



MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
GOIANO - CAMPUS RIO VERDE

HIGOR HENRIQUE LOPES GUIMARÃES

**TÉCNICAS DE VISÃO COMPUTACIONAL A PARTIR DA
DETECÇÃO DE PADRÕES COMPRIMIDOS EM IMAGENS
DIGITAIS**

RIO VERDE

2019



HIGOR HENRIQUE LOPES GUIMARÃES

**TÉCNICAS DE VISÃO COMPUTACIONAL A PARTIR DA
DETECÇÃO DE PADRÕES COMPRIMIDOS EM IMAGENS
DIGITAIS**

Trabalho de Conclusão de Curso apresentado ao Instituto Federal de Educação, Ciência e Tecnologia Goiano - Campus Rio Verde ligado ao Ministério da Educação, como requisito parcial da disciplina Tópicos de Pesquisa em Computação para a obtenção do título de Bacharel em Ciência da Computação.

Orientador: Adriano Soares de Oliveira Bailão
Instituto Federal de Educação, Ciência e
Tecnologia Goiano - Campus Rio Verde

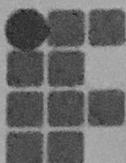
RIO VERDE
2019

Sistema desenvolvido pelo ICMC/USP
Dados Internacionais de Catalogação na Publicação (CIP)
Sistema Integrado de Bibliotecas - Instituto Federal Goiano

G963t Guimarães, Higor Henrique Lopes
 TÉCNICAS DE VISÃO COMPUTACIONAL A PARTIR DA
DETECÇÃO DE PADRÕES COMPRIMIDOS EM IMAGENS DIGITAIS /
Higor Henrique Lopes Guimarães; orientador Adriano
Soares de Oliveira Bailão. -- Rio Verde, 2019.
 47 p.

 Monografia (em Ciência da Computação) --
 Instituto Federal Goiano, Campus Rio Verde, 2019.

 1. Reconhecimento de Padrões. 2. Visão
 Computacional. 3. Inteligência Artificial. 4.
 Compressão. 5. Classificação de Imagens. I. Bailão,
 Adriano Soares de Oliveira, orient. II. Título.



ATA DE DEFESA DO TRABALHO DE CURSO (TC)

ANO	SEMESTRE
2019	02

No dia 16 do mês de dezembro de 2019, às 09 horas e 30 minutos, reuniu-se a banca examinadora composta pelos docentes Adriano Bailão, Fábio Montano Ramos e Douglas Cedrim Oliveira, para examinar o Trabalho de Curso (TC) intitulado Técnicas de Visão Computacional a Partir da detecção de Padrões Comprimidos em imagens digitais. do(a) acadêmico(a) Higer Henrique Lopes Guimarães, Matrícula nº 2016102192010269 do curso de Ciência da Computação do IF Goiano – Câmpus Rio Verde. Após a apresentação oral do TC, houve arguição do candidato pelos membros da banca examinadora. Após tal etapa, a banca examinadora decidiu pela APROVAÇÃO do(a) acadêmico(a). Ao final da sessão pública de defesa foi lavrada a presente ata, que segue datada e assinada pelos examinadores.

Rio Verde, 16 de dezembro de 19.

Adriano S. J. Bailão

Nome:
Orientador(a)

Fábio Montano Ramos

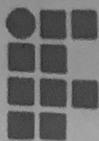
Nome:
Membro

Cláudio Henrique Amorim

Nome:
Membro

Observação:

() O(a) acadêmico(a) não compareceu à defesa do TC.



TERMO DE CIÊNCIA E DE AUTORIZAÇÃO PARA DISPONIBILIZAR PRODUÇÕES TÉCNICO-CIENTÍFICAS NO REPOSITÓRIO INSTITUCIONAL DO IF GOIANO

Com base no disposto na Lei Federal nº 9.610/98, AUTORIZO o Instituto Federal de Educação, Ciência e Tecnologia Goiano, a disponibilizar gratuitamente o documento no Repositório Institucional do IF Goiano (RIIF Goiano), sem ressarcimento de direitos autorais, conforme permissão assinada abaixo, em formato digital para fins de leitura, download e impressão, a título de divulgação da produção técnico-científica no IF Goiano.

Identificação da Produção Técnico-Científica

- Tese
- Artigo Científico
- Dissertação
- Capítulo de Livro
- Monografia - Especialização
- Livro
- TCC - Graduação
- Trabalho Apresentado em Evento
- Produto Técnico e Educacional - Tipo: _____

Nome Completo do Autor: HEGOR HENRIQUE LOPES GUIMARÃES
 Matrícula: 2016102192010263
 Título do Trabalho: TÉCNICAS DE VISÃO COMPUTACIONAL A PARTIR DA DETECÇÃO DE PADRÕES COMPLEXOS DOS EM IMAGENS DIGITAIS

Restrições de Acesso ao Documento

Documento confidencial: Não Sim, justifique: _____

Informe a data que poderá ser disponibilizado no RIIF Goiano: 14/02/2020

O documento está sujeito a registro de patente? Sim Não
 O documento pode vir a ser publicado como livro? Sim Não

DECLARAÇÃO DE DISTRIBUIÇÃO NÃO-EXCLUSIVA

- O/A referido/a autor/a declara que:
- o documento é seu trabalho original, detém os direitos autorais da produção técnico-científica e não infringe os direitos de qualquer outra pessoa ou entidade;
 - obteve autorização de quaisquer materiais inclusos no documento do qual não detém os direitos de autor/a, para conceder ao Instituto Federal de Educação, Ciência e Tecnologia Goiano os direitos requeridos e que este material cujos direitos autorais são de terceiros, estão claramente identificados e reconhecidos no texto ou conteúdo do documento entregue;
 - cumpriu quaisquer obrigações exigidas por contrato ou acordo, caso o documento entregue seja baseado em trabalho financiado ou apoiado por outra instituição que não o Instituto Federal de Educação, Ciência e Tecnologia Goiano.

Rio Verde/GO, 14/02/2020
Local Data

Higor Henrique L. Guimarães
Assinatura do Autor e/ou Detentor dos Direitos Autorais

Ciente e de acordo:
[Assinatura]
Assinatura do(a) orientador(a)

RESUMO

GUIMARÃES, Higor. Técnicas de Visão Computacional a partir da Detecção de Padrões Comprimidos em Imagens Digitais. 2019. 47 f. Trabalho de Conclusão de Curso – Bacharelado em Ciência Da Computação, Instituto Federal de Educação, Ciência e Tecnologia Goiano - Campus Rio Verde. Rio Verde, 2019.

A visão computacional, a ciência que busca emular algum aspecto da visão humana com uso de hardware e/ou software, tem tido enorme contribuição nas últimas décadas. Este trabalho apresenta uma nova abordagem livre de parâmetros baseada na compressão de dados para a construção de um modelo genérico de representação de imagens através da adaptação e extensão da metodologia DAMICORE. Foi desenvolvida uma aplicação com as técnicas propostas e foram realizados testes com alguns conjuntos de imagens, bons resultados foram obtidos, e no geral apresentam boa capacidade de generalização. Apesar de bons resultados não terem sido obtidos em todas as categorias de experimentos realizados, modificações na abordagem são apresentadas como trabalho futuro, além de ideias sobre a extensão da técnica para tarefas de classificação e para outros tipos de dados.

Palavras-chave: Reconhecimento de Padrões. Inteligência Artificial. Compressão. Visão Computacional. Classificação de Imagens. DAMICORE.

ABSTRACT

GUIMARÃES, Higor. Computer Vision Techniques from the Detection of Compressed Patterns in Digital Images. 2019. 47 f. Trabalho de Conclusão de Curso – Bacharelado em Ciência Da Computação, Instituto Federal de Educação, Ciência e Tecnologia Goiano - Campus Rio Verde. Rio Verde, 2019.

Computer vision, the science that seeks to emulate aspects of human vision through the use of hardware and/or software, had massive contribution in past decades. This work presents a new data compression based and parameter-free approach for the construction of a generic model of representation of images through the adaptation and extension of the DAMICORE methodology. An application was developed, and tests ran with some image sets, good results were obtained, and in general, good generalization capacity is shown. Though good results were not obtained in all experiment categories, possible modifications in the technique are presented as future work, besides ideas about the extension of the technique to classification tasks and to other kinds of data.

Keywords: Pattern Recognition. Computer Vision. Artificial Intelligence. Compression. Image Classification, DAMICORE.

LISTA DE FIGURAS

Figura 1 – Visão Humana	3
Figura 2 – Visão Computacional	3
Figura 3 – Comparação entre códigos de tamanho fixo e variável	6
Figura 4 – Exemplo de árvore binária de decisão para previsão de caractere em posição aleatória na <i>string</i> '01002301'	10
Figura 5 – Exemplos de imagens coletadas na categoria de frutas	15
Figura 6 – Exemplos de imagens coletadas na categoria de plantações	16
Figura 7 – Exemplos de imagens coletadas na categoria de animais	16
Figura 8 – Exemplos de imagens coletadas na categoria de testes CAPTCHA	18
Figura 9 – Exemplos de imagens coletadas na categoria de faces humanas	18
Figura 10 – Exemplos de imagens coletadas na categoria de pinturas	20
Figura 11 – Comparação do fluxo operacional	21
Figura 12 – Pintura do momento cubismo - Comparações	22
Figura 13 – Plantação de girassóis - Comparações	23
Figura 14 – Pré-processamento da imagem	24
Figura 15 – Exemplo: árvore original e árvore a ser avaliada	28
Figura 16 – Médias da variação da informação	37
Figura 17 – Comparação dos resultados com diferentes números de cores na etapa de quantização	38
Figura 18 – Comparação dos resultados	40
Figura 19 – Comparação dos piores e melhores resultados	41
Figura 20 – Diferença entre resultados mínimos e máximos obtidos com a aplicação proposta e com a ferramenta damicorepy	42

LISTA DE TABELAS

Tabela 1 – Comparação entre códigos de tamanho fixo e variável	7
Tabela 2 – Probabilidades dos símbolos (<i>string</i> '01002301')	9
Tabela 3 – Conjunto de dados 1 - Frutas	14
Tabela 4 – Conjunto de dados 2 - Plantações	14
Tabela 5 – Conjunto de dados 3 - Animais	17
Tabela 6 – Conjunto de dados 4 - CAPTCHA	17
Tabela 7 – Conjunto de dados 5 - 'ORL Database Of Faces'	18
Tabela 8 – Conjunto de dados 7 - Movimentos Artísticos	19
Tabela 9 – Resultados - Frutas	31
Tabela 10 – Resultados - Animais	32
Tabela 11 – Resultados - Plantações	33
Tabela 12 – Resultados - CAPTCHA	34
Tabela 13 – Resultados - Faces Humanas	35
Tabela 14 – Resultados - Pinturas	36
Tabela 15 – Médias da variação da informação	37
Tabela 16 – Média de resultados com diferentes números de cores	39

LISTA DE ABREVIATURAS E SIGLAS

RP	Reconhecimento de Padrões
DAMICORE	Data-Mining of Code Repositories
NID	Normalized Information Distance
NCD	Normalized Compression Distance
NJ	Neighbor Joining
FN	Fast Newman
CPE	Curva Preenhedora de Espaço
MIT	Massachusetts Institute of Technology
VLC	Variable Length Code
CAPTCHA	Completely Automated Public Turing test to tell Computers and Humans Apart
KNN	K Nearest Neighbors
JPEG	Joint Photographic Experts Group

SUMÁRIO

1 – INTRODUÇÃO	1
2 – FUNDAMENTAÇÃO TEÓRICA	3
2.1 Visão computacional	3
2.2 Aprendizado de Máquina através de uma métrica	4
2.3 DAMICORE	4
2.3.1 Compressão/Codificação dos dados	5
2.3.1.1 Compressão e Codificação de Huffman	5
2.3.2 Normalized Compression Distance - NCD	7
2.3.3 Neighbor Joining - NJ	8
2.3.4 Fast Newman - FN	8
2.4 Entropia da Informação	9
2.5 Trabalhos Relacionados	11
2.5.1 Estudo e extensão da metodologia DAMICORE para tarefas de classificação	11
2.5.2 UTILIZANDO ÁRVORES FILOGENÉTICAS PARA A IDENTI- FICAÇÃO DE SIMILARIDADES EM PINTURAS DIGITALIZADAS	12
3 – MATERIAIS E MÉTODOS	13
3.1 Área de Estudo	13
3.2 Materiais	13
3.2.1 Conjunto de imagens de frutas	14
3.2.2 Conjunto de Imagens de Plantações	14
3.2.3 Conjunto de Imagens de Animais	15
3.2.4 Conjunto de Imagens de CAPTCHA	15
3.2.5 Conjunto de Imagens de Faces Humanas - 'ORL Database Of Faces'	17
3.2.6 Conjunto de Pinturas (Diferentes Movimentos Artísticos)	19
3.3 Construção do Modelo de Inteligência	20
3.3.1 Extração de Características	20
3.3.1.1 Combinação de Objetos	25
3.3.2 Codificação das características	25
3.3.3 Cálculo de distâncias com a NCD	27
3.3.4 Formação da árvore filogenética com o NJ	27
3.3.5 Avaliação dos resultados	28
4 – RESULTADOS OBTIDOS	30

4.1	Resultado dos experimentos (Imagens quantizadas para 32 cores)	30
4.1.1	Frutas	30
4.1.2	Animais	31
4.1.3	Plantações	31
4.1.4	CAPTCHA	34
4.1.5	Faces Humanas	34
4.1.6	Pinturas	34
4.2	Resultados da variação de informação (ϵ) nas imagens com a quantização para 32 cores	36
4.3	Resultados empíricos (diferentes valores na etapa de quantização)	37
4.4	Comparações com a ferramenta <i>damicorepy</i>	39
5	– CONCLUSÃO	43
5.1	Trabalhos Futuros	44
5.1.1	Possíveis modificações no modelo de representação das imagens .	44
5.1.2	Extensão para Classificação	44
5.1.3	Extensão para outros tipos de dados	45
	Referencias Bibliográficas	46

1 INTRODUÇÃO

O grande volume de imagens digitais proporcionado pela era da informação tem as suas utilidades e aplicações, principalmente com o uso de ferramentas computacionais capazes, de assim como o homem com a sua visão, extrair conhecimento dessas informações visuais. Tais ferramentas podem por exemplo: permitir a navegação de veículos autônomos, busca de imagens através da associação de similaridades, identificação facial e até mesmo realizar processamento médico de imagens.

Um grande problema é que as ferramentas disponíveis para a análise de dados, inclusive para imagens, são direcionadas a domínios específicos, com restrição a poucas tarefas e portanto possuem um custo muito alto. Nesse contexto, surge uma metodologia genérica capaz de funcionar livre de parâmetros para qualquer tipo de dado chamada DAMICORE(SANCHES; CARDOSO; DELBEM, 2011). Esse método tem demonstrado bons resultados no Reconhecimento de Padrões (RP) em diferentes domínios(CESAR, 2016; TORRES, 2018) e um grande potencial de generalização.

