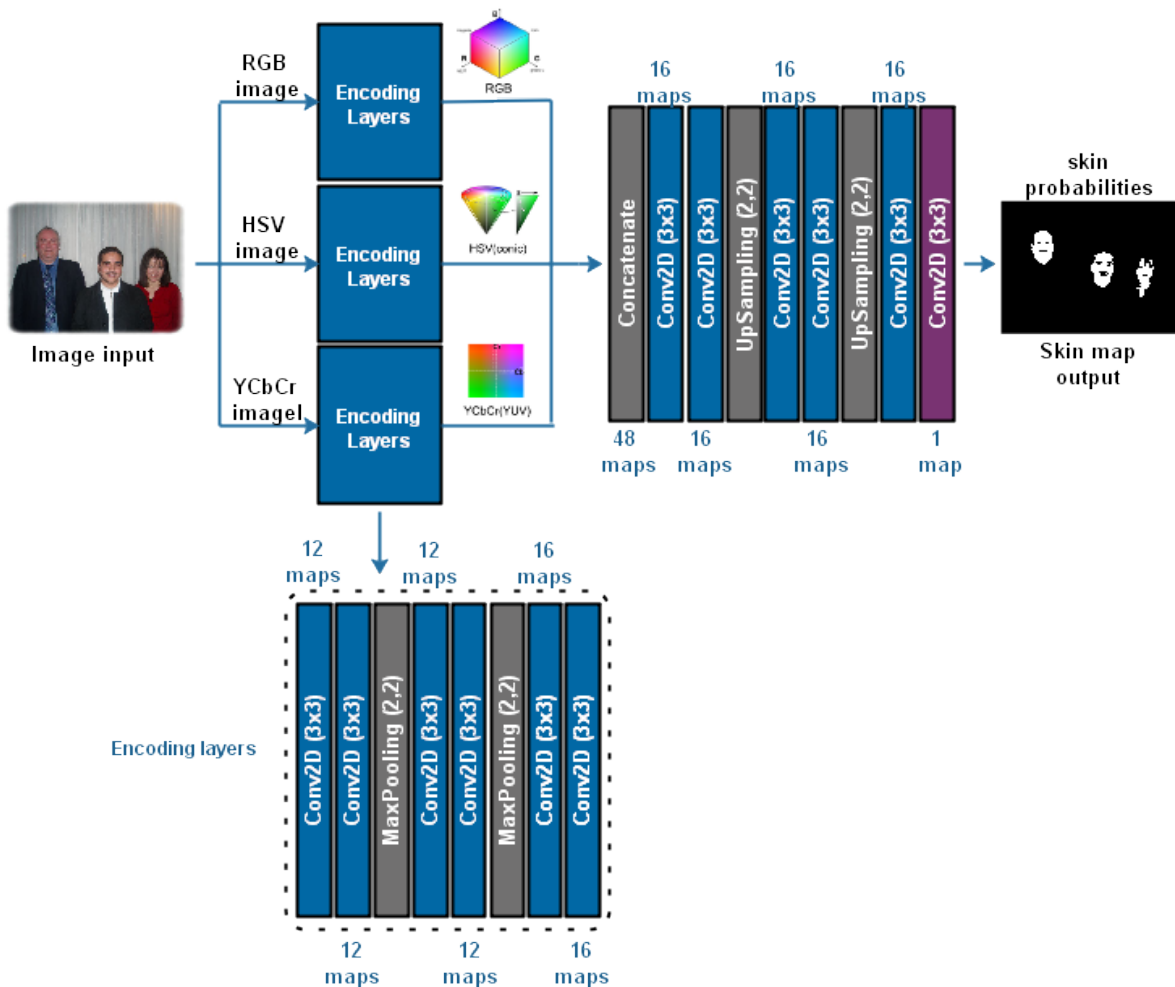


Figure 6.1: Model for skin detection in images. Layers employ ReLU (blue) and Sigmoid (purple) activations. On the bottom part of the Figure, the structure of *Encoding Layers* block is illustrated.

On the discriminative step of SALMORA, described in Section 4.2.2, a binary classifier was incorporated to the models using the Large-margin softmax loss [Liu and



Figure 6.2: Images and their corresponding skin maps from Pratheepan dataset [Tan et al., 2012].

Liu, 2016]. To compensate the class-imbalance for this classifier, since it separates instances from a single gesture class (SKIG contains 10 gesture classes, for example) from the remaining samples (and classes) of the dataset, a balance coefficient is computed. This coefficient is based on the occurrence of this gesture class in comparison to the remaining classes of the dataset, equalizing the contribution of the two classes considered by this binary classifier.

To train MORA and SALMORA models, the learning rate was experimentally set to 0.001. All convolutional layers of the presented architectures employ ReLU activation, except for the last ones shown in Figures 4.3 and 4.4, which employ sigmoid activation. The GRU layer uses a sigmoid activation for the output and a hard sigmoid for the recurrence.

6.1.2 Evaluating on the SKIG Dataset

The first dataset to evaluate MORA and SALMORA was the SKIG dataset. The initial step was the adjustment of SKIG videos to our models. As aforementioned, MORA and SALMORA models present a recurrent layer to deal with four different timesteps. For each timestep, 8-frame clips (RGB frames, Farneback Optical Flow and Depth) are used as inputs. In the case of SKIG, videos were uniformly sub-sampled to contain 32 frames and, consequently, four timesteps.

Since SKIG comprises ten gesture classes, it is necessary to train ten MORA autoencoder models, each one representing a class of the dataset and trained to minimize the reconstruction error for inputs of its respective class. On test, the model that presents the lowest reconstruction error, considering the multiple inputs and outputs, indicates the class. The first test conducted on SKIG was based on the employment of the original MORA models depicted in Figure 4.3, without hyperparameter adjustment with Hyperband. With this configuration, MORA obtained an average accuracy of 93.61%. After adjusting the hyperparameters and retraining the models, MORA obtained 93.80% as average accuracy. The small difference between these results can be associated with the fact that the original MORA models, depicted in Figure 4.3,

had their architecture assembled through experiments on validation subsets of SKIG and, consequently, it was expected that these configurations would achieve accurate performances on the dataset.

Table 6.2 shows the selected values from the search space used to determine hyperparameters for the SKIG dataset. Since MORA employs a different model for each class, hyperband had to be executed for each model, turning the setting of hyperparameters into a time-costly operation. This issue emphasizes the importance of a starting architecture, responsible for limiting the search space and reducing the time spent at this stage of the approach.

Table 6.2: Hyperband determined parameters for each model of SKIG dataset. *Despite the hyperband indication for MAE as the loss function for model 5, MSE was used to not create any discrepancy between the error computation for the reconstruction of this model and the remaining ones.

Model 1	Model 2	Model 3	Model 4	Model 5	Model 6	Model 7	Model 8	Model 9	Model 10
normal	uniform	normal	uniform	uniform	uniform	normal	uniform	uniform	uniform
RMSProp	Adagrad	RMSProp	Adam	Adagrad	RMSProp	RMSProp	RMSProp	Adam	RMSProp
1.0	1.5	1.0	1.5	1.5	0.75	1.0	1.0	1.5	1.0
1.0	1.0	1.0	1.5	1.0	1.5	1.5	1.0	1.0	0.75
16	16	9	16	4	16	16	9	4	9
1.0	1.5	1.5	1.0	1.0	1.0	1.0	1.5	1.0	1.0
Addition	Addition	Addition	Preserve	Removal	Addition	Preserve	Addition	Addition	Addition
MSE	MSE	MSE	MSE	MAE*	MSE	MSE	MSE	MSE	MSE

In a consecutive experiment, activations of the convolutional layer 10 (shown in Figure 4.3) were associated with a 15 hidden neurons multilayer perceptron classifier. The application of this classifier tends to enforce discriminative characteristics of learned representations. Since the autoencoder models are not discriminative, it is expected that outcomes from a classification model would surpass the ones purely obtained with reconstruction error. It is worth mentioning that this experiment claims for the application of a very simple and cheap-to-train classifier, with almost no impact on scalability. Differently from a deep classification model that aims at learning discriminative features over the input data, this simple classifier intends to separate data considering features already learned by autoencoder models, a much faster process. Thus, to correctly produce feature vectors representing videos without any bias, the activations of all ten models were concatenated to compose the vector for each video. Since the chosen layer employs a ReLU activation and models are trained to generate a response only for the class they are trained, the obtained vector is extremely descriptive and sparse. Then, the use of a simple classifier was enough to achieve an accuracy of 97.20% for original MORA models and 97.31% for models with hyperparameter adjustment.

Regarding SALMORA, the first conducted experiment considered the incorpora-

tion of the skin task to MORA models, leading to a performance improvement, with the approach achieving an average accuracy of 97.87%. This result can be justified by the higher focus given on the reconstruction of the subject performing the gesture, to the detriment of the background reconstruction. Since this background is common to several gestures, its reconstruction is not useful to properly characterize the dataset classes. Figure 6.3(a) shows a frame of a video of SKIG dataset, with MORA reconstruction without (b) and with (c) the skin task. Both models were trained for 25 epochs. A better reconstruction of the subject’s arm is noticeable after incorporating the skin task.

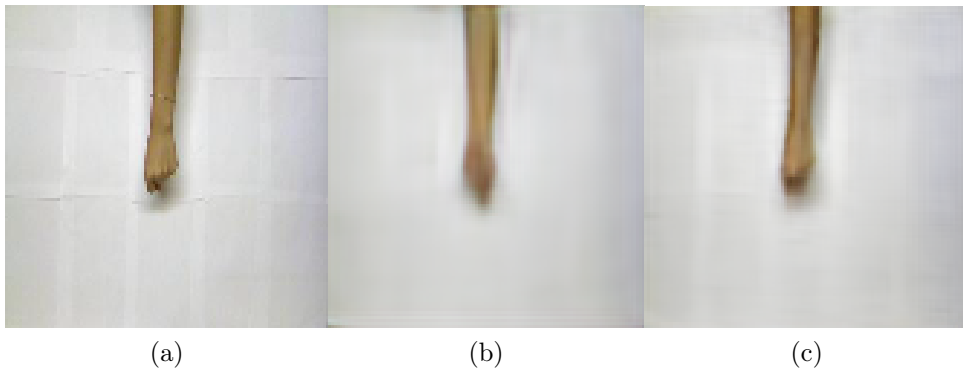


Figure 6.3: (a) Frame of a video of SKIG dataset. (b) MORA reconstruction without skin task. (c) MORA reconstruction with skin task.

