UNIVERSIDADE FEDERAL DO AMAZONAS
INSTITUTO DE COMPUTAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

Segmentação automática de demonstrações através da modelagem de séries temporais por processos beta

Gabriel Góes Rodrigues

Manaus – AM
Novembro de 2023

Gabriel Góes Rodrigues

Segmentação automática de demonstrações através da modelagem de séries temporais por processos beta

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Informática do Instituto de Computação da Universidade Federal do Amazonas (PPGI/ICOMP, UFAM) como parte dos requisitos necessários à obtenção do título de Mestre em informática.

Orientador(a)

Prof. Dr. José Reginaldo Hughes Carvalho

Universidade Federal do Amazonas – UFAM

Instituto de Computação – IComp

Manaus-AM

Novembro de 2023

Ficha Catalográfica

Ficha catalográfica elaborada automaticamente de acordo com os dados fornecidos pelo(a) autor(a).

Rodrigues, Gabriel Góes
R696s Segmentação automática de demonstrações através da modelagem de séries temporais por processos beta / Gabriel Góes Rodrigues . 2023
91 f.: il. color; 31 cm.

Orientador: José Reginaldo Hughes Carvalho
Dissertação (Mestrado em Informática) - Universidade Federal do Amazonas.

1. Reconhecimento de atividades. 2. Aprendizagem de robôs. 3. Aprendizagem por demonstração. 4. Reconhecimento de habilidades. 5. Segmentação de ações. I. Carvalho, José Reginaldo Hughes. II. Universidade Federal do Amazonas III. Título

Ministério da Educação
Universidade Federal do Amazonas
Coordenação do Programa de Pós-Graduação em Informática

FOLHA DE APROVAÇÃO

"SEGMENTAÇÃO AUTOMÁTICA DE DEMONSTRAÇÕES ATRAVÉS DA MODELAGEM DE SÉRIES TEMPORAIS POR PROCESSOS BETA"

GABRIEL GOES RODRIGUES

Dissertação de Mestrado defendida e aprovada pela banca examinadora constituída pelos Professores:

Prof. Dr. José Reginaldo Hughes Carvalho - PPGI/UFAM - PRESIDENTE

Prof. Dr. Edson de Araújo Silva - MEMBRO EXTERNO

Prof. Dr. Felipe Gomes de Oliveira - MEMBRO INTERNO

Manaus, 14 de novembro de 2023

Documento assinado eletronicamente por **José Reginaldo Hughes Carvalho**, **Professor do Magistério Superior**, em 16/11/2023, às 17:47, conforme horário oficial de Manaus, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

Documento assinado eletronicamente por **Felipe Gomes de Oliveira**, **Professor do Magistério Superior**, em 23/11/2023, às 00:01, conforme

horário oficial de Manaus, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

Documento assinado eletronicamente por **Edson de Araújo Silva**, **Professor do Magistério Superior**, em 03/01/2024, às 20:02, conforme horário oficial de Manaus, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

A autenticidade deste documento pode ser conferida no site https://sei.ufam.edu.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **1783878** e o código CRC **00E0C26E**.

Avenida General Rodrigo Octávio, 6200 - Bairro Coroado I Campus Universitário Senador Arthur Virgílio Filho, Setor Norte - Telefone: (92) 3305-1181 / Ramal 1193
CEP 69080-900, Manaus/AM, coordenadorppgi@icomp.ufam.edu.br

Referência: Processo nº 23105.050024/2023-31 SEI nº 1783878

Dedico este trabalho a meu pai, obrigado por tudo.

Agradecimentos

A Ayná Vieira, pela sua paciência e amor que me fizeram ver a vida com novas lentes. Sem seu suporte emocional eu nunca teria dado os passos que dei. Agradeço de coração por ter encontrado a parceira de uma vida inteira.

A meu pai, Ivo Seixas pelo amor e lições de disciplina, sem os quais não conseguiria ter caminhado uma distância tão grande. Suas lições de respeito e integridade foram passadas através do exemplo, devo meus valores a você. Que Deus o tenha.

A minha irmã Júlia Góes por seu amor e carinho mesmo nos momentos difíceis.

A minha tia Ione Araújo pelo amor e carinho, e por ter me acolhido quando eu mais precisei.

A minha vó Ivone Araújo por todo amor e generosidade, e ao meu avô Ivan pelo amor e carinho.

Ao Prof. Dr. José Reginaldo Hughes pela sua amizade, confiança em meu trabalho e principalmente pelos ensinamentos.

A todos os professores do Instituto de Computação pelos ensinamentos.

Ao corpo técnico e aos professores da Universidade Federal do Amazonas, sem os quais nada disso seria possível.

"*Você sempre pode reconhecer a verdade por sua beleza e simplicidade.*"

Richard P. Feynman

Segmentação automática de demonstrações através da modelagem de séries temporais por processos beta

Autor: Gabriel Góes Rodrigues

Orientador(a): Prof. Dr. José Reginaldo Hughes Carvalho

RESUMO

Este trabalho aborda a questão da segmentação automática de demonstrações de tarefas de manipulação dentro do contexto de Learning from Demonstration. Este é um conjunto de técnicas para programação de robôs e aprendizagem de políticas, baseado na observação de demonstrações de tarefas fornecidas por um professor humano ou robô. Para que um programador de robôs aplique o Learning from Demonstration, ele deve dividir as demonstrações em atividades ou ações antes de utilizá-las para o ensino real do robô, após esta segmentação os dados organizados são alimentados em algoritmos de aprendizagem, que são o ponto principal do LfD. Neste trabalho, foca-se na parte de segmentação do problema e propõe-se uma ferramenta de segmentação de tarefas de manipulação baseada em vídeo. A abordagem baseada em dados aqui proposta não requer nenhum conhecimento prévio de programação de robôs ou segmentação manual por parte do professor humano. Usam-se ferramentas de aprendizado de máquina prontas para uso para anotação automática de imagens e modelos de Markov ocultos autorregressivos de processo beta que aproveitam uma representação infinita baseada em recursos para criar uma ferramenta contínua e fácil de usar que atinge a segmentação semântica relevante de uma forma completamente não supervisionada. Como resultado, este método atinge uma acurácia de alinhamento temporal máxima de 96% e acurácia de classificação máxima de 86,8%, sem conhecimento prévio das ações sendo segmentadas, deixando claro que esta técnica pode ser usada para segmentar tarefas de manipulação sem a necessidade de qualquer trabalho manual além da demonstração em si.

Palavras-chave: Reconhecimento de atividades, aprendizagem de robôs, aprendizagem por demonstração, reconhecimento de habilidades, segmentação de ações, modelos ocultos de

Markov.

Automatic segmentation of demonstrations through time series modeling by beta processes

Author: Gabriel Góes Rodrigues

Advisor: Prof. Dr. José Reginaldo Hughes Carvalho

ABSTRACT

This work addresses the issue of automatic segmentation of manipulation task demonstrations in the context of Learning from Demonstration. Learning from Demonstration is a set of techniques for robot programming and policy learning, based on observation of task demonstrations provided by a human or robot teacher. For a robot programmer to apply Learning from Demonstration, he must divide the demonstrations into activities or actions before using them for actual teaching of the robot. After this segmentation, the organized data is fed into learning algorithms, which are the main point of the LfD. In this work, we focus on the segmentation part of the problem and propose a video-based manipulation task segmentation tool. The proposed data-driven approach does not require any prior knowledge of robot programming or manual segmentation on the part of the human teacher. It uses out-of-the-box machine learning tools for automatic image annotation and beta-process autoregressive hidden Markov models that leverage an infinite feature-based representation to create a seamless, easy-to-use tool that achieves relevant semantic segmentation of a completely unsupervised manner. As a result, this method reaches a maximum temporal alignment accuracy of 96% and maximum classification accuracy of 86.8%, without prior knowledge of the actions being segmented, making it clear that this technique can be used to segment manipulation tasks without the need for any manual work beyond of the demonstration itself.

Keywords: Activity recognition, robot learning, learning from demonstration, skill recognition, action segmentation, hidden Markov models..

Lista de figuras

Figura 1. Número de publicações, entre 2008 a 2018, sobre programação e aprendizagem por demonstração (RAVICHANDAR et al., 2020). 22

Figura 2. Divisão de abordagens de programação robótica (BIGGS e MACDONALD, 2023) 23

Figura 3. Densidade de probabilidade e realização de processo beta (BRODERICK, JORDAN , & PITMAN, 2011). 27

Figura 4. Exemplo de realização do processo beta (BRODERICK, JORDAN , & PITMAN, 2011) 28

Figura 5. Exemplo de realização do processo de Bernoulli conjugado ao processo beta (BRODERICK, JORDAN , & PITMAN, 2011). 28

Figura 6. Representação gráfica do BP-AR-HMM. 30

Figura 7. Algoritmo BP-AR-HMM 33

Figura 8. Estrutura da solução proposta. 38

Figura 9. Marcadores gerados pela API Mediapipe (a). Nomenclatura dos marcadores gerados (b). 39

Figura 10. Séries temporais. 40

Figura 11. Representação gráfica da estrutura de segmentação automática. 41

Figura 12. Exemplo ilustrativo da métrica de desempenho de segmentação. 42

Figura 13. Exemplo ilustrativo da métrica de acurácia de classificação 43

Figura 14. Movimentos realizados no cenário 1. 45

Figura 15. Movimentos realizados no cenário 2. 45

Figura 16. Rótulos definidos para segmentação manual – Cenário 1. 47

Figura 17. Mapa de segmentação manual – Verdade absoluta. Cenário 1. 48

Figura 18. Mapa de segmentação automática do cenário 1 - 5 *features*. Cenário 1. 51

Figura 19. Variação métrica com inicialização de - 5 *features*. Cenário 1. 51

Figura 20. Mapa de segmentação automática de 10 *features*. Cenário 1. 55

Figura 21. Variação métrica com inicialização de 10 *features*. Cenário 1. 55

Figura 22. Mapa de segmentação dos padrões de subtarefas encontrados para a ação "Repousar a mão". 59

Figura 23. Estados detectados como subtarefas para a ação de "Repousar a mão". 59

Figura 24. Mapa de segmentação automática do cenário 1 – 20 *features*. Cenário 1. 59

Figura 25. Variação métrica com inicialização - 20 *features*. Cenário 1. 60

Figura 26. Mapa de segmentação automática – 30 *features*. Cenário 1. 63

Figura 27. Variação métrica com inicialização de 30 *features*. Cenário 1. 63

Figura 28. Rótulos definidos para segmentação manual – Cenário 2. 64

Figura 29. Mapa de segmentação manual do cenário 2. 64

Figura 30. Mapa de segmentação automática - 5 *features*. Cenário 2. 68

Figura 31. Variação métrica com inicialização de 5 *features*. Cenário 2. 68

Figura 32. Mapa de segmentação automática - 10 *features*. Cenário 2. 71

Figura 33. Variação métrica com inicialização de 10 *features*. Cenário 2. 71

Figura 34. Mapa de segmentação automática – 20 *features*. Cenário 2. 74

Figura 35. Variação métrica com inicialização de 20 *features*. Cenário 2. 75

Figura 36. Mapa de segmentação automática - 10 *features*. Cenário 2. 79

Figura 37. Variação métrica com inicialização de 30 *features*. Cenário 2. 79

Figura 38. Resumo dos resultados 80

Lista de tabelas

Tabela 1. Matriz de confusão - 5 *features*. Cenário 1. 48

Tabela 2. Acurácia de classificação por ação demonstrada - 5 *features*. Cenário 1. 50

Tabela 3. Acurácia de classificação por estado gerado - 5 *features*. Cenário 1. 50

Tabela 4. Matriz de confusão - 10 *features*. Cenário 1. 53

Tabela 5. Acurácia de classificação por ação demonstrada - 10 *features*. Cenário 1. 54

Tabela 6. Acurácia de classificação por estado gerado - 10 *features*. Cenário 1. 54

Tabela 7. Matriz de confusão - 20 *features*. Cenário 1. 56

Tabela 8. Acurácia de classificação por ação demonstrada - 20 *features*. Cenário 1. 57

Tabela 9. Acurácia de classificação por estado gerado – 20 *features*. Cenário 1. 57

Tabela 10. Matriz de confusão - 30 *features*. Cenário 1. 61

Tabela 11. Acurácia de classificação por ação demonstrada - 30 *features*. Cenário 1. 62

Tabela 12. Acurácia de classificação por estado gerado - 30 *features*. Cenário 1. 62

Tabela 13. Matriz de confusão - 5 *features*. Cenário 2. 65

Tabela 14. Acurácia de classificação por ação demonstrada - 5 features. Cenário 2. 66

Tabela 15. Acurácia de classificação por estado gerado - 5 *features*. Cenário 2. 67

Tabela 16. Matriz de confusão - 10 *features*. Cenário 2. 69

Tabela 17. Acurácia de classificação por ação demonstrada - 10 *features*. Cenário 2. 70

Tabela 18. Acurácia de classificação por estado gerado - 10 *features*. 70

Tabela 19. Matriz de confusão - 20 *features*. Cenário 2. 72

Tabela 20. Acurácia de classificação por ação demonstrada - 20 *features*. Cenário 2. 73

Tabela 21. Acurácia de classificação por estado gerado 20 *features*. Cenário 2. 74

Tabela 22. Matriz de confusão - 30 *features*. Cenário 2. 76

Tabela 23. Acurácia de classificação por ações demonstrada - 30 *features*. Cenário 2. 77

Tabela 24. Acurácia de classificação por estado gerado - 30 *features*. 78

Lista de equações

Equação 1. Medida de Levy. 27

Equação 2. Descrição da realização do processo beta. 27

Equação 3. Realização do processo de beta dada a medida base $\boldsymbol{B0}$ 30

Equação 4. Descrição da realização do processo beta. 30

Equação 5. Realização do processo de Bernoulli com medida atômica dada pelo processo beta. 30

Equação 6. Distribuição a priori da probabilidade de transição do modelo BP-AR-HMM. 30

Equação 7. Distribuição de probabilidade de transição de estados do modelo BP-AR-HMM. 30

Equação 8. Distribuição de probabilidade de emissão do modelo BP-AR-HMM. 30

Equação 9. Métrica de acurácia de segmentação. 43

Equação 10. Métrica de acurácia de classificação. 43

Lista de abreviaturas e siglas

BP-AR-HMM – Beta Process Auto-regressive Hidden Markov Model

DMP – Dynamic Movement Primitives

FP - False positive

FPS – Frames por segundo

GMM – Gaussian Mixture Models

HMM - Hidden Markov Model

IComp – Instituto de Computação

LfD - Learning from Demonstration

MASD - Multimodal Assembly Skill Decoding System

MCMC - Markov Chain Monte Carlo

PfD - Programming from Demonstration

PME - Pequenas e Médias Empresas

RGB – Red Green Blue

SME – Small and Medium Enterprises

SVM - Support Vector Machines

TP - True Positive

UFAM – Universidade Federal do Amazonas

Sumário

1 Introdução 17

1.1 Contextualização 17

1.2 Objetivos 21

1.3 Organização do trabalho 21

2 Fundamentação teórica 22

2.1 Programação por demonstração 22

2.2 BP-AR-HMM 26

2.2.1 Processos beta e processo de bernoulli para a estratificação de comportamentos 26

2.2.2 Modelagem do BP-AR-HMM 29

2.2.3 Cálculo do posterior (BP-AR-HMM-MCMC) 31

3 Metodologia 34

4 Solução proposta 37

4.1 Estrutura 37

4.2 Definição do problema 40

4.3 Métricas de desempenho de segmentação 41

4.4 Setup e tarefas demonstradas 43

4.4.1 Cenário 1 – Movimentação do peão 44

4.4.2 Cenário 2 – Jogada Ruy Lopes 45

5 Resultados 46

5.1 Cenário 1 – Movimentação do peão 46

5.1.1 Segmentação automática com inicialização de 5 *features* 48

5.1.2 Segmentação automática com inicialização de 10 *features* 52

5.1.3 Segmentação automática com inicialização de 20 *features* 55

5.1.4 Segmentação automática com inicialização de 30 *features*..... 60

5.2 Cenário 2 – Jogada Ruy Lopes..... 63

5.2.1 Segmentação automática com inicialização de 5 *features* 65

5.2.2 Segmentação automática com inicialização de 10 *features*..... 69

5.2.3 Segmentação automática com inicialização de 20 *features*..... 72

5.2.4 Segmentação automática com inicialização de 30 *features*..... 75

6 Considerações finais.....80

7 Referências.....82

APÊNDICE A – Amostragem de *features* compartilhados 87

APÊNDICE B – Amostragem de sequência de estados 88

APÊNDICE C – Amostragem dos parâmetros de emissão.....89

APÊNDICE D – Amostragem de nascimento e morte de *features*.....90

APÊNDICE E – Amostragem de pesos de transição.....91

APÊNDICE F – Amostragem dos Hiper parâmetros do processo beta 92

APÊNDICE G – Amostragem dos Hiper parâmetros do modelo HMM 93

1 Introdução

1.1 Contextualização

Com a quarta revolução industrial, a integração de componentes sensoriais e robóticos com os sistemas de computação de alto desempenho alcança a fronteira da interação entre o homem e a máquina no processo produtivo. Dessa forma, cresce o número de sistemas robóticos colaborativos em processos de manufatura e aumenta proporcionalmente o número de robôs interagindo e trabalhando lado a lado com seres humanos (KRÜGER, et al., 2017). Essa evolução também vem tornando os processos produtivos mais eficientes e com melhor qualidade, uma vez que atividades de alta precisão e repetibilidade crítica estão sendo destinadas aos agentes robóticos. Como consequência desse avanço, proliferam-se também as soluções de programação baseadas em *Programming from Demonstration* (PfD), técnica cuja premissa é de que o robô seja programado através de demonstrações de um especialista. Esta abordagem permite que haja a transferência de habilidades e tarefas de seres humanos a agentes robóticos através de uma interface intuitiva. Apesar desta possibilidade, tradicionalmente a programação de robôs usando PfD é feita em ambientes laboratoriais por roboticistas que atuam como usuário final durante o período de validação. (CHERNOVA & THOMAZ, 2014)

Em paralelo, os sistemas de fabricação modernos, vem passando por uma crescente na demanda por produtos personalizados. Empresas de manufatura, cada vez mais tem optado por oferecer produtos individualizados para seus clientes em uma nova tendência que vai de encontro com a produção de grandes quantidades dos mesmos produtos, praticadas nos anos anteriores. Nesse contexto, há uma transição notável nos últimos anos, da fabricação de grandes lotes de produtos para lotes menores. (KUMAR, 2007)

Lotes menores trazem consigo uma consequência clara de maior variação nos cenários de implementação, pois novos produtos têm maneiras de produzir diferentes, processos diferentes, linhas produtivas de formatos diferentes. Ambientes dinâmicos e objetivos variáveis que, de acordo com o que foi discutido acima, fazem com que soluções robóticas sejam difíceis de implementar. Observando estes fatos é claro que a aplicação de métodos tradicionais de

programação nestes cenários seria ineficiente, especialmente para empresas menores, incapazes de lidar com os grandes ônus da variabilidade (KUMAR, 2007)

Unidades robóticas são hoje uma das formas mais complexas e completas de maquinário, sendo capazes de explorar desde as profundezas do mar até a superfície de um planeta remoto. Mais recentemente, a tecnologia que antes era aplicada apenas ao estado da arte da manufatura e exploração, agora também se torna disponível no cotidiano da maioria das pessoas, na forma de robôs aspiradores, carros autônomos (SILVER, BAGNELL, & STENTZ, 2010), entre outras aplicações domésticas (DONG, et al., 2021). Apesar dessa grande variedade de possibilidades para a robótica e de sua expansão nos últimos anos, alguns problemas ainda podem ser considerados impossíveis para este campo da engenharia. Tarefas com ambientes dinâmicos, ruidosos e com objetivos variáveis tendem a ser exponencialmente mais difíceis de implementar. (CHERNOVA & THOMAZ, 2014)

Podemos exemplificar esta realidade com o robô aspirador de pó. Este agente robótico, opera apenas em ambientes altamente controlados, que engloba exclusivamente o chão liso, sem objetos de pano, cadarços, brinquedos ou mudanças de elevação bruscas. É comum que os fabricantes incluam nos manuais de instrução alertas para que se remova quaisquer objetos estranhos do chão antes de iniciar os ciclos de aspiração, que se faça a limpeza periódica do robô e que não o permita chegar em regiões molhadas da residência, entre outras limitações. Devido a essas restrições a implementação de robôs aspiradores são simples e tem sido um sucesso desde sua invenção (CHERNOVA & THOMAZ, 2014)

Observando um outro exemplo no outro extremo de complexidade, um robô industrial multieixos industrial é uma solução extremamente complexa, capaz de realizar uma variedade enorme de movimentos precisos. Como observado por Ravichandar et al. (2020) eles podem ser utilizados nas mais diversas aplicações, como soldagem, pintura, transporte, montagem, dentre outros. Por esse motivo, a implementação deste tipo de agente é complexa e requer mão de obra especializada, muito diferente do único botão de acionamento do robô aspirador citado no exemplo anterior. Isso faz com que, apesar de extremamente versáteis, sua adaptação a novos cenários seja cara e morosa.

O paradigma da programação por demonstração vem para trazer uma solução criativa para situações em que a adaptação de um robô a novos cenários é difícil. Ela propõe que o robô seja treinado através das demonstrações de um especialista humano, capaz de executar a tarefa

alvo, seja através de demonstrações cenestésicas, operação remota do agente robótico ou mesmo através de execução direta, utilizando seu próprio corpo. Essa abordagem é oposta ao método tradicional de programação robótica, onde é necessário que o usuário final do robô programe uma sequência clara de ações para completar a atividade em questão. Portanto, a programação por demonstração atinge um objetivo muito interessante, o de facilitar o acesso a soluções robóticas por usuários que não necessariamente são especialistas em robótica (CHERNOVA & THOMAZ, 2014).

Sendo assim, esta transição é responsável por uma procura cada vez maior por soluções de automação através de robôs que sejam ao mesmo tempo ágeis e fáceis de programar (KUMAR, 2007). Isso é ainda mais relevante para as Pequenas e Médias Empresas (PME), elas têm grande interesse em melhorias de eficiência e qualidade para reduzir custos e sobreviver em um mercado competitivo. As soluções robóticas podem ajudá-las a satisfazer estes desejos no entanto, existem barreiras à implementação de robôs na sua produção (BI, et al., 2015)

As técnicas de programação de robôs exigem amplo treinamento e conhecimento prévio dos desenvolvedores, o que pode ser desgastante para as pequenas empresas, (BI, et al., 2015). Além disso, em tais ambientes, os sistemas robóticos devem se adaptar a diversas ferramentas e peças, este problema não é facilmente resolvido pelas abordagens tradicionais de programação robótica sem um investimento de tempo dispendioso (GUERIN, et al., 2015)

Para atender à demanda por soluções robóticas intuitivas, rápidas e facilmente programáveis, o *Learning from Demonstration* (LfD) fornece recursos de desenvolvimento de políticas para não especialistas e permite a interação intuitiva entre humanos e robôs. LfD é uma abordagem de aprendizagem de políticas de robôs em que as habilidades são transferidas para um agente robô por meio de dados de demonstração fornecidos por um professor. Isso permite que as políticas sejam baseadas em habilidades executadas por humanos, e não em programas codificados, e fornece aos usuários iniciantes uma maneira fácil de usar para transferir suas habilidades especializadas específicas de campo para agentes robóticos (SI, WAMG, & YANG, 2021).

A aplicação LfD exige baixos requisitos de conhecimento do usuário final e tempos de implantação mais rápidos. Em geral, é mais lucrativo empregar robôs que possam aprender com simples demonstrações humanas, do que perder tempo com programação complicada e longos períodos de inatividade. Devido às suas muitas vantagens na indústria, o interesse pela

investigação do LfD tem aumentado constantemente nos últimos anos (RAVICHANDAR, POLYDOROS, CHERNOVA, & BILLARD, 2020) Dado que o principal objetivo do LfD é facilitar a transferência de ações complicadas e repetitivas de um especialista humano para um agente robótico, a abordagem ao ensino deve ser o menos envolvente possível do ponto de vista do professor. Em resumo, quanto menos demonstrações, menos sensores aplicados e menor esforço exercido pelo usuário final, melhor (HEDLUND & JOHNSON, 2021).

A maioria das abordagens LfD são demonstradas em sessões contínuas, no entanto, os algoritmos de aprendizagem exigem que os dados sejam bem-organizados em tarefas ou blocos de habilidades, tornando a segmentação da demonstração um problema crucial a ser resolvido (RAVICHANDAR, POLYDOROS, CHERNOVA, & BILLARD, 2020). Contudo, na maioria dos trabalhos, a segmentação é normalmente realizada manualmente, o que é um trabalho tedioso e aumenta muito a necessidade de experiência em aplicações LfD. Com isso em mente, a principal contribuição deste trabalho é um novo método para realizar automaticamente a segmentação de habilidades a partir de dados de captura de movimento.