O framework é uma técnica de agrupamento genérico, livre de parâmetros, capaz de produzir agrupamentos para qualquer tipo de dado usando como métrica uma técnica da teoria da informação chamada de **distância normalizada de compressão** (*Normalized Compression Distance - NCD*). Para isso se utiliza *compactadores* para a medição de informações em objetos de fontes de dados não estruturados. Compactadores utilizam técnicas adaptativas e estatísticas de compressão *sem perdas*. Embora, frameworks baseados em *compactadores* como o DAMICORE possuam alta capacidade de generalização para o tratamento das fontes de dados em aplicações de RP, o modelo de representação dos objetos de dados com base neste tipo de aproximação torna a abordagem dependente do algoritmo que o compactador encapsula, levando a implicações como por exemplo, ambiguidade na representação das amostras, falta de precisão em problemas de classificação e resultados indeterminísticos.

Uma forma de contornar o problema é desmembrando o algoritmo implementado pelo compactador em duas fases, uma etapa probabilística encarregada da identificação e contabilização das ocorrências dos componentes elementares de um objeto de dados e outra etapa responsável por sua codificação. Este trabalho apresenta um **método de agrupamento de imagens** através da adaptação e extensão do fluxo de operação do framework DAMICORE de forma a direcionar o *bias* de uma aplicação de RP a partir de uma fase para *Extração de características* e outra fase chamada *Codificação* além da extensão do modelo para a aplicação de técnicas de compressão *com perdas*, com o objetivo de maior generalização para diferentes domínios. Como resultado, ganho qualitativo na formação e visualização dos agrupamentos e melhora da delimitação das fronteiras (limites de decisão) entre grupos de objetos.

Foram coletadas imagens de seis categorias diferentes para realização de testes com um protótipo desenvolvido durante a pesquisa. Entre os experimentos realizados foi testado a capacidade do algoritmo de reconhecer e distinguir faces humanas, pinturas de diferentes movimentos artísticos, plantações, animais, frutas e imagens de teste CAPTCHA.

Bons resultados foram obtidos com a técnica proposta, e uma comparação com o fluxo de operações tradicional mostra que a extensão proposta alcançou maior capacidade de generalização, proporcionando resultados melhores em todos as categorias de experimentos realizados.

O restante do documento está organizado da seguinte forma:

- O capítulo 2 apresenta conceitos e fundamentos por trás da aplicação desenvolvida, como a visão computacional e a metodologia DAMICORE. Além disso também contém uma breve revisão de trabalhos relacionados;
- O capítulo 3 apresenta os conjuntos de imagens coletados para teste, descreve o funcionamento e detalhes de implementação da aplicação proposta, e também o método de avaliação dos resultados.;
- O capítulo 4 apresenta os resultados obtidos com a aplicação desenvolvida com os conjuntos de dados coletados, além disso também é feita uma comparação com a metodologia DAMICORE tradicional;
- O capítulo 5 apresenta considerações finais sobre a contribuição do trabalho, e também, ideias para possíveis pesquisas e experimentações futuras.

2 FUNDAMENTAÇÃO TEÓRICA

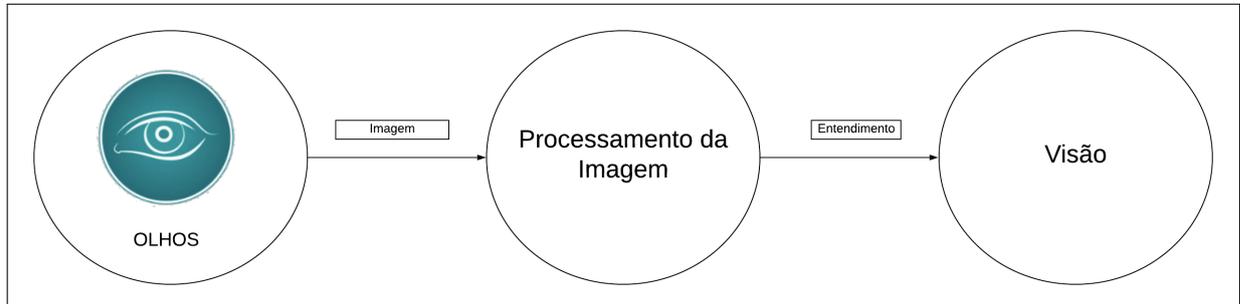


Figura 1 – Visão Humana

Diagrama autoral feito com a ferramenta lucid chart, imagem utilizada disponível no link: <https://www.livescience.com/60752-human-senses.html>

A visão, assim como qualquer outra percepção que temos das coisas ao nosso redor, está diretamente ligada aos nossos sentidos, sentidos que atuam como sensores na captação de informações ao nosso redor e fornecem essa informação como entrada para um processamento em nosso cérebro para que então possamos experienciar algo como a visão (WILDING, 2017; NORVIG; RUSSELL, 2014).

Na visão humana, tem-se os olhos como sensores que capturam uma imagem, e o cérebro que processa essa imagem, nos levando a experiência da visão (Figura 1). Hoje os computadores possuem acesso a um enorme conjunto de imagens proporcionado pela invenção das câmeras digitais, porém, processá-las para gerar entendimento de alguma forma tem se mostrado uma tarefa muito mais árdua (VISION, 2015) (Figura 2).

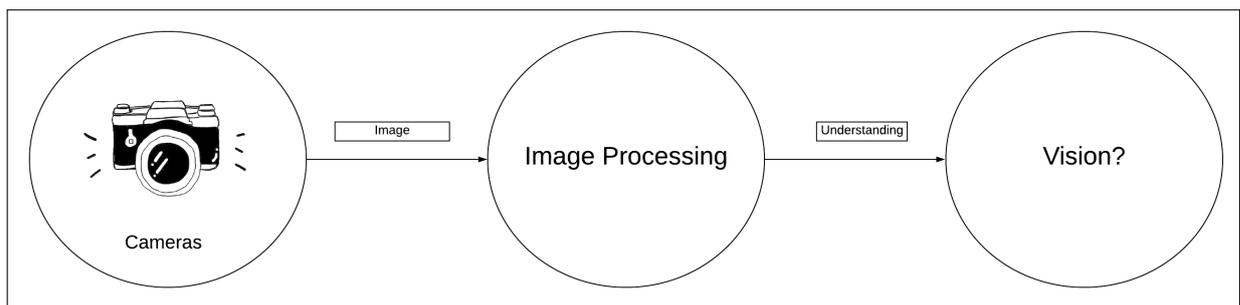


Figura 2 – Visão Computacional

Diagrama autoral feito com a ferramenta lucid chart

2.1 Visão computacional

Máquinas são cegas por natureza. Assim como perceber um som não é o mesmo que estar consciente do que se ouve, ter acesso a informação visual como uma imagem

não é o mesmo que vê-la. A visão em humanos começa nos olhos, mas o lugar em que realmente acontece é no cérebro. (LI, 2015)

A **visão computacional** é a ciência que estuda como as máquinas podem enxergar e entender informações visuais como um ser humano, por exemplo:

O homem, ao olhar uma flor sabe claramente que aquilo capturado por seus olhos é uma flor. A ciência da visão artificial por outro lado, busca transpor tal habilidade para um computador.

Uma câmera consegue agir como um olho e capturar a informação de uma cena em uma imagem ao guardar esses dados em matrizes bidimensionais cujo valores chamamos de pixels. Mas como processar essa informação visual e gerar algum entendimento da imagem por parte do computador?

Nas últimas décadas muitos estudos e contribuições tem sido feitos para esse objetivo, e um grande problema para aplicações de visão computacional é que as soluções existentes mais eficientes, incluindo o estado da arte, as Redes Neurais Artificiais (RNAs), sofrem com grande viés em seus resultados, isso por dependerem de configurações prévias bem específicas para atuar bem em um domínio.

Este trabalho busca uma abordagem mais genérica para o problema ao usar um modelo de representação de dados baseado em um histograma das imagens, imagens pré-processadas com uma quantização para redução de cores e maior generalização. Outros modelos de representação são sugeridos como trabalhos futuros no capítulo de conclusão.

2.2 Aprendizado de Máquina através de uma métrica

Para se pensar em uma estratégia que faça com que um computador aprenda a reconhecer e classificar dados com certos padrões, tratando-se não apenas de imagens, mas de qualquer tipo de informação, normalmente se faz uso de uma métrica para calcular quão distante está uma informação de outra.

Sobre a escolha de uma métrica:

”Na área de Aprendizado de Máquina, diversos métodos dependem da definição de uma métrica para fornecer a distância entre instâncias de dados. A escolha da métrica é um dos passos mais sensíveis ao desenvolver um algoritmo efetivo, dado que ela define como os elementos são similares/dissimilares entre si e irá impactar em como elementos são considerados “próximos” ou “distantes” ao obter agrupamentos e/ou classificar instâncias.”(CESAR, 2016, p. 28)

2.3 DAMICORE

A metodologia de mineração de dados DAMICORE, introduzida em 2011 (SANCHES; CARDOSO; DELBEM) faz uso da *Normalized Compression Distance* (NCD), a

métrica de distância de compressão normalizada e tem demonstrado bons resultados em diversos domínios (SILVA; SALVINI, 2018; MORO et al., 2014; TORRES, 2018; CESAR, 2016; SANCHES; CARDOSO; DELBEM, 2011; LOPES; DELBEM, ; PINTO; DELBEM; MONACO, 2017).

A metodologia faz uso de um conjunto de soluções de campos como a NCD da Teoria da Informação, o Neighbor Joining (NJ) da Filogenia e o Fast Newman (FN) das Redes Complexas (TORRES, 2018; CESAR, 2016). Inicialmente os dados a serem analisados por esse método devem ser comprimidos através de algum compressor, esses dados comprimidos são utilizados para criar uma **matriz de distâncias** com a métrica NCD, essa matriz gerada é então utilizada para a geração de uma **árvore filogenética** que expressa distância entre cada um dos dados envolvidos, e por fim, o algoritmo FN faz a divisão da árvore em **grupos**.

De forma resumida esse procedimento pode ser descrito em 4 etapas:

- Compressão/Codificação dos dados para servirem de entrada para a NCD;
- Criação de uma matriz de distâncias com a **NCD**;
- Geração de uma árvore filogenética para representação de distância usando o **NJ**;
- Detecção de comunidades na árvore com o **FN**.

2.3.1 Compressão/Codificação dos dados

O cálculo da NCD é feito em cima do tamanho de uma representação unidimensional normalmente comprimida ou codificada, dos dados. SANCHES; CARDOSO; DELBEM, (2011) ao fazerem uso da métrica para calcular distâncias entre códigos fonte na linguagem C, utilizam uma tabela de transformações para gerar o que chamam de **representação DNA** para cada código e ainda comparam os resultados dessa abordagem com a utilização do código binário do programa sem nenhuma transformação, que acaba produzindo melhores resultados.

Acontece que essa abordagem da representação DNA, tem os seus problemas: comumente gera ambiguidade na representação dos dados e conseqüentemente falta de precisão em problemas de classificação. Este trabalho propõe uma etapa chamada de **extração de características** onde será feito um pré-processamento em cada imagem para um cálculo probabilístico da ocorrência de cada valor de pixel presente na imagem, e posteriormente em outra etapa de **codificação**, esses valores probabilísticos serão ainda comprimidos usando o algoritmo de Huffman, gerando uma representação que com base na entropia da informação (SHANNON, 1948), será utilizada para produzir a representação final da imagem.

2.3.1.1 Compressão e Codificação de Huffman

O algoritmo de codificação e compressão de Huffman é um método se aproveita da frequência de símbolos no conjunto de dados a serem comprimidos, para determinar

códigos binários que representem tais símbolos de forma que os símbolos mais frequentes utilizem menos bits para serem armazenados, buscando conseguir armazenar a mesma informação **sem nenhuma perda**, com um espaço de armazenamento menor.

A codificação foi descrita por David Huffman em seu trabalho de doutorado no MIT (HUFFMAN, 1952), intitulado *A Method for the Construction of Minimum-Redundancy Codes*. O procedimento recebe um conjunto de símbolos (características de uma informação), como por exemplo os caracteres de um texto, e constrói uma representação comprimida e codificada em binário do mesmo conjunto sem perder nenhuma informação sobre o mesmo, essa representação construída pelo algoritmo é descrita como "Código de Redundância Mínima", pois quanto maior a redundância de um símbolo, menor será o espaço ocupado por sua representação.

Isso é possível devido a codificação usar como código de representação, prefixos de acesso a informações organizadas em uma árvore binária chamada árvore de Huffman, esses prefixos são comumente chamados de códigos de tamanhos variáveis (do inglês *Variable Length Code - VLC*).

A figura 3 e a tabela 1 mostram uma comparação entre a codificação de tamanho fixo e de tamanho variável com o algoritmo de Huffman. O exemplo utilizado¹ é um texto de 100 caracteres com apenas 6 caracteres diferentes entre si: 'a', 'b', 'c', 'd', 'e' e 'f'. Como resultado da comparação temos que com uma codificação tradicional de tamanho fixo são utilizados 300000 bits para armazenar a string, enquanto com a representação de tamanho variável do algoritmo de huffman, apenas 224000 bits, representando pouco mais de 25% em economia de memória.

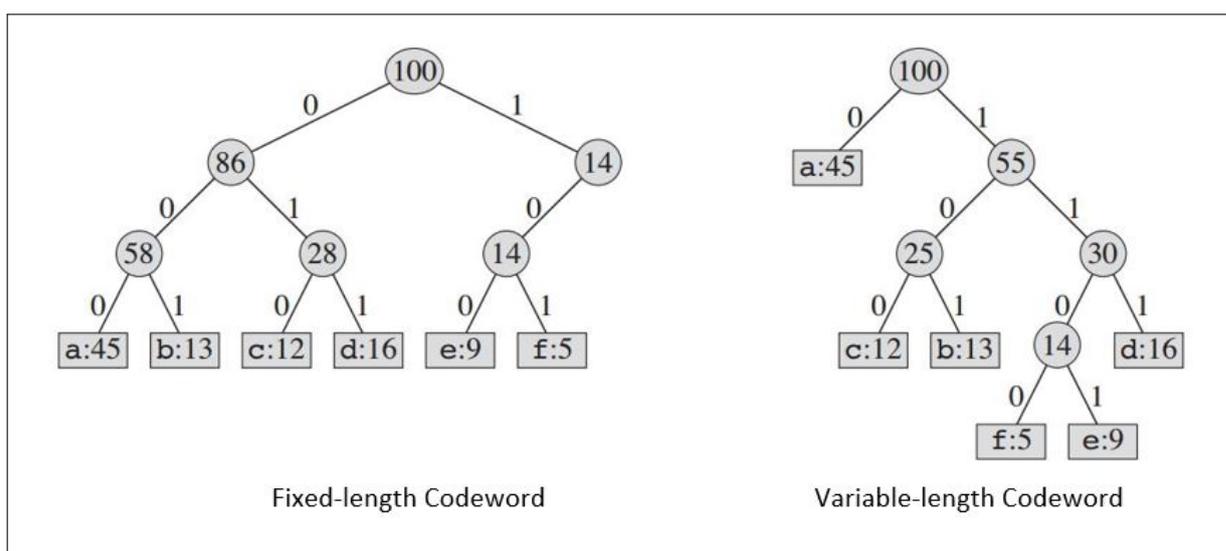


Figura 3 – Comparação entre códigos de tamanho fixo e variável¹

¹ Conteúdo do exemplo(ideia, imagem e tabela) disponível no seguinte endereço sobre a licença CC BY-SA 3.0: <<https://riptutorial.com/algorithm/example/23995/huffman-coding>>

Tabela 1 – Comparação entre códigos de tamanho fixo e variável¹

Caractere	a	b	c	d	e	f
Frequência	45	13	12	16	9	5
Código de Tamanho Fixo	000	001	010	011	100	101
Código de Tamanho Variável(Huffman)	0	101	100	111	1101	1100

O princípio dessa compressão é usar um número menor de bits para representar símbolos que aparecem com mais frequência. A codificação Huffman usa um método específico para escolher a representação de cada símbolo, resultando em uma árvore de prefixos que expressa os caracteres mais comuns usando cadeias de bits menores do que originalmente. (SHARMA, 2010)

2.3.2 Normalized Compression Distance - NCD

A *Normalized Compression Distance* é uma métrica para se medir a distância entre dados baseada na compressão dos mesmos. A NCD faz uso das informações em codificação binária produzida por um compressor, como por exemplo os compressores *gzip*, *ppmd* e *bzip2*, utilizados por CESAR, ou a codificação de Huffman, utilizado por este trabalho.