At last, to complete SALMORA pipeline, weights are frozen and the binary classifier, showed in Figure 4.5, is attached to convolutional layer 10 of the models. The employment of this discriminator reduces the scalability of the approach, since it is trained with all gesture instances of the dataset for each MORA model. To reduce this negative impact on scalability, we trained this composition *MORA + discriminator* for only ten iterations. After that, the discriminator was removed, weights were unfrozen (except for convolutional layer 10) and the models were trained once again to minimize reconstruction error for their respective classes. As a result, the approach obtained 99.25% as average accuracy and, after training with the Large-margin Softmax loss ($m = 2$) [Liu and Liu, 2016], it reached 99.53%. This experiment suggests that performing this discriminative step on MORA’s pipeline increases the distance between the representation of autoencoder models, which helps to recognize gestures using the reconstruction error as criterion. In addition, tuning the weights of a single layer (a middle layer of the model) for few iterations is enough to enhance the outcomes of the approach. Tests associating more than one layer with the discriminator produced less accurate results, mostly due to the significant increment of the reconstruction er-

Table 6.3: Accuracies of different approaches applied to the SKIG dataset.

	Approach	Acc (%)
Results	RGGP+RGB-D [Liu and Shao, 2013]	88.70
	4DCOV [Cirujeda and Binefa, 2014]	93.80
	Depth Context [Liu and Liu, 2016]	95.37
	Tung and Ngoc. [Tung and Ngoc, 2014]	96.70
	D3D-LWDM. [Azad et al., 2019]	97.31
	MRNN [Nishida and Nakayama, 2016]	97.80
	3DCNN+RNN+CTC [Molchanov et al., 2016]	98.60
	Li et al. [Li et al., 2018b]	99.05
	Zhang et al. [Zhang et al., 2017]	99.53
	Imran and Raman. [Imran and Raman, 2019]	98.24
	Li et al. [Li et al., 2019]	100.00
Our Results	MORA reconstruction	93.61
	Hyperband MORA reconstruction	93.80
	MORA activations + 15N MLP	97.20
	Hyperband MORA activations + 15N MLP	97.31
	MORA reconstruction + skin	97.87
	SALMORA (custom loss)	99.25
	SALMORA (large-margin)	99.53

ror for all models. Figure 6.4 depicts the disposition of MORA representations for samples of different classes of SKIG dataset, considering a model trained for class 1. These representations were obtained through the application of *PCA* (2 components) on activations of the last convolutional layer before the outputs of the models showed in Figures 4.3 and 4.4. A very similar behavior was noticed on models representing the remaining classes. In Figure 6.4, the representations for some samples of different classes were close to each other and even intersecting for MORA without the discriminative step (a). With this step, the representations are spread and intersection zones were reduced (b). At last, with the employment of a large-margin loss, no intersection can be perceived (c).

Table 6.3 shows the results achieved by several methods on the SKIG dataset. MORA reaches high accuracies, being comparable to state-of-the-art methods, with SALMORA matching the approach of Zhang et al. [2017], the second best reported on the database. It is important to mention that all results presented on Table 6.3 are based on the same evaluation protocol, making their outcomes fairly comparable.

Several approaches reach accuracies next to 99% on SKIG, evidencing the saturation of the database. However, it is important to evaluate MORA and SALMORA on this dataset to check some points. First, since SKIG is a 10-class balanced dataset, the advantages of MORA are not that evident. It would be interesting to check the

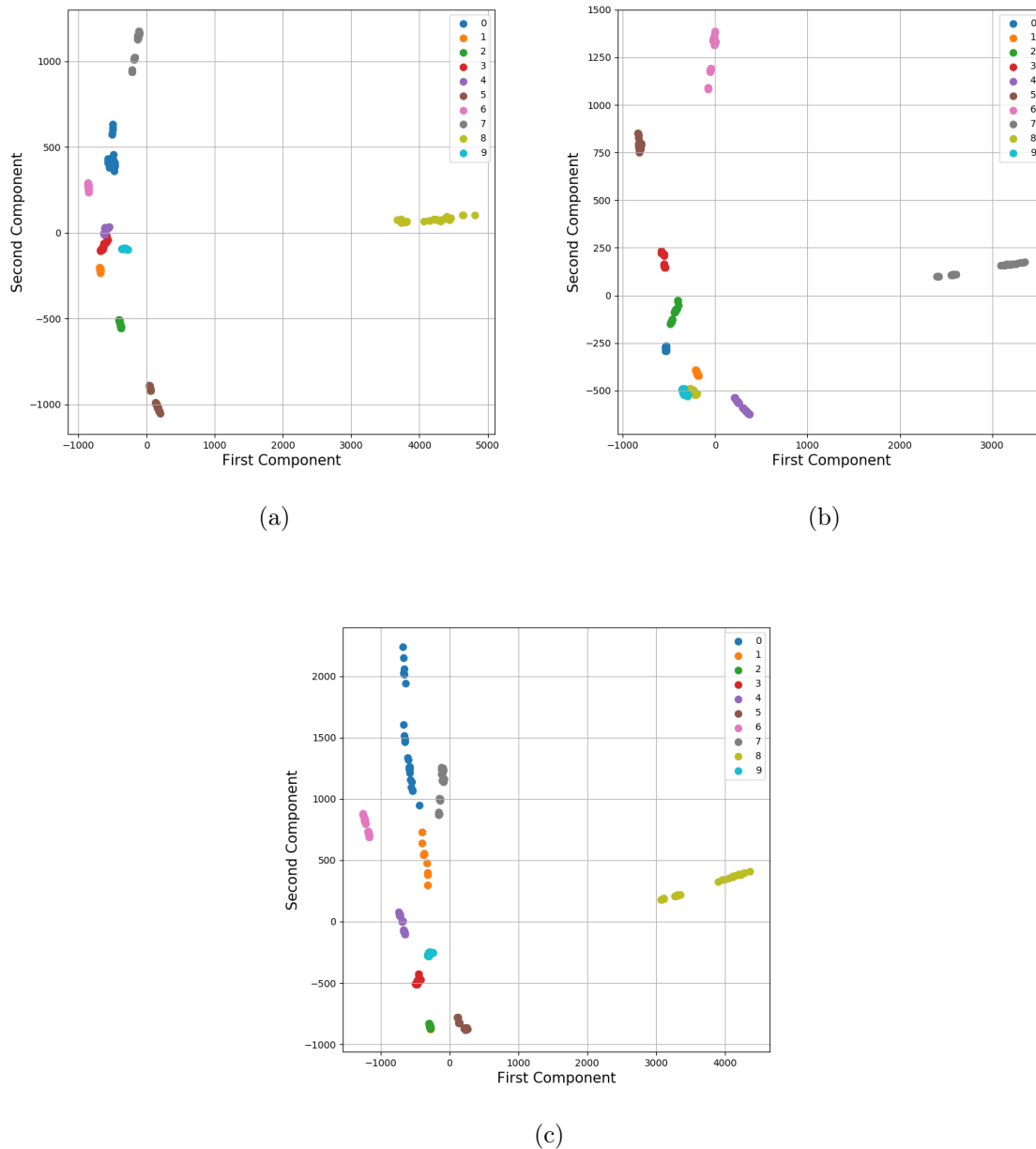


Figure 6.4: (a) Disposition of features from MORA models. (b) Disposition of features from MORA models after using a discriminator with custom binary-crossentropy loss. (c) Disposition of features from MORA models after using a discriminator with Large-margin softmax loss ($m=2$).

results obtained by the method in an unfavorable scenario. However, despite being surpassed by other approaches, when the complexity of models is considered besides the consequences of training a huge number of parameters, MORA shows to be valuable even when considering the outcomes obtained with only the reconstruction of

inputs. In addition, MORA is scalable in terms of number of classes, an advantage in this context. The approaches that surpassed MORA in terms of accuracy present a much higher complexity/capacity and sophisticated fusion techniques to obtain these outcomes. Second, incorporating an attention mechanism based on skin led to a huge improvement on MORA results, with its accuracy being superior to the activations associated to a MLP classifier. Third, enhancing a discriminative behavior in a single layer promoted a huge impact on the recognition of the approach, boosted by the employment of a large-margin loss. Even deteriorating the scalability aspect of the approach, a huge improvement was obtained with few extra training iterations. Fourth, furthermore, since the employed protocol for SKIG dataset does not demand the computation of dispersion metrics and/or significance tests, it is difficult to assert the superiority of the state-of-the-art method, proposed by Li et al. [2019], over SALMORA.

6.1.3 Evaluating on ChaLearn IsoGD

To evaluate MORA on ChaLearn IsoGD [Wan et al., 2016], it was necessary to adjust the dataset videos to be suitable to the models. Similarly to SKIG, ChaLearn IsoGD videos were sub-sampled and divided into four timesteps. Since ChaLearn videos are very short when compared to SKIG ones (some videos present only 13 frames), the sub-sampling produced 12-frame videos and each timestep is related to a 3-frame clip.

The experiments on ChaLearn IsoGD dataset intend to evaluate MORA’s behavior on a large class-unbalanced scenario. Thus, it is possible to analyze whether the reconstruction error of recurrent non-discriminative models provides enough information to separate a large number of classes, besides the robustness of MORA to class imbalance. In addition, we could analyze the improvement of accuracy resultant from the application of discriminative steps on MORA pipeline, with the employment of a classifier associated with activations of the autoencoders; and with SALMORA discriminative stage.

The first experiment conducted on ChaLearn IsoGD is related to the training and testing of 249 autoencoders, one for each dataset class. In this experiment, a 56.37% average accuracy was obtained. After that, hyperband was used to compute the best set of hyperparameters considering the search space showed in Table 6.1. Executing Hyperband is a time-costly operation and, performing it for all 249 classes of ChaLearn IsoGD would be very expensive in terms of time and machine processing. Instead of performing Hyperband for every model of the dataset, we randomly selected 20 models (classes) and computed the most voted set of hyperparameters, applying this set to all

models of the dataset. The most voted set of hyperparameters is showed in Table 6.4. The accuracy of MORA was increased to 57.01%.

Table 6.4: Most voted set of hyperparameters with Hyperband for ChaLearn IsoGD.

	Most voted
Weight Initializer	uniform
Optimizer	RMSProp
Enc. Factor	1.0
Dec. Factor	1.0
Batch Size	16
Rec. Layer Factor	1.5
Layer Growth	Addition
Loss	MSE

Following the experiments conducted on SKIG, we associated the activations of *convolutional layer 10* with a cheap 50-neurons Multilayer Perceptron classifier. In addition, we performed the steps of SALMORA, with the incorporation of skin task and employment of a classifier to tune weights of the model. Table 6.5 shows MORA outcomes and reference methods on the dataset. MORA accuracies are comparable to state-of-the-art methods, with SALMORA being surpassed only by the FOANet method proposed by Narayana et al. [2018]. It is important to mention that FOANet employs a different set of inputs, providing to their model information from RGB, optical flow and depth, besides the pose estimation, computed with OpenPose [Cao et al., 2017], and coordinates of the hands of the subject performing the gesture.

The SALMORA results showed in Table 6.5 are based on the employment of a different classifier to tune the weights of each of the 249 autoencoder models trained for ChaLearn IsoGD. Despite the accurate outcomes, this operation is expensive, since every classifier needs to handle the whole dataset to separate each class from the remaining ones on ChaLearn IsoGD. Consequently, even considering few iterations (ten) for this classifier, the time-to-train scalability appeal of MORA is crumbled.