Muitos trabalhos atualmente vêm buscando resolver este problema, alguns utilizando abordagens baseadas em tarefas com comportamentos e nomes pré-definidos, que são classificados utilizando condições bem definidas (EIBAND et al., 2023) (RAMIREZ-AMARO , YANG , & CHENG , 2019). Outros, em contrapartida, buscam soluções baseadas em dados, que identificam os padrões para segmentação a partir de modelos pré-treinados ou encontrando padrões em tempo real em um *dataset* (EIBAND et al., 2023)(WANG et al., 2021). O segundo método possui características que se alinham com a filosofia do LfD, especialmente quando analisando algumas soluções não supervisionadas como a utilização de Modelos Ocultos de Markov Auto-regressivos e processos beta (BP-AR-HMM), que elimina a necessidade de conhecimento do que está sendo demonstrado. (NIEKUM et al., 2012); (FOX et al., 2014)

Dentro deste contexto, propõe-se uma abordagem para resolver este problema com o mínimo de incômodo possível para o usuário final. Combinando o uso de ferramentas disponíveis no mercado para anotar automaticamente um vídeo de uma tarefa de manipulação com uma abordagem não supervisionada utilizando BP-AR-HMM para segmentar essa tarefa em blocos de ação simples.

1.2 Objetivos

Desenvolver um framework de segmentação automática de demonstração de tarefa manipulação baseado em visão computacional utilizando câmera RGB e algoritmo de modelagem estocástica BP-AR-HMM.

Para atingir esse objetivo geral desta pesquisa, um conjunto de resultados intermediários deve ser alcançado:

- Gerar um conjunto de demonstrações da tarefa de manipulação realizadas pelo especialista humano utilizando câmera RGB.
- Implementar algoritmo de anotação automática da demonstração.
- Implementar algoritmo de segmentação baseado em BP-AR-HMM.
- Realizar testes de desempenho comparativos.

1.3 Organização do trabalho

No Capítulo 2 é introduzida a fundamentação teórica necessária para o entendimento deste trabalho, nesta sessão são apresentados os principais conceitos sobre programação por demonstração, Modelos ocultos de Markov e BP-AR-HMM. No Capítulo 3 apresenta-se a metodologia aplicada para a construção do presente trabalho, etapas de desenvolvimento e ferramentas utilizadas. Já no Capítulo 4 expõe-se a arquitetura da solução proposta para a segmentação automática de atividades de manipulação demonstradas por um ser humano, incluindo o fluxograma de processamento das séries temporais, os algoritmos aplicados e métricas utilizadas para aferir a performance do algoritmo proposto.

Dentro do Capítulo 5 são apresentados e discutidos os resultados obtidos nos experimentos. Finalmente, no Capítulo 6 são explanadas as considerações finais, as limitações do método e os possíveis trabalhos futuros.

2 Fundamentação teórica

2.1 Programação por demonstração

Nos últimos dez anos, tem havido um aumento constante no interesse de pesquisa no ensino de robôs através de demonstrações. O número de publicações relacionadas com o assunto aumentou significativamente, como pode ser visto na **Figura 1**. Este processo de aprendizagem robótica também é conhecido na literatura por diversos nomes, aprendizagem por imitação, programação por demonstração ou clonagem comportamental. No presente trabalho, referenciar-se-á a este conjunto de técnicas através da sigla em inglês, PfD (*Programming from Demonstration*) (RAVICHANDAR et al., 2020).

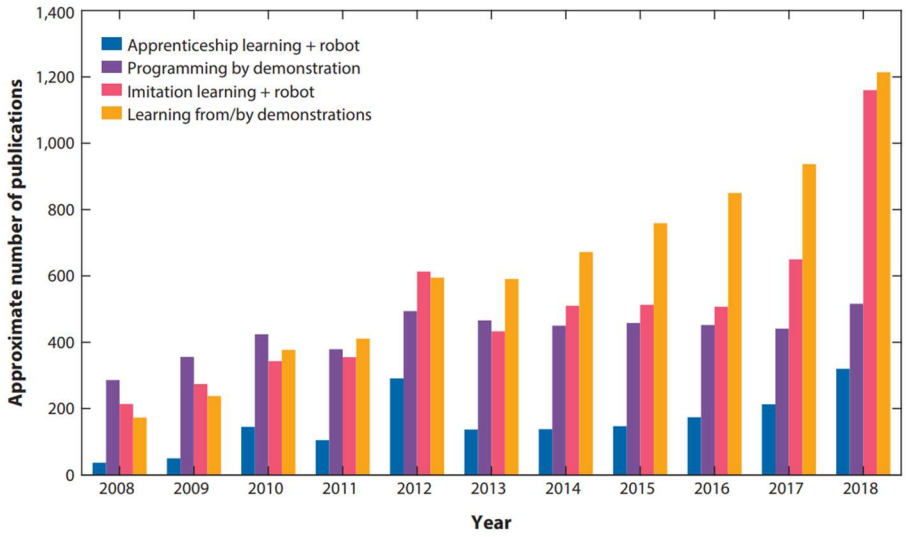

Figura 1. Número de publicações, entre 2008 a 2018, sobre programação e aprendizagem por demonstração (RAVICHANDAR et al., 2020).

LfD é um conjunto de ferramentas de aprendizado de máquina que pode ser categorizado como um subconjunto de aprendizado supervisionado. Neste tipo de abordagem, o sistema de aprendizado recebe como entrada um conjunto de demonstrações realizadas por um especialista humano ou, em alguns casos menos comuns um outro robô. Sendo assim, a definição formal dos problemas de PfD tem 2 componentes básico, um conjunto de estados S, um conjunto de ações A.

S representa o estado dos elementos que compõem o mundo no qual o robô se insere. Por exemplo, os ângulos de um braço robótico ou a posição translacional de um objeto alvo de manipulação. Por outro lado, as ações A representa as ações a serem tomadas pelo agente robótico, que dependendo da aplicação podem englobar um conjunto de ações de baixo nível ou alto nível.

As técnicas de programação de robôs tradicionais exigem amplo treinamento e conhecimento prévio dos desenvolvedores, o que pode ser desgastante para as pequenas empresas (BI et al., 2015). Além disso, em tais ambientes, os sistemas robóticos devem se adaptar a diversas ferramentas e peças, este problema não é facilmente resolvido pelas abordagens tradicionais de programação robótica sem um investimento de tempo dispendioso (GUERIN et al., 2015).

As abordagens de programação de uma unidade robótica podem ser divididas de acordo demonstrada na Figura 2 Segmentando-se em abordagens manuais, automáticas e arquiteturas de software. (BIGGS e MACDONALD, 2023)

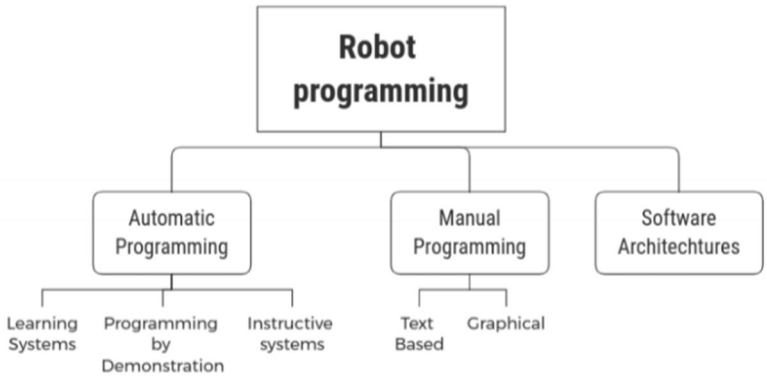

Figura 2. Divisão de abordagens de programação robótica (BIGGS e MACDONALD, 2023)

Programação por demonstração trata-se do campo da robótica onde pressupõe-se a interação com um especialista humano, capaz de prover demonstrações de uma atividade ou tarefa a um agente robótico, que por sua vez utiliza estas demonstrações para assimilar parcial ou completamente as habilidades demonstradas.

Sistemas que utilizam *programming by demonstration* seguem uma estrutura baseada em três conceitos fundamentais, propostos inicialmente por Kuniyoshi, Inaba e Inoue (1994), que definem os requisitos mínimos para que um sistema robótico possa realizar a imitação das ações ou habilidades realizadas por um especialista. O autor define inicialmente a função “Ver”,

referindo-se à habilidade do sistema robótico de reconhecer o estado inicial do ambiente, encontrar e seguir objetos de interesse e buscar o alvo da tarefa em questão. Em seguida, a função "Entender" é explorada como a capacidade de segmentar uma determinada tarefa contínua em unidades significativas, classificar as operações em tipos de movimentação, de objetos alvo e seus efeitos no alvo da tarefa. Finalmente, a função "Fazer" que descreve a habilidade do agente de instanciar uma ação planejada, planejar um caminho de execução e detecção de erros. (KUNIYOSHI, INABA E INOUE, 1994)

Os modos de demonstração LfD geralmente se dividem em três categorias: ensino cinestésico, teleoperação e captura de movimento (SI, WAMG, & YANG, 2021); (ZHU & HU, 2018); (HEDLUND & JOHNSON, 2021). O ensino cinestésico pressupõe que um robô físico esteja presente e que um professor humano manipule seus efetores finais para ensinar uma tarefa. Esse tipo de programação exige um grande investimento inicial, considerando a compra do próprio agente robô, software e treinamento de especialistas em robótica.

A seguir, a abordagem de teleoperação envolve um operador treinado controlando remotamente o agente robótico para ensinar uma habilidade alvo, tornando esta, de longe, a mais complicada das três, uma vez que os usuários acham significativamente mais difícil concluir tarefas (HEDLUND & JOHNSON, 2021). Finalmente, a captura de movimento envolve um sistema de observação passiva que observa a tarefa executada por um professor habilidoso. Essas soluções permitem uma experiência mais simples e intuitiva para o usuário final e, por esse motivo, representam o método de instrução menos difícil de ser manuseado pelo usuário final (HEDLUND & JOHNSON, 2021).

Devido a essas características, optamos por construir nossa solução utilizando um sistema baseado em vídeo, à mão livre, que observa passivamente as demonstrações.

Em relação aos métodos de segmentação de tarefas, pesquisas anteriores abordaram o problema de duas maneiras principais: abordagens simbólicas (ou sintáticas) e abordagens baseadas em dados. Numa abordagem Simbólica, os estados do robô são categorizados em blocos de construção básicos de acordo com a sua conformidade com certas regras, que são comumente referidas como pré e pós-condições, estes blocos recebem um rótulo semanticamente relevante, como "mover" ou "escolha" (EIBAND et al., 2023).

Park e Aggarwal (2004), apresentam uma aplicação de segmentação de demonstração através de uma abordagem simbólica. Eles apresentam uma metodologia que reúne uma demonstração

em vídeo e atribui um dos 9 tipos de interação pré-determinados a cada quadro, isso é feito adotando uma estrutura linguística de "argumento verbal" que representa as ações realizadas pelos humanos em termos de <agente-movimento- alvo> trigêmeos. O trabalho em Steinmetz, Nitsch e Stulp (2019) também apresenta uma solução simbólica. No entanto, eles interagem com o robô cinestesicamente, reunindo os estados do robô e os estados do mundo para alimentar um algoritmo de reconhecimento de habilidades semânticas. Este algoritmo consulta o modelo mundial em busca de relações entre os objetos, que são então usados para reconhecer as habilidades ensinadas em tempo real.

As abordagens baseadas em dados, por outro lado, dependem de conjuntos de dados para segmentar habilidades, o que pode ser alcançado por um modelo pré-treinado ou por outras técnicas de modelagem de dados, como *Support Vector Machines* (SVM) e *Hidden Markov Model* (HMM), respectivamente. Wang et al. (2018), implementam este tipo de solução introduzindo um novo *Multimodal Assembly Skill Decoding System* (MASD). Ele usa câmeras estéreo para adquirir informações multimodais e depois as processa para extrair informações de gestos e trajetórias, que são então usadas para treinar um classificador SVM.

Alguns trabalhos encontraram soluções combinando as duas abordagens para resolver cada uma de suas quedas. Eiband et. al. (2018), realizam segmentação usando pré e pós-condições para reconhecer habilidades apropriadas durante as demonstrações e então atribuem o rótulo simbólico correto. Após esta primeira etapa, repetem a segmentação utilizando uma máquina de vetores de suporte (SVM) para rotular o mesmo conjunto de dados. Finalmente, eles combinam essas duas segmentações resultantes em um conjunto de segmentos e o utilizam para criar a representação de saída.

Em resumo, as soluções simbólicas se esforçam para dar um significado semanticamente relevante aos estados do mundo real e robóticos, o que ajuda a representar as habilidades de uma maneira flexível e adaptável (RAMIREZ-AMARO , YANG , & CHENG , 2019). No entanto, estas técnicas precisam ser projetadas manualmente (PARK & AGGARWAL, emantic-level Understanding of Human Actions and Interactions using Event Hierarchy, 2004; PARK, KIM, & KEMP, 2019), e iriam contra o propósito de um sistema LfD se aplicadas a ele. Eles tornariam o aplicativo muito mais envolvente para o usuário final do que deveria.

Algumas abordagens baseadas em dados, entretanto, resolvem esse problema encontrando padrões nos dados, chegando até a propor métodos não supervisionados, como mostrado em Krishnan et al. (2017). Neste trabalho, eles criam uma ferramenta de segmentação de dados não supervisionada usando modelos hierárquicos de mistura gaussiana do processo Dirichlet (GMM). Esta técnica permite a segmentação mesmo que o número de segmentos não seja conhecido a priori.

O trabalho realizado em Niekum et al. (2012) atinge o mesmo fim. Aqui eles usam um modelo de Markov oculto autorregressivo de processo beta (BP-AR-HMM) para segmentar dados adquiridos cinestesicamente e, em seguida, usam-no para criar uma política Primitiva de Movimento Dinâmico (DMP). Depois que a política estiver pronta, eles a utilizam para guiar um manipulador móvel PR 2 e realizar uma tarefa de escrita em um quadro branco.

Nesta pesquisa, optamos por seguir a abordagem não supervisionada e orientada por dados para a segmentação de demonstração, uma vez que nos esforçamos principalmente para reduzir a carga de trabalho do usuário final.

2.2 BP-AR-HMM

2.2.1 Processos beta e processo de bernoulli para a estratificação de comportamentos

Um problema comum na literatura de análise de grandes séries temporais é a estratificação de características comuns presentes em subconjuntos dos *datasets*, trechos do *dataset* original que possuem comportamentos similares. Características muitas vezes são mais homogêneas entre si do que os *datasets* originais. A literatura apresenta muitas formas de resolver este problema, uma delas é a utilização de processos beta em conjunto com realizações do processo de bernoulli (BRODERICK, JORDAN , & PITMAN, 2011). De maneira intuitiva, uma coleção de K *features* binários são usados para descrever os dados e onde cada *feature* é modelado como uma variável aleatória de bernouli cujos parâmetros são obtidos a partir de um processo beta (BRODERICK, JORDAN , & PITMAN, 2011).

Os processos betas e o processo de bernoulli são uma classe especial dos processos estocásticos conhecidos como medidas completamente aleatórias (KINGMAN, 1967). Broderick, Jordan e Pitman (2011) definem uma medida completamente aleatória H como um

processo estocástico de medida aleatória que ocorre no espaço de probabilidade $(\Psi \times U)$ de tal forma que para qualquer disjunto $(\mathrm{A}_1, \ldots, \mathrm{A}_n) \in U$ as variáveis $\mathrm{H}(\mathrm{A}_1), \ldots, \mathrm{H}(\mathrm{A}_n)$ são independentes. A fim de facilitar o entendimento do conceito do processo beta, na **Figura 3** é exemplificada uma realização dele (BRODERICK, JORDAN , & PITMAN, 2011).

A **Figura 3** ilustra no espaço produtório $\Psi \times \mathbb{R}$ onde é a medida de Levy ν é representada pela superfície cinza. Define-se a medida de Levy ν que é definida na equação 1 (BRODERICK, JORDAN , & PITMAN, 2011).

Equação 1. Medida de Levy.

$$\nu(d\omega, d\theta) = c\omega^{-1}(1 - \omega)^{c-1} d\omega \mathrm{B}_0(d\theta)$$

Utilizando esta distribuição de probabilidade, realiza-se os átomos $\Pi = \{(\psi_i, U_i)\}_i$, onde o índice i pode assumir valores em uma inifidade contável. A medida aleatória, portanto, é definida como na **Equação 2** (FOX et al., 2014).

Equação 2. Descrição da realização do processo beta.

$$B = \sum_{i=1}^{\infty} U_i \, \delta_{\psi_i}$$

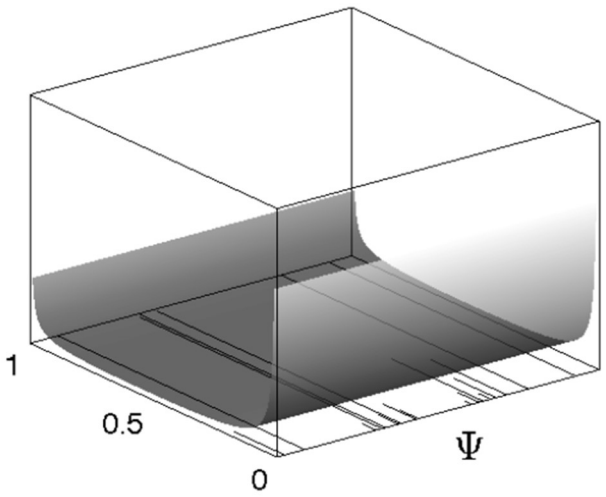

Figura 3. Densidade de probabilidade e realização de processo beta **(BRODERICK, JORDAN , & PITMAN, 2011)**.

Neste contexto, uma realização do processo beta é notada por B ~ BP(c, B0), onde c é um parâmetro de concentração e $B_0(\Psi)$ descreve a massa do espaço de probabilidade. Mais claramente, uma realização do processo Beta aqui descrito, tomaria a forma apresentada no gráfico da **Figura 4** (FOX et al., 2014).

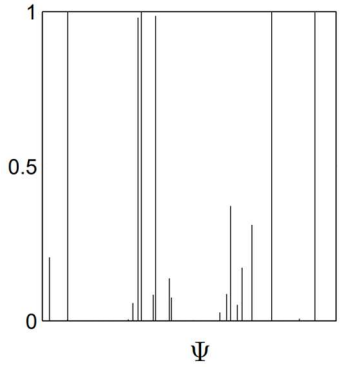

Figura 4. Exemplo de realização do processo beta **(BRODERICK, JORDAN , & PITMAN, 2011)**.

A partir deste ponto, introduz-se a utilização de um processo de Bernouli, que será denominado por BeP(B), este é conjugado ao processo beta de forma que suas realizações Xi ~ BeP(B) terão medida probabilística atômica pois as realizações do processo B ~ BP(c, B0) também o são. Sendo assim, o processo de Bernoulli terá realização como demonstrado na **Figura 5** (FOX et al., 2009). Aqui o eixo horizontal representa o espaço amostral e o eixo vertical demonstra subsequentes realizações do processo de Bernoulli, a partir disso é possível construir um vetor infinito de variáveis indicadoras binárias $f_i = [f_{i1}, f_{i2}, \ldots]$, onde $f_{ik} = 1$ se e somente se a uma série temporal i do *dataset* analisado possuir uma determinada característica k.

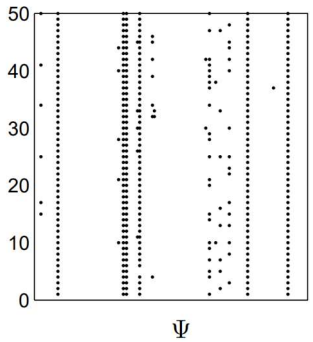

Figura 5. Exemplo de realização do processo de Bernoulli conjugado ao processo beta **(BRODERICK, JORDAN , & PITMAN, 2011)**.

Pode-se, portanto resumir estas realizações por um vetor infinito de variáveis indicadoras binárias fi = [fi1, fi2, fi3, ...] onde fik = 1 se e somente se uma determinada série temporal possuir a feature k (FOX et al., 2009). Esta técnica probabilística pode ser utilizada para inferir comportamentos de séries temporais em sistemas de aprendizado de máquina de maneira não-supervisionada.

2.2.2 Modelagem do BP-AR-HMM

O modelo básico do HMM assume que as séries temporais podem ser descritas por meio de transições de Markov entre um conjunto de comportamentos dinâmicos latentes individuais modelado via dinâmica linear temporalmente independente. No entanto, o HMM apresenta algumas limitações, como emissões condicionalmente independentes, dada a sequência de estados latentes (FOX et al., 2014).

Como resultado, o HMM não pode representar o movimento humano fluxos de dados e suas dependências temporais. No entanto, as emissões autorregressivas podem modelar dependências de estado e tempo, devido à sua observação de emissões passadas. Portanto, AR-HMM surge como uma aposta melhor solução para representar tarefas de manipulação (FOX et al., 2014).

Apesar de suas vantagens, o AR-HMM possui um número pré-definido de estados, que é determinado durante modelagem. Isto é bastante limitante, pois pressupõe que o usuário conhece o número de estados ou comportamentos a priori, o que é impraticável para aplicações LfD. Dito isso, o uso de um a prior não paramétrico bayesiano, permite ao modelo permanecer flexível ao comportamento do sistema. A representação gráfica na **Figura 6** mostra o resultante BP-AR-HMM.

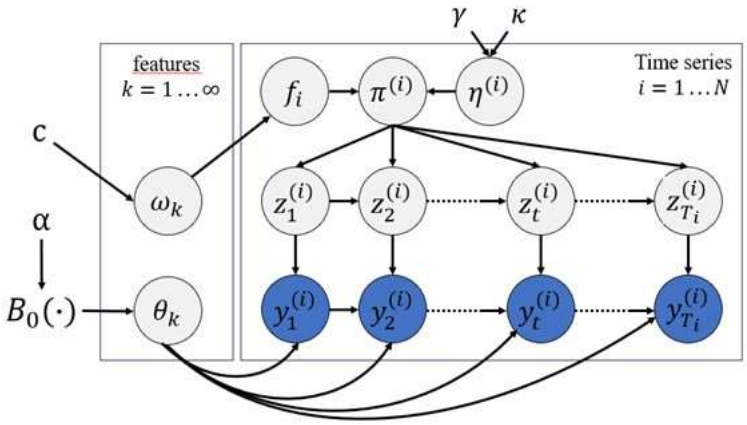

Figura 6. Representação gráfica do BP-AR-HMM.

As equações a seguir definem o funcionamento interno do modelo BP-AR-HMM:

Equação 3. Realização do processo de beta dada a medida base $\boldsymbol{B_0}$

$$B|B_0 \sim BP(1, B_0)$$

Equação 4. Descrição da realização do processo beta.

$$B = \sum_{i=1}^{\infty} U_i \, \delta_{\psi_i}$$

Equação 5. Realização do processo de Bernoulli com medida atômica dada pelo processo beta.

$$X_i|\mathrm{B} \sim \mathrm{BeP(B)}$$

Equação 6. Distribuição a priori da probabilidade de transição do modelo BP-AR-HMM.

$$\pi_j^{(i)}|\, f_i, \gamma, \kappa \sim Dir([\gamma, \ldots, \gamma + \kappa, \gamma, \ldots] \otimes f_i)$$

Equação 7. Distribuição de probabilidade de transição de estados do modelo BP-AR-HMM.

$$z_t^{(i)} \sim \pi^{(i)}_{z_{t-1}^{(i)}}$$

Equação 8. Distribuição de probabilidade de emissão do modelo BP-AR-HMM.

$$y_t^{(i)} = \sum_{j=1}^{r} A_{j,z_t^{(i)}} y_{t-j}^{(i)} + e_t^{(i)}\,(z_t^{(i)})$$

De acordo com a Equação 4. Descrição da realização do processo beta um sorteio do Processo Beta $BP(1, B_0)$ produz uma coleção de pontos $\{\psi_k, U_k\}$, onde ψ_k define a localização

dos átomos no espaço mensurável Ψ, sendo U_k seus respectivos pesos. A interpretação da localização ψ_k em nossa aplicação é uma coleção de características ou comportamentos que são compartilhados pela série temporal. Esses átomos definem a medida completamente aleatória B, que a Equação 5. Realização do processo de Bernoulli com medida atômica dada pelo processo beta utiliza para realizar as emissões X_i. Um vetor infinito $f_i = [f_{i1}, f_{i2}, \ldots]$ organiza as realizações de Xi de tal forma que $f_{ik} = 1$ se e somente se a série temporal i exibe o comportamento k .

A Equação 6. Distribuição a priori da probabilidade de transição do modelo BP-AR-HMM representa uma probabilidade de transição de estado anterior a priori $\pi_j^{(i)}$, que tem uma distribuição Dirichlet com viés de autotransição κ. Esta equação mostra que os estados nunca atingem características onde $f_{ik} = 0$ devido ao produto vetorial elemento a elemento, ou Hadamard, denotado por $\otimes$, portanto o comportamento não será exibido pela série temporal modelada.

Finalmente, a Eq. (6) define as emissões do modelo da série temporal i no tempo t, onde a matriz de defasagem $A_k = \{A_{1k}, \ldots, A_{rk}]$ relaciona a emissão do estado atual com as emissões passadas. Além disso, $e_t^{(i)}$ é um termo de ruído aditivo de distribuição normal. Neste contexto, r representa a ordem de autoregressividade do modelo. (NIEKUM et al., 2012); (FOX et al., 2014).