Sobre a NCD:

”Para compreendermos a essência do NCD precisamos conhecer o conceito de similaridade de objetos, dizemos que dois objetos x e y são similares se x pode ser comprimido utilizando a informação expressiva extraída a partir de y , e vice-versa. Esta ideia de informação está diretamente relacionada ao conceito de entropia oriundo da Teoria da Informação, a entropia representa a quantidade de informação necessária para predizer uma resposta, neste caso, se a entropia de x for expressiva em relação a y , então, torna-se mais fácil descrever y apenas referenciando x . A fonte de entropia utilizada para descrever essas relações será dada a partir de um compressor.” (TORRES, 2018, p. 29)

A distância normalizada de compressão entre objetos é calculada com base na teoria da complexidade de Kolmogorov: A complexidade de Kolmogorov $K(x)$ de um objeto x , é o comprimento da **menor** string que descreve o objeto x em uma linguagem turing completa. (adaptado de TORRES, 2018).

Suponhamos duas informações, x e y :

A partir de $K(x)$ e $K(y)$, poderíamos calcular a *Normalized Information Distance* (NID), a distância normalizada das informações. Infelizmente, a complexidade de Kolmogorov de uma informação x , $K(x)$ é incomputável. Porém, devido as suas diversas aplicações, um aproximação baseada na compressão foi apresentada por CILIBRASI; VITÁNYI (2005)

e demonstrado bons resultados ao ser utilizada para medir distância entre informações em diferentes domínios (LAN; HARVEY, 2005; AXELSSON, 2010; HUDSON et al., 2015).

Com essa aproximação, ao invés de tentar calcular o tamanho da menor representação possível de uma informação \mathbf{x} , calcula-se o tamanho da representação comprimida de \mathbf{x} , denominada $\mathbf{C}(\mathbf{x})$, utilizando algum compactador. Com essa aproximação conseguimos calcular a NCD com a seguinte equação:

$$NCD(x,y) = \frac{C(xy) - \min(C(x), C(y))}{\max(C(x), C(y))} \quad (1)$$

Onde xy representa a concatenação de x e y . Essa distância é normalmente calculada para todos elementos do conjunto de dados selecionado, formando uma matriz de distâncias.

2.3.3 Neighbor Joining - NJ

O Neighbor Joining (Junção de Vizinhos) é um algoritmo utilizado na bioinformática que recebe como entrada uma matriz de distâncias de diferentes espécies, e fornece como saída uma árvore filogenética que expressa ideias de ancestralidade e descendência, o que podemos considerar em outras palavras como distância entre as espécies envolvidas. A matriz de distâncias é uma matriz $n \times n$, sendo n a quantidade de espécies. O algoritmo tenta gerar uma árvore de evolução mínima, o que nem sempre consegue, ainda assim é um dos algoritmos mais eficientes para essa tarefa.(REIS; ZUBEN, 2007)

Sobre o algoritmo Neighbor Joining:

”O algoritmo trabalha com uma ”árvore-estrela (portanto, uma árvore sem raiz). É realizada uma junção de dois nós da árvore-estrela, os quais são substituídos, na árvore-estrela, pelo seu ancestral comum. Com isso, a cada junção, a árvore-estrela perde dois nós e ganha um, sendo que o comprimento do ramo que liga este novo nó à estrela deve ser calculado. Estas junções são realizadas em seqüência, até que a árvore-estrela fique com apenas dois nós. Logo, o número de junções vai ser $N-2$.”(REIS; ZUBEN, 2007, p. 1)

Ao final do algoritmo tem-se como saída, uma árvore filogenética, onde se espera que os elementos mais próximos sejam os mais similares entre si, e conseqüentemente, que os mais distantes sejam aqueles que compartilham menos características em comum.

2.3.4 Fast Newman - FN

O algoritmo Fast Newman é um algoritmo de clusterização que recebe uma árvore filogenética e busca identificar sub-árvores com um alto grau de independência, definindo grupos a partir dessas sub-árvores.

2.4 Entropia da Informação

Entropia ou incerteza da informação, conceito definido por SHANNON (1948), é a medida chave na teoria da informação, área que surge a partir do seu trabalho. A entropia de uma informação é medida em quantidade de bits, que por sua vez podem ter apenas 2 valores: 0 ou 1.

A entropia dentro da teoria da informação mede o quão imprevisível são as características de uma informação, implicando diretamente na capacidade de compressão dessa informação, pois a imprevisibilidade ou incerteza das características de uma informação exige uma representação mais bem definida que elimine possíveis ambiguidades na representação de características.

Exemplificando, considere a *string* '00000000' (oito zeros), essa informação é extremamente previsível, podemos claramente afirmar que todos caracteres que a compõem são 0, portanto, pode-se dizer que a entropia dessa informação é zero, pois não há incerteza alguma sobre o seu conteúdo (características), e conseqüentemente é uma informação com grande potencial de compressão.

Ao alterar essa *string*, adicionando mais símbolos, como por exemplo: '01002301', não podemos mais simplesmente dizer que é apenas um caractere se repetindo 8 vezes como anteriormente, pois essa nova informação carrega incerteza devido a existência de diferentes símbolos em diferentes posições, e conseqüentemente possui um potencial de compressão menor em relação a *string* anterior.

Para representar fielmente essa informação é necessário não apenas representar outros símbolos, mas também a sua posição corretamente, limitando a capacidade de compressão dessa informação por causa de sua incerteza (entropia).

Para a medir a entropia de uma informação como a *string* '01002301' mencionada, é fundamental uma observação das probabilidades de cada símbolo (caractere) presente na informação, o que nos dá 50% de probabilidade para o símbolo 0, 25% para o símbolo 1, e 12,5% para os símbolos 2 e 3 (Tabela 2).

Símbolo	0	1	2	3
Probabilidade	0.5	0.25	0.125	0.125

Tabela 2 – Probabilidades dos símbolos (*string*'01002301')

Com as probabilidades calculadas, podemos nos perguntar, dada uma posição aleatória nessa *string*, quantas perguntas em **média** temos de fazer para adivinhar um símbolo qualquer dessa informação?

Uma solução ótima para essa pergunta pode ser representada por uma árvore binária de decisão baseada na probabilidade de cada símbolo, como por exemplo as árvores geradas pelo algoritmo de codificação de Huffman. A figura 4 ilustra uma possível árvore otimizada de acordo com as probabilidades dos símbolos: os símbolos de maior frequência se

encontram mais próximos da raiz, exigindo um número menor de bits (ou perguntas) para a sua representação, enquanto que os símbolos de menor frequência exigem um número maior de bits.

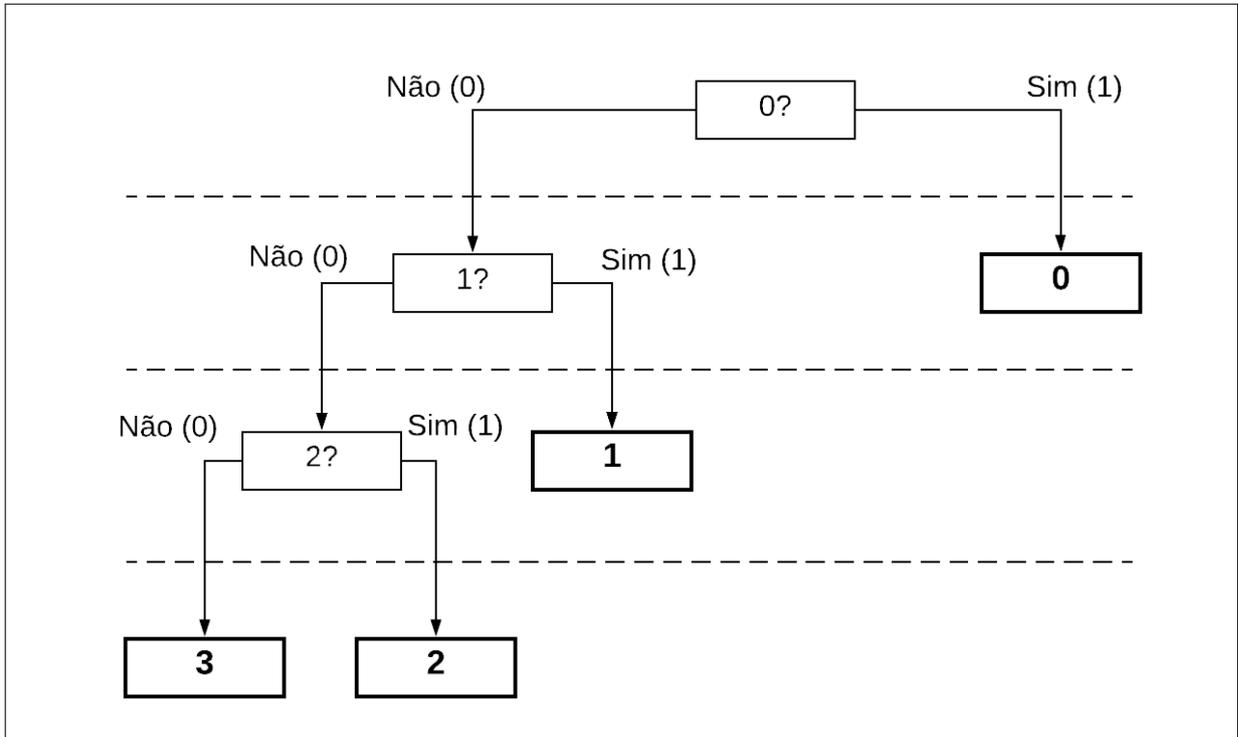


Figura 4 – Exemplo de árvore binária de decisão para previsão de caractere em posição aleatória na *string* '01002301'

Diagrama autoral feito com a ferramenta lucidchart

Com essa árvore temos que a média (M) de perguntas necessárias para adivinhar corretamente um símbolo em alguma posição aleatória da string é uma média ponderada representada pelo somatório da probabilidade de ocorrência (p) multiplicado pelo número de passos necessários para se atingir cada símbolo na árvore a partir da raiz (passos):

$$M('01002301') = \sum_{i=0}^3 p_i * passos_i \tag{2}$$

$$M('01002301') = (0.5 * 1) + (0.25 * 2) + (0.125 * 3) + (0.125 * 3) \tag{3}$$

$$M('01002301') = 1.75bits \tag{4}$$

O resultado dessa média é expresso em bits devido ao número de passos ser na verdade o número de bits(0 ou 1) utilizados para se chegar até tal folha a partir da raiz. A entropia dessa informação é equivalente a essa média de perguntas necessárias para se

adivinhar um símbolo, com a fórmula anterior é perceptível a equivalência com a equação genérica proposta por SHANNON (2010):

$$H(S) = \sum_{i=1}^n p_i * \log_2 \frac{1}{p_i} \quad (5)$$

onde H é o termo utilizado para representar a entropia de uma informação S, e H é igual ao somatório das probabilidades de todas as características multiplicadas pelo logaritmo de 1 dividido pela respectiva probabilidade de característica na base 2, o que é equivalente ao número de passos para encontrar um símbolo em uma árvore binária a partir da raiz no exemplo anterior.

2.5 Trabalhos Relacionados

2.5.1 Estudo e extensão da metodologia DAMICORE para tarefas de classificação

CESAR, (2016) fez sua própria implementação do fluxo de operações DAMICORE em *python*. Nessa sua implementação explorou diferentes formas de se trabalhar com o conjunto de métodos presentes no fluxo de operações DAMICORE, além disso, vale destacar a extensão da metodologia para um número ilimitado de classes e também a extensão da metodologia para tarefas de classificação.

O sistema desenvolvido apresentou ótimos resultados com o método de classificação kNN (k Nearest Neighbours), com precisão acima de 90% em todos testes expostos. Apesar da aplicação ser capaz de funcionar para qualquer tipo de dado, o autor apresenta apenas testes realizados com textos, mais precisamente quanto a capacidade de classificação de spam e identificação de línguas.

Durante a pesquisa aqui desenvolvida, foram realizados experimentos com imagens na aplicação de CESAR, disponível em sua página do GitHub¹. Com imagens o algoritmo não foi capaz de produzir resultados muito bons, uma possível causa é que, como mencionado anteriormente, o modelo de representação genérico dos compressores não produz uma representação adequada para a detecção de padrões em imagens.

Dito isso, o trabalho aqui apresentado propõe o desmembramento da etapa de compactação em duas: uma etapa responsável pela extração de características da imagem, e outra pela a codificação dessas características. Esse desmembramento produziu bons resultados em comparação com a aplicação de CESAR como pode ser visto no capítulo de resultados.

¹Aplicação disponível em: <<https://github.com/brunokim/damicore-python>>

2.5.2 UTILIZANDO ÁRVORES FILOGENÉTICAS PARA A IDENTIFICAÇÃO DE SIMILARIDADES EM PINTURAS DIGITALIZADAS

TORRES faz uso da metodologia DAMICORE para identificação de similaridades em pinturas de diferentes movimentos artísticos. Com esse objetivo, o autor faz testes com **3 descritores de imagens** durante uma etapa de extração de características que precede a etapa de compressão do framework:

- GIST (OLIVA; TORRALBA, 2001)
- Local Binary Pattern - LBP (OJALA; PIETIKAINEN; HARWOOD, 1994)
- COLOR, também conhecido como PLSA-bg (WEIJER; SCHMID; VERBEEK, 2007)

Com as características extraídas utilizando algum dos descritores mencionados, tem-se um vetor de características como entrada para o fluxo DAMICORE tradicional: esse vetor é comprimido com o algoritmo gzip para o cálculo da matriz de distâncias com a NCD, seguido pela geração da árvore filogenética com o NJ, e por fim é realizado uma detecção de comunidades na árvore com o algoritmo FN.

Todos os resultados apresentam elementos de diferentes classes nas comunidades detectadas, em todos os casos com um grande percentual dos elementos pertencentes a diferentes movimentos, como exemplo dessa mistura tem-se uma das comunidades obtidas com 6 pinturas pertencentes ao Cubismo, 5 pertencentes ao Expressionismo, e uma ao Romantismo.