To reduce the impact on scalability, we conducted additional experiments considering the training of the classifier with less classes of the dataset. On this experiment, the most confused classes on the validation subset of ChaLearn IsoGD were selected and a classifier was associated only with the models related to these classes. Two training modalities of this classifier were employed: (i) differentiating the selected classes to increase only the distance between them, i.e., if ten classes are selected, the classifier intends to discriminate between these ten classes (*intra* modality); and (ii) the classifier intends to separate selected classes from all remaining ones of the dataset (*inter* modality). After training the classifier and tuning the weights of selected models, all

Table 6.5: Accuracies on the test subset of the ChaLearn IsoGD dataset.

	Approach	Acc (%)
Results	Pyramidal C3D. [Zhu et al., 2016]	50.93
	MMFTSN [Zheng et al., 2019]	60.02
	Zhang et al. [Zhang et al., 2017]	62.14
	GL-PAM. [Li et al., 2018a]	67.02
	2SCVN-3DDSN. [Duan et al., 2016]	67.26
	C3D+Spt. Attention. [Li et al., 2019]	68.14
	FOANet [Narayana et al., 2018]	82.07
Our Results	MORA reconstruction	56.37
	Hyperband MORA reconstruction	57.01
	MORA reconstruction + skin	57.12
	MORA activations + 50N MLP	66.16
	Hyperband MORA activations + 50N MLP	66.28
	SALMORA (custom loss)	68.96
	SALMORA (large-margin)	69.44

Table 6.6: SALMORA recognition accuracy considering a variable amount of selected classes associated to the classifier. These experiments considered the Large-Margin Softmax Loss [Liu and Liu, 2016] and *inter* and *intra* modalities.

Classifier Epochs	10	30	50	Classifier Epochs	10	30	50
10 Classes (inter)	68.89	68.71	68.90	10 Classes (intra)	68.68	68.96	68.83
30 Classes (inter)	69.14	69.14	69.18	30 Classes (intra)	68.84	68.93	69.12
60 Classes (inter)	69.29	69.30	69.34	60 Classes (intra)	69.14	69.16	69.21
120 Classes (inter)	69.38	69.38	69.36	120 Classes (intra)	69.27	69.30	69.31
180 Classes (inter)	69.38	69.42	69.41	180 Classes (intra)	69.30	69.30	69.28
249 Classes (inter)	69.44	69.36	69.44	249 Classes (intra)	69.44	69.36	69.44

249 the models were trained again to minimize the reconstruction error. Table 6.6 shows the outcomes of this experiment and also presents the recognition accuracy for a higher number of iterations on the training of the classifiers. The impact of the number of iterations and even of the number of selected classes is not that relevant, making it possible to use a low number of selected classes, obtain accurate outcomes and preserve most of the time-to-train scalability of the approach.

6.1.4 Evaluating Time-to-Train Scalability on ChaLearn IsoGD

Finally, we performed an experiment to evaluate MORA’s time-to-train scalability as a function of the number of gestures. This experiment consists in the application of MORA (original architecture depicted in Figure 4.3) on 50 randomly selected classes from the ChaLearn IsoGD dataset. The validation subset of ChaLearn was consid-

ered and an increasing number of gestures was employed, with their accuracies being shown on Figure 6.5. In addition, the accuracy of a widely employed gesture/activity recognition classification model (C3D [Tran et al., 2015]) is also shown.

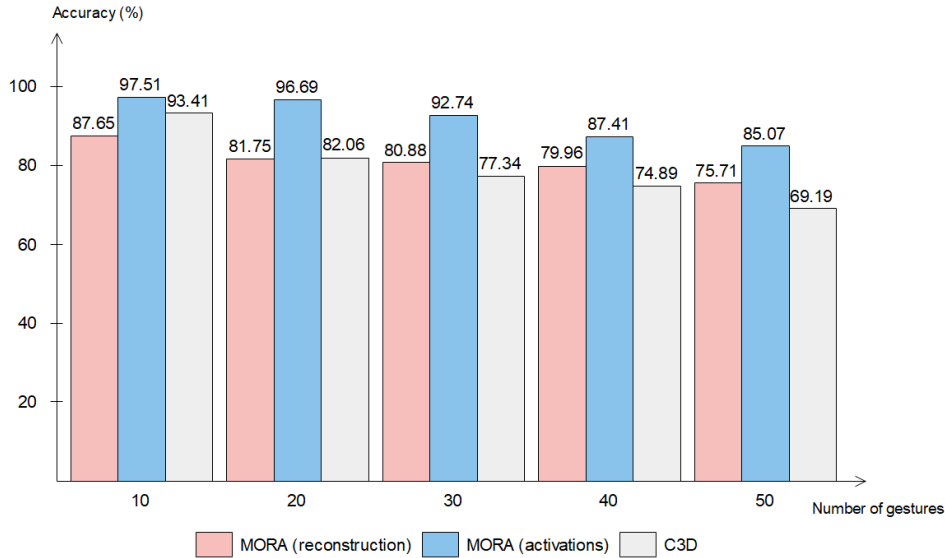


Figure 6.5: Accuracy on 50 classes of Chalearn.

According to the results, the accuracy drop obtained with MORA outcomes shows a soft behavior. In contrast, the C3D model shows a high accuracy for the first test (ten gestures), but much lower values on the subsequent ones. This behavior can be associated with the following aspects: (i) C3D model is a classification (discriminative) network and it is highly impacted by class imbalance, which is more evident when a higher number of classes is considered; (ii) differently from MORA, C3D is a fixed-capacity model. Thus, the same architecture was employed for all tests, with no variation on the number of parameters, while MORA’s capacity scales according to the number of classes. It is important to notice that MORA with activations employs a custom Multilayer Perceptron classifier, which is susceptible to class imbalance. However, since the features used by this classifier were learned in an unsupervised way, the impact of class imbalance is reduced.

Table 6.7 lists a set of parameters from MORA and C3D models, obtained with tests on a NVIDIA GeForce GTX 1060M. According to the table, MORA employs a variable number of parameters depending on the task and presents a low time requirement regarding the addition of a new class (i.e., new gesture). In addition, even presenting higher accuracy outcomes than C3D, MORA is, for most cases, a much less complex model and presents a lower time per iteration for training. Even though MORA’s complexity is similar to C3D for the classification of 50 gestures, the advan-

Table 6.7: Comparison between MORA and a discriminative model (C3D) for tests on ChaLearn IsoGD.

	MORA	C3D
Number of Parameters	2.7M per model	111M ¹
Average Time per Epoch (minutes)	4.89 ²	7.83
Average Time to insert new class (minutes)	75	1263 ³
Time to Train Custom Classifier (minutes)	8 (10 gestures)	Not required
	13 (20 gestures)	
	15 (30 gestures)	
	20 (40 gestures)	
	24 (50 gestures)	

tage of the present approach relies on incremental growth on number of parameters, which improves the capacity of MORA depending on the number of classes. Finally, the cost of inserting a discriminative classifier is also presented, evidencing the low impact on MORA’s scalability, since the cost to train this network is very low.

6.2 MLRRN evaluation

To evaluate MLRRN, experiments are conducted on ChaLearn Montalbano [Escalera et al., 2014] and on ChaLearn CongGD [Wang et al., 2016] datasets. For both, detection and recognition of gestures are performed following the standard evaluation protocols.

6.2.1 Experimental Setup

Most parameters of the MLRRN architecture (illustrated in Figure 4.10), such as the choice for a bidirectional LSTM layer, the employment of residual modules and activation function of layers, were determined by tests on the validation set of the ChaLearn Montalbano [Escalera et al., 2014]. However, since ChaLearn CongGD [Wan et al., 2016] is a more complex dataset, containing more videos and gesture classes than SKIG, the architecture depicted in Figure 4.10 was adjusted before conducting experiments on it, with the insertion of one extra residual block and the increment of the number of feature maps in some layers. In addition, the output of the softmax layer, responsible for the gesture recognition, had its size adjusted to contemplate the classes of ChaLearn CongGD and the no-gesture class.

¹This value is adjusted to receive ChaLearn videos as inputs.

²This average value represents the time to train 50 MORA models.

³Even with a similar time per epoch, C3D requires more iterations since it contains much more parameters to be trained.

MLRRN also had its hyperparameters adjusted by hyperband algorithm [Li et al., 2016, 2017]. The initial architecture, manually assembled and depicted in Figure 4.10(a), was used as a starting point, from which we executed hyperband ($max - iter = 18$ and $eta = 3$) to adjust the hyperparameters. The hyperband search space is shown in Table 6.8, with $v1$, $v2$ and $v3$ representing the possible values a hyperparameter can assume.

Table 6.8: Hyperband search space for MLRRN approach. Conv. Factor represents an adjustment on the number of filters on convolutional layers, while Rec. Layer Factor stands for an adjustment of units on recurrent layer. ELU layer factor represents an adjustment on the number of units on ELU layers.

	v1	v2	v3
Weight Initializer	normal	uniform	glorot normal
Optimizer	SGD	Adam	Adagrad
Batch Size	32	50	64
Conv. Factor	0.75	1.00	1.50
Rec. Layer Factor	0.75	1.00	1.50
ELU Layer Factor	0.75	1.00	1.50

To train the model, the learning rate was experimentally set to 0.0001, considering the validation subset of ChaLearn Montalbano. All convolutional layers of the model depicted in Figure 4.3 employ ReLU activation (shown in blue), except for some on residual blocks (shown in green), which employ ELU. LSTM and fully-connected layers employ sigmoid activation (shown in purple). The initial model evaluated on ChaLearn Montalbano contains 53.1Mi parameters while the one evaluated on ChaLearn ConGD contains 60.2Mi parameters, both trained on a NVIDIA GeForce 1080Ti.

As aforementioned, the evaluation of MLRRN considered the average temporal Jaccard metric, which takes into account both the accuracy of the network responses and the overlap between the responses and ground-truth annotations.

6.2.2 Ablation Study of MLRRN

Since MLRRN is composed of several components, an ablation study shows to be valuable. Table 6.9 presents results obtained with the ablation evaluation of MLRRN on ChaLearn Montalbano for recognition and detection tasks. The results obtained with this study justified our choices on the assembling of the final architecture, evidencing the contribution of the different modules incorporated in MLRRN. From this study, some points need to be highlighted, as: (i) the huge impact of recurrent layers, indicating the importance of temporal information and disposition of events for gesture

recognition, (ii) the complementarity of appearance and skeleton (joints) inputs and (iii) detection and recognition losses, with improvements obtained from the combination of them. Detection outcomes do not correspond to the final response of the approach. Presenting them intends to highlight the improvement, even for this complementary task, obtained with our multi-task strategy. A deeper evaluation of the method is presented in the following sections.

Table 6.9: Ablation study performed on ChaLearn Montalbano dataset.