2.2.3 Cálculo do posterior (BP-AR-HMM-MCMC)

Fox et al. (2014) propõe um método de cálculo de distribuição posterior baseado no algoritmo de Markov Chain Monte Carlo (MCMC), que chama de BP-AR-HMM-MCMC. Este método incorpora as características de amostragem do algoritmo MCMC puro e adiciona outras que permitem a amostragem das variáveis do modelo BP-AR-HMM tais como o conjunto de *features* f_i e a sequência de estados $z_t^{(i)}$.

O algoritmo proposto por Fox et al. (2014) é mostrado na **Figura 7**. Ele modifica o estado atual do modelo BP-AR-HMM, modificando todas as suas variáveis. Ao finalizar a execução do algoritmo, o sistema finaliza uma iteração. O modelo da **Figura 7**, inclui o processo de divisão e mesclagem, que foi introduzido no trabalho supracitado, porém não incluiu-se no presente trabalho.

O algoritmo BP-AR-HMM-MCMC inicia iterando entre todas as N séries temporais i e entre todas as K_+ *features* k, onde K_+ simboliza o número total de *features* compartilhadas entre mais de uma série temporal (Linhas 1 e 2). É importante notar que este algoritmo considera sempre que mais de uma série temporal, com comportamentos (*features*) compartilhados serão fornecidos como input.

Segue-se então para a atribuição do vetor $m_k^{(-i)} = [m_1\, m_2\, m_3 \ldots\, m_{K_+}]$, um vetor que indica a quantidade de séries temporais onde cada feature k está presente, para toda série temporal diferente daquela sendo iterada e onde $f_{jk} = 1$ (linha 3). De posse dessa informação é realizada a amostragem do estado posterior dos *features* compartilhados, este processo é descrito pelo algoritmo no APÊNDICE A – Amostragem de *features* compartilhados (Linha 4). Em seguida são amostrados os estados posteriores da sequencia de estados $z^{(i)}$ e são amostrados os nascimentos e mortes de novos *features* (linhas 5 e 6).

O algoritmo agora realiza o processo de divisão e mesclagem (linhas 7 e 8), no entanto no presente trabalho esta parte do algoritmo foi desconsiderada. Em seguida são amostrados os estados posteriores dos parâmetros de emissão θ_k, os parâmetros de transição $\eta^{(i)}$, os hiper parâmetros do modelo do processo beta $\alpha\, e\, c$ e finalmente os hiper parâmetros do modelo HMM $\gamma\, e\, \kappa$ (linhas de 9 a 14).

Algorithm C.1 BP-AR-HMM-MCMC($\boldsymbol{y}, \Psi, \alpha, c, \gamma, \kappa, \lambda$)

Input:

$\boldsymbol{y}, \tilde{\boldsymbol{y}}$: N sequences of observed data (current and lagged observations)
$\Psi^{t-1} = (\boldsymbol{F}, \boldsymbol{z}, \theta, \eta, \alpha, c, \gamma, \kappa)$: input state of Markov chain (iter. $t-1$)
$\boldsymbol{F}$: N-by-K_+ binary feature matrix
$\boldsymbol{z}$: N discrete state sequences. Each $\boldsymbol{z}_t^{(i)} = k \in \{1, 2, \dots K_+\}$ s.t. $f_{ik} = 1$
$\theta = \theta_1, \dots \theta_k$: Emission parameters (one set per feature k)
$\eta = \eta^{(1)} \dots \eta^{(N)}$: Transition weights (one set per sequence i)
α, c : BP hyperparameters
γ, κ : HMM transition hyperparameters
$\lambda = (n_0, S_0, M, K)$: MNIW prior hyperparameters

Output:

$\Psi^t = (\boldsymbol{F}, \boldsymbol{z}, \theta, \eta, \alpha, c, \gamma, \kappa)$: next state of Markov chain (iter. t)

Procedure:

for sequence $i \in \{1, 2, \dots N\}$:
  for feature $k \in \{1, 2, \dots K_+\}$ owned by ≥ 1 other seq. ($\exists j \neq i$ s.t. $f_{jk} = 1$):
    $m_k^{(-i)} \leftarrow \sum_{i' \neq i} f_{i'k}$
    $f_{ik} \sim$ `SampleSharedFeature`($\boldsymbol{y}^{(i)}$, $\boldsymbol{f}_i$, $m_k^{(-i)}$, θ, $\eta^{(i)}$, (α, c)) (Alg. C.2)
  $\boldsymbol{z}^{(i)} \sim$ `SampleStateSequence`($\boldsymbol{y}^{(i)}$, $\boldsymbol{f}_i, \theta, \eta$) (Alg. C.3)
  $\boldsymbol{f}_i, \boldsymbol{z}^{(i)} \sim$ `SampleBirthDeath`($\boldsymbol{y}^{(i)}$, $\boldsymbol{f}_i$, $\boldsymbol{z}$, (α, c), (γ, κ), λ (Alg. D.1)
for trial $r = 1,2, \dots R$:
  $\boldsymbol{F}, \boldsymbol{z} \sim$ `SampleSplitMerge`($\boldsymbol{y}$, $\boldsymbol{F}, \boldsymbol{z}$, (α, c), (γ, κ), λ)
for feature $k \in \{1, 2, \dots K_+\}$:
  $\theta_k \sim$ `SampleEmissionParams`($\{\boldsymbol{y}_t^{(i)} : z_t^{(i)} = k\}$, λ) (Alg. C.4)
for sequence $i \in \{1, 2, \dots N\}$:
  $\eta^{(i)} \sim$ `SampleTransitionWeights`($\boldsymbol{z}^{(i)}$, (γ, κ)) (Alg. C.5)
$\alpha, c \sim$ `SampleBPHypers`($\boldsymbol{F}$, (α, c)) (Alg. F.1)
$\gamma, \kappa \sim$ `SampleHMMTransitionHypers`($\boldsymbol{z}$, (γ, κ)) (Alg. F.2)

Figura 7. Algoritmo BP-AR-HMM

3 Metodologia

Neste trabalho, propõe-se uma metodologia dividida em um total de 6 etapas, essas etapas são como segue:

1. Seleção do método
2. Preparação do ambiente de demonstrações
3. Implementação dos algoritmos
4. Análise dos resultados
5. Escrita da dissertação
6. Publicação dos resultados

A primeira e mais importante etapa da metodologia de pesquisa é a seleção dos métodos de coleta de dados e de segmentação. Como já foi esclarecido no capítulo de fundamentação teórica, a escolha destes métodos tem grande impacto nos tipos de problemas que deverão ser resolvidos durante o projeto. Por esse motivo, foram selecionadas algumas características desejadas para o sistema PfD que será implementado, levando-se sempre em consideração a hipótese a ser validada no trabalho.

A primeira característica diz respeito à facilidade de uso. Uma vez que a contextualização do problema a ser resolvido envolve montagem industrial, focando na transferência de habilidades manuais para agentes robóticos, é imprescindível que o método escolhido permita que o especialista demonstrando as habilidades permaneça com as mãos livres durante o processo. Segundo Hedlund e Johnson (2021), em aplicações de programação por demonstração, métodos que permitem uma abordagem de mãos livres apresentam uma carga de trabalho menor para o especialista e permitem que ele execute as habilidades de maneira mais confiante. Estas características são desejáveis para o contexto proposto, principalmente em relação a demonstradores que não sejam especialistas em robótica.

O segundo aspecto importante dos métodos escolhidos é que sejam não - supervisionados. A visão original do problema aqui definido é que pessoas sem treinamento possam demonstrar habilidades da maneira mais intuitiva e com a mínima experiência em

aplicações robóticas possível. Dessa forma, a aplicação de métodos não-supervisionados maximizará a facilidade do uso.

O terceiro e último pré-requisito para a seleção de um método de aprendizado é o baixo custo. O objetivo deste trabalho está intrinsecamente ligado ao desenvolvimento da indústria 4.0 e a acessibilidade destas tecnologias às SME. Por esse motivo, é crucial que o projeto seja conduzido com o baixo custo como prioridade, utilizando sempre que possível materiais de prateleira e tecnologias *open source.*

A segunda etapa da metodologia de pesquisa tem relação com a preparação do ambiente em que os testes de validação dos algoritmos serão realizados. Para este fim, deve ser preparada a bancada de trabalho, equipamento de captura de demonstrações e ambiente de desenvolvimento adequado para construção do software.

Uma bancada de trabalho simples foi selecionada para realizar as demonstrações, trata-se de uma escrivaninha de 130 cm de largura, 50 cm de comprimento e 76,5 cm de altura. Um tabuleiro de xadrez foi posicionado sobre a mesa, com as peças de xadrez dispostas no arranjo tradicional. A captura das demonstrações fica a cargo de uma câmera RGB de smartphone que foi posicionada alinhada à borda da mesa a uma distância de 60 cm da superfície da mesma de forma que as mãos do especialista estejam visíveis. Finalmente, o ambiente de desenvolvimento selecionado para a construção do software foi o Mathworks MATLAB®.

A implementação dos algoritmos, terceira fase da metodologia deste trabalho, inicia-se pelo desenvolvimento da interface de captura da posição das mãos do especialista. Para tal foi utilizada a biblioteca MediaPipe para anotar as imagens geradas pela câmera RGB. Esta etapa foi considerada completa uma vez que o sistema realizou a aquisição de uma série temporal de representações confiáveis dos movimentos das mãos do demonstrador. Em seguida, foi realizada a implementação do algoritmo de segmentação, que foi capaz de segmentar os movimentos gerados pelo ser humano utilizando o ambiente de desenvolvimento MATLAB®.

Durante a fase de análise de resultados, que é a quarta desta metodologia, as demonstrações foram primeiro anotadas manualmente, utilizando um conjunto de habilidades definidas arbitrariamente pelo autor a fim de servir de verdade absoluta para o cálculo das métricas de performance do algoritmo. Na sequência, foram comparados os resultados de segmentação das demonstrações para todos os cenários propostos com um resultado de verdade absoluta. A partir desta comparação foram geradas as métricas F1-score para acurácia de

alinhamento temporal e acurácia de classificação.

4 Solução proposta

Neste trabalho, busca-se uma solução de baixo custo e de simples implementação para a segmentação de demonstrações realizadas através do método de observação remota. Para tal, propõe-se um sistema que emprega a anotação automática de pontos chave dos frames gravados nas demonstrações alvo e que em seguida utiliza estas anotações para treinar um modelo BP-AR-HMM. Posteriormente, os estados do modelo são utilizados para distinguir as habilidades sendo demonstradas e, por consequência, as ações que as compõem e seus instantes de transição.

4.1 Estrutura

O sistema proposto é descrito na **Figura 8**, que também demonstra que tipo de informação é transmitida entre cada um dos estágios da metodologia. O sistema pressupõe um input de demonstração de habilidade manual em vídeo no formato MP4 a 30fps. Em seguida é realizada a anotação automática dos pontos relevantes das mãos, que são descritos na **Figura 9** (a). Este processo fica a cargo da API *mediapipe*, uma ferramenta *opensource* com modelos de aprendizagem de máquina pré-treinados no reconhecimento de objetos, anotação de membros humanos e reconhecimento de gestos.

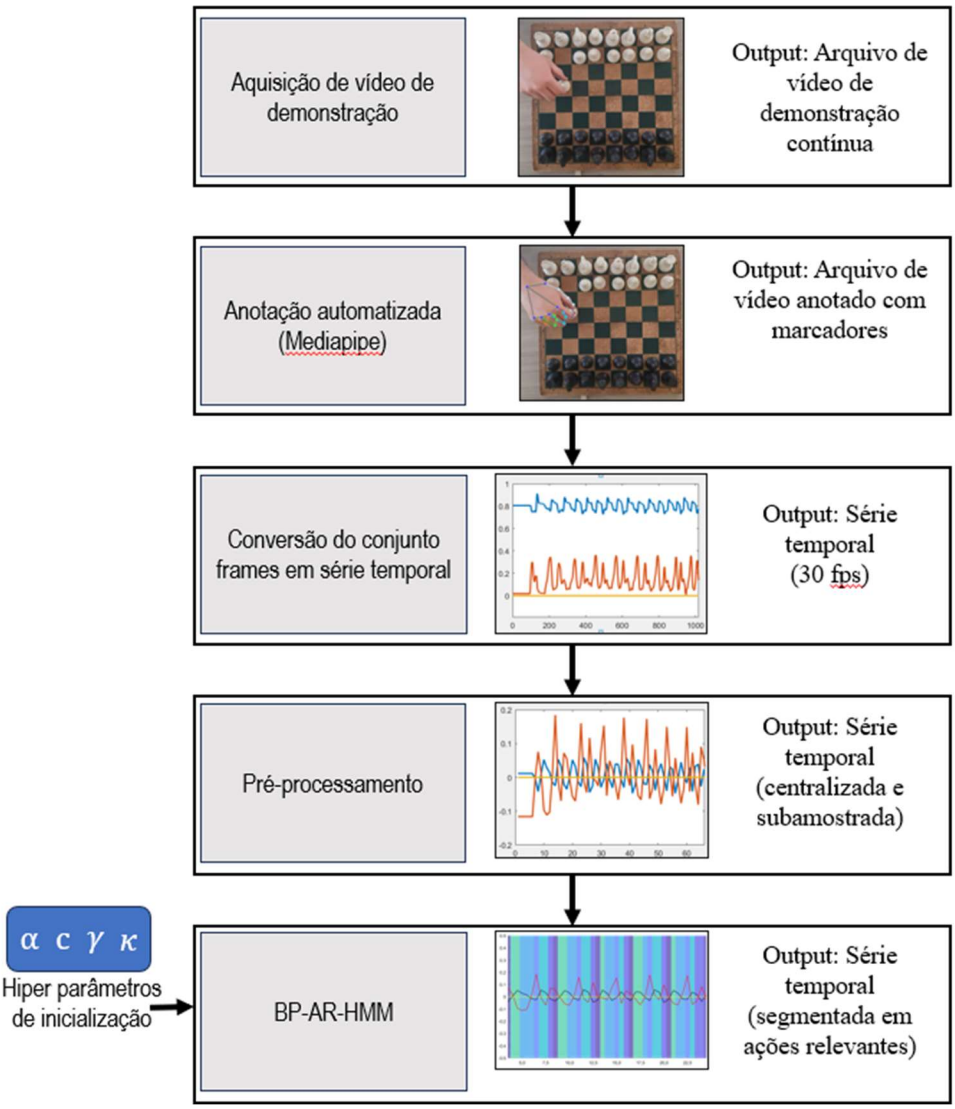

Figura 8. Estrutura da solução proposta.

Esta API, tem como saída um total de 20 pontos de anotação por mão detectada. Cada anotação recebe um vetor tridimensional, gerando o espaço de estados $p \in \mathbb{R}^{h\times 20\times 3}$, onde h representa o número de mãos que o sistema detectou em cada frame. Os valores das 3 dimensões correspondem aos eixos x, y e z.

Os eixos x e y são normalizados em [0.0 ; 1.0] em função da posição do marcador na imagem. Um valor 0 no eixo x, por exemplo, significa que o marcador se encontra à extrema esquerda da imagem anotada. Já o eixo y, ao assumir um valor 0, indica que o marcador se

encontra na região superior da imagem.

O eixo z, no entanto tem uma função diferente. Ele representa a profundidade que cada marcador está em relação à origem, que sempre é considerada no marcador do pulso. Por esse motivo, o marcador 0 sempre terá o valor do eixo z em 0.0

A **Figura 9**, apresenta a distribuição dos pontos detectados nas mãos e suas respectivas nomenclaturas. Um espaço de estados composto por um total de 4 marcadores foi selecionado para realização das segmentações automáticas das habilidades neste trabalho. Os marcadores 0, 4, 8 e 12 foram os escolhidos.

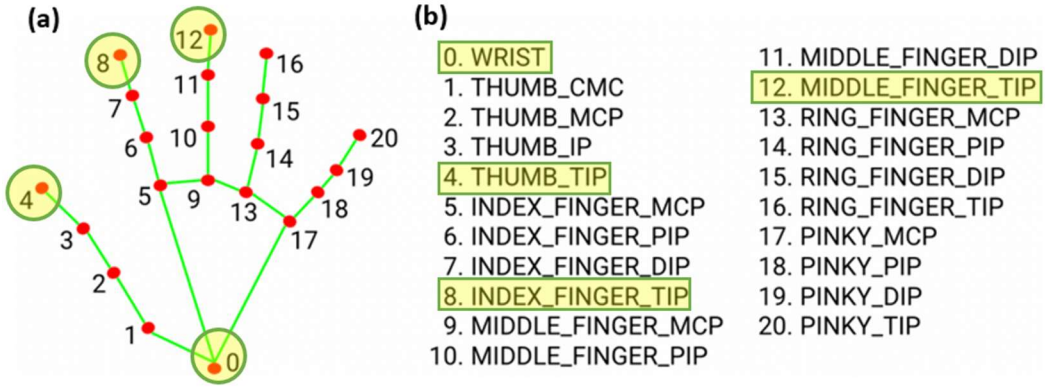

Figura 9. Marcadores gerados pela API Mediapipe (a). Nomenclatura dos marcadores gerados (b).

Repete-se este processo de anotação para todos os frames de uma determinada demonstração alvo e cada rótulo é rastreado ao longo do vídeo, permitindo a criação de uma série temporal $p(t)$. A utilização de séries temporais se faz pertinente neste contexto devido ao seu extenso interesse acadêmico e fácil disponibilidade de ferramentas de manipulação e modelos preditivos.

Após a geração da série temporal, ela é tratada em um estágio de pré-processamento, que realiza a redução da taxa amostral através da média em blocos com janelas não sobrepostas de 3 frames, efetivamente reduzindo a taxa amostral original de 30 fps para 10 fps. Este estágio também ajusta cada dimensão da série para que tenha média zero. A **Figura 10** mostra as 3 dimensões detectadas no marcador 0.

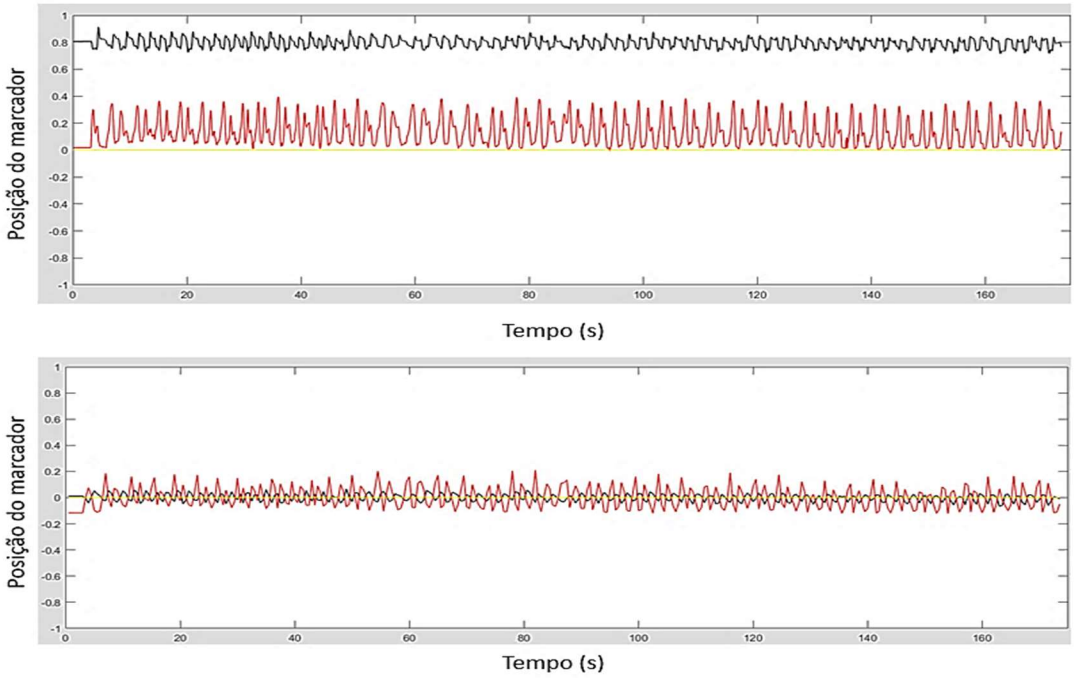

Figura 10. Séries temporais.

Seguimos então para o modelo BP-AR-HMM. Assim como qualquer outra cadeia oculta de Markov, esta é regida por distribuições de transição e de emissão, definidas como π e θ respectivamente. Porém, este modelo também utiliza vetor binário de comportamentos f_i, uma característica particular sua. Estas distribuições são inicializadas através dos parâmetros c, α, γ e κ, que definem os estados probabilísticos a priori. Uma vez iniciado, o modelo passa pelo algoritmo de *Markov Chain Monte Carlo* (MCMC), utilizado para realizar a amostragem posterior da distribuição probabilística.

Ao final de N iterações, o BP-AR-HMM já é capaz de modelar a série temporal, atribuindo a cada instante de tempo da série temporal um comportamento único, escolhido automaticamente pelo modelo a partir da lista de comportamentos f_i. Estes comportamentos então definem as ações realizadas pelo especialista e definem os instantes de transição de uma ação para outra.

4.2 Definição do problema

O principal objetivo da abordagem aqui proposta é segmentar automaticamente um vídeo de demonstração de tarefa de manipulação em conjuntos que contenham o mesmo tipo de ação representada. Com isso em mente, propõe-se a seguinte estrutura de segmentação.

Dado o vídeo de demonstração da tarefa de manipulação $v = \{h_1, \cdots, h_{N_v}\}$, onde h_i representa um frame dentro do vídeo e N_v a quantidade total de frames, Figura 11. Define-se um conjunto de ações f_{ki} como uma matriz binária representativa de K ações observadas no frame h_i. Considera-se que uma ação f_{ki} está presente no frame se e somente se $f_{ki} = 1$. Neste contexto, não é permitido que mais de uma ação esteja ativa para um determinado frame simultaneamente. Também vale ressaltar que dada a característica não supervisionada da abordagem, a quantidade de ações K também permanecerá variável.

Portanto, o problema consiste em otimizar f_{ki} e K a fim de maximizar as métricas de acurácia de segmentação e de classificação.

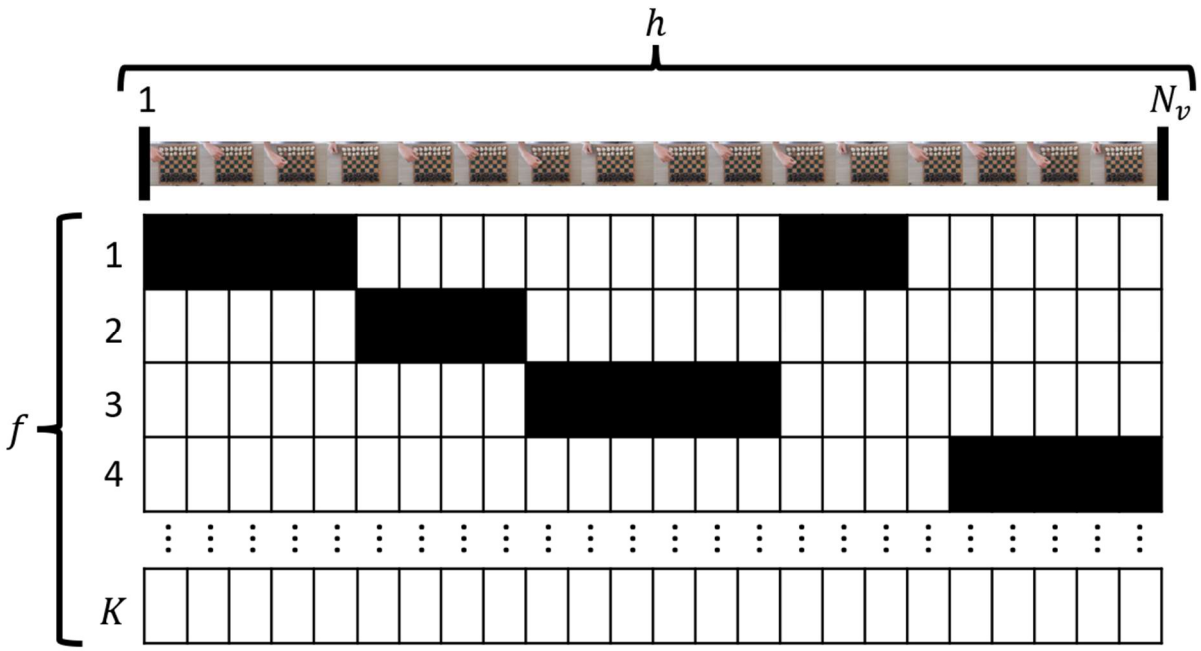

Figura 11. Representação gráfica da estrutura de segmentação automática.

4.3 Métricas de desempenho de segmentação

A fim de mensurar a performance do sistema proposto, serão utilizadas as métricas descritas por Eiband et al. (2023). O autor propõe duas, uma que mede o desvio temporal dos segmentos atribuídos pelo sistema e outra que mede a precisão das classificações. Ambas são obtidas através da comparação entre o resultado de classificação gerado pelo algoritmo e uma segmentação de referência, que se obtém através da anotação manual por um ser humano. Este procedimento será repetido para cada um dos experimentos propostos.

A primeira métrica, mede o desvio temporal dos segmentos e será referida = trabalho como acurácia de segmentação. Ela basicamente indica o quão longe as transições de comportamento estimadas estão da verdade absoluta no tempo. Essa métrica é obtida através do cálculo do F1-score da seguinte forma.