A avaliação dos resultados não apresenta nenhuma métrica para a medição da precisão dos agrupamentos gerados. Porém, pela mistura de classes visível nas comunidades apresentadas como resultados, os resultados obtidos com a metodologia proposta neste trabalho, descrito em detalhes no capítulo 3, aparentemente demonstram resultados com maior precisão.

3 MATERIAIS E MÉTODOS

Este capítulo apresenta detalhes do desenvolvimento da aplicação proposta, dos dados coletados para teste e a metodologia utilizada para avaliação dos resultados.

3.1 Área de Estudo

Emulação da capacidade humana de reconhecimento de padrões através da extensão da metodologia DAMICORE. Com esse objetivo foi desenvolvido uma aplicação em **python** capaz de gerar de maneira não supervisionada uma árvore filogenética que expressa relações de distância entre diferentes instâncias de imagens. A árvore de saída pode servir futuramente como modelo de inteligência para um agente classificador capaz de inferir rótulos para novas instâncias, ideias são discutidas para trabalhos futuros.

Foram coletadas imagens na internet para realização de experimentos com a aplicação desenvolvida, esses dados estão descritos na próxima seção, nomeada de **materiais**.

3.2 Materiais

Nesta seção estão descritos as diferentes categorias de imagens coletadas, as categorias estão organizadas em subseções que possuem informações do conjunto de dados, além de uma tabela com as classes e quantidade de instâncias coletada por classes.

Grande parte das imagens foram coletadas manualmente através de serviços de busca como o *google search*, *pixabay*, *flickr* e *shutterstock*, outras vieram de conjuntos de dados disponibilizados por terceiros, e para uma categoria em especial, a categoria de imagens do teste CAPTCHA, as imagens foram coletadas através da captura de tela em sites que fazem uso do mesmo.

No total foram coletadas 3435 imagens, sendo elas pertencentes as seguintes categorias:

- Frutas (102)
- Animais (1232)
- Plantações (1322)
- CAPTCHA (124)
- Faces Humanas (400)
- Pinturas (255)

Nem todas as imagens coletadas foram utilizadas nos experimentos realizados, mas todas elas estão disponíveis em um repositório no github¹, com os devidos créditos quando se tem uma única origem de maior relevância. Além disso, as fontes utilizadas nos experimentos descritos na seção de resultados também estão disponíveis no mesmo

¹Repositório no github: <https://github.com/higorhlg/IMAGEDATASET_ML>

repositório para possibilitar a replicação dos mesmos experimentos aqui realizados. A seguir tem-se uma descrição de todos esses conjuntos de imagens.

3.2.1 Conjunto de imagens de frutas

Foram coletadas um total de 102 imagens de frutas através de buscas na internet em provedores como *google search*, *pixabay*, *flickr* e *shutterstock*. As imagens estão divididas em 5 classes.

Tabela 3 – Conjunto de dados 1 - Frutas

CLASSE	OBJETOS
BANANA	20
LARANJA	20
LIMÃO	20
MAÇA	20
MORANGO	22
TOTAL	102

As imagens estão em diversas resoluções, com diferentes perspectivas de capturas e **podem conter uma ou várias unidades de alguma fruta**. A figura 5 mostra exemplos de imagens coletadas nessa categoria.

3.2.2 Conjunto de Imagens de Plantações

Foram coletadas um total de 1322 imagens de plantações através de provedores de busca na internet como *google search*, *pixabay*, *flickr* e *shutterstock*. As imagens estão divididas em 12 classes.

Tabela 4 – Conjunto de dados 2 - Plantações

CLASSE	OBJETOS
ALGODÃO	109
ARROZ	116
MILHO	110
GIRASSOL	101
MANDIOCA	100
CANAVIAL	106
MAMAO	110
BANANA	113
JABUTICABA	111
ABACAXI	117
TOMATE	130
ALFACE	110
TOTAL	1322



Figura 5 – Exemplos de imagens coletadas na categoria de frutas

As imagens estão em diversas resoluções e com diferentes perspectivas de captura. As imagens possuem grandes ou pequenas plantações (que podem ter frutos visíveis ou não), e algumas (uma pequena parcela) apenas frutos provenientes da plantação. A figura 6 mostra alguns exemplos de imagens que foram coletadas nessa categoria.

3.2.3 Conjunto de Imagens de Animais

Foram coletadas um total de 1232 imagens de animais através de provedores de busca na internet como *google search*, *pixabay*, *flickr* e *shutterstock*. As imagens estão divididas em 11 classes.

As imagens estão em diversas resoluções e com diferentes perspectivas de captura, algumas possuem partes do animal, outras o animal inteiro. A figura 7 mostra exemplos de imagens coletadas nessa categoria.

3.2.4 Conjunto de Imagens de CAPTCHA

Foram coletadas imagens através de sites que utilizam o **CAPTCHA** (*Completely Automated Public Turing test to tell Computers and Humans Apart*) como medida de segurança contra ataques automatizados por máquinas.



Figura 6 – Exemplos de imagens coletadas na categoria de plantações



Figura 7 – Exemplos de imagens coletadas na categoria de animais

Tabela 5 – Conjunto de dados 3 - Animais

CLASSE	OBJETOS
HOMEM	110
MACACO	116
SERPENTE	110
TUBARÃO	111
CROCODILO	110
CAVALO	115
ELEFANTE	112
BEIJA-FLOR	115
GATO	113
GIRAFÁ	110
AVESTRUZ	111
TOTAL	1232

Para essa categoria foram capturadas imagens de testes CAPTCHA através da captura de tela do computador, e então as imagens envolvidas em cada um dos testes capturados foram separadas manualmente através de um recorte em novos arquivos, totalizando 124 imagens diferentes.

As imagens foram separadas em apenas 2 classes para simulação do teste "Selecione Um Veículo", onde o objetivo é apenas classificar as imagens como contendo ou não veículos, as mesmas imagens poderiam ser organizadas em outros conjuntos binários (2 classes) para simular outros tipos de situações que ocorrem nesse teste.

Tabela 6 – Conjunto de dados 4 - CAPTCHA

CLASSE	OBJETOS
VEÍCULOS	54
NÃO_VEÍCULOS	70
TOTAL	124

As imagens possuem resoluções similares com várias perspectivas de capturas e podem possuir apenas uma cor sólida, apenas um objeto ou uma grande variedade de objetos misturados na mesma imagem. A figura 8 mostra exemplos de imagens coletadas nessa categoria.

3.2.5 Conjunto de Imagens de Faces Humanas - 'ORL Database Of Faces'

Este conjunto de imagens, disponibilizado pela **Universidade de Cambridge**, chamado de '*ORL Database of Faces*' (CAMBRIDGE, 1994), contém imagens da face de 40 indivíduos diferentes, 10 imagens por indivíduo, totalizando 400 imagens.

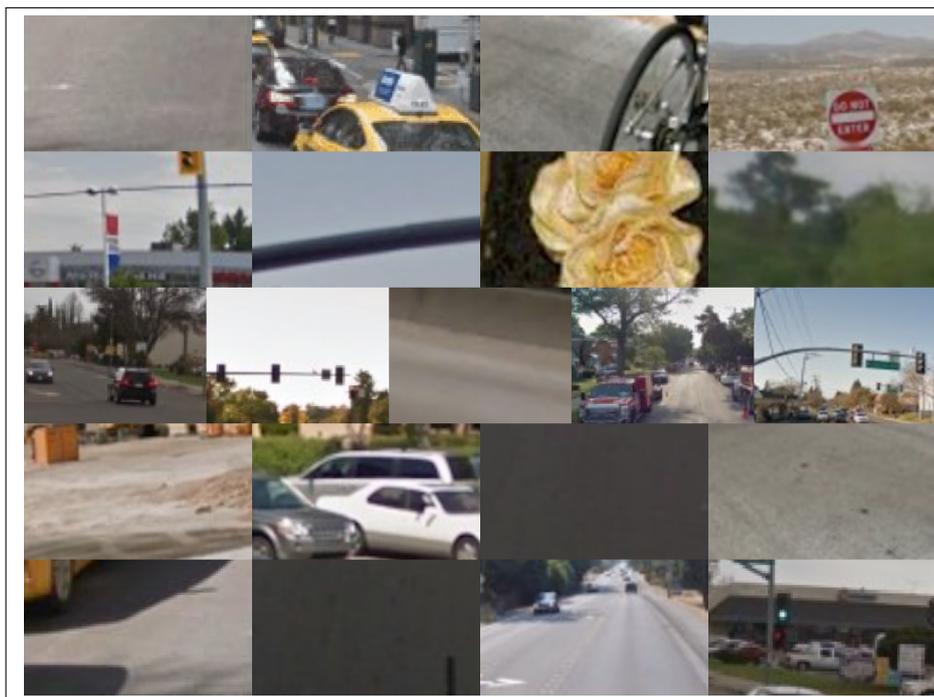


Figura 8 – Exemplos de imagens coletadas na categoria de testes CAPTCHA

Tabela 7 – Conjunto de dados 5 - 'ORL Database Of Faces'

CLASSE	OBJETOS
INDIVÍDUO0	10
INDIVÍDUO1	10
...	...
INDIVÍDUO39	10
TOTAL	400



Figura 9 – Exemplos de imagens coletadas na categoria de faces humanas

(CAMBRIDGE, 1994) Essas imagens foram capturadas entre 1992 e 1994 no Laboratório de Pesquisas Olivetti (ORL - Olivetti Research Laboratory) em Cambridge para testes de reconhecimento facial, estão todas dispostas na mesma resolução com a mesma perspectiva de captura.

Tradução e adaptação de um trecho informativo disponível junto ao conjunto de dados:

”Para alguns dos indivíduos, as suas 10 imagens foram capturadas em momentos diferentes, variando levemente a luz, expressões faciais e detalhes faciais. Todas as imagens foram tiradas em um fundo escuro homogêneo e todos os indivíduos estão encarando a câmera com possivelmente uma pequena variação de movimentos e inclinações laterais da cabeça.”

Dados e informações disponíveis em:

<<https://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html>>

3.2.6 Conjunto de Pinturas (Diferentes Movimentos Artísticos)

Foram coletadas da enciclopédia de artes visuais, **WikiArt**² imagens de pinturas de 6 movimentos artísticos diferentes, cada classe (movimento artístico) possui no mínimo 31 amostras e totalizam 255 imagens.

Foram utilizadas pinturas de 6 movimentos de 3 eras diferentes:

- Medieval (Pinturas dos movimentos Bizantino e Gótico);
- Pós Renascença (Pinturas dos movimentos Barroco e Romantismo);
- Moderna (Pinturas dos movimentos Cubismo e Expressionismo).

Tabela 8 – Conjunto de dados 7 - Movimentos Artísticos

CLASSE	OBJETOS
BIZANTINO	37
GOTICO	31
BARROCO	49
ROMANTISMO	52
CUBISMO	43
EXPRESSIONISMO	43
TOTAL	255

As imagens variam bastante em resolução, e cada uma contém apenas uma pintura pertencente predominantemente a algum movimento artístico específico, podendo também exibir alguma características de outros movimento na mesma.

²Endereço para acesso: <<https://www.wikiart.org/>>



Figura 10 – Exemplos de imagens coletadas na categoria de pinturas

3.3 Construção do Modelo de Inteligência

Propõe-se o uso da árvore filogenética gerada pelo NJ como modelo de conhecimento a ser utilizado para predições de rótulos futuramente, não se fazendo necessário a detecção de comunidades na árvore com um algoritmo como o Fast Newman.

O modelo de inteligência é construído a partir de uma fonte de dado que seria um diretório contendo imagens, a sua construção será baseada no framework DAMICORE com algumas alterações no fluxo de operação do mesmo. A figura 11 ilustra as diferenças entre o fluxo de operação DAMICORE e o proposto.

O compactador utilizado no método DAMICORE determina a representação das informações de objetos disponíveis na fonte de dados utilizada, essa representação pode ser crucial para o resultado do algoritmo, pois é a partir dessa representação que será calculada a distância entre cada objeto da fonte de dados em questão. Pretende-se desmembrar esse processo de compactação em 2 etapas para um maior controle da representação gerada:

- **Extração de Características**
- **Codificação das características**

3.3.1 Extração de Características

Munido de uma fonte de imagens, a primeira tarefa do algoritmo para construir o modelo é aplicar um pré-processamento que consiste em uma técnica de **compressão** com perda para **redução** de informação e conseqüentemente maior **generalização** de valores de pixels. Isso é feito com base na visão humana que faz algo semelhante ao descartar

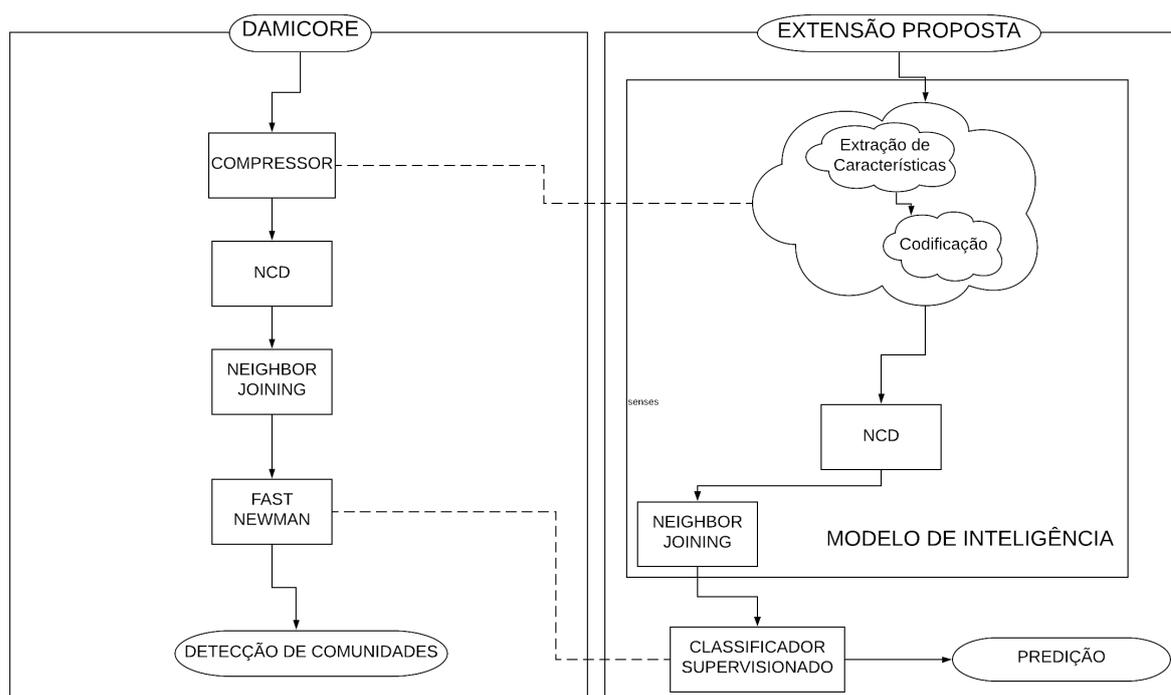


Figura 11 – Comparação do fluxo operacional
Diagrama autoral feito com a ferramenta lucid chart

o excesso de informação (em especial as cores) visual das imagens capturadas pelo olho humano(WILDING, 2017; CANARIA, 2014).