	Approach variation	Jaccard Score (recognition)
Results	Only appearance input	0.830
	Only skeleton input	0.659
	No residual blocks (skip-connections removed)	0.881
	No recurrent layers	0.547
	Single-directional recurrent layers	0.861
	Only recognition task (no detection)	0.806
	<i>Full model</i>	0.919
	Approach variation	Jaccard Score (detection)
Results	Only detection task (no recognition)	0.949
	<i>Full model</i>	0.982

6.2.3 Evaluating on ChaLearn Montalbano

MLRRN is an approach that relies on frame-level inputs. Consequently, it is not necessary to perform any subsampling. For each frame, an RGB input tensor is assembled along with the joint response of the technique of Cao et al. [2017]. In addition, once MLRRN presents a bidirectional recurrent layer, it takes into account previous and future frames to produce a response for every frame of the input. An important point on this approach is the batch size, since it must be large enough to provide information that reflects long-term dependency that exists between frames. However, the larger this batch size, the higher must be the number of parameters of this recurrent layer, leading to problems such as higher time to train and training data requirement, proclivity to overfitting and struggling convergence. On ChaLearn Montalbano, a batch size of 50 was initially used.

Since ChaLearn Montalbano comprises 20 gesture classes and one no-gesture class, it was necessary to train a 21-class classification model for the recognition task. For detection, this model acts as a binary classifier, outputting labels that indicate whether a frame is part of a gesture or not. Based on that, we evaluated our approach on the test subset of ChaLearn Montalbano, which provided frame-level recognition accuracy of 96.87%. This accurate result led to a high average temporal Jaccard re-

sponse, with the approach achieving a value of 0.914 in terms of the output of the recognition task of the model. With the application of hyperband regarding the parameters showed in Table 6.10, the approach presented a lower recognition accuracy and a lower average temporal Jaccard, with 96.12% and 0.896, respectively.

Table 6.10: Selected search parameters obtained with Hyperband for ChaLearn Montalbano.

	Selected Parameter
Weight Initializer	uniform
Optimizer	Adagrad
Batch Size	50
Conv. Factor	0.75
Rec. Layer Factor	1.00
ELU Layer Factor	1.00

Besides the results shown, we also executed a post-processing on MLRRN recognition output, obtained with the original model (no hyperband) illustrated in Figure 4.10. On that, we performed a majority voting around each frame response, using masks with different sizes (encompassing different number of frames) and making the label of each frame to be the most common response of the own frame and its neighbors. The employment of this post-processing stems from the frame-level output of MLRRN, which makes the approach sensible to wrongly recognized frames, as depicted in Figure 6.6, where colors indicate class label of a frame. One could notice incorrect responses (dark blue) among the frames of a gesture (represented in orange).

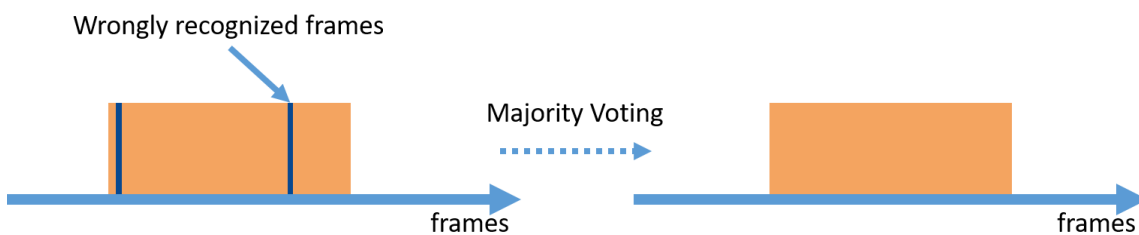


Figure 6.6: Post processing on MLRRN response. Colors indicate the label of the frame. The orange box indicates the duration, in terms of frames, of a gesture related to a specific class (orange). The dark blue lines indicate the existence of wrongly recognized frames during the recognition of the orange class.

Table 6.11 shows the results of the proposed MLRRN and state-of-the-art approaches on ChaLearn Montalbano [Escalera et al., 2014], considering the conventional and post-processed outputs with different mask sizes. According to the results, it is possible to see that larger masks tend to degrade the Jaccard response of the model,

Table 6.11: Average temporal Jaccard score on ChaLearn Montalbano dataset.

	Approach	Jaccard Score
Results	MRF, KK, PCA, HOG [Chang, 2015]	0.827
	AdaBoost, HOG [Monnier et al., 2015]	0.834
	Multi-scale DNN [Neverova et al., 2016]	0.870
	TempConv + LSTM [Pigou et al., 2017]	0.906
	PM-EGD [Gupta et al., 2019]	0.910
	3DCNN + ConvLSTM [Zhu et al., 2018]	0.915
Our Results	Hyperband MLRNN	0.896
	MLRNN	0.914
	MLRRN + 3-size mask	0.916
	MLRRN + 5-size mask	0.919
	MLRRN + 7-size mask	0.912
	MLRRN + 9-size mask	0.908

since transition zones (between different gestures and gesture to non-gesture frames) become corrupted. With a 5-frame mask, we achieved state-of-the-art results, surpassing the method proposed by Zhu et al. [2018]. It is important to mention that the research proposed by Molchanov et al. [2016] obtained a higher average Jaccard score on this dataset. However, since this approach uses ground-truth annotations to perform gesture detection, their outcomes cannot be compared to ours.

To better understand the performance of MLRRN, we evaluated the recognition outcome of the approach considering different Jaccard values, as showed in Figure 6.7. For Jaccard values below 0.65, MLRRN executes a perfect recognition on ChaLearn Montalbano. In addition, even for high values such as 0.90, the approach presents accurate results.

6.2.4 Evaluating on ChaLearn ConGD

The evaluation of MLRRN on ChaLearn ConGD presented few changes in comparison to ChaLearn Montalbano. Most of these changes are related to the increment of the model capacity to be able to handle a more complex dataset and the increase on the number of classes for classification, which goes to 250. In addition, since ChaLearn ConGD is composed of shorter videos when compared to SKIG, the batch size was experimentally set to 40. Finally, hyperband was executed to estimate hyperparameters of the model for this dataset, resulting in the selected search space parameters showed in Table 6.12.

Since ChaLearn ConGD presents a 1-frame distance between different gestures, the impact of the detection task was extremely mitigated due to the absence of no-

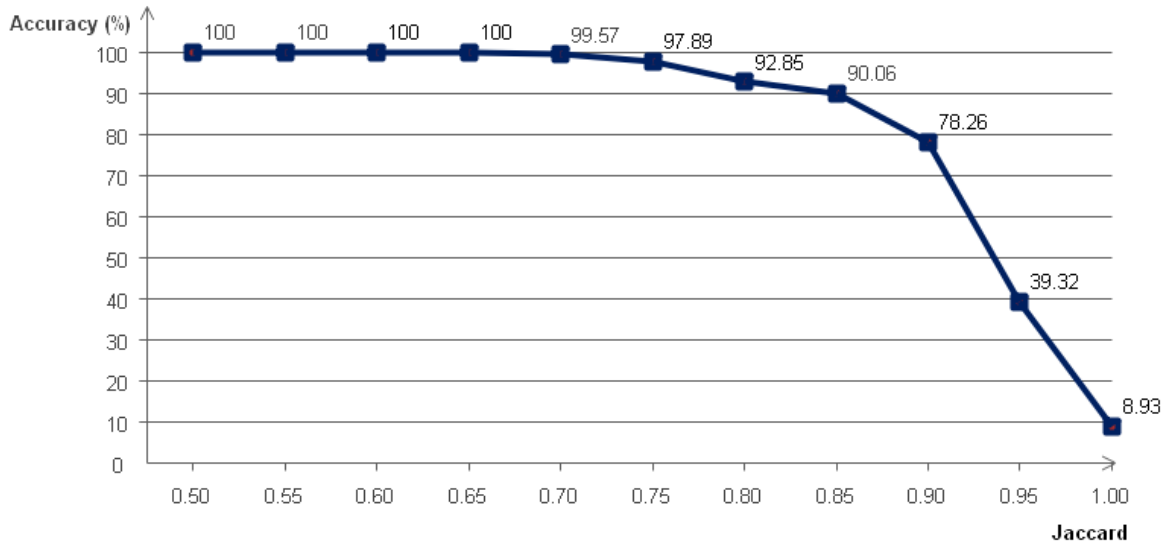


Figure 6.7: MLRRN recognition outcomes considering different Jaccard scores.

Table 6.12: Selected search parameters obtained with Hyperband for ChaLearn ConGD.

	Selected Parameter
Weight Initializer	uniform
Optimizer	Adagrad
Batch Size	32
Conv. Factor	1.00
Rec. Layer Factor	1.00
ELU Layer Factor	1.50

gesture frames. As a result, on ChaLearn ConGD, the approach obtained a frame-level recognition accuracy of 73.23% and a temporal Jaccard score of 0.5627. With the hyperparameter adjustment, MLRRN performance was boosted, obtaining frame-level recognition accuracy of 76.06% and temporal Jaccard score of 0.5692. Table 6.13 shows MLRRN results along with state-of-the-art approaches on the dataset. It is important to notice that we performed an additional evaluation on this dataset to enhance the outcomes of MLRRN, in which we assigned different labels for classes on the detection task, creating groups on the dataset. In a first test, for example, five groups were created. Thus, gestures from classes 0-49 are assigned to label 1, classes 50-99 to label 2, and so on. This strategy aimed to improve outcomes from the detection task, as it could now determine intervals related to gestures from different groups, gathering some information about their length and transitions that exist between them. As a consequence, a significant improvement on temporal Jaccard score was noticed, evidencing the impact of detection task for the proper recognition of gestures.

Table 6.13: Average temporal Jaccard score on ChaLearn ConGD dataset.

	Approach	Jaccard Score
Results	Two-Stream ConvNets + Ensemble learning [Wang et al., 2017b]	0.5307
	Faster-RCNN + Heterogeneous networks [Wang et al., 2017a]	0.5950
	3D Finger-Joints [Hoang et al., 2019]	0.5523
	Faster RCNN+ C3D [Liu et al., 2017]	0.6103
	TS1-Res3D + Multiply Fusion [Zhu et al., 2018]	0.6435
	TS1-Res3D + Average Fusion Fusion [Zhu et al., 2018]	0.7163
	STF [Narayana et al., 2019]	0.7740
Our Results	MLRRN	0.5627
	Hyperband MLRRN	0.5692
	MLRRN + 5-detection classes	0.6204
	MLRRN + 5-detection classes + 5-mask	0.6231
	Hyperband MLRRN + 5-detection classes + 5-mask	0.6287
	MLRRN + 10-detection classes + 5-mask	0.6217
	MLRRN + 20-detection classes + 5-mask	0.6083

On the ChaLearn ConGD, there are similar problems to the ones found on ChaLearn Montalbano, such as the existence of some noise between frame outputs of a class. Besides that, since the annotation of this dataset does not include no-gesture frames, the performance of MLRRN is deteriorated, with our model acting in a similar way to a standard classifier, getting few contribution from the detection task. However, with the assignment of class groups for detection, the impact of this multi-task was greatly enhanced, even with this separation being performed with a very simple criterion.