Primeiramente, atribui-se uma região definida por $\pm t_{err}$ ao redor das transições entre os comportamentos da segmentação de verdade absoluta. Em seguida, é avaliado se a transição gerada pelo algoritmo está dentro ou fora de $\pm t_{err}$. Se a transição automática ocorrer dentro de t_{err} considera-se uma ocorrência de verdadeiro positivo (*True Positive* – TP), se ocorrer fora da região considera-se um falso positivo (*False positive* – FP) e finalmente se a transição automática ocorrer apenas na segmentação de verdade absoluta e não for gerada pela segmentação automática, considera-se um falso negativo (*False negative* – FN). A **Figura 12** mostra um exemplo ilustrativo desta avaliação. Neste trabalho será utilizado o mesmo valor proposto por Eiband et al. (2023) onde $t_{err} = 0{,}2s$

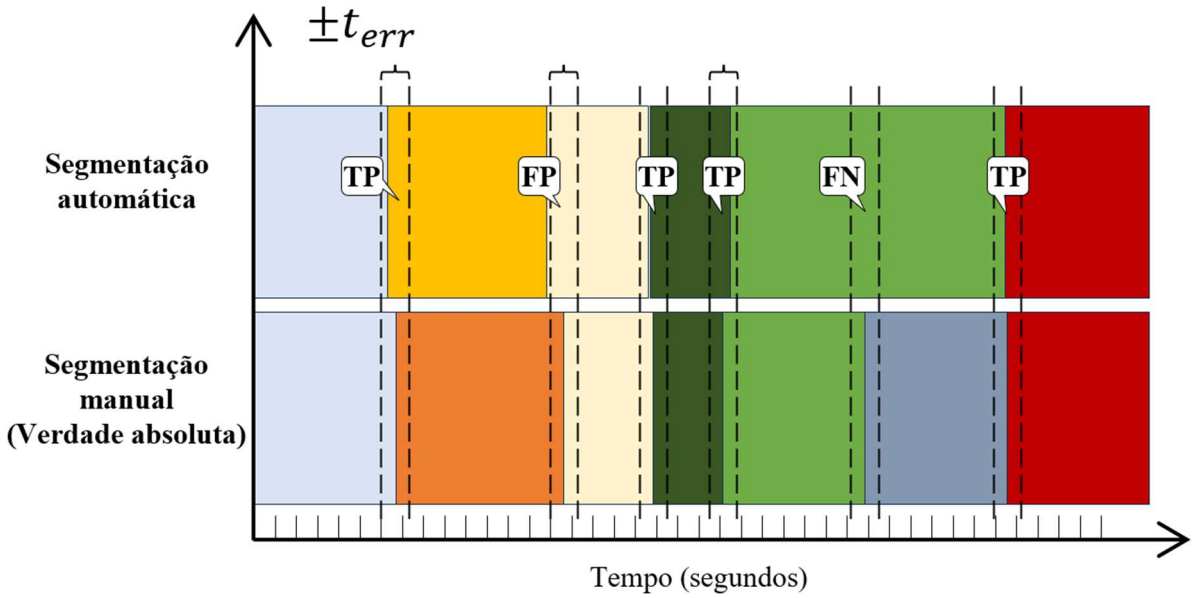

Figura 12. Exemplo ilustrativo da métrica de desempenho de segmentação.

A partir disso, o cálculo do F1-score será feito através da **Equação 9**.

Equação 9. Métrica de acurácia de segmentação.

$$F1 = \frac{2.TP}{2.TP + FN + FP}$$

É importante ressaltar que esta primeira métrica não é dependente da corretude das classificações feitas pelo algoritmo, apenas o seu alinhamento temporal. A avaliação da acuracidade das classificações fica a cargo da segunda métrica, que é calculada da **Equação 10**.

Equação 10. Métrica de acurácia de classificação.

$$C = \frac{N_{classificações\_corretas}}{N_{classificações\_totais}}$$

Esta segunda métrica será referida neste trabalho como acurácia de classificação. Para aferir as duas variáveis $N_{classificações\_corretas}$ e $N_{classificações\_totais}$, considera-se se uma classificação foi feita corretamente a cada ponto da série temporal. Sendo assim, já que as séries temporais foram pré-processadas para reduzir a taxa amostral para 10 fps, a verificação de classificação é feita a cada 0,1s, como mostra a **Figura 13**.

Tempo de demonstração (s)	Anotação manual (Verdade absoluta)	Anotação automatizada	Resultado de classificação
0,1	Mover mão para peão	Mover mão para peão	Correta
0,2	Mover mão para peão	Mover mão para peão	Correta
0,3	Mover mão para peão	Mover mão para peão	Correta
0,4	Mover mão para peão	Mover mão para peão	Correta
0,5	Mover mão para peão	Mover mão para peão	Correta
0,6	Mover mão para peão	Mover mão para peão	Correta
0,7	Mover mão para peão	Mover mão para peão	Correta
0,8	Mover peão	Mover mão para peão	Incorreta
0,9	Mover peão	Mover mão para peão	Incorreta
1	Mover peão	Mover cavalo	Incorreta
1,1	Mover peão	Mover cavalo	Incorreta
1,2	Repousar mão	Mover cavalo	Incorreta
1,3	Repousar mão	Repousar mão	Correta
1,4	Repousar mão	Repousar mão	Correta
1,5	Repousar mão	Repousar mão	Correta
1,6	Repousar mão	Repousar mão	Correta

Figura 13. Exemplo ilustrativo da métrica de acurácia de classificação

4.4 Setup e tarefas demonstradas

Os experimentos serão realizados em 2 cenários, utilizando um acervo de

demonstrações criadas pelo autor. O setup escolhido para a realização destas demonstrações é composto por uma simples câmera de smartphone, que grava as ações de manipulação do especialista em vídeo RGB no formato .mp4, a 30 fps e com uma resolução de 720p.

As demonstrações são todas realizadas sobre um tabuleiro de xadrez. As movimentações de peças são completamente visíveis e uma margem de aproximadamente 20cm ao redor do tabuleiro também é inclusa no frame, a fim de detectar os movimentos realizados ao redor da área de atuação.

Também vale ressaltar que aqui o jogo de xadrez é utilizado como um substrato para demonstração de habilidades manuais, alguns movimentos que não são permitidos dentro das regras do jogo serão executados a fim de retornar o estado original do jogo e rapidamente realizar uma nova demonstração. Como de fato é realizado no contexto de LfD.

4.4.1 Cenário 1 – Movimentação do peão

Inicialmente, deseja-se verificar se o sistema proposto é capaz de segmentar satisfatoriamente demonstrações nas condições mais simples possíveis. Desta forma, define-se o seguinte experimento para averiguar esta condição.

Em um tabuleiro de xadrez com todas as peças dispostas na arrumação padrão para qualquer jogo, o especialista descansa a mão direita ao lado do tabuleiro. A partir disso, move sua mão direita e pega o peão da posição g2 e o avança em direção a posição g4, onde deixa o peão e posiciona a mão para descansá-la novamente. Após esse primeiro passo, o especialista ergue a mão novamente, pega o peão na posição g4 e o recua a posição g2, retornando-o a sua posição original e descansando a mão novamente. Este ciclo é repetido durante aproximadamente 3 minutos, o que resulta em 42 repetições da ação demonstrada.

A **Figura 14** mostra detalhadamente cada um dos 6 movimentos que o professor deve executar neste experimento.

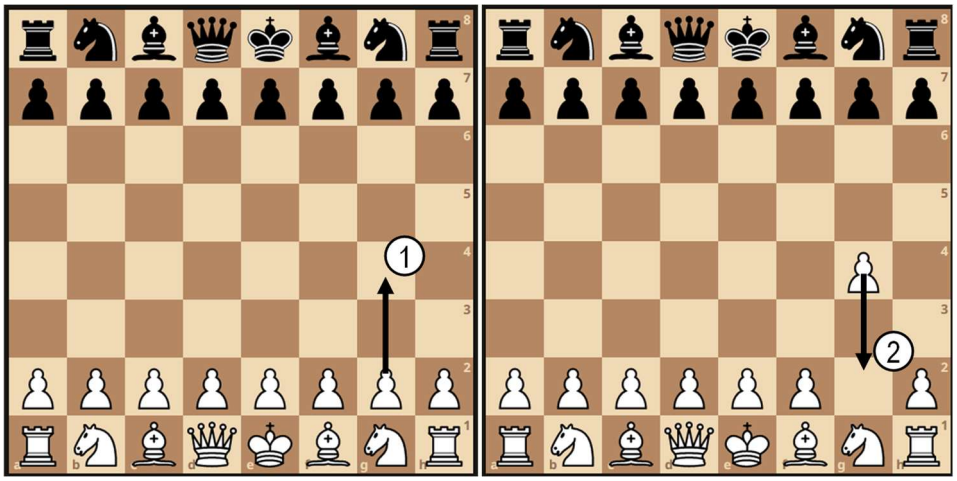

Figura 14. Movimentos realizados no cenário 1.

4.4.2 Cenário 2 – Jogada Ruy Lopes

No cenário 2 propõe-se um experimento com uma gama maior de movimentações. Neste cenário, o especialista realiza os movimentos da clássica abertura de xadrez Ruy Lopez, descrita na **Figura 15**. Ele avança o peão da posição E2 até a posição E4, em seguida move o cavalo da posição G1 até a posição F3 e por fim, move o bispo da posição F1 até a posição B5. Após a realização destes movimentos, o especialista retorna as peças a suas posições originais, uma de cada vez. Durante toda a demonstração, o especialista deve descansar sua mão entre cada uma das atividades.

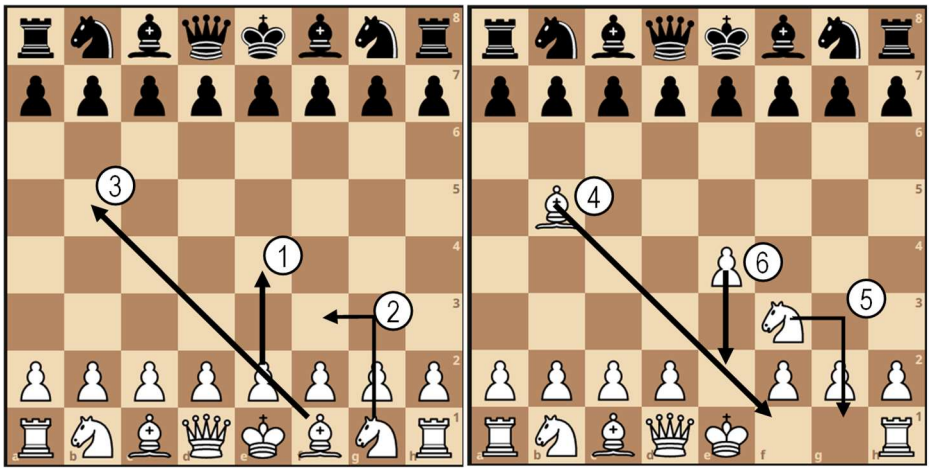

Figura 15. Movimentos realizados no cenário 2.

5 Resultados

Esta sessão descreve os experimentos realizados, resultados obtidos utilizando as métricas indicadas.

5.1 Cenário 1 – Movimentação do peão

De acordo com o que foi previamente descrito na sessão de solução proposta, neste primeiro experimento o especialista realiza o movimento de um simples avanço e recuo de um peão. Essa habilidade foi segmentada manualmente e classificada em 6 ações distintas com o objetivo de gerar a referência de verdade absoluta, que será posteriormente utilizada para mensurar a performance do segmentador proposto. A segmentação manual (**Figura 16 e Figura 17**) foi realizada em um total de 60 segundos de vídeo das demonstrações propostas e é importante notar que em nenhum momento os rótulos pré-definidos foram transmitidos ao sistema segmentador, sendo apenas utilizado para aferir as métricas após a classificação.

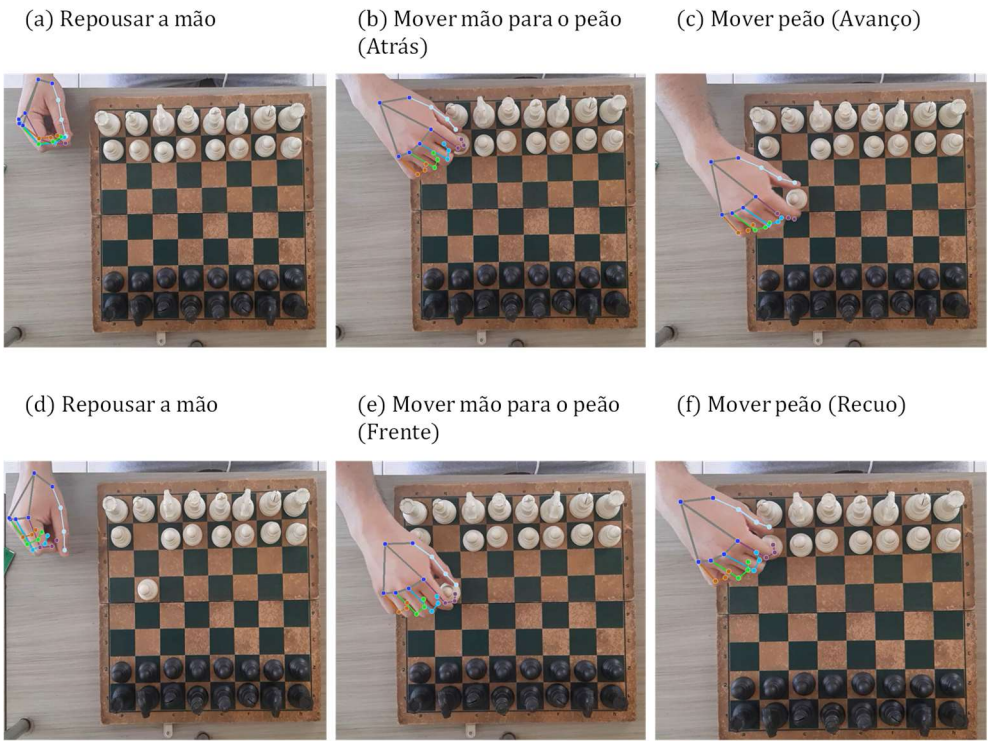

Figura 16. Rótulos definidos para segmentação manual – Cenário 1.

A **Figura 16** mostra os rótulos selecionados manualmente para estas demonstrações. A partir da análise manual do vídeo foi gerado um mapa de segmentação de verdade absoluta, representado na **Figura 17**. No mapa foram indicadas cada uma das ocorrências do ciclo de ações mencionado, totalizando 15 demonstrações da habilidade. Estes ciclos têm duração média de 3,75 segundos, com o maior ciclo chegado a 5,0 segundos e o menor a 2,9 segundos.

É importante notar aqui que no estado inicial da demonstração o peão já se encontra em posição de avanço e a mão do especialista já se encontra levantada. Por esse motivo na **Figura 17** a primeira ação realizada é justamente a movimentação da mão para o peão na posição avançada, representado pela cor verde.

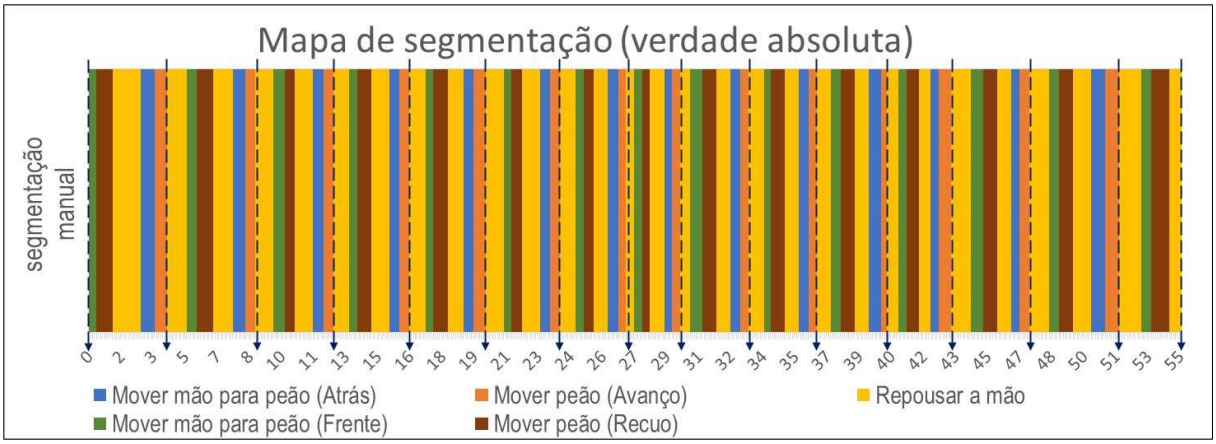

Figura 17. Mapa de segmentação manual – Verdade absoluta. Cenário 1.

5.1.1 Segmentação automática com inicialização de 5 *features*

O experimento realizado com a inicialização de 5 *features* gerou um total de 6 estados. Estes foram comparados com as ações manualmente definidas na classificação de verdade absoluta e, a partir de seu comportamento, tiveram um significado semântico atribuído. Com estes resultados foi montada a matriz de confusão da **Tabela 1.**

Tabela 1. Matriz de confusão - 5 *features*. Cenário 1.

		Classificação real (Verdade absoluta)				
		Mover mão para peão (Atrás)	Mover peão (Avanço)	Repousar a mão	Mover mão para peão (Frente)	Mover peão (Recuo)
Classificação gerada						
Estados gerados	Significado semântico					
-	Mover mão para peão (Atrás)	0	0	0	0	0
-	Mover peão (Avanço)	0	0	0	0	0
1; 3; 4	Repousar a mão	4	10	229	5	6
2; 6	Mover mão para peão (Frente)	49	0	11	61	6
5	Mover peão (Recuo)	27	66	4	3	87

É importante ressaltar que os significados semânticos não são gerados pelo sistema, eles são atribuídos na fase de análise dos dados. Neste trabalho, atribuiu-se a um dado estado o

significado semântico da ação em que o mesmo ocorre com mais frequência. Por exemplo, o estado “1”, ocorreu 75 vezes sob a ação “repousar a mão” aparecendo ocasionalmente durante a execução de outras ações, sendo assim, “repousar a mão” foi atribuído como o significado semântico do estado “1”.

Ao observar a matriz de confusão, percebe-se que nem todas as ações previstas pela segmentação manual foram descritas corretamente pelo sistema proposto. Apenas 3 ações do conjunto de 6 definidas para a classificação manual tiveram estados que as representam semanticamente. Como consequência, algumas ações foram sub-representação no modelo, fazendo com que o resultado de acurácia de classificação seja afetado negativamente.

A acurácia de classificação teve como resultado o valor máximo 67,5% e valor mínimo de 58,9%, com uma variação 8,6 pontos percentuais entre o melhor e pior seeds, o que são resultados péssimos para um classificador. Comparativamente, os experimentos de Vögele, Krüger e Klein (2017), por exemplo, mostram resultados de acurácia de classificação melhores do que aqueles aqui apresentados, atingindo 88% de acurácia de classificação. Vögele, Krüger e Klein (2017) realizaram em seu trabalho a segmentação de uma sequência de dados obtidos a partir de sensores de captura de movimento que demonstram ações de corpo todo, como “caminhar”, “socar”, “correr” e agachar.

Wang et al. (2018), por sua vez, realizou a segmentação de um *dataset* de demonstrações manuais muito similares ao *dataset* utilizado neste trabalho, composto de vídeos RGB de demonstrações simples. O autor obteve uma acurácia de classificação global de 85,7%. Wang et al. (2018) propôs um modelo próprio para segmentação de ações baseado em Support Vector Machines e reconhecimento de *features* multimodais para realizar sua segmentação.

Analisando de maneira detalhada, a

Tabela 2 mostra a acurácia de classificação por ação, deixando clara a não representatividade das ações “Mover mão para peão (Atrás)” e “Mover peão (Avanço)”.

Tabela 2. Acurácia de classificação por ação demonstrada - 5 *features*. Cenário 1.

Ação	Acurácia de classificação	Estados gerados
Mover mão para peão (Atrás)	0,0%	-
Mover peão (Avanço)	0,0%	-
Repousar a mão	93,9%	1; 3; 4
Mover mão para peão (Frente)	88,4%	2; 6
Mover peão (Recuo)	87,9%	5

Na **Tabela 3** são apresentados os resultados de acurácia por estado gerado. O estado 2, por exemplo, representa majoritariamente a ação "Mover mão para peão (Frente)", sendo alvo de 47,6% de suas instâncias, no entanto, 38,9% das ocorrências deste estado foram classificadas como "Mover mão para peão (Atrás)", o que mostra uma ambiguidade de classificação. O mesmo comportamento está presente no estado 5, que tem 46,5% das classificações alocadas na ação "Mover peão (Recuo)" ao mesmo tempo que 35,3% das vezes foi classificado como "Mover peão (Avanço)".

Tabela 3. Acurácia de classificação por estado gerado - 5 *features*. Cenário 1.

Estados gerados	Significado semântico	Acurácia de classificação
1	Repousar a mão	81,5%
3	Repousar a mão	94,5%
4	Repousar a mão	100,0%
2	Mover mão para peão (Frente)	47,6%
6	Mover mão para peão (Frente)	100,0%
5	Mover peão (Recuo)	46,5%

A partir dos estados gerados para cada instante da demonstração, foi construído o mapa de segmentação da **Figura 18**. Calculando-se a métrica de alinhamento temporal, que corresponde a um F1-score máximo de 93,8% e mínimo de 75,3%, com uma variação de 18,5 pontos percentuais entre o melhor e pior seeds, **Figura 19**. O resultado aqui obtido é superior ao

obtido por Eiband et al., (2023) em 19 dos 25 seeds gerados, em seu resultado de alinhamento temporal, Eiband et al., (2023) obteve F1-score de 90,0% utilizando $t_{err} = 0{,}2s$

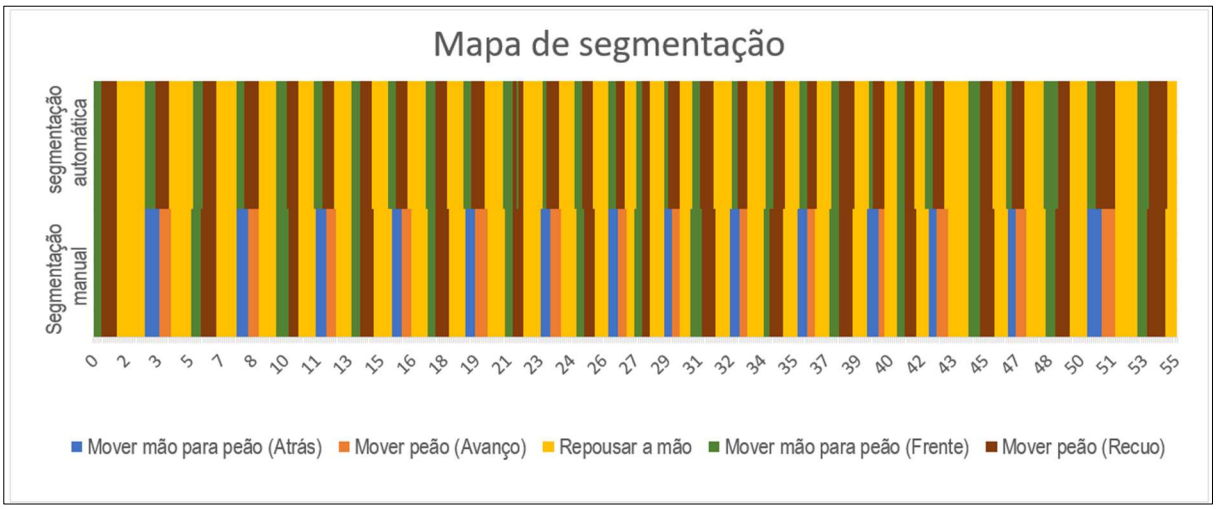

Figura 18. Mapa de segmentação automática do cenário 1 - 5 *features*. Cenário 1.

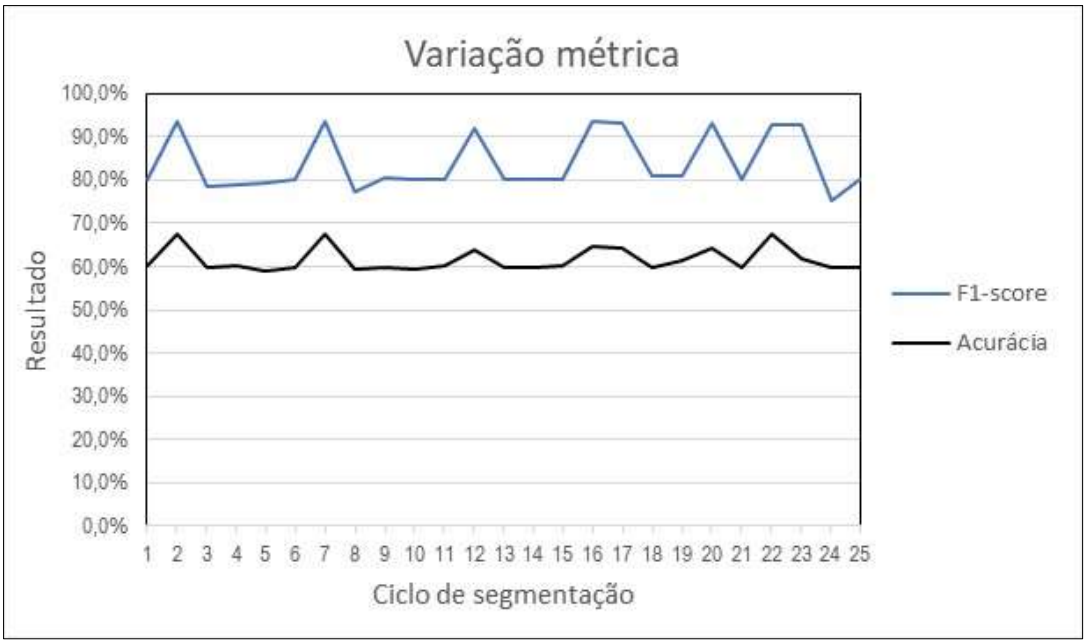

Figura 19. Variação métrica com inicialização de - 5 *features*. Cenário 1.