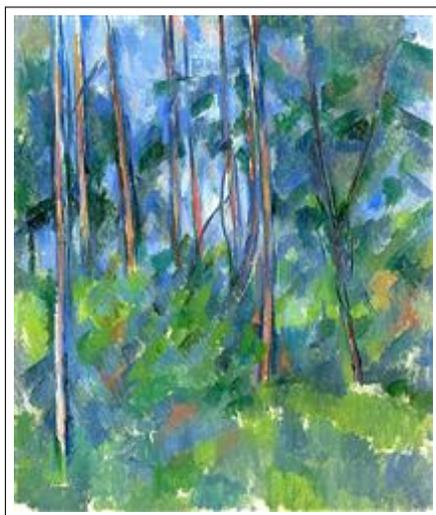
O pré-processamento definido começa com um simples redimensionamento das imagens para o tamanho 256x256, garantindo que todas imagens possuem a mesma quantidade de pixels, e posteriormente uma quantização para apenas um único canal de cor, diminuindo consideravelmente a quantidade excessiva de cores presente nas imagens, e conseqüentemente a sua dimensionalidade.

As figuras 12 e 13 mostram comparações das imagens que foram utilizadas nos experimentos em seu estado original e no estado após o pré-processamento (redimensionamento e quantização para 32 cores), percebe-se que o número de cores (32) é suficiente para demonstrar as diferenças entre cores mais relevantes para a percepção da cena, porém em alguns casos (exemplo na parte superior da figura 13) pode-se perder alguns detalhes com esse valor.

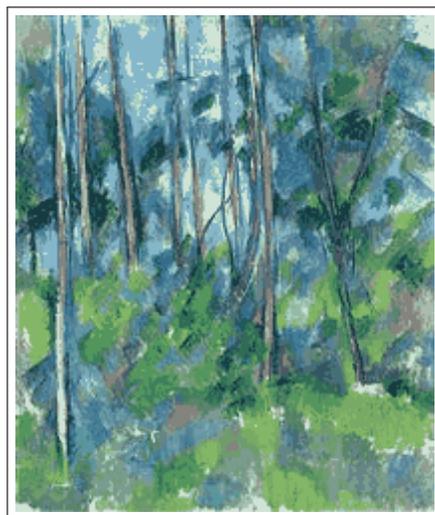
Para a medição da perda de informação ϵ em uma imagem (X), após o processamento, foi determinada uma expressão baseada na variação da informação original comparada com a informação **reconstruída** após o processamento realizado (redimensionamento e quantização) esse cálculo é feito através da seguinte equação:

$$\epsilon(X) = \frac{1}{n \cdot 3} \cdot \sum_{j=1}^n \left(\sum_{c=1}^3 \left(\frac{\max(f_{jc}, g_{jc}) - \min(f_{jc}, g_{jc})}{\max(f_{jc}, g_{jc})} \right) \right) \quad (6)$$

onde:



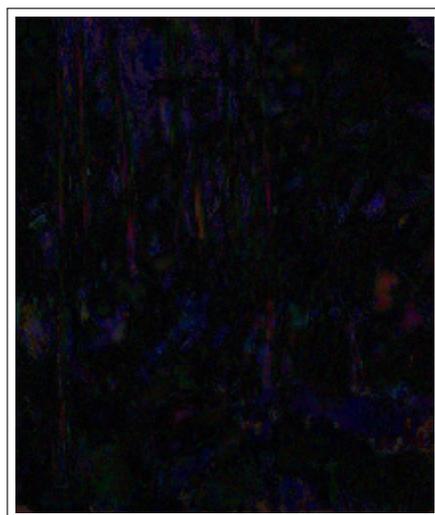
(a) Imagem original



(b) Imagem redimensionada e quantizada



(c) Imagem reconstruída



(d) Diferença entre imagem real e reconstruída



(e) Diferença entre imagem real e reconstruída (cores invertidas)

Figura 12 – Pintura do momento cubismo - Comparações

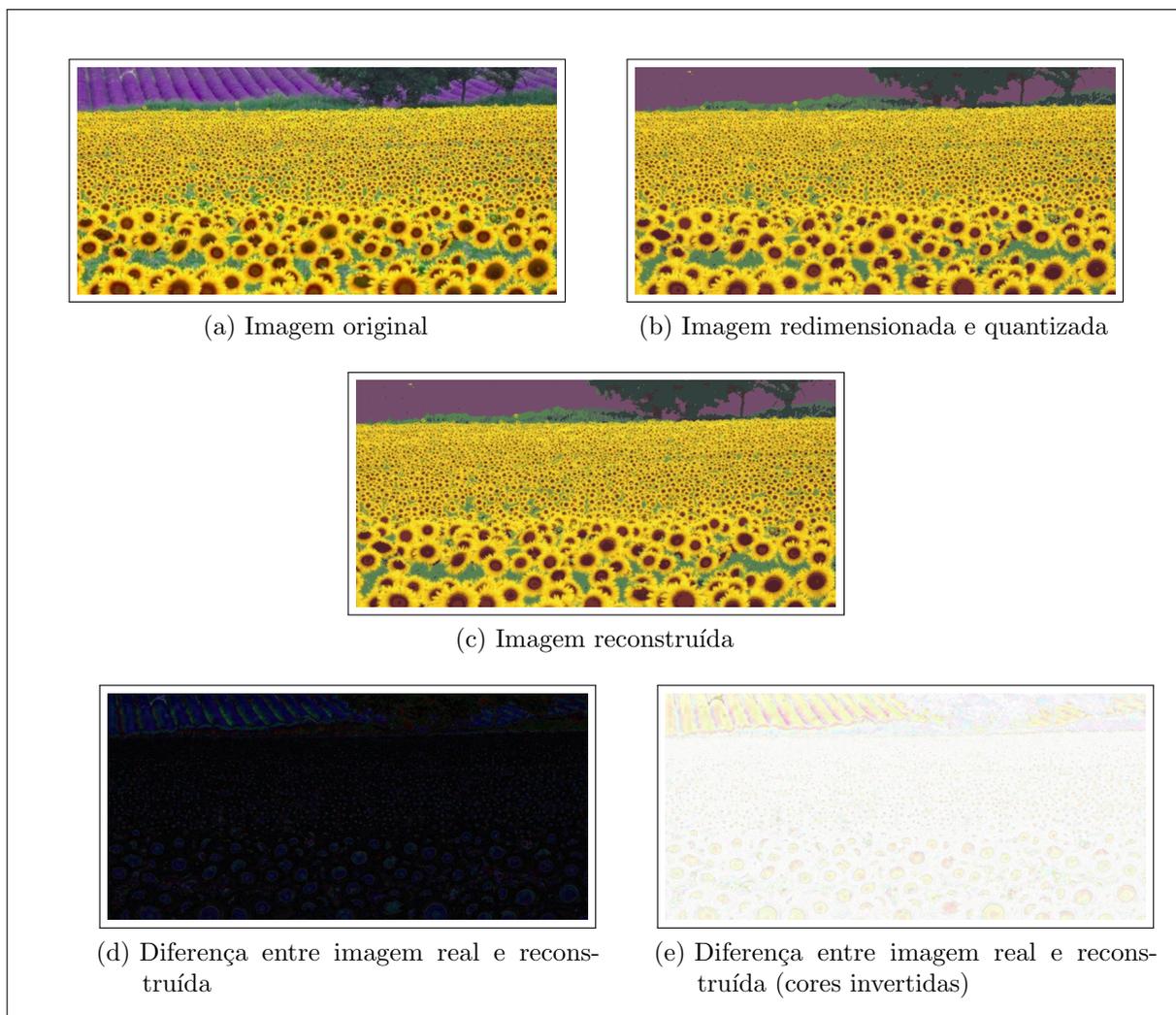


Figura 13 – Plantação de girassóis - Comparações

- n é o total de número de pixels na imagem original X ;
- f é uma matriz de $n * 3$ posições e que contém todos os valores de pixels da **imagem original** X em 3 canais de cores diferentes (RGB);
- g é uma matriz de $n * 3$ posições que contém todos os valores de pixels da imagem X em canais de cores diferentes (RGB) **após o processamento e reconstrução** para a quantidade de cores e resolução original.

O valores de $\epsilon(X)$ caracterizam a abordagem de compressão da seguinte forma:

- sem perdas ($\epsilon(X) = 0$): a amostra original e a reconstruída são idênticas;
- com perdas ($\epsilon(X) > 0$): valores baixos para $\epsilon(X)$ resultam em amostras reconstruídas muito próximas das reais.

Médias dos valores de ϵ obtidos para cada amostra em cada categoria estão expostas na seção de resultados obtidos. Para as imagens das figuras 12 e 13, os valores de variação (ϵ) entre a informação original e a processada são respectivamente 0.084 (8.4%) e 0.136 (13.6%), o que faz sentido de forma empírica ao observar as duas imagens após o

pré-processamento:

Na figura 12, percebe-se o efeito gradiente de cores sendo eliminado, porém, em nenhuma porção da figura se percebe uma perda muito grande de informação relevante para o entendimento da cena, diferentemente da figura 13, que apresenta uma grande perda de detalhes na porção superior, onde a textura de fundo é definida por uma coloração gradiente e completamente substituída por uma única cor sólida após a redução de cores, causando uma perda mais influente na percepção da cena quando comparada com a figura 12.

Na etapa de quantização foram realizados testes com diferentes valores para o número de cores, e na seção de resultados estão expostos as médias dos resultados obtidos com cada valor testado. A figura 14 ilustra a metodologia executada pelo pré-processamento realizado nas imagens, especificamente em uma situação em que o número de cores na etapa de quantização é 32.

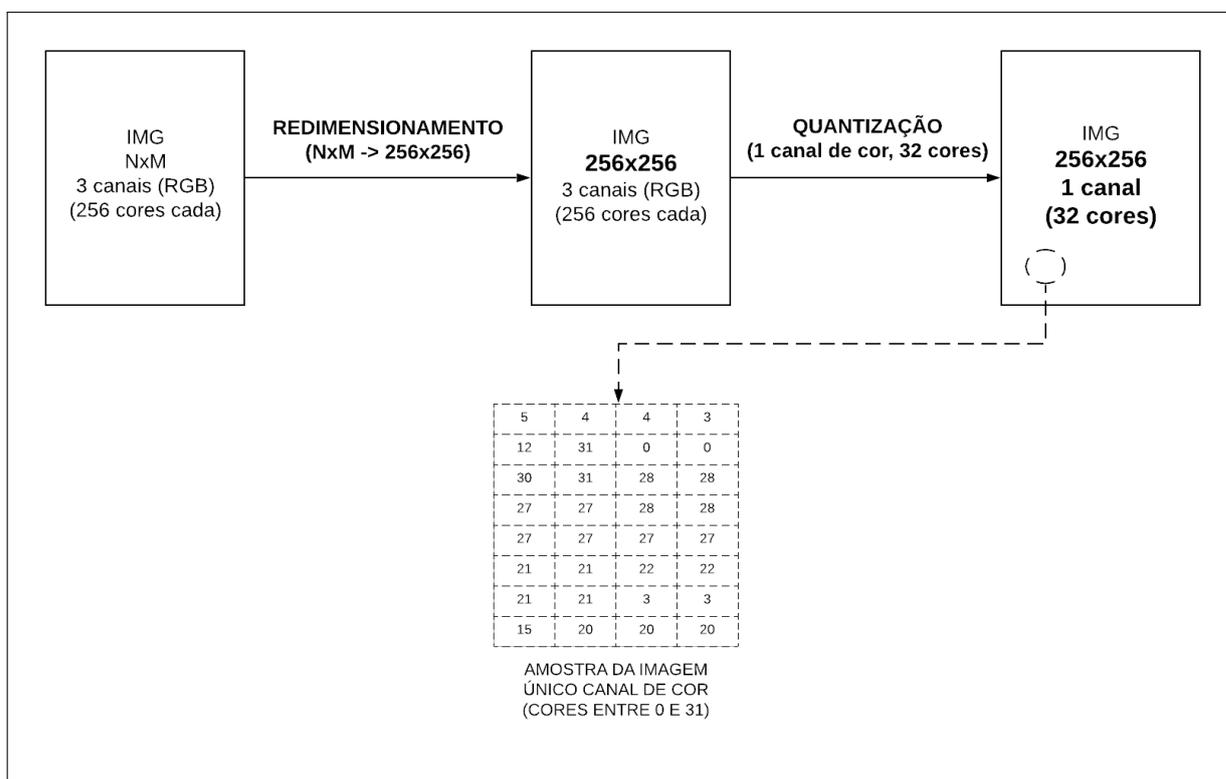


Figura 14 – Pré-processamento da imagem
Diagrama autoral feito com a ferramenta lucid chart

Como pode ser visto na figura 14, após esse pré-processamento, realizado com a biblioteca *pillow* da linguagem python, tem-se como resultado, um único canal de cores, que pode ser representado por uma matriz de tamanho 256x256 que possui valores de pixels que variam de 0 até 31. Com essa matriz resultante é feito uma contagem da ocorrência de cada valor de pixel.

Para a contagem de cada cor x pertencente ao conjunto de cores $\{0...31\}$, temos

que:

$$Contagem(x) = N^{\circ}Ocorrencias(x) \quad (7)$$

Sendo o número de ocorrências o número de vezes que a cor x aparece na matriz que possui os valores de pixels da imagem.

Forma-se então um dicionário que contém a contagem de quantas vezes cada valor de pixel ocorreu, a partir daí é realizado um simples cálculo probabilístico para cada um desses valores de contagem com a divisão desse valor pela quantidade total de pixels presentes na imagem (256x256).

Dessa forma, para cada elemento x pertencente ao conjunto de cores $\{0...31\}$ temos que:

$$Probabilidade(x) = Contagem(x)/ContagemTotal \quad (8)$$

Sendo **ContagemTotal** a quantidade total de pixels na imagem, nesse caso 256x256 (65 536).

3.3.1.1 Combinação de Objetos

Após os cálculos de probabilidades para cada imagem serem realizados, as imagens são combinadas entre si de todas as formas possíveis (imagem 1 com imagem 2, imagem 1 com imagem 3...). O resultado de uma combinação é um novo objeto que possui a soma da contagem das cores de cada um dos objetos combinados. Por exemplo o objeto (1,2) formado pela combinação do objeto 1 com o objeto 2, tem seus valores de contagem de cores definidos da seguinte forma:

$$ContagemObjeto[1,2](x) = ContagemObjeto[1](x) + ContagemObjeto[2](x) \quad (9)$$

Enfim, com esses objetos combinados é feito o mesmo cálculo de probabilidade feito individualmente para cada imagem, porém agora com as imagens combinadas, utilizando os valores de contagem combinados.

$$ProbabilidadeCombinada[1,2](x) = ContagemObjeto[1,2](x)/ContagemTotal \quad (10)$$

onde a contagem total nesse caso se refere a soma da contagem total de pixels de cada um dos objetos envolvidos, ou seja, das duas imagens que na metodologia proposta possuem 65 536 pixels cada, totalizando 131 072 pixels.