As for ChaLearn Montalbano, we evaluated the recognition outcome of the approach on ChaLearn ConGD considering different Jaccard values, as showed in Figure 6.8. The performance presented in Figure 6.8 regards the outcome obtained with 5-detection classes and a 5-mask for majority voting.

6.2.5 Qualitative Evaluation of MLRRN Tasks

Results on ChaLearn Montalbano and ChaLearn ConGD evidenced the high performance of MLRRN and the positive impact of the employment of correlated tasks. The superiority of the approach on ChaLearn Montalbano, for which we achieved state-of-the-art performance, is greatly related to the gesture disposition of this dataset and their annotations, which provide no-gesture frame intervals between the gesture instances. On ChaLearn ConGD, the performance of MLRRN is mitigated mostly due to annotations. On ChaLearn ConGD, frames associated with relaxing postures of performers that should be annotated as non-gesture frames, are still annotated as part of gesture intervals.

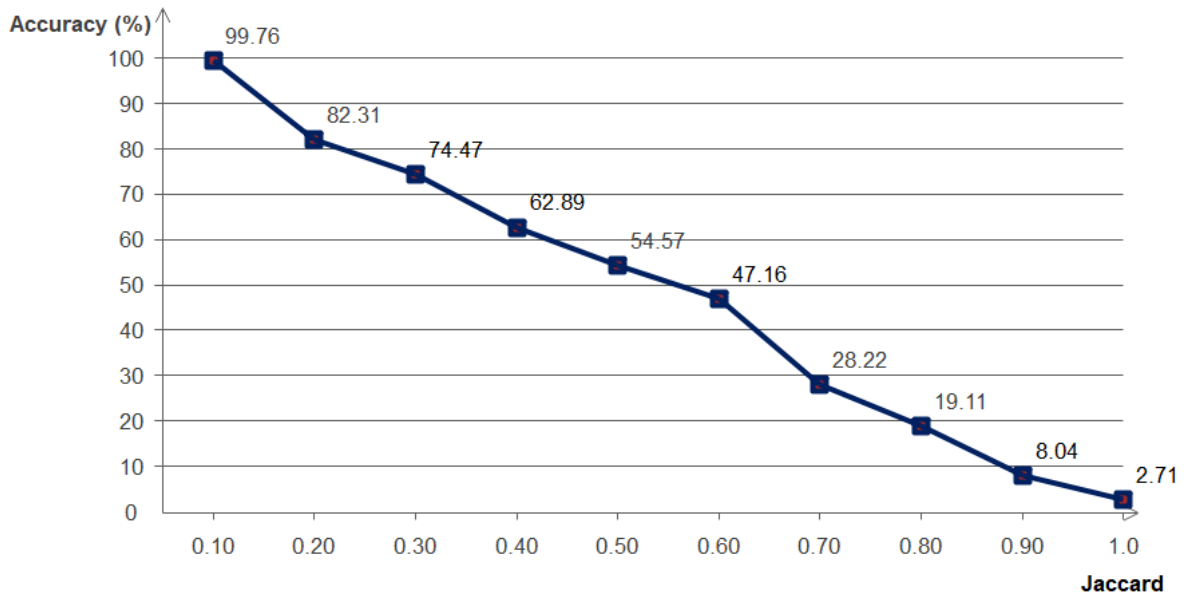


Figure 6.8: MLRRN recognition outcomes considering different Jaccard scores on ChaLearn ConGD.

In addition to the quantitative results showed, it is interesting to notice, in a qualitative way, how the method performed for both tasks. Figure 6.9 depicts the response of the temporal detection task for a video of ChaLearn Montalbano compared to the ground-truth response. In this figure, low-responses (next to 0) indicate the absence of gesture and high-responses (next to 1) indicate the presence of a gesture. One could notice the existence of some noise on the detection, which is filtered by the employment of an empirical threshold (0.5) used to approximate the responses to 0 or 1. The abrupt transition between non-gesture and gesture frames is well-detected by the model in most cases in this dataset.

The detection and recognition tasks produced accurate outcomes for most videos in the datasets. However, the last frames of some gestures were predicted as no-class frames, what produced a shortening effect on recognition. This issue can be associated with the similarity, in terms of appearance and motion, between frames of retraction phase (final part of a gesture) and relaxing phase (post gesture), with the latter corresponding to no-class frames. In addition, the lack of no-class standard postures and inconsistent behavior of performers could have contributed to this result, illustrated in Figure 6.10.

On the evaluation of the proposed MLRRN, even on ChaLearn Montalbano for which the approach presented very accurate results, some frames belonging to all classes of the dataset are recognized as class-0 (i.e., no-gesture class). This result is mostly perceived on the retraction frames of each gesture, since these frames are similar, in

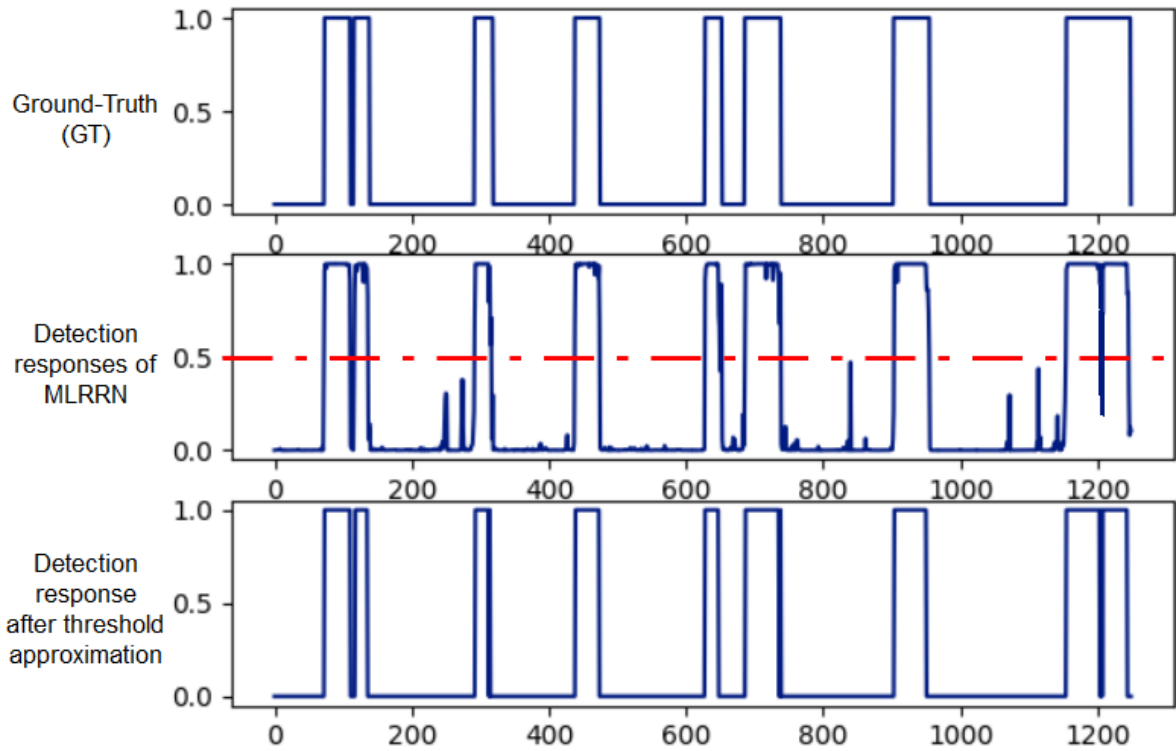


Figure 6.9: Detection responses of MLRRN. Dashed red line represents the threshold used to approximate responses.

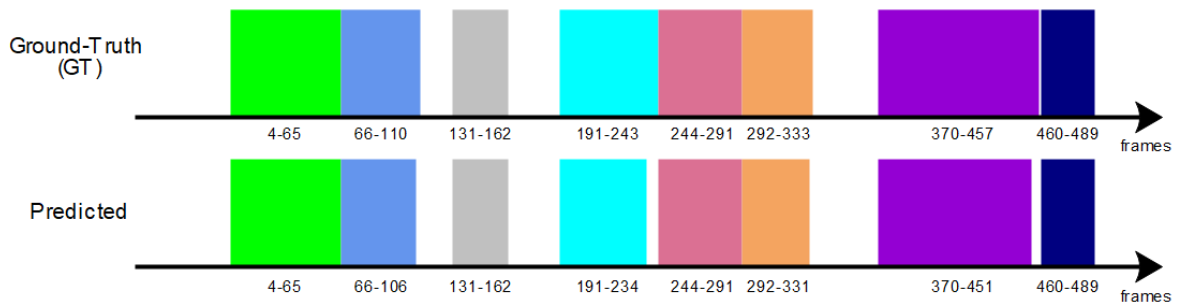


Figure 6.10: Shortening effect on the recognition of gestures by MLRRN.

terms of appearance and even motion, to the relaxing postures that are common on post-gesture frames. Figure 6.11 depicts the frame-level confusion matrix of MLRRN responses on ChaLearn Montalbano, for which the approach presented more than 96% frame-level accuracy. It is possible to see that for almost all classes, some frames are predicted as non-gesture (class 0).

Finally, we performed a cross-dataset test, in which we used a model trained on ChaLearn Montalbano to act over videos of ChaLearn ConGD. Since the class-recognition labels are not useful in this scenario, we only qualitatively verified whether the detection task was able to produce reasonable results. Figure 6.12 depicts the