Eiband et al., (2023) aplicou um método inovador com processamento *On-line*, baseado em uma combinação de métodos semânticos e métodos focados em dados. Em seu trabalho, realizou a segmentação de uma tarefa de aplicação de carimbo, onde obteve o resultado supracitado, e uma tarefa de montagem, onde a acurácia de segmentação temporal atingiu 84%.

A atividade descrita por Eiband et al. (2023) que gerou estes resultados tem características muito similares àquela apresentada no cenário aqui discutido. No entanto, vale ressaltar que o trabalho de Eiband et al. (2023) entra em um contexto de demonstrações cinestésicas, onde erros provindos de sensores são baixos ou nulos, diferente do presente trabalho onde os ruídos gerados pela captura de dados são sempre presentes.

Comparativamente, os resultados de alinhamento temporal de Lin e Kulić (2014) atingiram um resultado de 77%, usando um valor de $t_{err} = 0{,}2s$, e 87% utilizando $t_{err} = 0{,}3s$. Lin e Kulić (2014) inseriu seu trabalho em um contexto de segmentação automatizada de exercícios de reabilitação de pacientes em fisioterapia. Para atingir seu fim, utilizou dispositivos de captura inerciais montados aos corpos dos pacientes como input em seu sistema. Em seguida, modelou os movimentos utilizando características da velocidade, como valor de pico e instante de cruzamento zero para treinar um modelo HMM e consequentemente segmentar as demonstrações utilizando o algoritmo de maximização de expectativa de Baum-Welch.

A fim de averiguar a repetibilidade dos resultados aqui apresentados, o algoritmo MCMC foi executado um total de 25 vezes registrando os resultados de alinhamento temporal e acuracidade de classificação na **Figura 19**.

5.1.2 Segmentação automática com inicialização de 10 *features*

O experimento de segmentação do cenário 1 foi repetido, desta vez iniciando o modelo com 10 *features* a priori.

Em relação à capacidade de classificação desta nova configuração, a confusão presente anteriormente entre as classificações “Mover mão para peão (Atrás)” e “Mover mão para peão (Frente)” já não está mais presente. Isso é evidenciado na matriz de confusão,

Tabela 4, que mostra que o sistema gerou um estado com significado semântico “Mover mão para peão (Atrás)”, que não estava presente na inicialização com 5 *features*. Essa melhora também se reflete na acurácia de classificação, que atinge agora um resultado máximo de 76,5%, e mínimo de 65,1%, evidenciando que os resultados nesta configuração têm variação de 11,4 pontos percentuais.

É importante notar aqui que apesar desta melhoria, o sistema também gerou novos

estados redundantes para “repousar a mão”, um total de 5 estados foram gerados para esta ação.

Tabela 4. Matriz de confusão - 10 *features*. Cenário 1.

		Classificação real (Verdade absoluta)				
Classificação gerada		Mover mão para peão (Atrás)	Mover peão (Avanço)	Repousar a mão	Mover mão para peão (Frente)	Mover peão (Recuo)
Estados gerados	Significado semântico					
4	Mover mão para peão (Atrás)	50	0	2	0	0
-	Mover peão (Avanço)	0	0	0	0	0
2; 1; 5; 9; 6	Repousar a mão	7	10	235	6	5
3	Mover mão para peão (Frente)	0	0	3	60	8
7; 8	Mover peão (Recuo)	23	66	4	3	86

Fica clara também uma melhoria nos resultados de acurácia por ação, houve melhora significativa na acurácia da ação “Mover mão para peão (Atrás)”, que saiu de um a situação de não representatividade para 62,5% de acurácia. Isso ocorre pois com o aumento de *features* de inicialização de 5 para 10, um *feature* foi alocado para os frames desta ação, **Tabela 5**. Além disso, também há melhoria na acurácia por estado,

Tabela 6, o estado 3, em particular, teve melhoria significativa, chegando à acurácia de 85,0%.

Tabela 5. Acurácia de classificação por ação demonstrada - 10 *features*. Cenário 1.

Ação	Acurácia de classificação	Estados gerados
Mover mão para peão (Atrás)	62,5%	4
Mover peão (Avanço)	0,0%	-
Repousar a mão	96,3%	2; 1; 5; 9; 6
Mover mão para peão (Frente)	87,0%	3
Mover peão (Recuo)	86,9%	7; 8

Tabela 6. Acurácia de classificação por estado gerado - 10 *features*. Cenário 1.

Estados gerados	Significado semântico	Acurácia de classificação
4	Mover mão para peão (Atrás)	96%
2	Repousar a mão	93%
1	Repousar a mão	81%
5	Repousar a mão	100%
9	Repousar a mão	86%
6	Repousar a mão	67%
3	Mover mão para peão (Frente)	85%
7	Mover peão (Recuo)	47%
8	Mover peão (Recuo)	100%

O aumento no número de *features* de inicialização (**Figura 20**) também proporcionou uma melhora nos resultados de alinhamento temporal. Com o sistema atingindo um F1-score de alinhamento temporal máximo de 96,6% e valor mínimo de 84,2%, **Figura 21**, houve melhoria em relação ao resultado com 5 *features* anterior. Além disso, este resultado também se manteve mais estável do que o anterior ao longo de vários seeds de inicialização, representando uma variação de apenas 12,4 pontos percentuais.

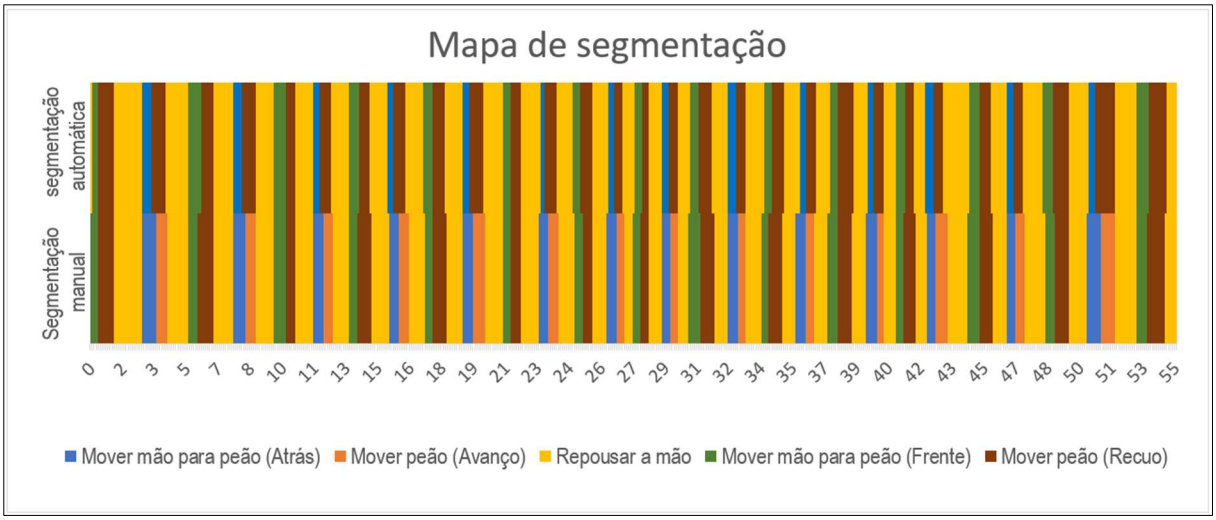

Figura 20. Mapa de segmentação automática de 10 *features*. Cenário 1.

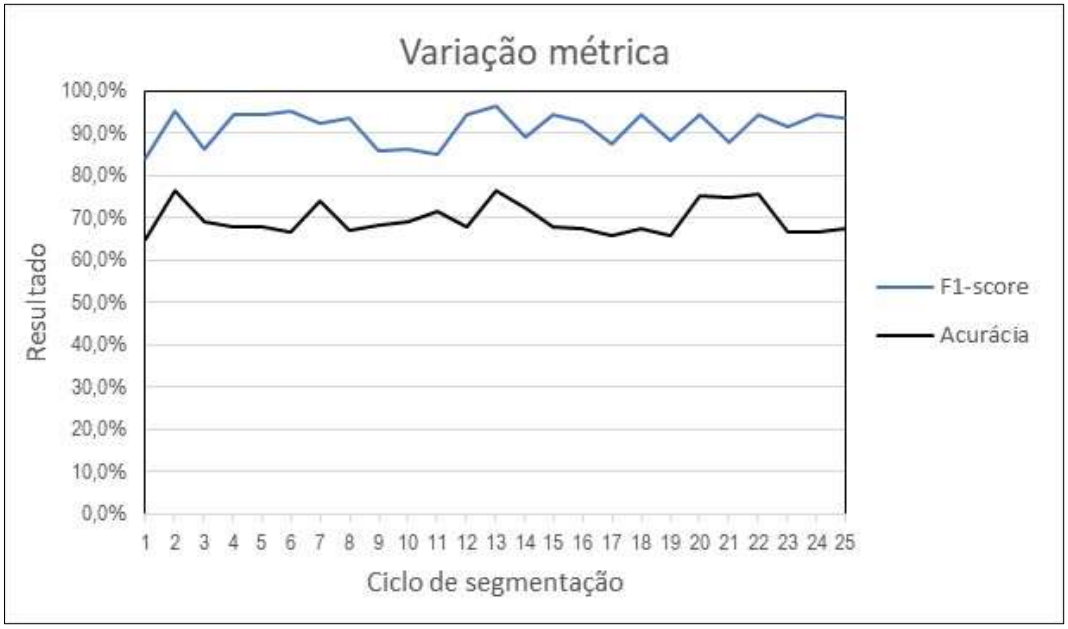

Figura 21. Variação métrica com inicialização de 10 *features*. Cenário 1.

5.1.3 Segmentação automática com inicialização de 20 *features*

O experimento de segmentação tem continuação com a inicialização de 20 *features* nesta nova configuração a priori.

De imediato, é possível notar a diferença entre a matriz de confusão deste experimento utilizando 20 *features* para inicialização (

Tabela 7), já que não existem ações sem pelo menos 1 estado gerado que a represente semanticamente. Em contrapartida, o aumento na redundância dos estados agora é visível, a ação "repousar a mão" é definida por um total de 6 estados, 3 a mais do que no experimento original com 5 *features*. Como consequência, temos um resultado de acurácia de classificação melhor do que as anteriores, sendo o melhor 86,8% e o pior 74,3% dentre os 25 seeds, ocasionando a variação de 12,5 pontos percentuais. Desta vez, alguns seeds foram capazes de superar os resultados de Wang et al. (2018), porém os resultados de Vögele, Krüger e Klein (2017) ainda se sobressaem.

Tabela 7. Matriz de confusão - 20 *features*. Cenário 1.

		Classificação real (Verdade absoluta)				
		Mover mão para peão (Atrás)	Mover peão (Avanço)	Repousar a mão	Mover mão para peão (Frente)	Mover peão (Recuo)
Classificação gerada						
Estados gerados	Significado semântico					
10	Mover mão para peão (Atrás)	46	0	2	0	0
12	Mover peão (Avanço)	27	66	1	0	0
4; 1; 7; 11; 8; 2; 3	Repousar a mão	7	10	235	5	6
6; 13	Mover mão para peão (Frente)	0	0	3	63	11
5	Mover peão (Recuo)	0	0	3	1	82

Analisando os resultados de acuracidade por ação, estes não apresentam nenhuma ação sem representatividade. A ação com o pior resultado foi "Mover mão para peão (Atrás)", com 57,5% e a melhor foi "Repousar a mão" com 96,3%, **Tabela 8**. Em contrapartida, a ação

“Repousar a mão” também apresentou a maior redundância de estados, um total de 7 estados são utilizados para a representar.

Tabela 8. Acurácia de classificação por ação demonstrada - 20 *features*. Cenário 1.

Ação	Acurácia de classificação	Estados gerados
Mover mão para peão (Atrás)	57,5%	10
Mover peão (Avanço)	86,8%	12
Repousar a mão	96,3%	4; 1; 7; 11; 8; 2; 3
Mover mão para peão (Frente)	91,3%	6; 13
Mover peão (Recuo)	82,8%	5

Tabela 9. Acurácia de classificação por estado gerado – 20 *features*. Cenário 1.

Estados gerados	Significado semântico	Acurácia de classificação
10	Mover mão para peão (Atrás)	96%
12	Mover peão (Avanço)	70%
4	Repousar a mão	92%
1	Repousar a mão	93%
7	Repousar a mão	100%
11	Repousar a mão	58%
8	Repousar a mão	100%
2	Repousar a mão	100%
3	Repousar a mão	100%
6	Mover mão para peão (Frente)	82%
13	Mover mão para peão (Frente)	100%
5	Mover peão (Recuo)	95%

Neste estado da análise, é interessante comentar que a redundância dos estados que são semanticamente representados pela ação “Repousar a mão” são uma consequência do fato de que o sistema gera os estados sem seguir diretrizes pré-estabelecidas de quantidade de ações ou número de segmentos na demonstração. Isso ocorre pois aqui aplicou-se um método não-

supervisionado que agrupa segmentos com o mesmo comportamento multidimensional, portanto, faz-se relevante realizar um estudo do significado latente dos estados redundantes.

Uma possível explicação para o crescimento na redundância dos estados é que o sistema pode estar subdividindo as ações em micro ações que também têm seu próprio significado semântico, porém não fazem parte do conjunto definido pelo ser humano que realizou a segmentação de verdade absoluta. Isso indica claramente que o sistema proposto é capaz de identificar e segmentar de maneira autônoma e não-supervisionada com variados níveis de precisão dependendo da quantidade de *features* inicializados. De fato, analisando em detalhes os estados usados na execução da ação "repousar a mão" foram identificados alguns padrões, ao exemplo da **Figura 22**.

As subtarefas descritas nas **Figura 22** mostram que a ação "Repousar a mão" foi dividida de maneira autônoma em 2 padrões descritos por 4 estados básicos cada um, os estados 1, 4 ,7 e 11. A primeira configuração, descrita na **Figura 22 (a)**, ocorre um total de 15 vezes dentro dos 60 segundos de vídeo analisado. Já a segunda configuração de estados, **Figura 22 (b)** ocorre 11 vezes dentro do vídeo. Deve-se atentar também que dentro deste período de tempo a ação "Repousar a mão" foi executada 30 vezes.

De posse destes dados, constata-se que 26 das 30 vezes que a ação "Repousar a mão" foi realizada, os estados selecionados para representá-la seguiram algum dos 2 padrões. As outras 4 repetições foram representadas por outras combinações dos estados 4, 1, 7, 11, 8, 2 e 3. Essas observações levantam a obvia necessidade de análise aprofundada do real significado semântico latente destes estados.

Ao comparar a filmagem da demonstração com os estados gerados é possível inferir o significado de cada estado, como mostra a **Figura 23**. A ocorrência do estado 11 indica o momento em que o especialista solta a peça, este é um comportamento generalizado independente da peça de xadrez sendo movimentada. O estado 1 indica o momento em que o especialista levanta a mão e a leva para a posição de repouso. O estado 4 ocorre quando o especialista posiciona a sua mão em descanso de lado e finalmente, o estado 7 ocorre quando o especialista posiciona a mão de costas para a mesa com a palma para cima. Quanto aos estados 8, 2 e 3, não foi possível encontrar um significado semântico para estes.

Fica, portanto, comprovado que o modelo BP-AR-HMM é capaz de prever significados semânticos latentes sem conhecimento prévio do especialista humano.

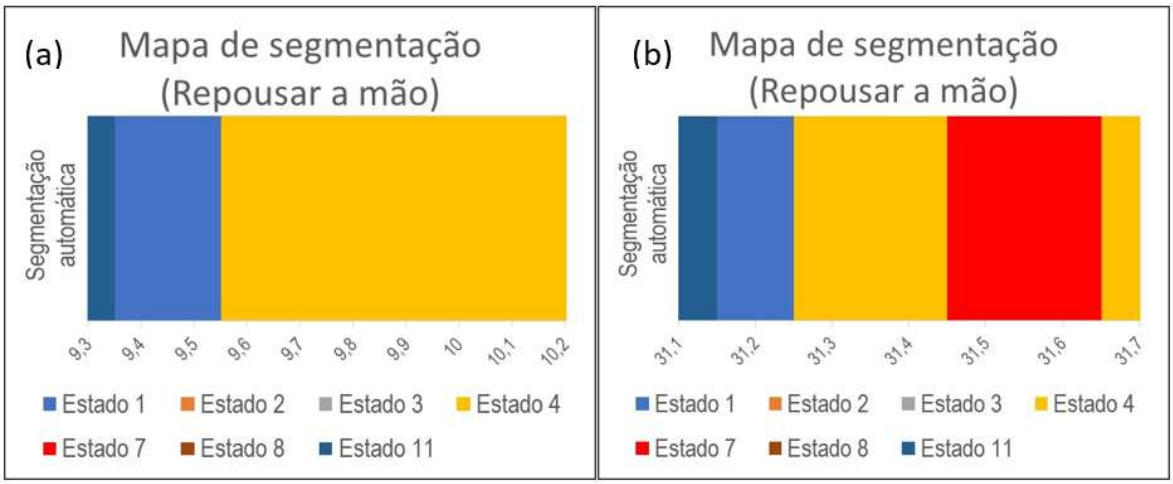

Figura 22. Mapa de segmentação dos padrões de subtarefas encontrados para a ação "Repousar a mão".

Figura 23. Estados detectados como subtarefas para a ação de "Repousar a mão".

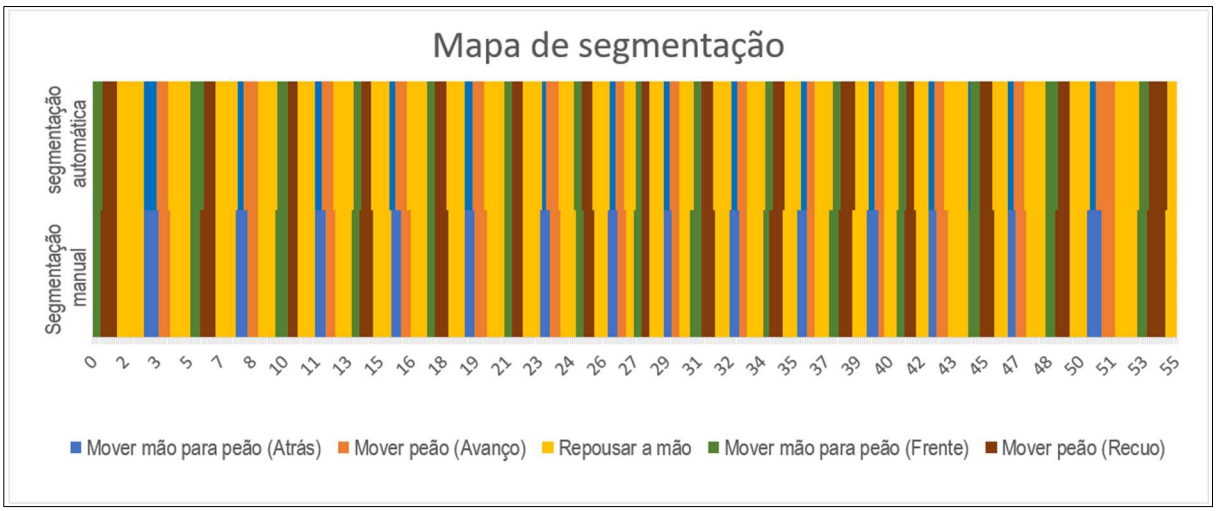

Figura 24. Mapa de segmentação automática do cenário 1 – 20 *features*. Cenário 1.

Nestas condições, foi obtido um resultado máximo de acurácia temporal de 96,0% e resultado mínimo de 81,9%, superando a inicialização com 10 *features* e reduzindo a variação entre seeds para 14,1 pontos percentuais. Estes resultados podem ser observados no mapa de segmentação **Figura 24** e na variação métrica, **Figura 25**.

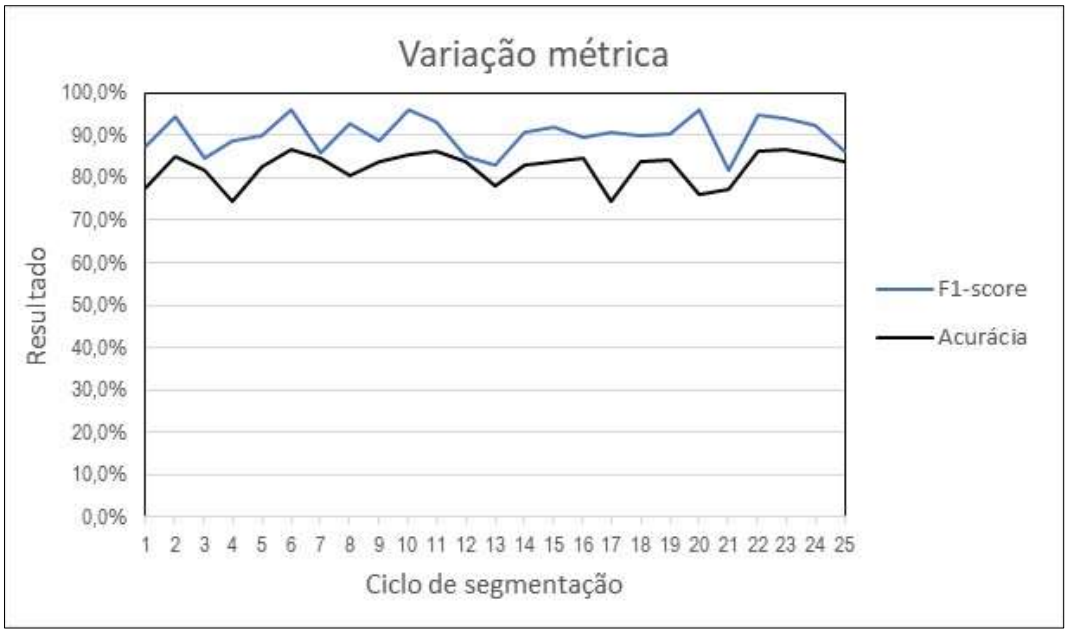

Figura 25. Variação métrica com inicialização - 20 *features*. Cenário 1.

5.1.4 Segmentação automática com inicialização de 30 *features*

Finalmente, ao realizar o experimento com inicialização de 30 *features*, observa-se uma estabilização dos resultados. A matriz de confusão (**Tabela 10**) novamente mostra que todas as ações têm pelo menos 1 estado que as representa semanticamente. Porém a característica de redundância de estados agora se intensifica, o estado "Repousar a mão" agora possui 10 estados que definem seu comportamento, porém os 4 estados que foram identificados na sessão 5.1.3 ainda são os mais relevantes em detrimento dos outros 6. Mostrando que o sistema não encontrou mais padrões relevantes.

Tabela 10. Matriz de confusão - 30 *features*. Cenário 1.

		Classificação real (Verdade absoluta)				
		Mover mão para peão (Atrás)	Mover peão (Avanço)	Repousar a mão	Mover mão para peão (Frente)	Mover peão (Recuo)
Classificação gerada						
Estados gerados	Significado semântico					
4	Mover mão para peão (Atrás)	49	0	2	0	0
13; 7	Mover peão (Avanço)	22	66	1	0	0
3; 12; 2; 6; 16; 19; 8; 14; 11; 18	Repousar a mão	9	10	238	12	6
1; 17	Mover mão para peão (Frente)	0	0	1	51	2
10; 5	Mover peão (Recuo)	0	0	2	6	91

Neste experimento, o resultado de acurácia de classificação atinge valor máximo de 86,8% e o mínimo 73,4%, com variação de 13,4%. Nessa condição, 1 seed foi capaz de superar os resultados de Wang et al. (2018), porém nenhum foi capaz de se equiparar aos resultados de Vögele, Krüger e Klein (2017).

Tabela 11. Acurácia de classificação por ação demonstrada - 30 *features*. Cenário 1.

Ação	Acurácia de classificação	Estados gerados
Mover mão para peão (Atrás)	61,3%	4
Mover peão (Avanço)	86,8%	13; 7
Repousar a mão	97,5%	3; 12; 2; 6; 16; 19; 8; 14; 11; 18
Mover mão para peão (Frente)	73,9%	1; 17
Mover peão (Recuo)	91,9%	10; 5

Tabela 12. Acurácia de classificação por estado gerado - 30 *features*. Cenário 1.