3.3.2 Codificação das características

Para cada imagem é calculada uma aproximação (C) da incomputável complexidade de Kolmogorov (K) utilizando a codificação de Huffman. Essa aproximação calculada é a incerteza ou entropia de uma representação comprimida dos valores de **probabilidade**,

calculados anteriormente em cima dos valores de pixels da imagem e as suas respectivas frequências.

Para esse cálculo de C , é utilizada uma máquina de Turing que possui como entradas:

- Alfabeto = Rótulo das cores (0 - N-1)
- Estado Inicial = Probabilidade da primeira cor (0)
- Conjunto de Estados Finais = Última Cor(Alfabeto[N-1])
- Alfabeto de Saída = [0,1]
- Função de Transição (F) = Cálculo da inversa da função Probabilidade(x), para cada x, através da codificação com o algoritmo de Huffman.

Esta etapa se inicia com a criação da máquina com esses parâmetros, e com um somatório dos valores da função de transição para cada cor, tem-se uma aproximação aplicável da complexidade de Kolmogorov para cada imagem.

Essa aproximação resultante baseia-se no conceito de **entropia da informação**(SHANNON, 1948) e pode ser entendida como uma **média do número de bits utilizado para o armazenamento de cada símbolo (cor)**, quanto maior essa média, maior a entropia (incerteza inversamente proporcional a capacidade de compressão da informação), quanto menor essa média, menor é a entropia.

A incerteza gerada como aproximação, por mais que diferente da complexidade de Kolmogorov, é capaz de representar propriedades semelhantes ao representar uma média do espaço de armazenamento utilizado por cada característica comprimida pelo algoritmo de Huffman, enquanto que por outro lado, a complexidade de Kolmogorov é o tamanho da **menor representação possível** de uma certa informação.

Dessa forma, a representação final das imagens consiste no somatório de uma função F com valores para cada x pertencente ao conjunto das cores $\{0..31\}$. O valor de F para determinada cor x , $F(x)$, é o tamanho do resultado da multiplicação entre a probabilidade dessa cor, $P(x)$, e a codificação binária dessa probabilidade com o algoritmo de Huffman, denotado na equação a seguir por $Huffman(P(x))$:

$$F(x) = TAM(Huffman(P(x))) * P(x) \quad (11)$$

Após ser calculado os valores da função de transição de saída da máquina de Turing, é finalmente calculada a aproximação da complexidade de Kolmogorov de cada imagem. A aproximação C , da complexidade de Kolmogorov de uma certa imagem pode ser expressa da seguinte forma:

$$C(imagem) = \sum_{x=0}^N F(x) \quad (12)$$

Sendo \mathbf{F} a função de transição com valores para cada cor \mathbf{x} calculados na etapa descrita anteriormente, assim temos que a representação comprimida \mathbf{C} de uma imagem é

o resultado do somatório de todos os valores de F e representa a **entropia** de uma imagem de acordo com o compactador utilizado na etapa de codificação.

Além disso, para a aplicação da NCD, também é construída a representação comprimida das combinações de imagens para cada combinação possível. Como os objetos combinados já possuem seus valores de probabilidades de cada cor, a função de transição F e até mesmo o cálculo da entropia (aproximação C) se torna idêntico, porém com o uso dos valores de probabilidade combinados ao invés dos valores individuais.

3.3.3 Cálculo de distâncias com a NCD

Com a representação comprimida C para cada imagem \mathbf{x} e para cada combinação possível de imagens \mathbf{xy} , em seguida é construída uma matriz de distâncias com o cálculo da distância normalizada de compressão para cada combinação de imagens possível, onde para cada entrada da matriz tem-se que:

$$NCD(x,y) = \frac{C(xy) - \min(C(x), C(y))}{\max(C(x), C(y))} \quad (13)$$

A distância normalizada de compressão entre duas instâncias x e y , é um valor entre 0 e 1 gerado como resultado da **subtração** da concatenação (soma) das representações comprimidas de x e y , $C(xy)$ **pelo menor valor entre $C(x)$ e $C(y)$, tudo isso dividido pelo maior valor entre $C(x)$ e $C(y)$.**

Cada distância calculada é parte da matriz de distâncias gerada ao final do cálculo de toda possível combinação, ao final dessa etapa tem-se uma matriz de distâncias que expressa quão distante estão as imagens envolvidas de acordo com a distância de compressão, em uma escala normalizada de 0 até 1. Onde uma distância 0 significa que não há distância entre as imagens, e provavelmente as imagens são iguais.

3.3.4 Formação da árvore filogenética com o NJ

Fazendo uso dessa matriz de distâncias calculada com a NCD, temos a entrada para o algoritmo Neighbor Joining. O algoritmo através de uma matriz de distância gera uma árvore filogenética que expressa relações entre as instâncias envolvidas, o procedimento do NJ nem sempre gera uma árvore perfeita que expressa essas relações da melhor forma possível, mas busca sempre manter próximas aquelas instâncias mais similares entre si. Por mais que o algoritmo seja incapaz de produzir **sempre** a melhor árvore, ainda é muito utilizado por ser um dos melhores métodos para essa tarefa.

Com essa árvore gerada, temos então o nosso modelo de conhecimento pronto, onde espera-se ter regiões da árvore com aglomerados de instâncias pertencentes a mesma classe, esse modelo pode ser utilizado para uma detecção de clusters, ou como é proposto como trabalho futuro, o modelo pode ser utilizado como conhecimento para um classificador supervisionado capaz de inferir rótulos a novas instâncias.

3.3.5 Avaliação dos resultados

Foi definido um coeficiente chamado de G para a avaliação dos modelos de conhecimento (árvores filogenéticas) gerados. Para o cálculo desse coeficiente, considera-se uma derivação da árvore original que pode ser vista na figura 15.

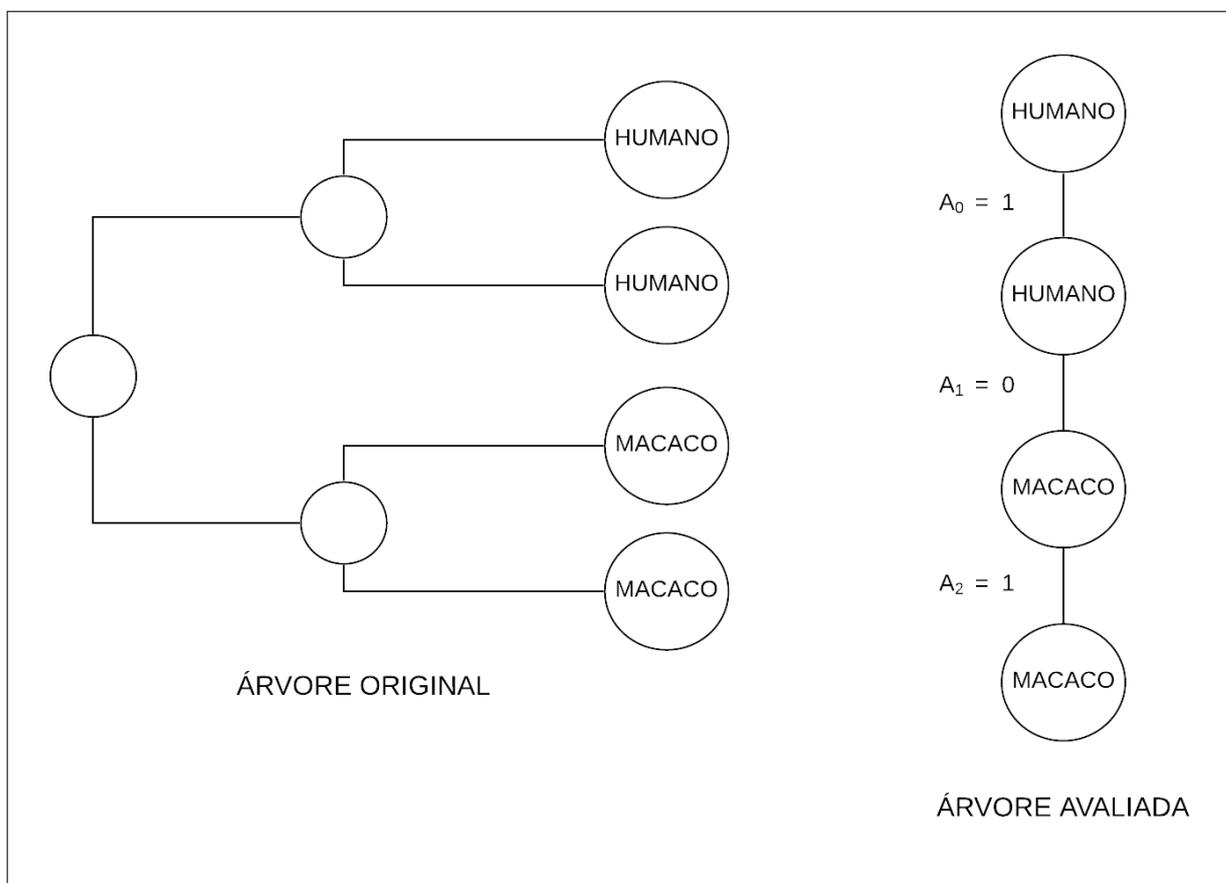


Figura 15 – Exemplo: árvore original e árvore a ser avaliada
Diagrama autoral feito com a ferramenta lucid chart

Essa derivação da árvore filogenética original é um grafo não direcionado composto pelos nós que representam as instâncias (imagens) e arestas que fazem ligações entre os vizinhos de acordo com a disposição na árvore original, as arestas são valoradas em 0 caso liguem vizinhos pertencentes a classes diferentes e 1 caso os vizinhos sejam da mesma classe.

O coeficiente normalizado é calculado com a seguinte equação:

$$G = \frac{\sum_{i=1}^N A_i}{Nv - Nc} \quad (14)$$

onde:

- A_i é o valor da i -ésima aresta;
- N é o número de arestas;
- Nv é o número de vértices (elementos na árvore);
- Nc é o número de classes.

O valor desse coeficiente é normalizado, ou seja, varia de 0 a 1, sendo 1 o melhor resultado possível (100% de acerto), representando uma árvore perfeita, com todos elementos de diferentes classes separados em diferentes sub-árvores.

4 RESULTADOS OBTIDOS

Com os materiais coletados, foram realizados os experimentos com imagens de frutas, plantações, animais, captcha, faces humanas e pinturas. Os experimentos foram realizados na aplicação desenvolvida durante a pesquisa e também na aplicação *damicorepy* (CESAR, 2016) para fins comparativos.

Resultados bons e até mesmo ruins foram alcançados com estes experimentos, este capítulo apresenta os resultados obtidos em diversas situações e experimentos, além de uma comparação no final do capítulo. Os resultados estão divididos em seções na seguinte ordem:

- Resultados dos experimentos (G) com as imagens quantizadas para 32 cores;
- Resultados da variação da informação (ϵ) nas imagens quantizadas para 32 cores;
- Resultados empíricos na etapa de quantização com o número de cores variando de 4 até 256;
- Comparações dos resultados (árvores filogenéticas) obtidos com a aplicação desenvolvida durante a pesquisa (32 cores na etapa de quantização) com os resultados dos mesmos experimentos realizados na ferramenta *damicorepy* (CESAR, 2016), que faz uso do fluxo tradicional DAMICORE.

O número de cores escolhido para a apresentação da maior parte dos resultados foi 32, a mediana que se encontra no meio da série ordenada dos valores utilizados para testes (4, 8, 16, **32**, 64, 128, 256), e representa aproximadamente uma média dos resultados obtidos nos diferentes experimentos realizados na etapa de quantização.

Dito isso, os melhores resultados obtidos com a aplicação foram com apenas 4 cores na etapa de quantização, todos os resultados com diferentes números de cores podem ser vistos na seção 4.3 deste capítulo, onde estão expostos todos os números de cores testados nessa etapa.

4.1 Resultado dos experimentos (Imagens quantizadas para 32 cores)

Nesta seção são apresentados os resultados obtidos com a aplicação proposta nas diferentes categorias de imagens coletadas, os resultados estão descritos de acordo com a forma em que foram organizados os experimentos, em diferentes fontes de dados (diretórios contendo imagens). Para cada categoria tem-se uma tabela para visualização dos resultados em cada uma das fontes de dados utilizadas.

4.1.1 Frutas

Foram utilizadas 102 imagens de frutas cujos experimentos foram divididos em 6 fontes de dados:

Fonte	Classes	G
FONTE1(40)	2: BANANA(20); LARANJA(20)	0.7631
FONTE2(60)	3: BANANA(20); LARANJA(20); MAÇA(20)	0.6842
FONTE3(80)	4: BANANA(20); LARANJA(20); MAÇA(20) E LIMÃO(20)	0.6184
FONTE4(100)	5: BANANA(20); LARANJA(20); MAÇA(20); LI- MÃO(20) E MANGA(20)	0.5474
FONTE5(122)	6: BANANA(20); LARANJA(20); MAÇA(20); LIMÃO(20); MANGA(20) E MORANGO(22)	0.5862

Tabela 9 – Resultados - Frutas

Esse experimento visa testar a capacidade do algoritmo de identificar diferentes frutas, como pode ser visto na tabela 9, as árvores resultantes demonstram uma precisão acima de 58% para todos os experimentos realizados, chegando até 76% no caso mais trivial com apenas 2 classes.

4.1.2 Animais

Foram utilizadas 360 imagens de animais cujos experimentos foram divididos em 9 fontes de dados:

Esse experimento visa testar a capacidade do algoritmo de identificar diferentes animais. Como pode ser visto na tabela 11, os resultados nessa categoria a partir de 8 classes foram ruins, as árvores resultantes demonstram uma precisão de apenas 38% no pior caso (11 classes), chegando até 89% no caso mais trivial com apenas 2 classes.

4.1.3 Plantações

Foram utilizadas 360 imagens de plantações cujos experimentos foram divididos em 9 fontes de dados:

Esse experimento visa testar a capacidade do algoritmo de identificar diferentes plantações. Como pode ser visto na tabela 11, as árvores resultantes demonstram resultados medianos, uma precisão acima de 54% para todos os experimentos realizados, chegando até 75% no experimento mais trivial, com 4 classes.