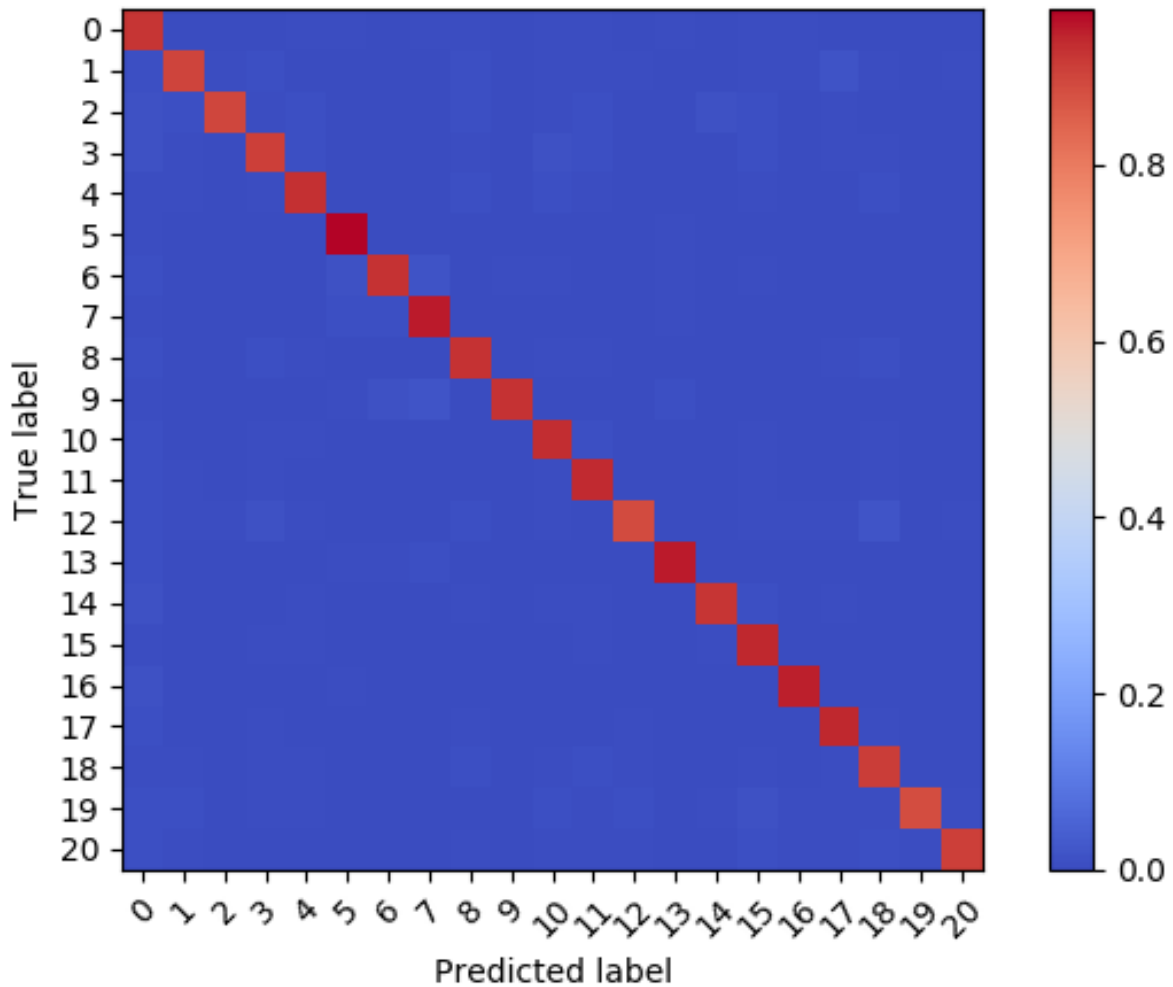


Figure 6.11: Frame-level confusion matrix of MLRRN on ChaLearn Montalbano. Indexes represent different dataset classes. Class-0 represents non-gesture class.

detection task on videos of ChaLearn ConGD. In this test, we manually annotated the preparation, nucleus and relaxing phases of gestures. According to the results, the model presents high responses for the nucleus part of the gestures, with oscillating responses on preparation and retraction and low responses on the relaxing postures, annotated on ChaLearn Montalbano as the non-gesture class. Even though not completely accurate, the results suggest that the trained model is able to indicate the separation between gestures in a different dataset, which is promising once that experiment emulates conditions similar to real-life scenarios.

Since the model trained on ChaLearn ConGD presents no impact on the detection task due to the lack of annotations associated with the no-gesture frames, the evaluation of detection task is not reasonable on the ChaLearn Montalbano dataset.

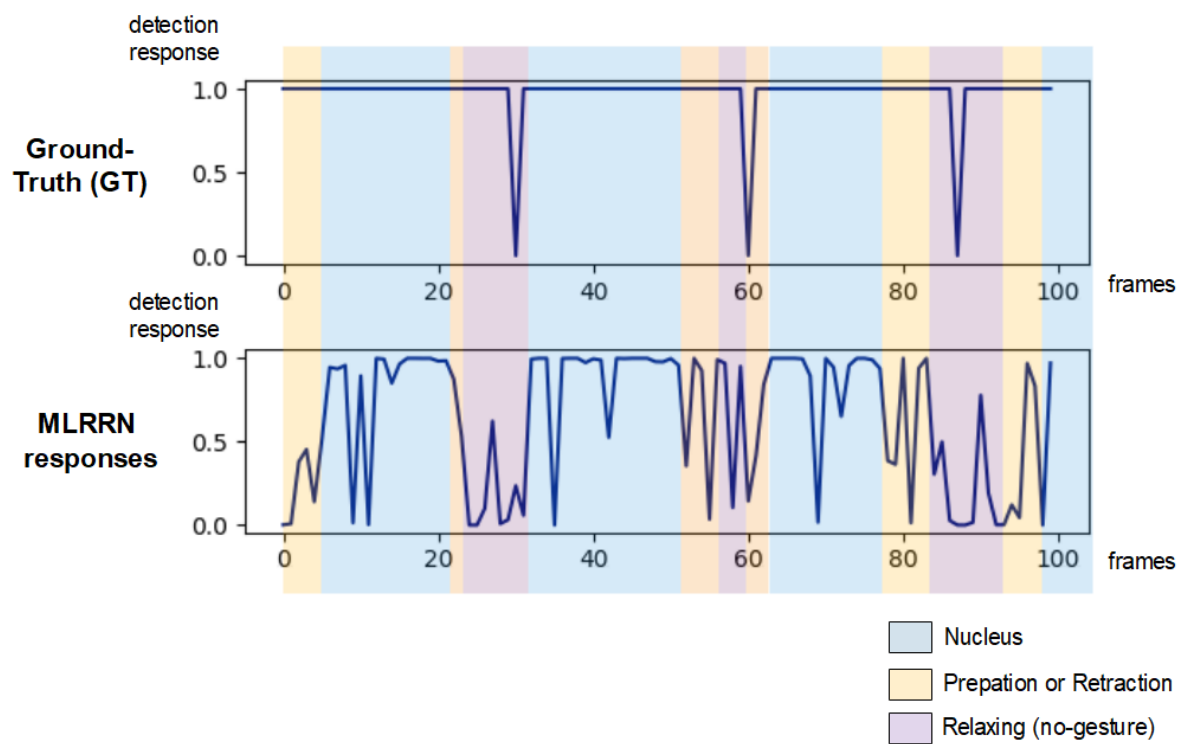


Figure 6.12: Detection responses of MLRRN on ChaLearn ConGD.

Chapter 7

Conclusions

In this chapter, we present conclusions related to the present work. Specifically, we discuss MORA, SALMORA and MLRRN approaches, presenting final considerations about the methods regarding their advantages and drawbacks (Section 7.1). In addition, we discuss some future work that have potential to enhance the outcomes obtained with the proposed methods (Section 7.2).

7.1 Final considerations

In this dissertation, we address the gesture recognition task aiming at filling gaps noticed on literature approaches, such as scalability (in terms of the number of gestures), time cost to incorporate new gestures; and actuation over unsegmented videos. To accomplish that, we proposed two approaches that tackle the problems. The first (MORA) employs a set of unsupervised models to represent each gesture class in an independent way, using reconstruction error as classification metric, focusing on scalability as a function of number of gestures and minimizing the time-to-train requirement of the approach to consider a new gesture. The second approach (MLRRN) focus on real-life communication scenarios, performing gesture recognition on a continuous stream of data. To properly recognize gestures, MLRRN also performs gesture temporal detection, exploring the correlation that exists between both tasks to enhance the results.

MORA evaluation led to accurate gesture recognition results, with this strategy scaling according to the number of classes of the dataset. Besides the scalability, our approach presents advantages in relation to literature methods, mainly regarding robustness to class imbalance, lower complexity, open-set applicability and class specificity. In addition to these advantages, the evaluation of MORA indicated a recognition

performance that could be compared to state-of-the-art methods. On SALMORA, the discriminative behavior of autoencoders was enhanced, in addition to the employment of a skin-attention mechanism, leading to a remarkable accuracy improvement. Even tuning a single layer of the autoencoders for few iterations, a great accuracy improvement was noticed on experiments. At the end, the contribution of the Large-Margin Softmax Loss [Liu et al., 2016] must be highlighted, leading SALMORA to rank 2th on both evaluated datasets.

MORA and SALMORA share disadvantages when compared to custom classification methods that employ a single network to perform gesture recognition. MORA and SALMORA are based on a set of different models that need to have their hyperparameters selected and validated through experiments. Employing hyperparameter search mechanisms, such hyperband, requires the execution of these techniques several times, what increases the time/computational cost of this stage. Disadvantages in terms of computational cost are more evident on SALMORA, which also requires insertion of an external classifier to separate weights of each model in relation to the remaining ones trained for a given dataset. Furthermore, MORA and SALMORA are dependent on stages that are external from the autoencoders, such as skin detection and classifier training. In addition, in terms of skin detection, the performance of the approach is hugely mitigated if the arms of the subject performing the gesture are not exposed. Although showing efficiency for gesture recognition task, MORA and SALMORA need to have their methodologies and architectures adjusted to be applied in the sign-language recognition field, in order to extract different input parameters (i.e., facial/body expression, inflection point) and traits of the language, such as dependency of previous terms and sentences and structure of phrases.

Our multi-task method MLRRN presented an accurate performance for detection and recognition of gestures on unsegmented videos. The evaluation of the method evidenced the capability to properly separate transition zones between different gesture classes (and non-gesture class), even in a cross-dataset scenario. Moreover, performing both tasks concomitantly showed a notorious improvement on their results, indicating the correlation that exists between them. The evaluation of MLRRN showed a good performance for both tasks, with state-of-the-art temporal Jaccard score on ChaLearn Montalbano dataset.

Despite the accurate outcomes for the evaluated datasets, MLRRN results demonstrated some inefficiency to detect and recognize the last frames regarding the gesture classes, commonly associated by MLRRN with the non-gesture class (relaxing frames). Moreover, a strong dependency of the approach in relation to the existence of frames separating gesture instances is noticed. Tests on ChaLearn ConGD showed an accu-

racy drop due to the absence of these frames (the annotation of these frames is not adequate), mitigating the contribution of the detection task and decreasing the final temporal Jaccard score of the method. The high dependency of recognition in relation to detection is also evidenced by this experiment and the grouping strategy employed to enhance the outcomes. Besides improving the results of the approach, the gain of this grouping mechanism relies on a consistent and hard to select clustering criterion.

7.2 Future Work

The purpose of this section is to analyze the open questions raised in this dissertation as future work.

7.2.1 Extend SALMORA's hyperparameter optimization

To enhance the outcomes of SALMORA on ChaLearn ConGD [Wan et al., 2016], we intend to execute hyperband for every class of the dataset. Due to the high cost of performing hyperband [Li et al., 2017] for all autoencoder models on ChaLearn ConGD [Wan et al., 2016], we randomly selected 20 classes and employed hyperband on these models. The most common hyperparameters were used to train all autoencoder models. Even presenting results that rival the state-of-the-art methods, the outcomes reported for SALMORA could be improved with the application of hyperband for all classes and employment of autoencoders with specific architectures for each class of the dataset.