Estados gerados	Significado semântico	Acurácia de classificação
4	Mover mão para peão (Atrás)	96%
13	Mover peão (Avanço)	74%
7	Mover peão (Avanço)	80%
3	Repousar a mão	100%
12	Repousar a mão	78%
2	Repousar a mão	65%
6	Repousar a mão	100%
16	Repousar a mão	89%
19	Repousar a mão	89%
8	Repousar a mão	100%
14	Repousar a mão	100%
11	Repousar a mão	100%
18	Repousar a mão	100%
1	Mover mão para peão (Frente)	94%
17	Mover mão para peão (Frente)	100%
10	Mover peão (Recuo)	90%
5	Mover peão (Recuo)	95%

Já a nova acurácia de alinhamento temporal chega a um máximo de 96,0% e mínimo de 81,2% em seeds diferentes, o que resulta em uma variação de 14,8 pontos percentuais. Esses resultados superam os resultados de Eiband et al. (2023) em 23 seeds e os Lin e Kulić (2014) em 25.

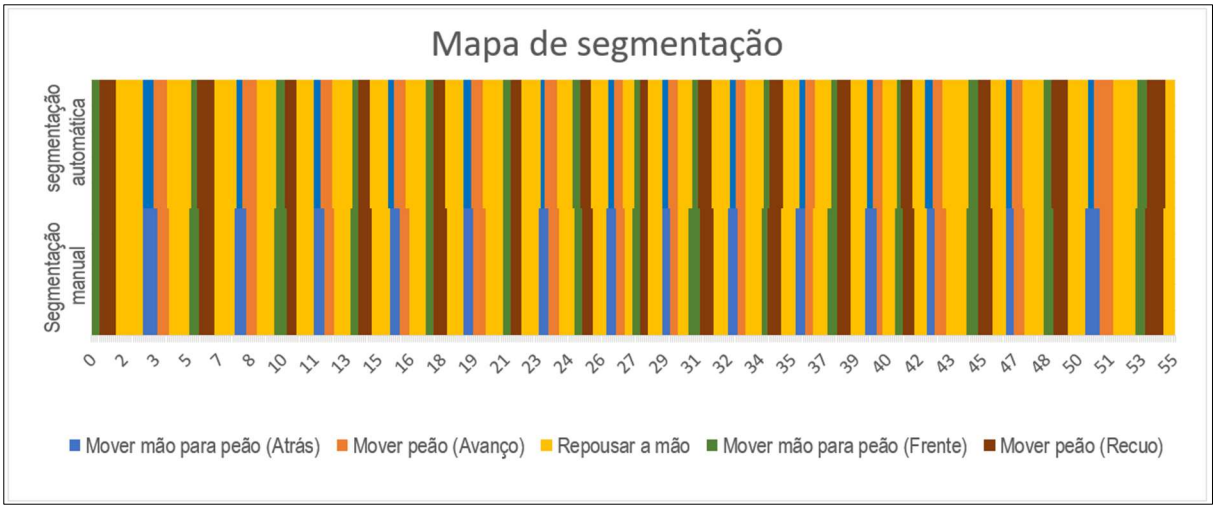

Figura 26. Mapa de segmentação automática – 30 *features*. Cenário 1.

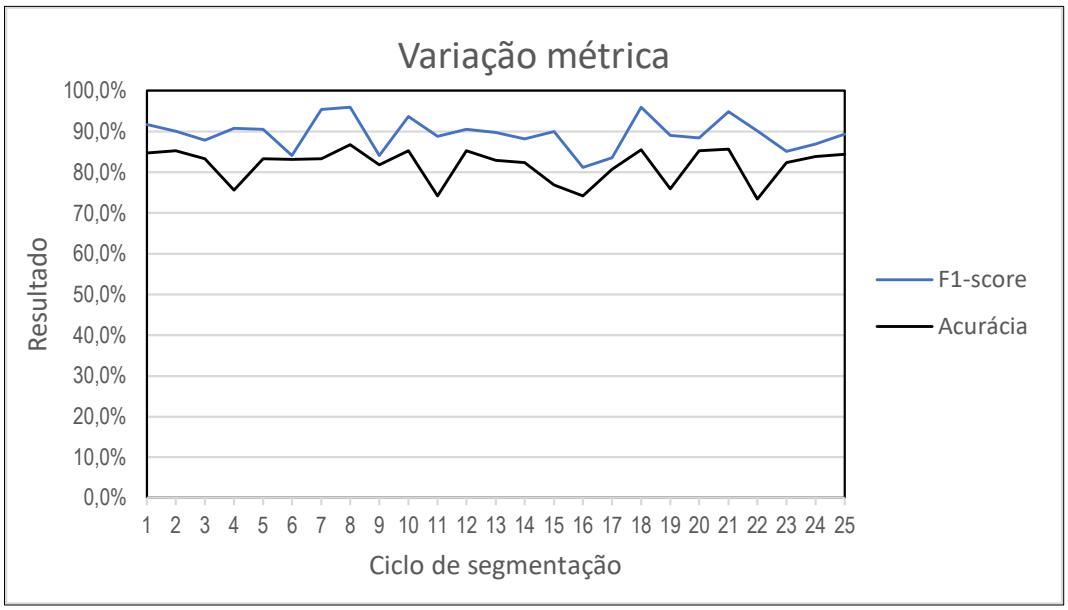

Figura 27. Variação métrica com inicialização de 30 *features*. Cenário 1.

5.2 Cenário 2 – Jogada Ruy Lopes

Nesta sessão serão apresentados os resultados do segundo cenário, onde o especialista realiza uma jogada Ruy Lopez. As mesmas configurações de segmentação manual e métricas foram mantidas para este segundo experimento.

Aqui movimentam-se um total de 3 peças de xadrez. Dessa forma, a classificação manual foi feita segmentando as tarefas em 13 ações, que formam um ciclo com 18 ações,

descritas na **Figura 28**. No mapa de segmentação da verdade absoluta, **Figura 29**, indicamos com linhas pontilhadas cada uma das ocorrências do ciclo de ações descrito, é possível identificar 4 ocorrências do ciclo no trecho da demonstração analisado, durando entre 11,6 e 15,0 segundos cada um.

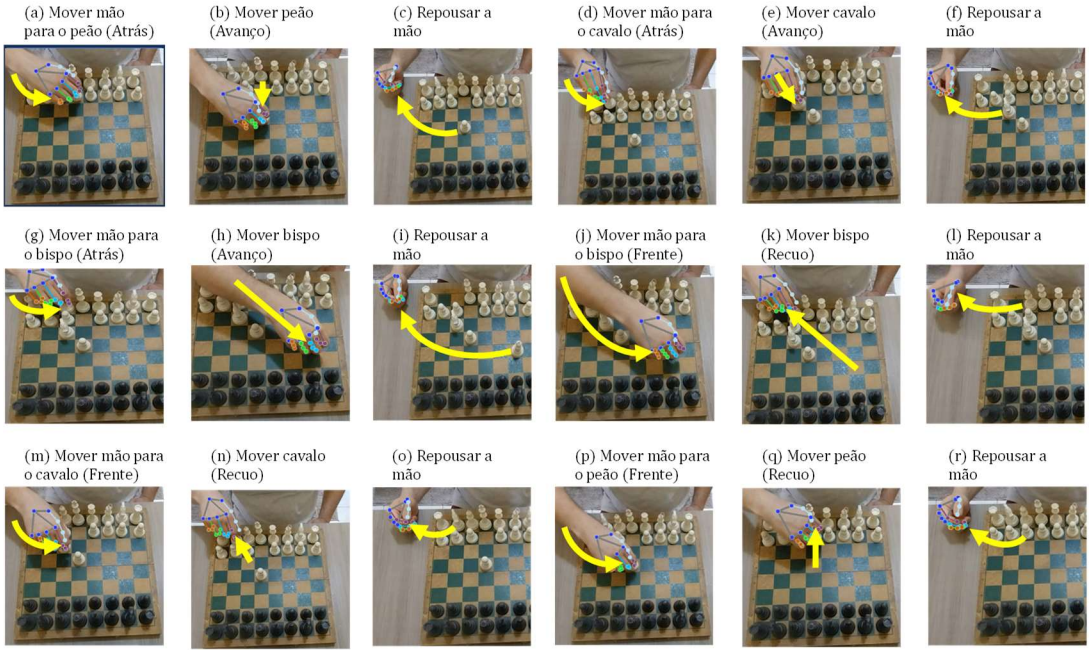

Figura 28. Rótulos definidos para segmentação manual – Cenário 2.

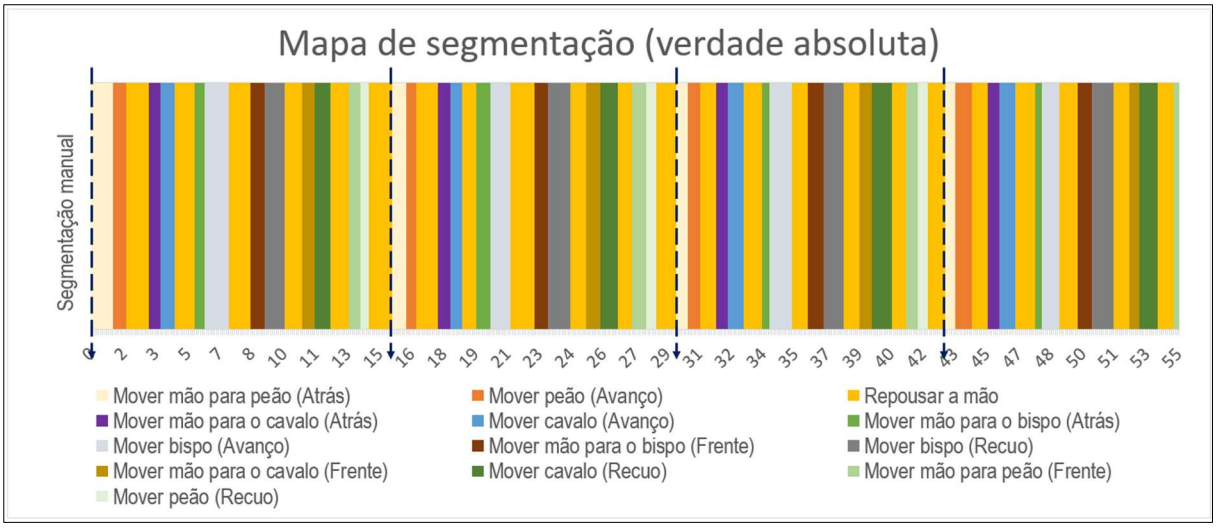

Figura 29. Mapa de segmentação manual do cenário 2.

5.2.1 Segmentação automática com inicialização de 5 *features*

Neste novo cenário com uma inicialização de 5 *features* ocorre o mesmo problema visto na sessão (5.1.1). Com apenas 6 estados gerados, a maioria das ações não possui um estado que a represente unicamente, **Tabela 13**. Apenas 5 do conjunto das 13 ações definidas possuem um estado atribuído. Como consequência, a acurácia de classificação aqui teve valor máximo de 51,6% e valor mínimo de 40,9%, resultando em uma variação de 10,6 pontos percentuais.

Tabela 13. Matriz de confusão - 5 *features*. Cenário 2.

		Classificação real (Verdade absoluta)												
Classificação gerada		Mover mão para peão (Atrás)	Mover peão (Avanço)	Repousar a mão	Mover mão para o cavalo (Atrás)	Mover cavalo (Avanço)	Mover mão para o bispo (Atrás)	Mover bispo (Avanço)	Mover mão para o bispo (Frente)	Mover bispo (Recuo)	Mover mão para o cavalo (Frente)	Mover cavalo (Recuo)	Mover mão para peão (Frente)	Mover peão (Recuo)
Estados gerados	Significado semântico													
6	Mover mão para peão (Atrás)	8	7	5	0	0	0	0	0	0	0	0	0	0
-	Mover peão (Avanço)	0	0	0	0	0	0	0	0	0	0	0	0	0
1; 2	Repousar a mão	17	2	167	14	0	6	1	5	16	1	1	6	0
-	Mover mão para o cavalo (Atrás)	0	0	0	0	0	0	0	0	0	0	0	0	0
3	Mover cavalo (Avanço)	4	17	6	10	29	0	0	0	0	0	0	0	0
-	Mover mão para o bispo (Atrás)	0	0	0	0	0	0	0	0	0	0	0	0	0
5	Mover bispo (Avanço)	5	0	6	0	0	13	37	7	0	0	0	18	11
-	Mover mão para o bispo (Frente)	0	0	0	0	0	0	0	0	0	0	0	0	0
-	Mover bispo (Recuo)	0	0	0	0	0	0	0	0	0	0	0	0	0
-	Mover mão para o cavalo (Frente)	0	0	0	0	0	0	0	0	0	0	0	0	0
4	Mover cavalo (Recuo)	0	0	34	0	0	0	4	17	26	23	35	0	10
-	Mover mão para peão (Frente)	0	0	0	0	0	0	0	0	0	0	0	0	0
-	Mover peão (Recuo)	0	0	0	0	0	0	0	0	0	0	0	0	0

A

Tabela 14 mostra que a falta de estados para representar todas as ações causam a redução da acurácia por ação. Além disso, a acurácia por estado (

Tabela 15) também teve resultados ruins, com o estado 2 tendo o melhor resultado no valor de apenas 78,0%, chegando no valor mais baixo de 23,5% para o estado 4. A baixa representatividade dos estados gerados também fica clara no mapa de segmentação. Gerando assim uma acurácia temporal máxima de 68,0% e mínima de 24,2% em diferentes seeds o que resulta em uma variação de 43,8 pontos percentuais.

Tabela 14. Acurácia de classificação por ação demonstrada - 5 features. Cenário 2.

Ação	Acurácia de classificação	Estados gerados
Mover mão para peão (Atrás)	23,5%	6
Mover peão (Avanço)	0,0%	-
Repousar a mão	76,6%	1; 2
Mover mão para o cavalo (Atrás)	0,0%	-
Mover cavalo (Avanço)	100,0%	3
Mover mão para o bispo (Atrás)	0,0%	-
Mover bispo (Avanço)	88,1%	5
Mover mão para o bispo (Frente)	0,0%	-
Mover bispo (Recuo)	0,0%	-
Mover mão para o cavalo (Frente)	0,0%	-
Mover cavalo (Recuo)	97,2%	4
Mover mão para peão (Frente)	0,0%	-
Mover peão (Recuo)	0,0%	-

Tabela 15. Acurácia de classificação por estado gerado - 5 *features*. Cenário 2.

Estados gerados	Significado semântico	Acurácia de classificação
6	Mover mão para peão (Atrás)	40,0%
1	Repousar a mão	55,8%
2	Repousar a mão	78,0%
3	Mover cavalo (Avanço)	43,9%
5	Mover bispo (Avanço)	38,1%
4	Mover cavalo (Recuo)	23,5%

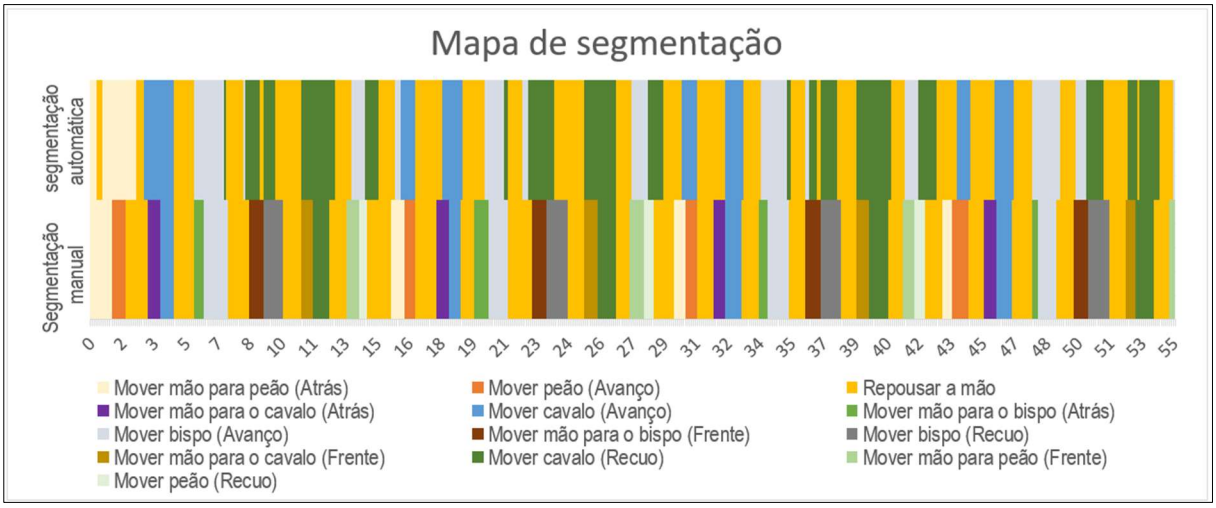

Figura 30. Mapa de segmentação automática - 5 *features*. Cenário 2.

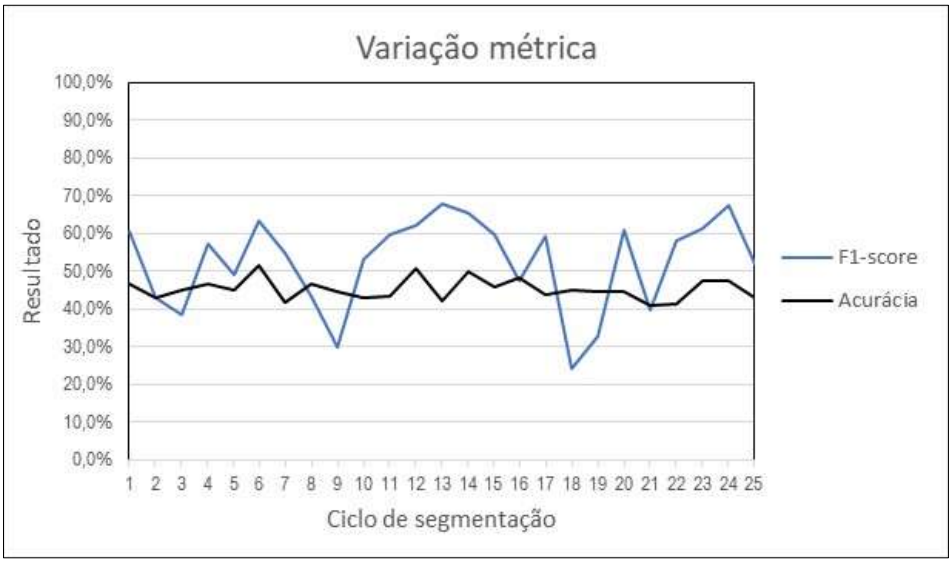

Figura 31. Variação métrica com inicialização de 5 *features*. Cenário 2.

5.2.2 Segmentação automática com inicialização de 10 *features*

Ao inicializar a simulação com 10 *features*, observa-se uma melhora no cenário, apesar de os mesmos números de comportamento serem representados pelos estados gerados, **Tabela 16**. Desta vez o resultado de classificação teve uma variação de 10,1%, variando de 45,7% no pior resultado até 55,8% no melhor, **Figura 33**.

Tabela 16. Matriz de confusão - 10 *features*. Cenário 2.

		Classificação real (Verdade absoluta)												
Classificação gerada		Mover mão para peão (Atrás)	Mover peão (Avanço)	Repousar a mão	Mover mão para o cavalo (Atrás)	Mover cavalo (Avanço)	Mover mão para o bispo (Atrás)	Mover bispo (Avanço)	Mover mão para o bispo (Frente)	Mover bispo (Recuo)	Mover mão para o cavalo (Frente)	Mover cavalo (Recuo)	Mover mão para peão (Frente)	Mover peão (Recuo)
Estados gerados	Significado semântico													
11	Mover mão para peão (Atrás)	11	7	0	0	0	0	0	0	0	0	0	0	0
-	Mover peão (Avanço)	0	0	0	0	0	0	0	0	0	0	0	0	0
3; 1; 7; 8	Repousar a mão	12	0	201	13	0	6	4	9	21	1	2	7	18
-	Mover mão para o cavalo (Atrás)	0	0	0	0	0	0	0	0	0	0	0	0	0
6	Mover cavalo (Avanço)	3	17	4	3	29	0	0	0	17	11	0	0	0
-	Mover mão para o bispo (Atrás)	0	0	0	0	0	0	0	0	0	0	0	0	0
4; 5	Mover bispo (Avanço)	8	2	5	0	0	13	38	9	4	8	0	17	3
-	Mover mão para o bispo (Frente)	0	0	0	0	0	0	0	0	0	0	0	0	0
-	Mover bispo (Recuo)	0	0	0	0	0	0	0	0	0	0	0	0	0
-	Mover mão para o cavalo (Frente)	0	0	0	0	0	0	0	0	0	0	0	0	0
2; 9	Mover cavalo (Recuo)	0	0	8	8	0	0	0	11	0	4	34	0	0
-	Mover mão para peão (Frente)	0	0	0	0	0	0	0	0	0	0	0	0	0
-	Mover peão (Recuo)	0	0	0	0	0	0	0	0	0	0	0	0	0

Tabela 17. Acurácia de classificação por ação demonstrada - 10 *features*. Cenário 2.

Ação	Acurácia de classificação	Estados gerados
Mover mão para peão (Atrás)	32,4%	11
Mover peão (Avanço)	0,0%	-
Repousar a mão	92,2%	3; 1; 7; 8
Mover mão para o cavalo (Atrás)	0,0%	-
Mover cavalo (Avanço)	100,0%	6
Mover mão para o bispo (Atrás)	0,0%	-
Mover bispo (Avanço)	90,5%	4; 5
Mover mão para o bispo (Frente)	0,0%	-
Mover bispo (Recuo)	0,0%	-
Mover mão para o cavalo (Frente)	0,0%	-
Mover cavalo (Recuo)	94,4%	2; 9
Mover mão para peão (Frente)	0,0%	-
Mover peão (Recuo)	0,0%	-

Tabela 18. Acurácia de classificação por estado gerado - 10 *features*.

Estados gerados	Significado semântico	Acurácia de classificação
11	Mover mão para peão (Atrás)	61,1%
3	Repousar a mão	77,7%
1	Repousar a mão	79,2%
7	Repousar a mão	55,0%
8	Repousar a mão	50,0%
6	Mover cavalo (Avanço)	34,5%
4	Mover bispo (Avanço)	44,2%
5	Mover bispo (Avanço)	29,7%
2	Mover cavalo (Recuo)	43,8%
9	Mover cavalo (Recuo)	76,5%
10	Mover mão para peão (Atrás)	0,0%

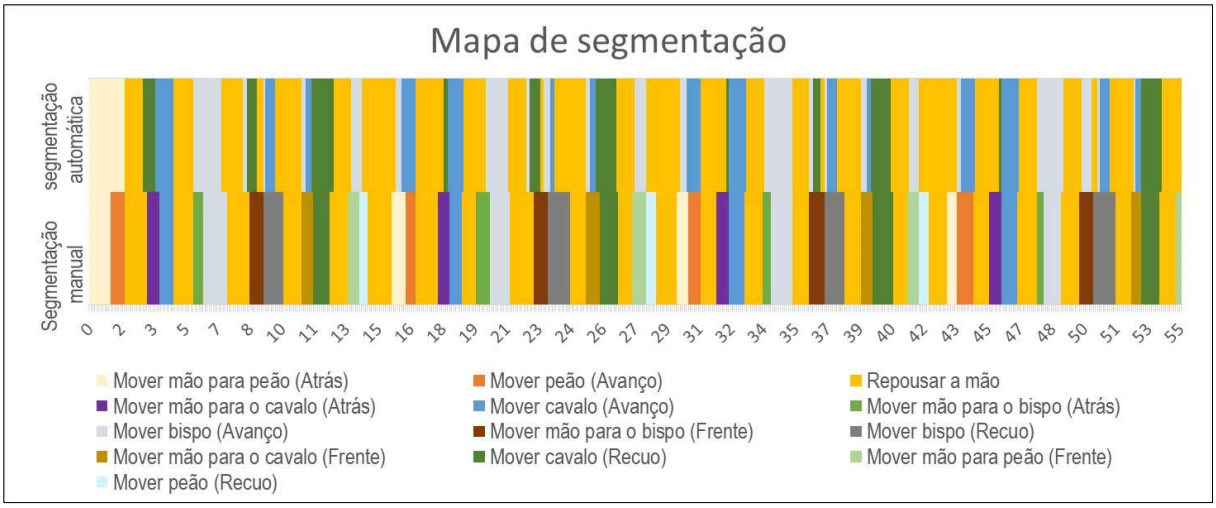

Figura 32. Mapa de segmentação automática - 10 *features*. Cenário 2.

O mapa de segmentação (**Figura 32**) indica uma acurácia de alinhamento temporal que vai do mínimo de 59,7% a um máximo de 72,0%, o que representa a variação de 12,3 pontos percentuais, **Figura 33**. Um resultado que apresenta menos variação do que aquele com inicialização de 5 *features*.