Fonte	Classes	G
FONTE11(30)	2: HOMEM(15); MACACO(15)	0.8928
FONTE12(60)	2: HOMEM(30); MACACO(30)	0.8793
FONTE13(120)	4: HOMEM(30); MACACO(30); SERPENTE(30); TUBARÃO(30)	0.6293
FONTE14(240)	8: HOMEM(30); MACACO(30); SERPENTE(30); TUBARÃO(30); CROCODILO(30); CAVALO(30); ELEFANTE(30); BEIJA-FLOR(30)	0.4827
FONTE15(270)	9: HOMEM(30); MACACO(30); SERPENTE(30); TUBARÃO(30); CROCODILO(30); CAVALO(30); ELEFANTE(30); BEIJA-FLOR(30); GATO(30)	0.4636
FONTE16(270)	9: HOMEM(30); MACACO(30); SERPENTE(30); TUBARÃO(30); CROCODILO(30); CAVALO(30); ELEFANTE(30); BEIJA-FLOR(30); GIRAFAS(30)	0.4367
FONTE17(270)	9: HOMEM(30); MACACO(30); SERPENTE(30); TUBARÃO(30); CROCODILO(30); CAVALO(30); ELEFANTE(30); BEIJA-FLOR(30); AVESTRUZ(30)	0.4444
FONTE18(300)	10: HOMEM(30); MACACO(30); SERPENTE(30); TUBARÃO(30); CROCODILO(30); CAVALO(30); ELEFANTE(30); BEIJA-FLOR(30); GATO(30); GIRAFAS(30)	0.4275
FONTE19(330)	11: HOMEM(30); MACACO(30); SERPENTE(30); TUBARÃO(30); CROCODILO(30); CAVALO(30); ELEFANTE(30); BEIJA-FLOR(30); GATO(30); GIRAFAS(30); AVESTRUZ(30)	0.3824

Tabela 10 – Resultados - Animais

Fonte	Classes	G
FONTE21(60)	4: ALGODAO(15); ARROZ(15); GIRASSOL(15); MILHO(15)	0.75
FONTE22(120)	4: ALGODAO(30); ARROZ(30); GIRASSOL(30); MILHO(30)	0.6896
FONTE23(150)	5: ALGODAO(30); ARROZ(30); GIRASSOL(30); MILHO(30); MANDIOCA(30)	0.6758
FONTE24(180)	6: ALGODAO(30); ARROZ(30); CANAVIAL(30); GIRASSOL(30); MANDIOCA(30); MILHO(30)	0.6494
FONTE25(210)	7: ALGODAO(30); ARROZ(30); CANAVIAL(30); GIRASSOL(30); MANDIOCA(30); MILHO(30); MAMÃO(30)	0.6256
FONTE26(240)	8: ALGODAO(30); ARROZ(30); CANAVIAL(30); GIRASSOL(30); MANDIOCA(30); MILHO(30); MAMÃO(30); BANANA(30)	0.612
FONTE27(270)	9: ALGODAO(30); ARROZ(30); CANAVIAL(30); GIRASSOL(30); MANDIOCA(30); MILHO(30); MAMÃO(30); BANANA(30); JABUTICABA(30)	0.5977
FONTE28(300)	10: ALGODAO(30); ARROZ(30); CANAVIAL(30); GIRASSOL(30); MANDIOCA(30); MILHO(30); MAMÃO(30); BANANA(30); JABUTICABA(30); ABACAXI(30)	0.5655
FONTE29(360)	12: ALGODAO(30); ARROZ(30); CANAVIAL(30); GIRASSOL(4); MANDIOCA(30); MILHO(30); MAMÃO(30); BANANA(30); JABUTICABA(30); ABACAXI(30); TOMATE(30); ALFACE(30)	0.5402

Tabela 11 – Resultados - Plantações

4.1.4 CAPTCHA

Foram utilizadas 124 imagens de testes CAPTCHA cujos experimentos foram divididos em 4 fontes de dados:

Fonte	Classes	G
FONTE31(16)	2: VEÍCULO(8); NÃO_VEÍCULO(8)	1
FONTE32(32)	2: VEÍCULO(16); NÃO_VEÍCULO(16)	0.9642
FONTE33(64)	2: VEÍCULO(32); NÃO_VEÍCULO(32)	0.85
FONTE34(124)	2: VEÍCULO(54); NÃO_VEÍCULO(70)	0.8278

Tabela 12 – Resultados - CAPTCHA

Esse experimento visa testar a capacidade do algoritmo de identificar imagens que possuam ou não veículos, assim como é feito nos testes CAPTCHA (Complete Automated Test to tell Computers and Humans Apart) uma tarefa supostamente exclusiva para humanos. Como pode ser visto na tabela 12, os resultados dessa categoria foram ótimos, provavelmente pela baixa quantidade de classes a serem identificadas(2), as árvores resultantes demonstram uma precisão acima de 85% para todos os experimentos realizados, chegando até 100% no experimento mais trivial, com apenas 16 imagens, 8 contendo veículos e 8 sem veículos, vale destacar também o experimento com maior quantidade de imagens (fonte34 com 124 imagens) que teve um resultado de aproximadamente 82%.

4.1.5 Faces Humanas

Foram utilizadas 400 imagens de faces humanas cujos experimentos foram divididos em 8 fontes de dados:

Esse experimento visa testar a capacidade do algoritmo de identificar diferentes indivíduos, com possível aplicação para o reconhecimento facial por exemplo. Como pode ser visto na tabela 13, os resultados desse experimento foram ótimos, as árvores resultantes demonstram uma precisão acima de 83% para todos os experimentos realizados, chegando até 98% em um dos experimentos com 10 classes e 80 imagens, vale destacar também o experimento com maior quantidade de imagens (fonte48 com 400 imagens) que teve um resultado de 83%.

4.1.6 Pinturas

Foram utilizadas 255 imagens de pinturas de diferentes movimentos artísticos cujos experimentos foram divididos em 9 fontes de dados.

Esse experimento visa testar a capacidade do algoritmo de identificar pinturas pertencentes a diferentes movimentos artísticos, uma possível aplicação é apontada por

Fonte	Classes	G
FONTE41(20)	2: INDIVIDUO_0(10); INDIVIDUO_1(10)	0.9444
FONTE42(30)	3: INDIVIDUO_0(10); INDIVIDUO_1(10); INDIVIDUO_2(10)	0.9629
FONTE43(40)	4: INDIVIDUO_0(10); INDIVIDUO_1(10); INDIVIDUO_2(10); INDIVIDUO_3(10)	0.9722
FONTE44(80)	8: INDIVIDUO_0(10); INDIVIDUO_1(10); INDIVIDUO_2(10); INDIVIDUO_3(10); INDIVIDUO_4(10); INDIVIDUO_5(10); INDIVIDUO_6(10); INDIVIDUO_7(10)	0.9861
FONTE45(80)	10: INDIVIDUO_0(10); INDIVIDUO_1(10); INDIVIDUO_2(10); INDIVIDUO_3(10); INDIVIDUO_4(10); INDIVIDUO_5(10); INDIVIDUO_6(10); INDIVIDUO_7(10); INDIVIDUO_8(10); INDIVIDUO_9(10)	0.9888
FONTE46(200)	20: INDIVIDUO_0(10); INDIVIDUO_1(10); INDIVIDUO_2(10); INDIVIDUO_3(10); INDIVIDUO_4(10); INDIVIDUO_5(10); INDIVIDUO_6(10); INDIVIDUO_7(10); INDIVIDUO_8(10); INDIVIDUO_9(10);; INDIVIDUO 19(10)	0.9666
FONTE47(300)	30: INDIVIDUO_0(10); INDIVIDUO_1(10); INDIVIDUO_2(10); INDIVIDUO_3(10); INDIVIDUO_4(10); INDIVIDUO_5(10); INDIVIDUO_6(10); INDIVIDUO_7(10); INDIVIDUO_8(10); INDIVIDUO_9(10) ;; INDIVIDUO_29(10)	0.8962
FONTE48(400)	40: INDIVIDUO_0(10); INDIVIDUO_1(10); INDIVIDUO_2(10); INDIVIDUO_3(10); INDIVIDUO_4(10); INDIVIDUO_5(10); INDIVIDUO_6(10); INDIVIDUO_7(10); INDIVIDUO_8(10); INDIVIDUO_9(10);; INDIVIDUO_39(10)	0.8361

Tabela 13 – Resultados - Faces Humanas

Fonte	Classes	G
FONTE61(20)	2: BIZANTINO(37); ROMANTICISMO(52)	0.9885
FONTE62(30)	2: GOTICO(31); BARROCO(49)	0.9615
FONTE63(40)	2: ROMANTISMO(52); CUBISMO(43)	0.9247
FONTE64(80)	2: BARROCO(49); EXPRESSIONISMO(43)	0.988
FONTE65(80)	2: CUBISMO(43); EXPRESSIONISMO(43)	0.94
FONTE66(200)	4: BIZANTINO(37); GOTICO(31); CUBISMO(43); EXPRESSIONISMO(43)	0.88
FONTE67(300)	4: BIZANTINO(37); GOTICO(31); ROMANTISMO(52); BARROCO(49)	0.8848
FONTE68(400)	4: ROMANTISMO(52); BARROCO(49); CUBISMO(43); EXPRESSIONISMO(43)	0.8251
FONTE69(400)	6: ROMANTISMO(52); BARROCO(49); CUBISMO(43); EXPRESSIONISMO(43); BIZANTINO(37); GOTICO(31)	0.7711

Tabela 14 – Resultados - Pinturas

TORRES: a busca de imagens por movimentos artísticos em uma base de dados de pinturas. com possível aplicação para busca de pinturas em uma vasta base de dados por movimento artístico, por exemplo.

Como pode ser visto na figura 14, os resultados desse experimento foram muito positivos, as árvores resultantes demonstram uma precisão acima de 77% para todos os experimentos realizados, chegando até 98% em alguns dos experimentos com apenas 2 classes.

4.2 Resultados da variação de informação (ϵ) nas imagens com a quantização para 32 cores

Foi calculado a variação de informação (ϵ) entre as imagens coletadas em seu formato real em relação e as imagens após o pré-processamento com perda realizado na etapa de **extração de características**, o objetivo com esse cálculo é medir através da variação entre as informações, um percentual de perda após o processo de redimensionamento e quantização.

A tabela 15 e o gráfico 16 apresentam a media dos resultados de variação obtidos, medias que não ultrapassam em nenhuma categoria o valor de 10%, uma baixa variação entre informação original e a informação após o pré-processamento. A equação utilizada

para o cálculo é descrita no capítulo 3.

Categoria	Média da variação da informação ϵ
Frutas	0.065
Animais	0.070
Plantações	0.099
CAPTCHA	0.045
Faces Humanas	0.017
Pinturas	0.075
Média Geral	0.061

Tabela 15 – Médias da variação da informação

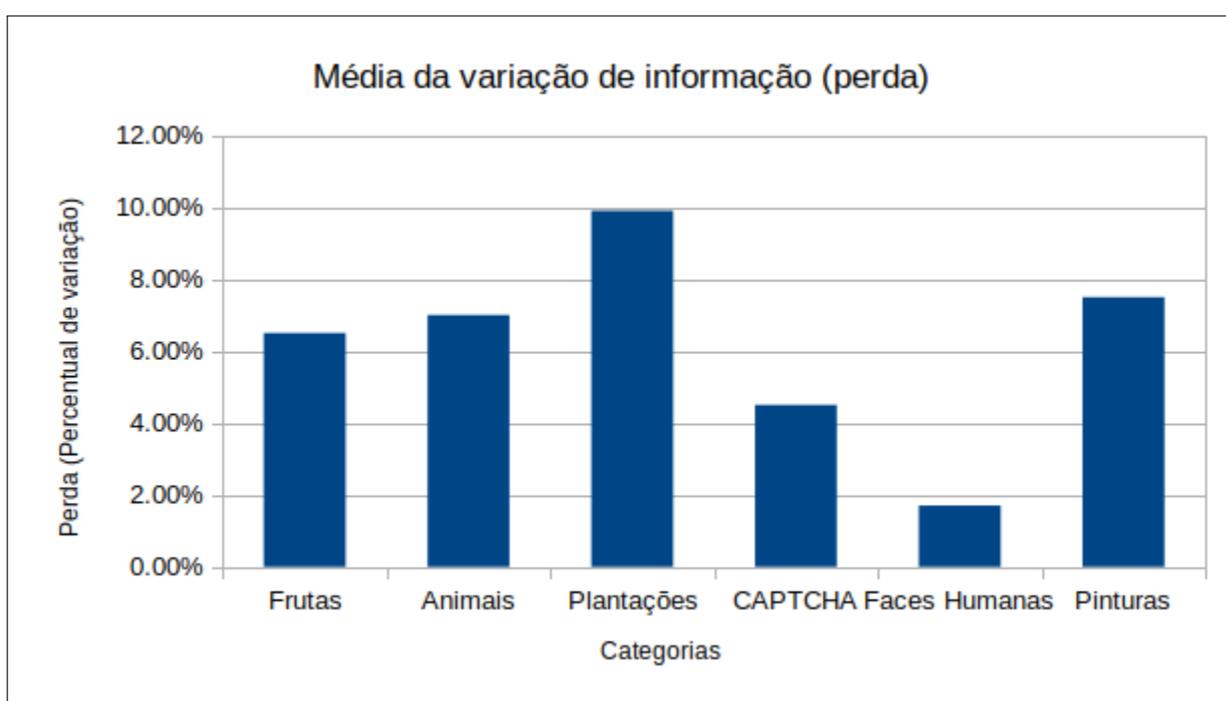


Figura 16 – Médias da variação da informação

4.3 Resultados empíricos (diferentes valores na etapa de quantização)

Foram realizados testes com diferentes valores para o número de cores na etapa de quantização, nesta seção os resultados são apresentados, vale a pena destacar o caso com apenas 4 cores em que se obteve uma **média** de resultados de 89%.

A tabela 16 contém a média de resultados obtidos nas diferentes categorias de experimentos realizados, com diferentes números de cores utilizados na etapa de quantização, além de uma média geral de todas as categorias para cada quantidade de cores testada.

A figura 17 mostra os gráficos obtidos com diferentes valores testados na etapa de quantização, de 4 até 256 cores. Com apenas quatro cores, observa-se os melhores resultados, com a maior parte dos resultados com precisão acima de 85%. Na categoria

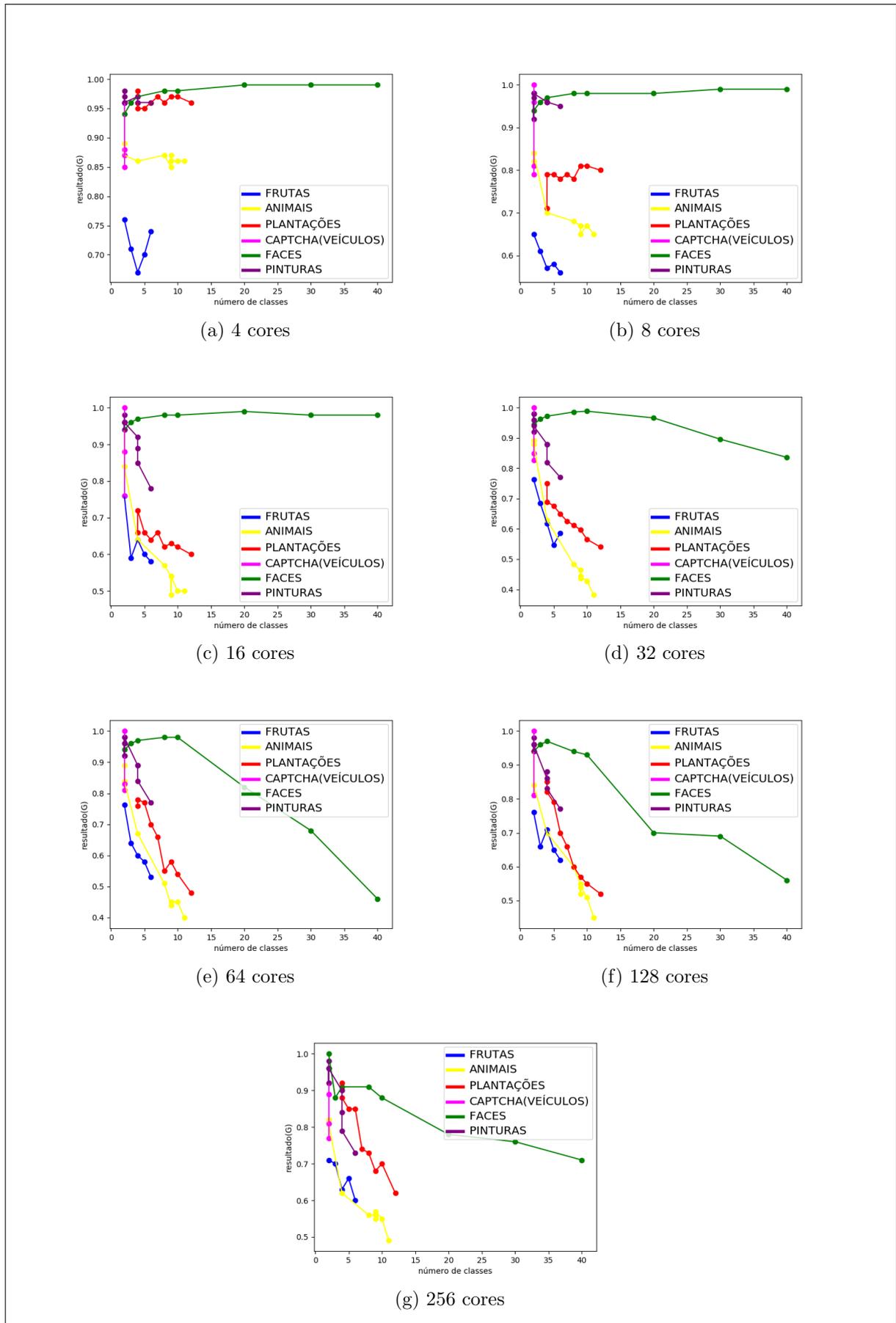


Figura 17 – Comparação dos resultados com diferentes números de cores na etapa de quantização