7.2.2 Improvement and better evaluation of MLRRN

MLRRN presented accurate results for ChaLearn Montalbano and ChaLearn ConGD datasets. As a future step, we intend to better evaluate MLRRN, replacing VGG-16 activations by a more complex network, applying more sophisticated strategies to group gestures and applying this technique on different datasets. In addition, we intend to evaluate the architecture in similar domains, such as activity recognition and sign-language recognition.

7.2.3 Libras dataset

During this research, we could notice the existence of few gesture datasets that target unsegmented videos. In addition, these datasets tend not to tackle sign languages, presenting only gestures used for limited communication. This point becomes alarming

when we bring this situation to Brazilian scenario. On that, we could not find any unsegmented video dataset related to Brazilian Sign Language (Libras) that encompasses multiple parameters, such as facial expressions, gesture configurations and inflection point. Thus, we intend to assemble a Libras dataset containing unsegmented videos and approaching different parameters of the language.

7.2.4 Alternatives for recurrent models

Despite the accurate results and the dominance of models based on recurrent layers for the gesture recognition task, we intend to evaluate alternative architectures to perform the task, such as the ones based on Transformers [Vaswani et al., 2017], widely employed on the natural language processing field [Vaswani et al., 2017; Devlin et al., 2019]. From Transformers, more complex architectures have been produced, such as BERT [Devlin et al., 2019], which are able to gather long-term temporal information during the operation of a model, besides imbuing a self-attention mechanism, responsible to weigh the contribution of different parts of input for the recognition process.

7.2.5 Adjustments on Proposed Architectures

Despite the employment of Hyperband to adjust the set of hyperparameters of most approaches presented in this document, we still intend to evaluate/validate architectural changes on the proposed models. A first point to be evaluated regards the inputs of MORA and SALMORA models, which are disposed into 3-channels for each input modality (i.e., RGB, Optical Flow and Depth). To achieve this 3-channel distribution, information had to be replicated. We intend to compose a different input distribution, with no channel replication, to evaluate the performance of the models. For the MLRRN approach, we intend to evaluate a different layer disposition for the residual blocks, which were assembled according to the research of Wu et al. [2016]. A point to be evaluated on these modules is the application of different activation functions for the ELU-based layers, depicted in Figure 4.10.

7.2.6 Grouping strategy to increase scalability

With the aim of increasing the scalability on test phase of MORA and SALMORA approaches, we intend to perform a grouping of the trained gestures classes. The criterion to group the different classes can be associated to the distance between spatiotemporal features of different classes, extracted from a mid-level layer of autoencoders. On test phase, video samples would be provided only to the group(s) which presents the

average feature activations with the lowest distance a test sample. This process would minimize the cost to recognize a gesture video.

Bibliography

- Azad, R., Asadi-Aghbolaghi, M., Kasaei, S., and Escalera, S. (2019). Dynamic 3d hand gesture recognition by learning weighted depth motion maps. *IEEE Transactions on Circuits and Systems for Video Technology*, 29:1729–1740. 57
- Baldi, P. (2011). Autoencoders, unsupervised learning and deep architectures. In *Proceedings of the 2011 International Conference on Unsupervised and Transfer Learning Workshop, UTLW'11*, pages 37--50. 13
- Bastos, I., Melo, V., and Schwartz, W. (2019). Multi-loss recurrent residual networks for gesture detection and recognition. In *2019 32nd SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*, pages 170–177. 30
- Bastos, I. L. O., Angelo, M. F., and Loula, A. (2015). Recognition of static gestures applied to brazilian sign language (libras). In *SIBGRAPI*, pages 305--312. xviii, 1, 2, 19, 20, 28, 30
- Bastos, I. L. O., Melo, V. H. C., Goncalves, G. R., and Schwartz, W. R. (2018). Mora: A generative approach to extract spatiotemporal information applied to gesture recognition. In *15th International Conference on Advanced Video and Signal-based Surveillance (AVSS)*. 30
- Camgoz, N., Hadfield, S., Koller, O., Ney, H., and Bowden, R. (2018). Neural sign language translation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 27, 28, 30
- Cao, C., Zhang, Y., Wu, Y., Lu, H., and Cheng, J. (2018). Egocentric gesture recognition using recurrent 3d convolutional neural networks with spatiotemporal transformer modules. In *2017 IEEE International Conference on Computer Vision (ICCV)*, volume 00, pages 3783–3791. 27, 30
- Cao, Z., Simon, T., Wei, S., and Sheikh, Y. (2017). Realtime multi-person 2d pose estimation using part affinity fields. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1302–1310. 22, 40, 60, 65
- Cardenas, E., Cerna, L., and Camara-Chavez, G. (2019). Dynamic sign language recognition based on convolutional neural networks and texture maps. In *XXXII Conference on Graphics, Patterns and Images - SIBGRAPI*. 22, 30
- Caruana, R. (1997). Multitask learning. *Machine Learning*, 28(1):41--75. xvii, 14, 15
- Chang, J. Y. (2015). Nonparametric gesture labeling from multi-modal data. In *Computer Vision - ECCV 2014 Workshops*, pages 503--517. Springer International Publishing. 67

- Chen, G. (2016). A gentle tutorial of recurrent neural network with error backpropagation. *ArXiv*, abs/1610.02583. xvii, 11
- Cho, K., van Merriënboer, B., Gülçehre, Ç., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734. 12
- Cirujeda, P. and Binefa, X. (2014). 4dcov: A nested covariance descriptor of spatio-temporal features for gesture recognition in depth sequences. In *3DV*, pages 657–664. 57
- Collobert, R. and Weston, J. (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning, ICML '08*, pages 160–167. ACM. 14
- Côté, S. and Beaulieu, O. (2019). Vr road and construction site safety conceptual modeling based on hand gestures. *Front. Robotics and AI*, 2019. xvii, 1
- Crasborn, O., Zwitterlood, I., and Ros, J. (2008). A digital open access corpus of movies and annotations of sign language of the netherlands. centre for language studies. 30
- Deng, l., Hinton, G., and Kingsbury, B. (2013). New types of deep neural network learning for speech recognition and related applications: An overview. In *ICASSP*, pages 8599–8603. 14
- Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186. 78
- Duan, J., Zhou, S., Wan, J., Guo, X., and Li, S. Z. (2016). Multi-modality fusion based on consensus-voting and 3d convolution for isolated gesture recognition. *CoRR*. xviii, 21, 22, 30, 61
- Dumas, T., Roumy, A., and Guillemot, C. (2016). Shallow sparse autoencoders versus sparse coding algorithms for image compression. In *2016 IEEE International Conference on Multimedia & Expo Workshops, ICME Workshops 2016, Seattle, WA, USA, July 11-15, 2016*, pages 1–6. 14
- Escalera, S., Baro, X., Gonzalez, J., Bautista, M., Madadi, M., Reyes, M., Ponce-López, V., Escalante, H., Shotton, J., and Guyon, I. (2014). ChaLearn Looking at People Challenge 2014: Dataset and Results. In *ECCV*. xvii, xviii, 1, 4, 24, 30, 40, 47, 51, 63, 66
- Farneback, G. (2003). Two-frame motion estimation based on polynomial expansion. In *Proceedings of the 13th Scandinavian Conference on Image Analysis, SCIA'03*, pages 363–370, Berlin, Heidelberg. 31
- Fong, S., Yan, Z., Fister, I., and Fister jr, I. (2013). A biometric authentication model using hand gesture images. *Biomedical engineering online*, 12:111. xvii, 1
- Forster, J., Schmidt, C., Koller, O., Bellgardt, M., and Ney, H. (2014). Extensions of the sign language recognition and translation corpus RWTH-PHOENIX-weather. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 1911–1916. 30

- Gavrila, D. M. (1999). The visual analysis of human movement: A survey. *Computer Vision and Image Understanding*, 73:82–98. 39
- Girshick, R. B. (2015). Fast r-cnn. In *Proceedings of the International Conference on Computer Vision*. 14
- Gondara, L. (2016). Medical image denoising using convolutional denoising autoencoders. In *IEEE 16th International Conference on Data Mining Workshops (ICDMW)*. 14
- Greff, K., Srivastava, R. K., KoutnĀk, J., Steunebrink, B. R., and Schmidhuber, J. (2015). Lstm: A search space odyssey. *CoRR*. 52
- Guo, X., Liu, X., Zhu, E., and Yin, J. (2017). Deep clustering with convolutional autoencoders. In *ICONIP*. 14
- Gupta, V., Dwivedi, S. K. and Dabral, R., and Jain, A. (2019). Progression modelling for online and early gesture detection. In *International Conference on 3D Vision (3DV)*, pages 289–297. 67
- Hadji, I. and Wildes, R. (2018). What do we understand about convolutional networks? xvii, 14
- Han, W., Chang, S., Liu, D., Yu, M., Witbrock, M., and Huang, T. S. (2018). Image super-resolution via dual-state recurrent networks. In *CVPR*, pages 1654–1663. xvii, 10
- Herreweghe, M., Vermeerbergen, M., Demey, E., De Durpel, N., and Verstraete, S. (2015). Het corpus vgt. een digitaal open access corpus van videos and annotaties van vlaamse gebarentaal. 30
- Hoang, N., Lee, G., Kim, S., and Yang, H. (2019). Continuous hand gesture spotting and classification using 3d finger joints information. In *IEEE International Conference on Image Processing (ICIP)*, pages 539–543. 69
- Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780. 10
- Howard, A., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., and Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. *ArXiv*, abs/1704.04861. 22, 27
- Imran, J. and Raman, B. (2019). Deep motion templates and extreme learning machine for sign language recognition. *The Visual Computer*. 22, 57
- Inoue, T., Chaudhury, S., Magistris, G. D., and Dasgupta, S. (2018). Transfer learning from synthetic to real images using variational autoencoders for precise position detection. In *2018 IEEE International Conference on Image Processing, ICIP 2018, Athens, Greece, October 7-10, 2018*, pages 2725–2729. 14
- Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., and Fei-Fei, L. (2014). Large-Scale Video Classification with Convolutional Neural Networks. In *CVPR*. 21
- Krizhevsky, A. (2009). Learning multiple layers of features from tiny images. Technical report. xvii, 17