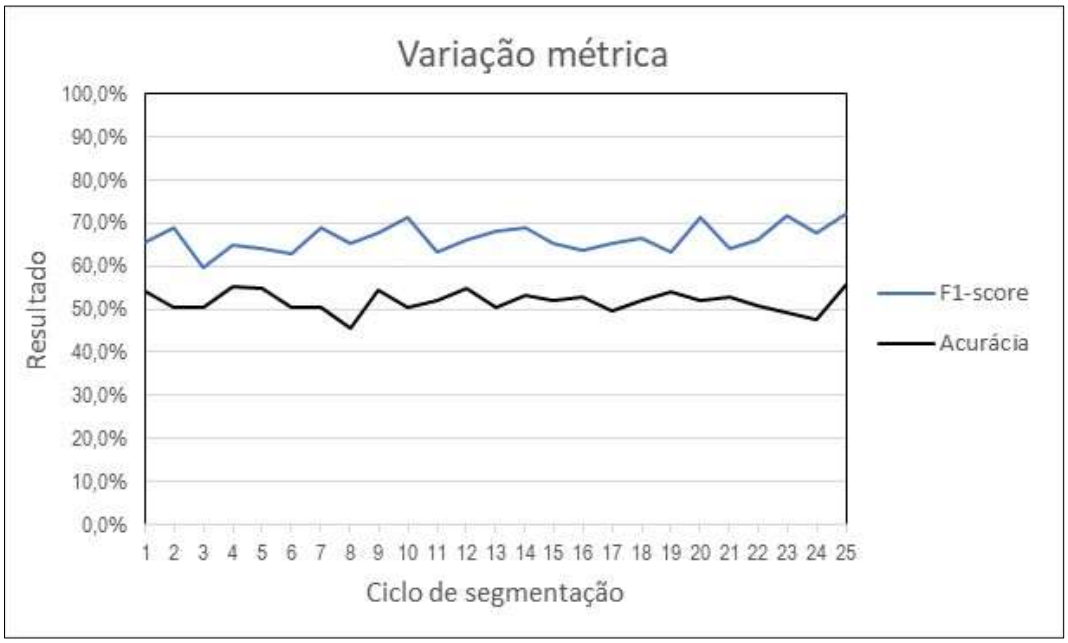

Figura 33. Variação métrica com inicialização de 10 *features*. Cenário 2.

5.2.3 Segmentação automática com inicialização de 20 *features*

Ao realizar o experimento com inicialização de 20 *features*, há um aumento na quantidade de ações representadas pelos estados, desta vez, um total de 9 ações das 13 são representadas por algum estado, **Tabela 19**. Essa melhoria ocasiona aumento na acuracidade de classificação, que atinge 64,8%, com mínimo de 53,6% e variação de 11,2 pontos percentuais, **Figura 35**.

Tabela 19. Matriz de confusão - 20 *features*. Cenário 2.

		Classificação real (Verdade absoluta)												
Classificação gerada		Mover mão para peão (Atrás)	Mover peão (Avanço)	Repousar a mão	Mover mão para o cavalo (Atrás)	Mover cavalo (Avanço)	Mover mão para o bispo (Atrás)	Mover bispo (Avanço)	Mover mão para o bispo (Frente)	Mover bispo (Recuo)	Mover mão para o cavalo (Frente)	Mover cavalo (Recuo)	Mover mão para peão (Frente)	Mover peão (Recuo)
Estados gerados	Significado semântico													
16	Mover mão para peão (Atrás)	11	7	0	0	0	0	0	0	0	0	0	0	0
-	Mover peão (Avanço)	0	0	0	0	0	0	0	0	0	0	0	0	0
13; 11; 12; 2; 18; 19	Repousar a mão	13	0	183	16	0	9	9	1	13	0	2	7	0
-	Mover mão para o cavalo (Atrás)	0	0	0	0	0	0	0	0	0	0	0	0	0
14;15	Mover cavalo (Avanço)	0	11	6	8	29	0	0	0	0	0	0	0	0
-	Mover mão para o bispo (Atrás)	0	0	0	0	0	0	0	0	0	0	0	0	0
10	Mover bispo (Avanço)	0	0	1	0	0	5	25	0	0	0	0	0	0
7	Mover mão para o bispo (Frente)	0	0	7	0	0	0	0	12	0	0	0	0	0
4; 3; 17	Mover bispo (Recuo)	0	0	1	0	0	0	0	2	25	20	1	0	0
-	Mover mão para o cavalo (Frente)	0	0	0	0	0	0	0	0	0	0	0	0	0
5; 8	Mover cavalo (Recuo)	0	0	7	0	0	0	0	14	4	4	33	0	0
9	Mover mão para peão (Frente)	10	8	0	0	0	0	0	0	0	0	0	17	2
1	Mover peão (Recuo)	0	0	13	0	0	5	8	0	0	0	0	0	19

Tabela 20. Acurácia de classificação por ação demonstrada - 20 *features*. Cenário 2.

Ação	Acurácia de classificação	Estados gerados
Mover mão para peão (Atrás)	32,4%	16
Mover peão (Avanço)	0,0%	-
Repousar a mão	83,9%	13; 11; 12; 2; 18; 19
Mover mão para o cavalo (Atrás)	0,0%	-
Mover cavalo (Avanço)	100,0%	14;15
Mover mão para o bispo (Atrás)	0,0%	-
Mover bispo (Avanço)	59,5%	10
Mover mão para o bispo (Frente)	41,4%	7
Mover bispo (Recuo)	59,5%	4; 3; 17
Mover mão para o cavalo (Frente)	0,0%	-
Mover cavalo (Recuo)	91,7%	5; 8
Mover mão para peão (Frente)	70,8%	9
Mover peão (Recuo)	90,5%	1

Tabela 21. Acurácia de classificação por estado gerado 20 *features*. Cenário 2.

Estados gerados	Significado semântico	Acurácia de classificação
16	Mover mão para peão (Atrás)	61,1%
13	Repousar a mão	76,3%
11	Repousar a mão	85,4%
12	Repousar a mão	54,8%
2	Repousar a mão	47,1%
18	Repousar a mão	41,7%
19	Repousar a mão	80,6%
14	Mover cavalo (Avanço)	41,7%
15	Mover cavalo (Avanço)	77,8%
10	Mover bispo (Avanço)	80,6%
7	Mover mão para o bispo (Frente)	63,2%
4	Mover bispo (Recuo)	48,8%
3	Mover bispo (Recuo)	100,0%
17	Mover bispo (Recuo)	50,0%
5	Mover cavalo (Recuo)	89,5%
8	Mover cavalo (Recuo)	37,2%
9	Mover mão para peão (Frente)	45,9%
1	Mover peão (Recuo)	42,2%
6	-	

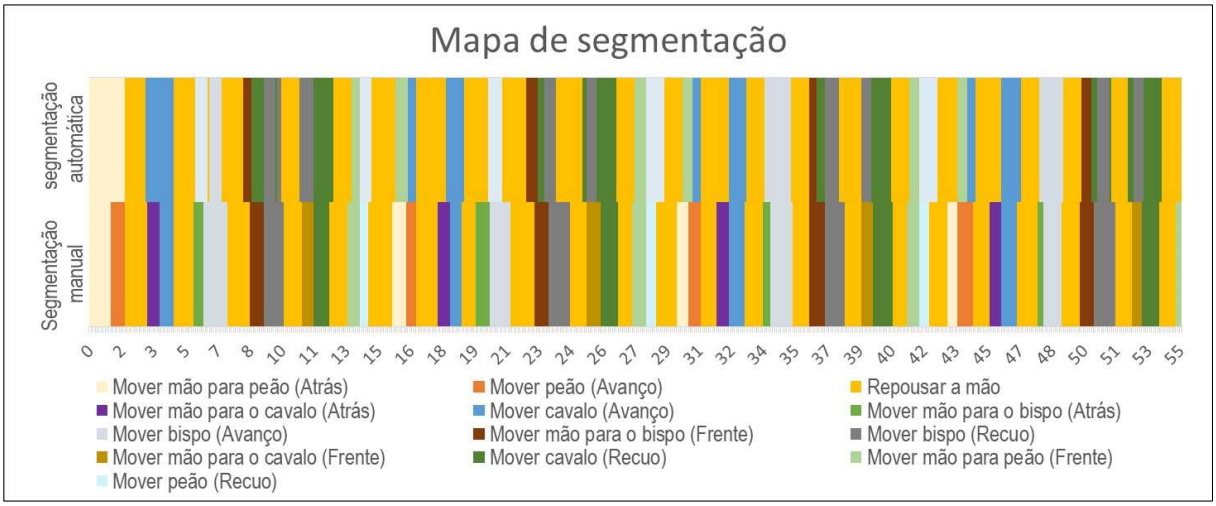

Figura 34. Mapa de segmentação automática – 20 *features*. Cenário 2.

A acurácia de alinhamento temporal nessa configuração atingiu máximo de 76,4%, mínimo de 57,1% e variação de 19,2 pontos percentuais.

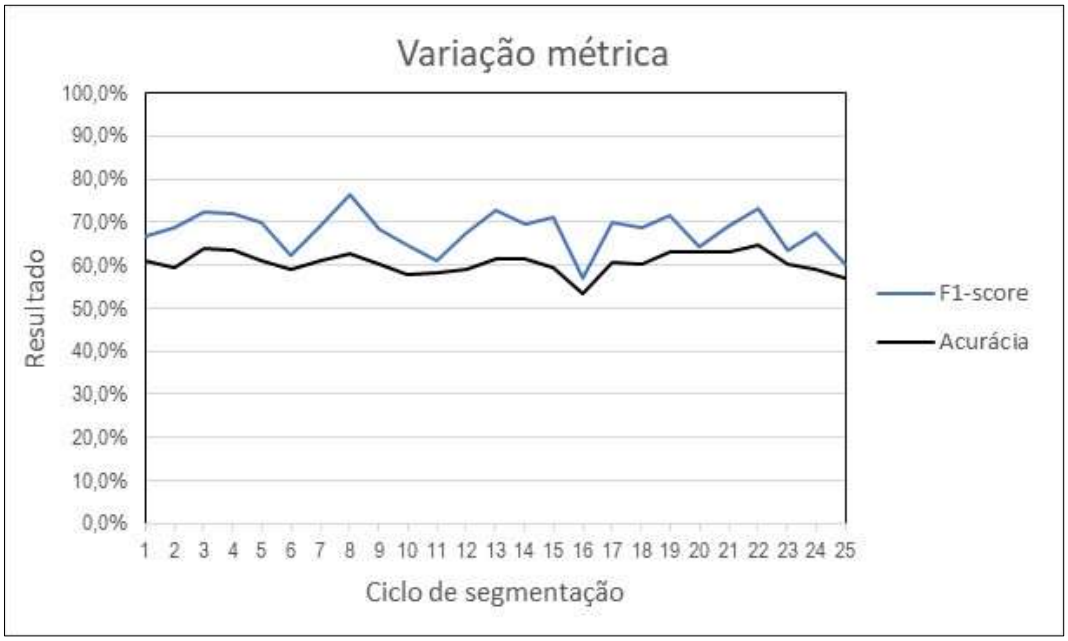

Figura 35. Variação métrica com inicialização de 20 *features*. Cenário 2.

5.2.4 Segmentação automática com inicialização de 30 *features*

Os resultados dos testes do cenário 2 com uma inicialização de 30 *features* apresentou grande melhoria em relação aos resultados dos testes anteriores. Neste experimento, quase todos os comportamentos exceto “Mover peão (Recuo)” possuem um estado que os representa, como pode ser observado na

Tabela 22.

Da mesma forma que foi constatado antes, na sessão 5.1.43, o efeito de todos os comportamentos serem representados por um estado é a melhoria da acurácia de classificação. Nesse caso fica entre 74,1% e 62,9%, apresentando variação de 11,2 pontos percentuais entre os 25 seeds, **Figura 37**.

Tabela 22. Matriz de confusão - 30 *features*. Cenário 2.

		Classificação real (Verdade absoluta)												
Classificação gerada		Mover mão para peão (Atrás)	Mover peão (Avanço)	Repousar a mão	Mover mão para o cavalo (Atrás)	Mover cavalo (Avanço)	Mover mão para o bispo (Atrás)	Mover bispo (Avanço)	Mover mão para o bispo (Frente)	Mover bispo (Recuo)	Mover mão para o cavalo (Frente)	Mover cavalo (Recuo)	Mover mão para peão (Frente)	Mover peão (Recuo)
Estados gerados	Significado semântico													
15	Mover mão para peão (Atrás)	13	0	0	0	0	0	0	0	0	0	0	0	0
8	Mover peão (Avanço)	9	26	3	0	0	0	0	0	0	0	0	0	13
14; 6; 3 ;7 ;26	Repousar a mão	7	0	194	4	1	4	0	6	0	8	1	6	1
13	Mover mão para o cavalo (Atrás)	0	0	0	20	14	0	0	0	0	0	0	0	0
22	Mover cavalo (Avanço)	0	0	1	0	3	0	0	0	0	0	0	0	0
19	Mover mão para o bispo (Atrás)	0	0	1	0	0	10	1	0	0	0	0	0	0
10; 25	Mover bispo (Avanço)	0	0	0	0	0	0	20	8	5	0	0	0	0
17	Mover mão para o bispo (Frente)	0	0	5	0	0	0	2	15	0	0	0	0	0
2; 5	Mover bispo (Recuo)	0	0	2	0	0	5	19	0	37	0	0	0	0
11	Mover mão para o cavalo (Frente)	0	0	1	0	11	0	0	0	0	16	1	0	0
4; 18	Mover cavalo (Recuo)	0	0	2	0	0	0	0	0	0	0	34	0	0
9	Mover mão para peão (Frente)	0	0	0	0	0	0	0	0	0	0	0	16	0
-	Mover peão (Recuo)	0	0	0	0	0	0	0	0	0	0	0	0	0

Apesar destes serem os melhores resultados para esse novo conjunto de atividades, não chegaram a superar os resultados de Wang et al. (2018) e Vögele, Krüger e Klein (2017). Uma possível explicação seria a diferença em complexidade das propostas.

Vögele, Krüger e Klein (2017) realizou a segmentação das ações dentro de um conjunto de no máximo 7 ações. Além disso, sua proposta é baseada em sensores de captura de movimento, onde ruído de captura é desprezível, principalmente quando comparado com captura a partir de vídeo. Já Wang et al. (2018), baseou sua segmentação em um conjunto de 5 ações, previamente conhecidas pelo sistema, sem necessidade de inferência por parte do seu modelo.

No que tange a acurácia de segmentação o modelo aqui proposto gerou para o cenário 2, com inicialização de 30 *features*, resultados entre 84,6% e 71,9%, como variação de 12,8

pontos percentuais entre os 25 seeds, **Figura 37**. Este resultado supera os de Eiband et. al. (2023) em seu cenário de montagem em apenas 2 seeds e os de Lin e Kulić (2014) em 5 seeds.

Pode-se observar que nos resultados da

Tabela 22, existem estados que, apesar de terem sido criados pelo modelo, não foram utilizados durante o trecho de vídeo, (*dataset* de testes). Esses estados representam variações na forma de execução das atividades, assim como foi constatado na sessão 5.1.3, referem-se à comportamentos latentes, que estavam presentes apenas no segmento de vídeo utilizado para treinar o modelo. O excesso de estados não utilizados no *dataset* de teste, indica que a quantidade de demonstrações neste conjunto foi baixa para que se pudesse enxergar todos os padrões de comportamento executados pelo especialista humano.

Tabela 23. Acurácia de classificação por ações demonstrada - 30 *features*. Cenário 2.

Ação	Acurácia de classificação	Estados gerados
Mover mão para peão (Atrás)	44,8%	15
Mover peão (Avanço)	100,0%	8
Repousar a mão	92,8%	14; 6; 3 ;7 ;26
Mover mão para o cavalo (Atrás)	83,3%	13
Mover cavalo (Avanço)	10,3%	22
Mover mão para o bispo (Atrás)	52,6%	19
Mover bispo (Avanço)	47,6%	10; 25
Mover mão para o bispo (Frente)	51,7%	17
Mover bispo (Recuo)	88,1%	2; 5
Mover mão para o cavalo (Frente)	66,7%	11
Mover cavalo (Recuo)	94,4%	4; 18
Mover mão para peão (Frente)	72,7%	9
Mover peão (Recuo)	0,0%	-

Tabela 24. Acurácia de classificação por estado gerado - 30 *features*.

Estados gerados	Significado semântico	Acurácia de classificação
15	Mover mão para peão (Atrás)	100,0%
8	Mover peão (Avanço)	51,0%
14	Repousar a mão	81,1%
6	Repousar a mão	96,0%
3	Repousar a mão	82,8%
7	Repousar a mão	100,0%
26	Repousar a mão	75,0%
13	Mover mão para o cavalo (Atrás)	58,8%
22	Mover cavalo (Avanço)	75,0%
19	Mover mão para o bispo (Atrás)	83,3%
10	Mover bispo (Avanço)	55,2%
25	Mover bispo (Avanço)	100,0%
17	Mover mão para o bispo (Frente)	68,2%
2	Mover bispo (Recuo)	61,5%
5	Mover bispo (Recuo)	54,2%
11	Mover mão para o cavalo (Frente)	55,2%
4	Mover cavalo (Recuo)	92,6%
18	Mover cavalo (Recuo)	100,0%
9	Mover mão para peão (Frente)	100,0%
1	-	0,0%
12	-	0,0%
16	-	0,0%
20	-	0,0%
21	-	0,0%
23	-	0,0%
24	-	0,0%

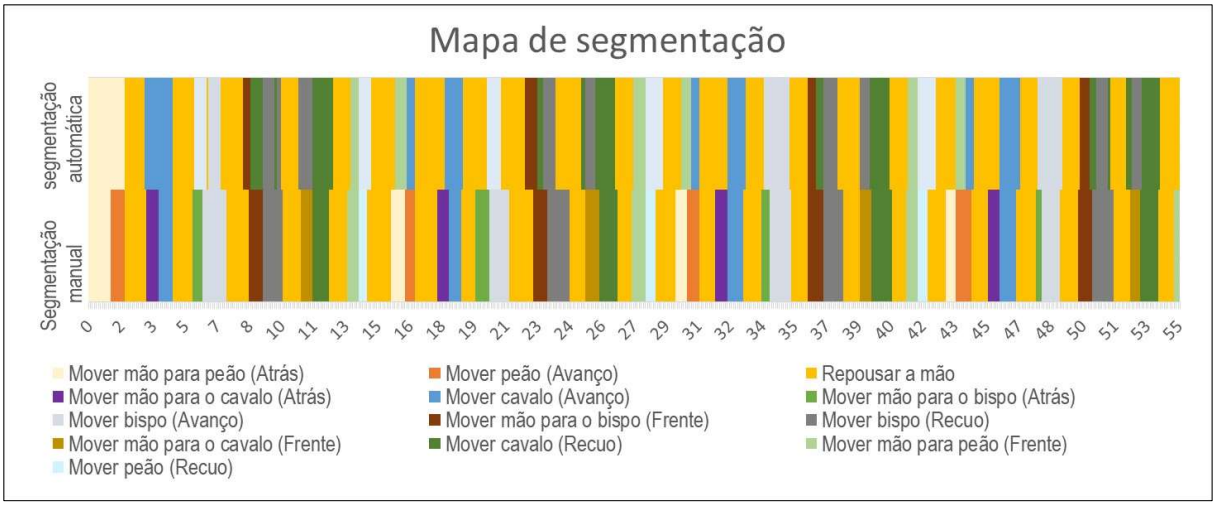

Figura 36. Mapa de segmentação automática - 10 *features*. Cenário 2.

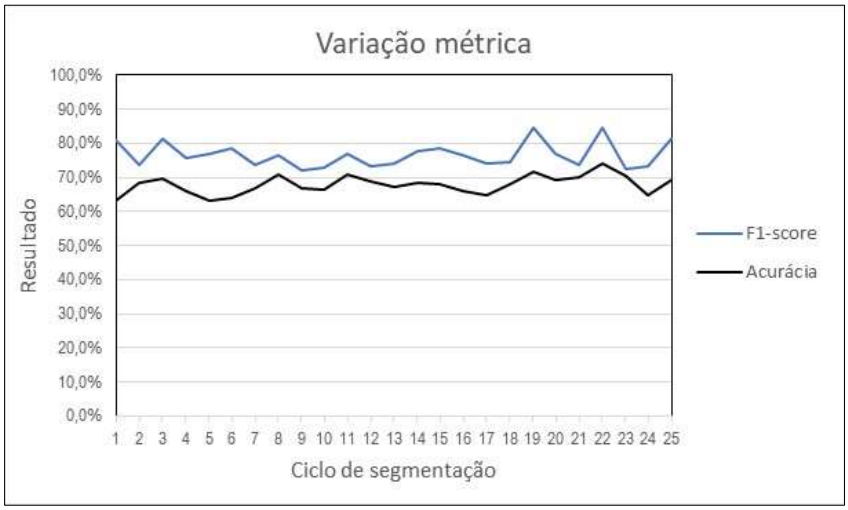

Figura 37. Variação métrica com inicialização de 30 *features*. Cenário 2.

6 Considerações finais

Neste trabalho propomos um método para segmentação automática de tarefas de manipulação. A vantagem mais relevante trazida por nossa abordagem é que ela identifica e segmenta os comportamentos e não requer equipamentos caros, além, é claro, da câmera RGB, o que reduz drasticamente o custo dos sensores. Além disso, a solução proposta permite que a segmentação seja feita sem conhecimento aprofundado de inteligência artificial ou robótica pelo usuário final. Do ponto de vista do LfD, isto permite ao professor expressar plenamente os seus conhecimentos na tarefa em questão.

Os resultados indicam que nosso método pode segmentar as demonstrações enquanto mantém a relevância semântica apesar do pequeno conjunto de dados, o que é útil para aplicações LfD. Um conjunto de rótulos foram criados para realizar a anotação manual e o algoritmo conseguiu prever estes rótulos com acurácia de classificação máxima de 86,8%. Isso mostra que apesar deste ser um método não-supervisionado baseado em uma abordagem de anotação de pontos chave automática, seus resultados chegam próximos aos de outros autores que tiveram propostas baseadas em sensores de movimento e com rótulos pré-estabelecidos.

Demonstração	Features inicializados	Acurácia de alinhamento temporal			Acurácia de classificação		
		Máximo	Mínimo	Variação	Máximo	Mínimo	Variação
Movimentação do peão	5	93,8%	75,3%	18,5%	67,5%	58,9%	8,6%
Movimentação do peão	10	96,6%	84,2%	12,4%	76,5%	65,1%	11,4%
Movimentação do peão	20	96,0%	81,9%	14,1%	86,8%	74,3%	12,5%
Movimentação do peão	30	96,0%	81,2%	14,8%	86,8%	73,4%	13,4%
Jogada Ruy Lopes	5	68,0%	24,2%	43,8%	51,6%	40,9%	10,6%
Jogada Ruy Lopes	10	72,0%	59,7%	12,3%	55,8%	45,7%	10,1%
Jogada Ruy Lopes	20	76,4%	57,1%	19,2%	64,8%	53,6%	11,2%
Jogada Ruy Lopes	30	84,6%	71,9%	12,8%	74,1%	62,9%	11,2%

Figura 38. Resumo dos resultados

Além disso, o método garantiu uma acurácia de alinhamento temporal máxima de 96,0%, dependendo da quantidade de *features* inicializados e do tipo de demonstração em questão. Este tipo de resultado tem o potencial de reduzir o tempo de preparação de *datasets* em aplicações de PfD.

No entanto, algumas limitações foram constatadas durante o desenvolvimento deste trabalho. Primeiramente, ficou clara que existe uma influência do número de *features* utilizados na inicialização do algoritmo nos resultados obtidos, por consequência, o usuário final deve prever de antemão quantos rótulos serão obtidos após a segmentação, o que é algo que vai contra a proposta do trabalho. Em segundo lugar, o sistema apresentou estados irrelevantes, ficando presentes no modelo apesar de não ser utilizado durante a demonstração. Isso ocorreu nas demonstrações da jogada Ruy Lopes com o modelo inicializado com 20 e 30 *features*, indicando que é necessária a implementação de um algoritmo de nascimento e morte de *features* mais eficaz.

Para trabalhos futuros, propõe-se a investigação da aplicabilidade da solução aqui proposta a demonstrações de corpo inteiro, realizando também a anotação automática. E com esta mesma premissa, realizar o estudo de aplicabilidade deste método para análises ergonômicas em contexto industrial. Também pensando em trabalhos futuros, deve-se estudar a implementação de uma extensão dos algoritmos propostos de maneira a resolver os problemas de quantidades de rótulos e de estados irrelevantes aqui apontados.

7 Referências

GUERIN, K. R., LEA, C., PAXTON, C., & HAGER , G. D. (2015). A framework for end-user instruction of a robot assistant for manufacturing. *International conference on robotics and automation*, pp. 6167-6174.

ALIBEIGI, M., RABIEE, S., & AHMADABADI, M. N. (2017). Inverse kinematics based human mimicking system using skeletal tracking technology. Journal of Intelligent & Robotic Systems. *Journal of Intelligent & Robotic Systems, 85*, pp. 27-45.

ARGALL , B. D., CHERNOVA , S., VELOSO , M., & BROWNING , B. (2009). A survey of robot learning from demonstration. *Robotics and Autonomous Systems, 57*, pp. 469–483.

BERNARDIN, K., OGAWARA, K., & IKEUCHI, K. (2005). A Sensor Fusion Approach for Recognizing. *IEEE Transactions on Robotics, 21*, pp. 47-57.

BI, Z. M., LIU, Y., BAUMGARTNER, B., CULVER, E., SOROKIN, J. N., PETERS, A., . . . O'SHAUGHNESSEY, S. (2015). Reusing industrial robots to achieve sustainability in small and medium-sized enterprises. *Industrial Robot: An International Journal,, 42*, pp. 264-273.

BRADSKI, G. (2000). *The Opencv Library* (Vol. 25). Doctor Dobbs Journal .

BRODERICK, T., JORDAN , M., & PITMAN, J. (2011). *Beta processes, stick-breaking, and power laws.* University of California at Berkeley, Berkeley.

CACCAVALE, R., SAVERIANO, M., FINZI, A., & LEE, D. (2019). Kinesthetic teaching and attentional supervision of structured tasks in human–robot interaction. *Special Issue: Learning for Human-Robot Collaboration, 43*, pp. 1291-1307.

CHEN, N., HU, Y., ZHANG, J., & ZHANG, J. (2014). Robot Learning of everyday object manipulation using Kinect. In: Foundations and Practical Applications of Cognitive Systems and Information Processing. *Springer, 215*, pp. 665-674.

CHERNOVA, S., & THOMAZ, A. L. (2014). *Robot Learning from Human Teachers* (1 ed., Vol. 1). Springer Cham. doi:978-3-031-01570-0

DONG, C., YU, L., TAKIZAWA, M., KUDOH, S., & SUEHIRO, T. (2021). Food peeling method for dual-arm cooking robot. *International Symposium on System Integration (SII)*, pp. 801-806.

DU, G., ZHANG, P., MAI, J., & LI, Z. (2012). Markerless Kinect-Based Hand. *International Journal of Advanced Robotic Systems, 9*, pp. 1-10.

EIBAND , T., LIEBL, J., WILLIBALD , C., & LEE , D. (2023). Online task segmentation by merging symbolic and data-driven skill recognition during kinesthetic teaching. *Robotics and Autonomous Systems, 162*, pp. 104-367.

EKVALL, S., & KRAGIC, D. (2004). Interactive grasp learning based on human demonstration. *IEEE International Conference on Robotics and Automation, 2004. Proceedings, 4*, pp. 3519-3524.

ELLIOTT, J. R., MOORE, B. J., & AGGOUN, L. (1995). *Hidden Markov Models.* New York: Springer New York, NY.

FAOUZI, J., & JANATI, H. (2020). pyts: A Python Package for Time Series Classification,. *Journal of Machine Learning Research, 21*, pp. 1-6.

FOX, E. B., HUGHES, M. C., SUDDERTH, E. B., & JORDAN , M. I. (2014). Joint modeling of multiple time series via the beta process with application to motion capture segmentation. *The Annals of Applied Statistics, 8*, pp. 1281-1313.

FOX, E. B., SUDDERTH, B. E., JORDAN, I. M., & WILLSHY, S. A. (2009). Sharing features among dynamical systems with beta processes. *Proceedings of the 22nd International Conference on Neural Information Processing Systems*, pp. 549-557.

GREWAL, L. J., KRZYWINSKI, M., & ALTMAN , N. (2019). Markov models—hidden Markov models. *Nature methods, 16*, pp. 795-796.

HEDLUND, E., & JOHNSON, M. (2021). The Effects of a Robot's Performance on Human Teachers for Learning from Demonstration Tasks. *International Conference on Human-Robot Interaction*, pp. 207–215.

HEIMANN , O., & GUHL, J. (2020). Industrial robot programming methods: A scoping review. *International Conference on Emerging Technologies and Factory Automation, 25*, pp. 696-703.

JHA, A., CHIDDARWAR, S. S., ALAKSHENDRA, V., & ANDULKAR, M. V. (2017). Kinematics-based approach for robot programming via human arm motion. *Journal of the Brazilian Society of Mechanical Sciences and Engineering, 39*, pp. 2659-2675.

KINGMAN, J. (1967). Completely random measures. *Pacific Journal of Mathematics, 21*, pp. 59-78.

KRISHNAN, S., GARG, A., PATIL, S., LEA, C., HAGER, G., ABBEEL , P., & GOLDBERG , K. (2017). Transition state clustering: Unsupervised surgical trajectory segmentation for robot learning. . *The International journal of robotics research, 26*, pp. 1595-1618.

KRÜGER, J., WANG, L., VERL, A., BAUERNHANSL, T., CARPANZANO, E., MAKRIS, S., . . . PELLEGRINELLI, S. (2017). Innovative control of assembly systems and lines. *66*, pp. 707-730.

KUMAR, A. (2007). From mass customization to mass personalization: a strategic transformation. *International Journal of Flexible Manufacturing Systems, 19*, pp. 533–547.

KUNIYOSHI, Y., INABA, M., & INOUE, H. (1994). *Trans. Robot. Autom., 10*, pp. 779-822.

KYRARINI, M., HASEEB, M., RISTIĆ-DURRANT, D., & GRÄSER, A. (2019). Robot learning of industrial assembly task via human demonstrations. *Autonomous Robots, 43*, pp. 239–257.

LEE, D., & OTT, C. (2011). Incremental kinesthetic teaching of motion primitives using the motion refinement tube. *Autonomous Robots, 31*, pp. 115-131.

LIN, J. F.-S., & KULIĆ, D. (2014). Online Segmentation of Human Motion for. *Transactions on neural systems and rehabilitation engineering, 22*, pp. 168-180.

LIU, W., CHEN, D., & STEIL, J. (2017). Analytical inverse kinematics solver for anthropomorphic 7-DOF redundant manipulators with human-like configuration constraints. *Journal of Intelligent & Robotic Systems, 86*, pp. 63-79.

LUGARESI, C., TANG, J., NASH, H., MCCLANAHAN, C., UBOWEJA, E., HAYS, M., . . . GRUNDMANN, M. (2019). A Framework for Building Perception Pipelines. *MediaPipe*, pp. 1-9.

MOR, B., GARHWAL, S., & KUMAR, A. (2021). A systematic review of hidden Markov models and their applications. *Archives of computational methods in engineering, 28*, pp. 1429-1448.

NG, A., & RUSSEL, S. J. (2000). Algorithms for Inverse Reinforcement Learning. *Proceedings of the Seventeenth International Conference on Machine Learning*, (pp. 663–670).

NIEKUM, S., OSENTOSKI, S., KONIDARIS, G., & BARTO, A. G. (2012). Learning and generalization of complex tasks from unstructured demonstrations. *International Conference on Intelligent Robots and Systems* (pp. 5239-5246). Vilamoura-Algarve: IEEE/RSJ.

NIKOLAIDIS, S., & SHAH, J. (2013). Human-robot cross-training: computational formulation, modeling and evaluation of a human team training strategy. *ACM/IEEE International Conference on Human-Robot Interaction (HRI* (pp. 33-40). IEEE.

PARK, D., KIM, H., & KEMP, C. C. (2019). Multimodal anomaly detection for assistive robots. *Autonomous Robots, 43*, pp. 611-62.

PARK, S., & AGGARWAL, J. K. (2004). emantic-level Understanding of Human Actions and Interactions using Event Hierarchy. *Conference on Computer Vision and Pattern Recognition Workshop*, pp. 12-12.

PARK, S., & AGGARWAL, J. K. (2004). Semantic-level understanding of human actions and interactions using event hierarchy. *Conference on Computer Vision and Pattern Recognition Workshop*, pp. 35-62.

QIU, Z., EIBAND, T., LI, S., & LEE, D. (2020). Hand Pose-based Task Learning from Visual Observations with. *IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*.

RAMIREZ-AMARO , K., YANG , Y., & CHENG , G. (2019). A survey on semantic-based methods for the understanding of human movements. *Robotics and Autonomous Systems, 119*, pp. 31–50.

RAVICHANDAR, H., POLYDOROS, A. S., CHERNOVA, S., & BILLARD, A. (2020). Recent Advances in Robot Learning from Demonstration. *Annual Review of Control, Robotics, and Autonomous Systems, 3*, pp. 297–330.

ROA-GARZÓN, M. A., GAMBARO, E. F., FIOREK-JASINSKA, M., ENDRES, F., RUESS, F., SCHALLER, R., . . . SUPPA, M. (2019). Vision-based solutions for robotic manipulation and navigation applied to object picking and distribution. *KI-Künstliche Intelligenz, 33*, pp. 171-180.

SI, W., WAMG, N., & YANG, C. (2021). A review on manipulation skill acquisition through teleoperation-based learning from demonstration. *Cognitive Computation and Systems, 3*, pp. 1-16.

SILVER, D., BAGNELL, A., & STENTZ, A. (2010). Learning from demonstration for autonomous navigation in complex unstructured terrain. *The International Journal of Robotics Research, 29*, pp. 1565-1592.

SPIERS, A., KHAN, S. G., & HERRMANN, G. (2016). Biologically inspired control of humanoid robot arms. *Springer International Publishing*, pp. XIX, 276.

STEINMETZ, F., NITSCH, V., & STULP , F. (2019). Intuitive task-level programming by demonstration through Semantic Skill Recognition. *Robotics and Automation Letters, 4*, pp. 3742–3749.

TEKA,, B., RAJA, R., & DUTTA, A. (2019). Learning based end effector tracking control of a mobile manipulator for performing tasks on an uneven terrain. *International Journal of Intelligent Robotics and Applications, 3*, pp. 102-114.

VIDAKOVIĆ, J., JERBIĆ, B., ŠEKORANJA, B., ŠVACO, M., & ŠULIGOJ, F. (2020). Learning from Demonstration Based on a Classification of Task Parameters and Trajectory Optimization. *Journal of Intelligent & Robotic Systems, 99*, pp. 261–275.

VÖGELE, A., KRÜGER, B., & KLEIN, R. (2017). Efficient Unsupervised Temporal Segmentation. *IEEE Transactions on Multimedia, 19*, pp. 797-812.

WANG, Y., JIAO, Y., XIONG, R., YU, H., ZHANG, J., & LIU, Y. (2018). A multimodal assembly skill decoding system for robot programming by demonstration. *Trans. Autom. Sci. Eng, 15*, pp. 1722-1734.

ZHANG, H.-D., LIU, S.-B., LEI, Q.-J., HE, Y., YANG, Y., & BAI, Y. (2020). Robot programming by demonstration: a novel system for robot trajectory programming based on robot operating system. *Advances in Manufacturing, 8*, pp. 216-229.

ZHU, Z., & HU, H. (2018). Robot Learning from Demonstration in Robotic Assembly: A Survey. *Robotics, 7*, p. 17.

APÊNDICE A – Amostragem de *features* compartilhados

Algorithm C.2 SampleSharedFeature($\boldsymbol{y}^{(i)}, \tilde{\boldsymbol{y}}^{(i)}, \boldsymbol{f}_i, m_k^{(-i)}, \theta, \eta^{(i)}, \alpha, c$)

MH proposal that attempts to flip single entry in binary matrix to its complement

Input:

$\boldsymbol{y}^{(i)}, \tilde{\boldsymbol{y}}^{(i)}$: observations and lagged observations for seq. i

$\boldsymbol{f}_i$: current binary feature vector for seq. i

$m_k^{(-i)}$: usage count of feature k in $\boldsymbol{F}$, ignoring sequence i

Output:

$f_{ik}^{(new)}$: new binary assignment for feature k in seq. i

Procedure:

1: Obtain proposed binary assignment

$$\boldsymbol{f}_i^* \leftarrow [f_{i1} \quad f_{i2} \quad \dots \quad \underset{\text{flip feat.}k}{\neg f_{ik}} \quad \dots \quad f_{iK_+}]$$

2: Compute probability of each feature configuration under IBP prior:

$$P^* \leftarrow f_{ik}^* p_{on} + (1 - f_{ik}^*)(1 - p_{on})$$

$$P \leftarrow f_{ik} p_{on} + (1 - f_{ik})(1 - p_{on}), \qquad p_{on} \leftarrow \frac{m_k^{(-i)}}{N - m_k^{(-i)} + c}$$

3: Construct feature-constrained transition distributions:

$$\pi_k^* \propto [\eta_{k1}^{(i)} \dots \eta_{kK_+}^{(i)}] \otimes \boldsymbol{f}_i^*, \qquad k \in \{k : f_{ik}^* = 1\}$$

$$\pi_k \propto [\eta_{k1}^{(i)} \dots \eta_{kK_+}^{(i)}] \otimes \boldsymbol{f}_i, \qquad k \in \{k : f_{ik} = 1\}$$

4: Compute likelihoods of observed data under each assignment [Alg. B.27]

$$L^* \leftarrow p(\boldsymbol{y}^{(i)} | \tilde{\boldsymbol{y}}^{(i)}, \boldsymbol{f}_i^*, \theta, \pi^*)$$

$$L \leftarrow p(\boldsymbol{y}^{(i)} | \tilde{\boldsymbol{y}}^{(i)}, \boldsymbol{f}_i, \theta, \pi)$$

5: Compute acceptance ratio $\rho = \frac{P^* L^*}{P L}$

6: Sample new assignment: rand $\sim$ Unif$(0, 1)$

$$f_{ik}^{(new)} \leftarrow \begin{cases} f_{ik}^* & \text{if rand } < \min(\rho, 1) \\ f_{ik} & \text{otherwise} \end{cases}$$

APÊNDICE B – Amostragem de sequência de estados

Algorithm C.3 SampleStateSequence($\boldsymbol{y}^{(i)}$, $\tilde{\boldsymbol{y}}^{(i)}$, $\boldsymbol{f}_i, \theta, \eta^{(i)}$)

Block sample discrete state assignments to each timestep in seq. i

Input:

$\boldsymbol{y}^{(i)}, \tilde{\boldsymbol{y}}^{(i)}$: observations and lagged observations for seq. i

$\boldsymbol{f}_i$: current binary feature vector for seq. i

$\theta = (\theta_1, \ldots \theta_{K_+})$: emission parameters (one set per feature)

$\theta_k = (A_k, \Sigma_k)$: VAR likelihood parameterized by coefs A_k and covariance Σ_k

Output:

$\boldsymbol{z}^{(i)}$: sample of i's discrete state sequence drawn from conditional posterior

Procedure:

1: Precompute soft evidence

for $t \in \{1, 2, \ldots T_i\}$:

$\ell_{tk} \triangleq p(\boldsymbol{y}_t^{(i)} | z_t^{(i)} = k) = \mathcal{N}(\boldsymbol{y}_t^{(i)} \mid \boldsymbol{A}_k \tilde{\boldsymbol{y}}_t^{(i)}, \ \Sigma_k) \quad k \in \{k : f_{ik} = 1\}$

2: Create feature constrained transition weights

$\pi_{k\cdot} \propto [\ \eta_{k1}^{(i)} \ldots \eta_{kK_+}^{(i)}\ ] \odot \boldsymbol{f}_i, \quad k \in \{k : f_{ik} = 1\}$

3: Initialize M: T_i-by-K_i table for dynamic programming

$M_{T_i,k} \leftarrow 1, \quad M_{t,k} \triangleq p(\boldsymbol{y}_{t+1:T_i}^{(i)} | z_t^{(i)} = k, \tilde{\boldsymbol{y}}_t^{(i)}, \pi, \theta)$

4: Backward pass of dynamic programming.

for $t \in \{T_i - 1, \ldots 1\}$:

for $k \in \{k : f_{ik} = 1\}$:

$M_{t,k} \leftarrow \sum_{k' : f_{ik'} = 1} [\pi_k(k') \cdot \ell_{t+1,k'} \cdot M_{t+1,k'}]$

5: Forward sample discrete state sequence (initialize in special start state $z_0^{(i)} = 0$):

for $t \in \{1, 2, \ldots T_i\}$:

$\vec{p}_k \leftarrow \pi_{z_{t-1}^{(i)}, k} \cdot \ell_{t,k} \cdot M_{t,k} \qquad$ for $k \in \{k : f_{ik} = 1\}$

$\vec{p} \leftarrow \vec{p} / \sum_k \vec{p}_k \qquad$ *normalize*

$z_t^{(i)} \sim \text{Cat}(\vec{p})$

APÊNDICE C – Amostragem dos parâmetros de emissão

Algorithm C.4 SampleEmissionParams($\boldsymbol{Y}_k, \tilde{\boldsymbol{Y}}_k, \lambda$)

Gibbs sampling algorithm for HMM emission parameter set θ_k for specific feature k

Input:

$\boldsymbol{Y}_k = \{y_t^{(i)} : z_t^{(i)} = k\}$: set of all observed data from any seq. assigned to feature k

$\tilde{\boldsymbol{Y}}_k = \{\tilde{y}_t^{(i)} : z_t^{(i)} = k\}$: set of all lagged observations from any seq. assigned to k

$\lambda = (n_0, S_0, M, K)$: MNIW prior hyperparameters

Output:

$\theta_k = (A_k, \Sigma_k)$: VAR likelihood parameters drawn from conditional posterior

Procedure:

1: Compute sufficient statistics $S_\cdot^{(k)}$ according to Eq. B.12.

2: Compute parameters of MNIW posterior distribution

degrees of freedom $n_k \leftarrow n_0 + |\boldsymbol{Y}_k|$

scale matrix $S_k \leftarrow S_{y|\tilde{y}}^{(k)}$

mean matrix $M_k \leftarrow S_{y\tilde{y}}^{(k)} [S_{\tilde{y}\tilde{y}}^{(k)}]^{-1}$

coef. scale matrix $K_k \leftarrow S_{\tilde{y}\tilde{y}}^{(k)}$

3: Sample new emission parameters $\theta_k = (A_k, \Sigma_k)$

$$\Sigma_k \sim \mathcal{IW}(n_k, S_k)$$

$$A_k \mid \Sigma_k \sim \mathcal{MN}(M_k, \Sigma_k, K_k)$$

APÊNDICE D – Amostragem de nascimento e morte de *features*

Algorithm D.1 SampleBirthDeath$(i, \boldsymbol{y}, \tilde{\boldsymbol{y}}, \boldsymbol{F}, \boldsymbol{z})$

Data-driven MH proposal that attempts to add/remove feature from seq. i

Input:

i : integer id of the sequence whose assignments we'll modify

$\boldsymbol{y}, \tilde{\boldsymbol{y}}$: observations and lagged observations for all sequences

$\boldsymbol{F} = \{\boldsymbol{f}_i\}_{i=1}^N$: binary feature assignments for all items

$\boldsymbol{z}$: state sequences for all items

Output:

$\boldsymbol{f}_i^{new}$, $\boldsymbol{z}^{(i)new}$: new assignments for item (sequence) i

Subprocedures:

$p_{\text{birth}}(\mathcal{U}_i)$: function that determines the probability of performing a birth move given the current configuration's set of unique feature ids $\mathcal{U}_i$

we set $p_{\text{birth}}(\mathcal{U}_i) = \begin{cases} \frac{1}{2} & \text{if } |\mathcal{U}_i| > 0 \\ 1 & \text{otherwise} \end{cases}$, though other possibilities exist

Procedure:

1: Identify set $\mathcal{U}_i$ of features unique to sequence i in $\boldsymbol{F}$

2: Select random contiguous window $\mathcal{W} = \{T_0, T_0+1, \ldots T_0+L-1\}$ of length L

$$L \sim \text{Unif}(L_{\text{min}}, L\text{max}), \quad T_0 | L \sim \text{Unif}(\{1, 2, \ldots T_i - L + 1\})$$

3: Choose either *birth* or *death* move

`movetype` $\sim$ Bern($p_{\text{birth}}(\mathcal{U}_i)$)

4: Propose new assignments using chosen move

keep track of forward and reverse transition probs Q

if `movetype` $= 1$: *birth move*

$\boldsymbol{f}_i^*, \boldsymbol{z}^{*(i)}, Q_{f-\text{fwd}}, Q_{z-\text{fwd}}, Q_{f-\text{rev}}, Q_{f-\text{rev}} \leftarrow$ `BirthProposal`($i, \boldsymbol{f}_i, \mathcal{U}_i, \boldsymbol{y}, \tilde{\boldsymbol{y}}, \boldsymbol{z}, \mathcal{W}$)

else: death move

$\boldsymbol{f}_i^*, \boldsymbol{z}^{*(i)}, Q_{f-\text{fwd}}, Q_{z-\text{fwd}}, Q_{f-\text{rev}}, Q_{f-\text{rev}} \leftarrow$ `DeathProposal`($i, \boldsymbol{f}_i, \mathcal{U}_i, \boldsymbol{y}, \tilde{\boldsymbol{y}}, \boldsymbol{z}, \mathcal{W}$)

5: Compute joint probability of current and proposed configurations (Eq. B.2)

$$L^* \leftarrow p(\boldsymbol{y}, \tilde{\boldsymbol{y}}, \boldsymbol{F}^*, \boldsymbol{z}^*)$$
$$L \leftarrow p(\boldsymbol{y}, \tilde{\boldsymbol{y}}, \boldsymbol{F}, \boldsymbol{z})$$

6: Decide whether to accept or reject candidate state

$$\rho = \frac{L^*}{L} \frac{Q_{z-\text{rev}} Q_{f-\text{rev}}}{Q_{z-\text{fwd}} Q_{f-\text{fwd}}}, \quad D \sim \text{Bern}\left(\min(\rho, 1)\right), \quad \boldsymbol{f}_i^{new}, \boldsymbol{z}^{(i)new} = \begin{cases} \boldsymbol{f}_i^*, \boldsymbol{z}^{*(i)} & \text{if } D = 1 \\ \boldsymbol{f}_i, \boldsymbol{z} & \text{o.w.} \end{cases}$$

APÊNDICE E – Amostragem de pesos de transição

Algorithm C.5 SampleTransitionWeights($\boldsymbol{f}_i, \boldsymbol{z}^{(i)}, \gamma, \kappa$)

Gibbs sampler for HMM transition weight set $\eta^{(i)}$ for available features of seq. i

Input:

$\boldsymbol{z}^{(i)} = [\, z_1^{(i)}, z_2^{(i)}, \ldots z_{T_i}^{(i)} \,]$: state trajectory for seq. i, $\quad z_t^{(i)} \in \{k : f_{ik} = 1\}$

γ : transition hyperparameter (pseudocounts for each possible transition)

κ : self-transition bias hyperparameter

Output:

$\eta^{(i)}$: set of transition weights drawn from conditional posterior

Procedure:

1: Compute sufficient statistics above Markovian transitions in $\boldsymbol{z}^{(i)}$

$$N_{kk'} = \sum_{t=2}^{T_i} \delta_{z_{t-1}^{(i)},k} \delta_{z_t^{(i)},k'} \quad \text{for } k, k' \in \{k : f_{ik} = 1\}$$

2: **for** each feature $k \in \{k : f_{ik} = 1\}$:

Sample constrained transition distribution for each feature:

$$\pi_{k,\cdot} \sim \text{Dir}(\, [\, \ldots N_{kk'} + \gamma + \delta_{k,k'}\kappa \ldots \,] \odot \boldsymbol{f}_i \,)$$

Sample scale factor

$$C_k \sim \text{Gamma}(K_i\gamma + \kappa)$$

Apply scale factor to distribution to create unconstrained transition weight

$\eta_{k,k'}^{(i)} \leftarrow C_k \cdot \pi_{kk'}$

APÊNDICE F – Amostragem dos Hiper parâmetros do processo beta

Algorithm F.1 SampleBPHypers()

Input:

$\boldsymbol{F} = \{\boldsymbol{f}_i\}$: binary feature assignments for all time series

α, c : IBP hyperparameters

Output:

α, c : new HMM hyperparameter values drawn from conditional posterior

Sampler Algorithm Settings:

σ_c^2 : proposal distribution variance

R_{IBP} : number of iterations to attempt proposals

Procedure:

1: **for** iterations $r = 1, 2, \ldots R_{\text{IBP}}$:

2: Sample $\alpha | c, \boldsymbol{F}$

$$\alpha \sim \text{Gamma}\left(a_\alpha + K_+, b_\alpha + \sum_{n=1}^{N} \frac{c}{c+n-1}\right)$$

3: Sample $c | \alpha, \boldsymbol{F}$

$$c' | c \sim \text{Gamma}(\text{mean} = c, \text{var} = \sigma_c^2)$$

$$c \leftarrow \begin{cases} c' \text{ with prob } \min(1, r), & r \text{ defined in Eq. } F.6 \\ c \text{ otherwise} \end{cases}$$

APÊNDICE G – Amostragem dos Hiper parâmetros do modelo HMM

Algorithm F.2 SampleHMMTransitionHypers()

Input:

$\boldsymbol{F} = \{\boldsymbol{f}_i\}$: binary feature assignments for all time series

$\boldsymbol{\eta} = \{\eta^{(i)}\}$: HMM transition weights for all time series

γ, κ : HMM transition hyperparameters

Output:

γ, κ : new HMM hyperparameters drawn from conditional posterior

Sampler Algorithm Settings:

$\sigma_\gamma^2, \sigma_\kappa^2$: proposal distribution variances

R_{HMM} : number of iterations to attempt proposals

Procedure:

1: **for** iterations $r = 1, 2, \ldots R_{\text{HMM}}$:

2: Sample $\gamma|\kappa, \boldsymbol{F}, \boldsymbol{\eta}$

$$\gamma'|\gamma \sim \text{Gamma}(\text{mean} = \gamma, \text{var} = \sigma_\gamma^2)$$

$$\gamma \leftarrow \begin{cases} \gamma' \text{ with prob. } \min(1, r_\gamma), & r_\gamma \text{ defined in Eq. } F.11 \\ \gamma \text{ otherwise} \end{cases}$$

3: Sample $\kappa|\gamma, \boldsymbol{F}, \boldsymbol{\eta}$

$$\kappa'|\kappa \sim \text{Gamma}(\text{mean} = \kappa, \text{var} = \sigma_\kappa^2)$$

$$\kappa \leftarrow \begin{cases} \kappa' \text{ with prob. } \min(1, r_\kappa), & r_\kappa \text{ defined in Eq. } F.12 \\ \kappa \text{ otherwise} \end{cases}$$