- Kuchaiev, O. and Ginsburg, B. (2017). Training deep autoencoders for collaborative filtering. *CoRR*. 14
- Le, T.-T.-H., Kim, J., and Kim, H. (2016). Classification performance using gated recurrent unit recurrent neural network on energy disaggregation. *2016 International Conference on Machine Learning and Cybernetics (ICMLC)*, pages 105–110. xvii, 12
- Li, B., Li, W., Tang, Y., Hu, J., and Zheng, W. (2018a). Gl-pam rgb-d gesture recognition. In *25th IEEE International Conference on Image Processing (ICIP)*, pages 3109–3113. 61
- Li, D., Chen, Y., Gao, M.-k., Jiang, S., and Huang, C. (2018b). Multimodal gesture recognition using densely connected convolution and blstm. *24th International Conference on Pattern Recognition (ICPR)*, pages 3365–3370. 57
- Li, L., Jamieson, K., DeSalvo, G., Rostamizadeh, A., and Talwalkar, A. (2017). Hyperband: A novel bandit-based approach to hyperparameter optimization. *J. Mach. Learn. Res.*, 18(1):6765–6816. 15, 16, 52, 64, 77
- Li, L., Jamieson, K. G., DeSalvo, G., Rostamizadeh, A., and Talwalkar, A. (2016). Efficient hyperparameter optimization and infinitely many armed bandits. *CoRR*, abs/1603.06560. 15, 16, 52, 64
- Li, W., Zhang, Z., and Liu, Z. (2010). Action recognition based on a bag of 3d points. *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops*, pages 9–14. 20, 30
- Li, Y., Miao, Q., Qi, X., Ma, Z., and Ouyang, W. (2019). A spatiotemporal attention-based resc3d model for large-scale gesture recognition. *Mach. Vis. Appl.*, 30(5):875–888. 57, 59, 61
- Liu, L. and Shao, L. (2013). Learning discriminative representations from rgb-d video data. In *IJCAI*, pages 1493–1500. xvii, xix, 4, 30, 38, 45, 46, 51, 57
- Liu, M. and Liu, H. (2016). Depth context. *Neurocomput.*, 175:747–758. xxi, 53, 56, 57, 61
- Liu, W., Wen, Y., Yu, Z., and Yang, M. (2016). Large-margin softmax loss for convolutional neural networks. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, page 507–516. xvii, 16, 17, 37, 76
- Liu, Z., Chai, X., Liu, Z., and Chen, X. (2017). Continuous gesture recognition with hand-oriented spatiotemporal feature. In *The IEEE International Conference on Computer Vision (ICCV) Workshops*. 22, 69
- Luo, Y., Ren, J. S. J., Wang, Z., Sun, W., Pan, J., Liu, J., Pang, J., and Lin, L. (2018). LSTM pose machines. In *CVPR*. 10
- Mitra, S. and Acharya, T. (2007). Gesture recognition: A survey. *Transactions Systems, Man and Cybernetics Part C*, 37(3):311–324. 1, 2

- Molchanov, P., Yang, X., Gupta, S., Kim, K., Tyree, S., and Kautz, J. (2016). Online detection and classification of dynamic hand gestures with recurrent 3d convolutional neural networks. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4207–4215. xviii, 1, 2, 3, 9, 19, 20, 21, 22, 23, 24, 26, 30, 38, 39, 57, 67
- Monnier, C., German, S., and Ost, A. (2015). A multi-scale boosted detector for efficient and robust gesture recognition. In Agapito, L., Bronstein, M. M., and Rother, C., editors, *Computer Vision - ECCV 2014 Workshops*. 67
- Narayana, P., Beveridge, J. R., and Draper, B. A. (2018). Gesture recognition: Focus on the hands. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5235–5244. 21, 60, 61
- Narayana, P., Beveridge, J. R., and Draper, B. A. (2019). Continuous gesture recognition through selective temporal fusion. In *International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. 69
- Neverova, N., Wolf, C., Taylor, G. W., and Nebout, F. (2016). Moddrop: Adaptive multi-modal gesture recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 38(8):1692–1706. 67
- Nishida, N. and Nakayama, H. (2016). Multimodal gesture recognition using multi-stream recurrent neural network. In *7th Pacific-Rim Symposium on Image and Video Technology - Volume 9431*, pages 682–694. xviii, 3, 9, 20, 21, 23, 24, 25, 57
- Pigou, L., Herreweghe, M. V., and Dambre, J. (2017). Gesture and sign language recognition with temporal residual networks. In *ICCV Workshops*, pages 3086–3093. IEEE Computer Society. xviii, 19, 21, 25, 26, 27, 30, 39, 42, 67
- Ren, Z., Yuan, J., and Zhang, Z. (2011). Robust hand gesture recognition based on finger-earth mover’s distance with a commodity depth camera. pages 1093–1096. 30
- Ronchetti, F., Quiroga, F., Estrebou, C., Lanzarini, L., and Rosete, A. (2016). Lsa64: An argentinian sign language dataset. In *XXII Congreso Argentino de Ciencias de la Computacin (CACIC)*. 22, 30
- Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1. In Rumelhart, D. E., McClelland, J. L., and PDP Research Group, C., editors, *MIT Cambridge, MA*, chapter Learning Internal Representations by Error Propagation, pages 318–362. MIT Press. 13
- Shahroudy, A., Liu, J., Ng, T., and Wang, G. (2016). Ntu rgb+d: A large scale dataset for 3d human activity analysis. In *IEEE CVPR*. 2
- Sharma, M. and Chawla, E. R. (2013). Gesture recognition: A survey of gesture recognition techniques using neural networks. *Global Journal of Computer Science and Technology: Neural and Artificial Intelligence*, 13(3). 2
- Simonyan, K. and Zisserman, A. (2014). Two-stream Convolutional Networks for Action Recognition in Videos. In *NIPS*. 21

- Song, Y., Demirdjian, D., and Davis, R. (2011). Tracking body and hands for gesture recognition: Natops aircraft handling signals database. In *FG*, pages 500–506. 20, 30
- Song, Y., Demirdjian, D., and Davis, R. (2012). Continuous body and hand gesture recognition for natural human-computer interaction. *ACM Transactions on Interactive Intelligent Systems*, 2:5. 19, 30, 38
- Soni, N., Nagmode, M., and Komati, R. (2016). Online hand gesture recognition classification for deaf dumb. In *2016 International Conference on Inventive Computation Technologies (ICICT)*, volume 3, pages 1–4. 1
- Souza, C., Pádua, F., Lima, V., Lacerda, A., and Carneiro, C. (2018). A computational approach to support the creation of terminological neologisms in sign languages. *Computer Applications in Engineering Education*. 2, 3
- Suganuma, M., Ozay, M., and Okatani, T. (2018). Exploiting the potential of standard convolutional autoencoders for image restoration by evolutionary search. In *ICML*, volume 80 of *JMLR Workshop and Conference Proceedings*, pages 4778--4787. 14
- Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going Deeper With Convolutions. In *CVPR*. 52
- Tan, W. R., Chan, C. S., Pratheepan, Y., and Condell, J. (2012). A fusion approach for efficient human skin detection. *IEEE Trans. Industrial Informatics*, 8(1):138–147. xix, 53, 54
- Tran, D., Bourdev, L., Fergus, R., Torresani, L., and Paluri, M. (2015). Learning Spatiotemporal Features With 3D Convolutional Networks. In *ICCV*. 20, 22, 24, 62
- Tran, D., Ray, J., Shou, Z., Chang, S.-F., and Paluri, M. (2017). Convnet architecture search for spatiotemporal feature learning. *CoRR*, abs/1708.05038. 26
- Tung, P. and Ngoc, L. (2014). Elliptical density shape model for hand gesture recognition. In *5th Symposium on Information and Communication Technology*, pages 186--191. 57
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems - NIPS*, pages 5998--6008. 78
- Wan, J., Li, S. Z., Zhao, Y., Zhou, S., Guyon, I., and Escalera, S. (2016). 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. xvii, xix, 4, 21, 22, 23, 30, 38, 46, 47, 48, 51, 59, 63, 77
- Wang, H., Klaser, A., Schmid, C., and Liu, C.-L. (2011). Action recognition by dense trajectories. In *CVPR*. 19
- Wang, H., Wang, P., Song, Z., and Li, W. (2017a). Large-scale multimodal gesture recognition using heterogeneous networks. In *ICCV Workshops*, pages 3129--3137. IEEE Computer Society. 19, 20, 21, 69

- Wang, H., Wang, P., Song, Z., and Li, W. (2017b). Large-scale multimodal gesture segmentation and recognition based on convolutional neural networks. In *The IEEE International Conference on Computer Vision (ICCV) Workshops*. 69
- Wang, J., Liu, Z., Chorowski, J., Chen, Z., and Wu, Y. (2012). Robust 3d action recognition with random occupancy patterns. In *Computer Vision – ECCV 2012*, volume 7573 of *Lecture Notes in Computer Science*, pages 872–885. Springer. 20, 30
- Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., and Van Gool, L. (2016). Temporal Segment Networks: Towards Good Practices for Deep Action Recognition. In *ECCV*. 63
- Wu, D., Pigou, L., Kindermans, P., Le, N. D., Shao, L., Dambre, J., and Odobez, J. (2016). Deep dynamic neural networks for multimodal gesture segmentation and recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 38(8):1583–1597. xviii, 5, 23, 24, 30, 78
- Wu, W., Yang, Q., Lv, J. and Li, A., and Liu, H. (2019). Investigation of remote sensing imageries for identifying soil texture classes using classification methods. *IEEE Transactions on Geoscience and Remote Sensing*, 57(3):1653–1663. 37
- Xu, J., Zhang, X., and Zhou, M. (2018). A high-security and smart interaction system based on hand gesture recognition for internet of things. *Security and Communication Networks*, 2018:1–11. 1
- Yang, X., Molchanov, P., and Kautz, J. (2018). Making convolutional networks recurrent for visual sequence learning. In *CVPR*, pages 6469–6478. 9
- Zhang, C. and Tian, Y. (2015). Histogram of 3d facets. *Computer Vision Image Understanding*, 139(C):29–39. 20
- Zhang, L., Zhu, G., Shen, P., Song, J., Shah, S. A., and Bennamoun, M. (2017). Learning spatiotemporal features using 3dcnn and convolutional lstm for gesture recognition. In *IEEE ICCV*. xviii, 2, 3, 15, 19, 20, 25, 27, 30, 45, 51, 57, 61
- Zhang, Y. and Yang, Q. (2017). A survey on multi-task learning. *CoRR*. 14, 15
- Zhao, Y., Deng, B., Shen, C., Liu, Y., Lu, H., and Hua, X.-S. (2017). Spatio-temporal autoencoder for video anomaly detection. In *Proceedings of the 25th ACM International Conference on Multimedia*, pages 1933–1941. ACM. 14
- Zheng, M., Tie, Y., Qi, L., and Jiang, S. (2019). Dynamic gesture recognition based on the multimodality fusion temporal segment networks. In *2019 8th International Symposium on Next Generation Electronics (ISNE)*, pages 1–3. 23, 30, 61
- Zhou, H. and Ruan, Q. (2006). A real-time gesture recognition algorithm on video surveillance. In *8th International Conference on Signal Processing*, volume 3. 2
- Zhu, G., Zhang, L., Mei, L., Shao, J., Song, J., and Shen, P. (2016). Large-scale isolated gesture recognition using pyramidal 3d convolutional networks. In *Proceedings of the 23rd International Conference on Pattern Recognition (ICPR)*. 1, 22, 30, 61

- Zhu, G., Zhang, L., Shen, P., Song, J., Shah, S., and Bennamoun, M. (2018). Continuous gesture segmentation and recognition using 3dcnn and convolutional lstm. *IEEE Transactions on Multimedia*. xviii, 26, 27, 28, 30, 67, 69