Quantidade de Cores	Frutas	Animais	Plantações	CAPTCHA	Faces Humanas	Pinturas	Media Geral
4 cores	0.71	0.86	0.96	0.89	0.97	0.96	0.89
8 cores	0.59	0.7	0.78	0.89	0.97	0.96	0.81
16 cores	0.63	0.62	0.64	0.90	0.97	0.91	0.78
32 cores	0.63	0.55	0.63	0.90	0.94	0.90	0.76
64 cores	0.62	0.56	0.64	0.90	0.84	0.90	0.74
128 cores	0.68	0.63	0.67	0.89	0.83	0.90	0.75
256 cores	0.66	0.61	0.77	0.84	0.85	0.89	0.77

Tabela 16 – Média de resultados com diferentes números de cores

de frutas, única categoria em que se tem resultados abaixo de 85%, os resultados variam entre aproximadamente 75% para o melhor resultado, e 65% para o pior resultado.

Percebe-se que, com exceção dos casos de transição para 128 e 256 cores, com o aumento do número de cores, que pode ser entendido como dimensão das características, há uma queda relevante na precisão dos resultados obtidos, o que pode ser entendido como eficácia da redução de características aplicada com o objetivo de maior generalização das cores.

4.4 Comparações com a ferramenta *damicoresp*

Os experimentos também foram realizados na aplicação **damicoresp** disponibilizada por CESAR em um repositório no github¹. A aplicação faz uso do fluxo tradicional da metodologia DAMICORE e demonstra ótimos resultados nos experimentos com textos realizados pelo autor.

Um problema nessa abordagem tradicional, é que sejam os dados: imagens, textos, áudio ou qualquer outro tipo, a representação dos mesmos é totalmente dependente do algoritmo encapsulado pelo compressor, levando a representações muitas vezes imprecisas, que não carregam informações relevantes dos dados para tarefas de reconhecimento de padrões.

As árvores obtidas com a alteração proposta para o fluxo de operações do framework: o desmembramento da etapa de compressão em uma fase de **extração de características** e outra de **codificação** (Figura 11), apresentam resultados de precisão (G) claramente superiores, como pode ser visto nas figuras 18 e 19, nos resultados da aplicação proposta utilizados para a comparação, as imagens foram quantizadas para 32 cores.

A figura 20 mostra a diferença entre os mínimos e máximos obtidos com as duas aplicações (aplicação proposta - *damicoresp*). Percebe-se que, com exceção do resultado máximo na categoria de faces (100% - 98% = -2%), todos os outros casos tiveram um aumento de precisão nos resultados, com aumentos de mais de 50% em alguns casos.

A aplicação proposta atinge o pior resultado na categoria de animais, com uma precisão de apenas 38% no experimento realizado com 11 classes, enquanto que a aplicação

¹Aplicação disponível em: <<https://github.com/brunokim/damicoresp>>

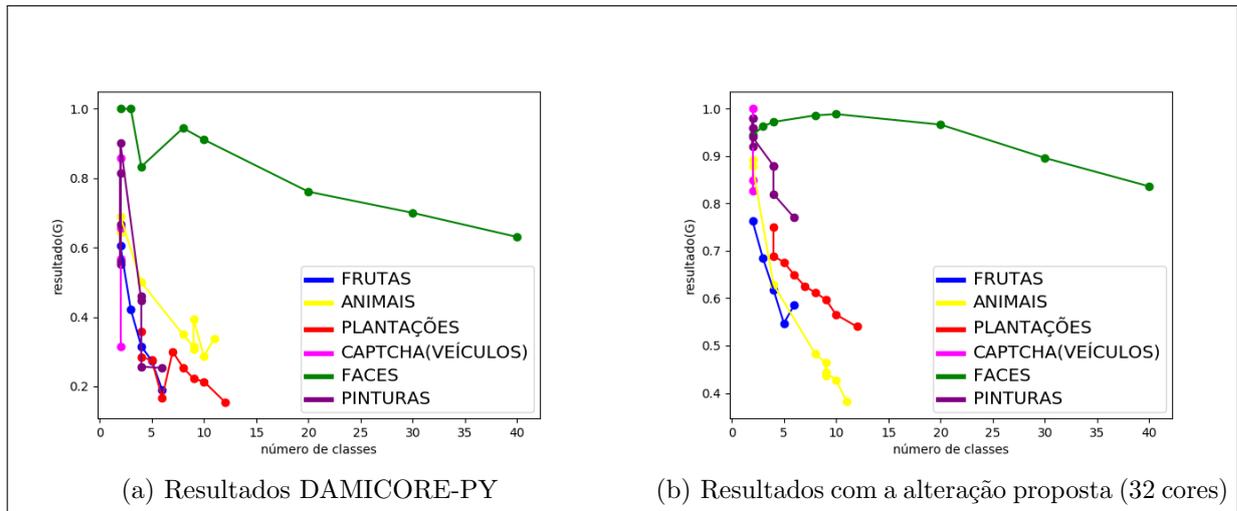


Figura 18 – Comparação dos resultados

damicorepy chega a 15% em um experimento com 12 classes na categoria de plantações (nesse mesmo experimento a aplicação proposta atinge 54%).

É perceptível nos dois casos que de acordo com o crescimento do número de classes, a precisão costuma ser prejudicada. Porém, também é perceptível que a aplicação proposta foi capaz de lidar melhor com o aumento do número de classes, com queda mais suaves nos valores de precisão.

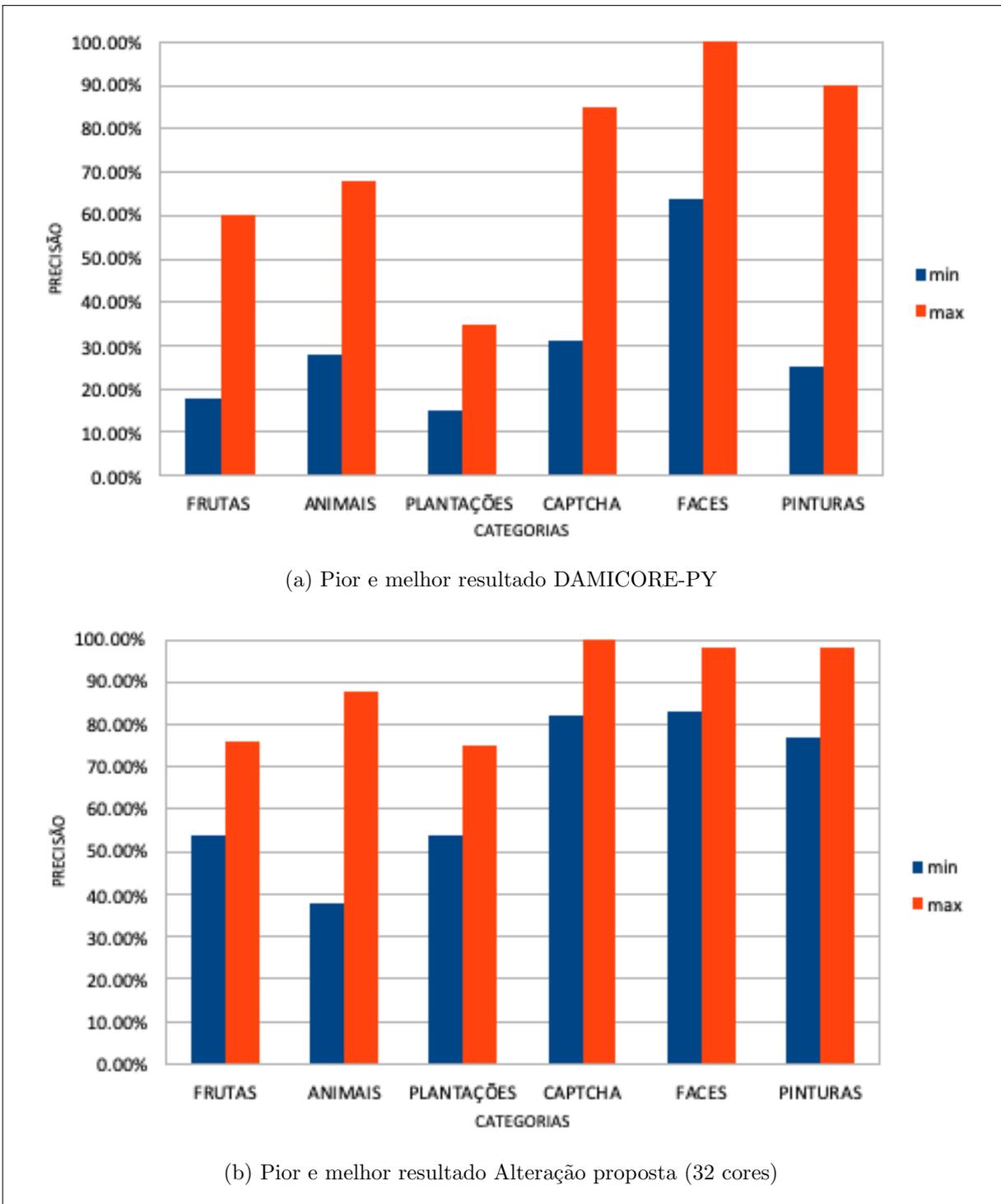


Figura 19 – Comparação dos piores e melhores resultados

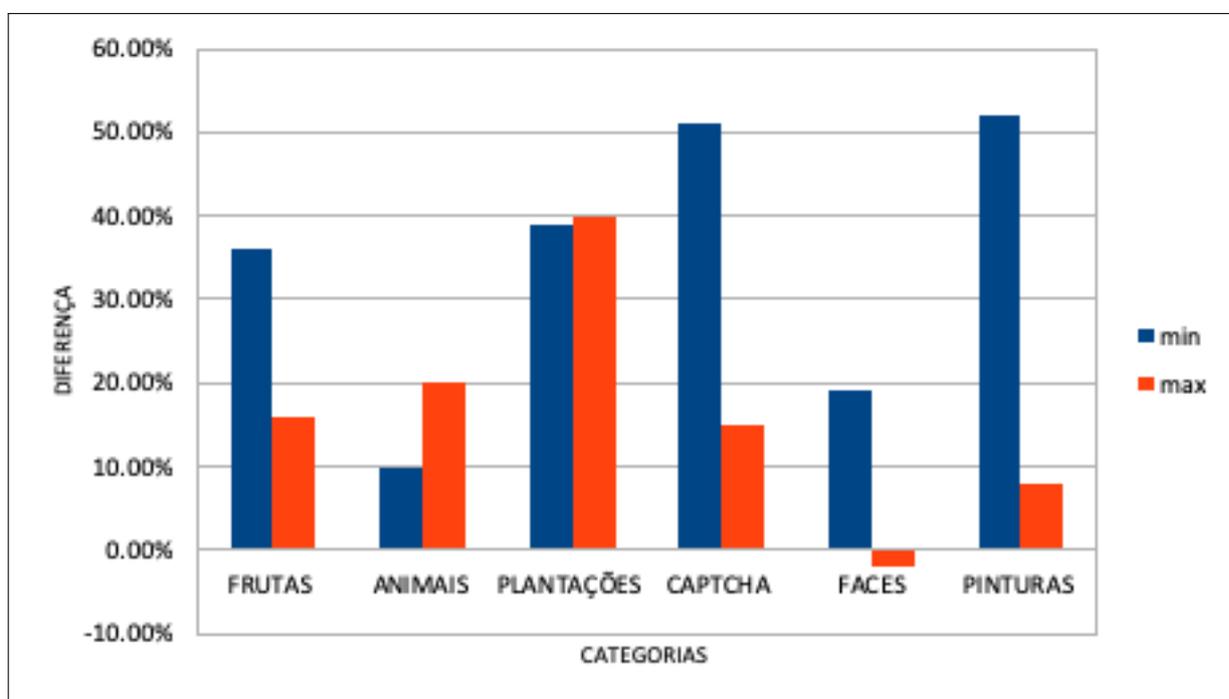


Figura 20 – Diferença entre resultados mínimos e máximos obtidos com a aplicação proposta e com a ferramenta damicorepy

5 CONCLUSÃO

A metodologia DAMICORE se mostra como uma ferramenta de baixa complexidade computacional capaz de produzir ótimos resultados em tarefas de agrupamentos e classificação, porém, em algumas situações o framework produz resultados ruins (CESAR, 2016; SANCHES; CARDOSO; DELBEM, 2011; SILVA; SALVINI, 2018; PINTO; DELBEM; MONACO, 2017; LOPES; DELBEM, ; MORO et al., 2014). Como possível causa dos resultados negativos, durante a pesquisa é apontado a dependência da representação dos dados com o algoritmo do compressor utilizado, o que pode levar a representações inadequadas.

A técnica proposta na pesquisa para a representação e codificação das imagens, demonstrou resultados significativamente melhores com imagens quando comparado ao método original. A aplicação mesmo tendo obtidos bons resultados, também obteve resultados ruins em alguns domínios de experimentos realizados. Possivelmente algumas modificações no pré-processamento e na extração de características das imagens poderiam trazer resultados melhores para esses casos, ideias são discutidas como trabalhos futuros.

Entre os experimentos realizados com a finalidade de testar a aplicação desenvolvida, vale a pena destacar os de faces humanas, imagens CAPTCHA e pinturas de diferentes movimentos artísticos. Nessas categorias foram obtidos os melhores resultados em todas as configurações testadas (diferentes números de cores na etapa de quantização).

Nessas categorias foram produzidas árvores filogenéticas com uma precisão acima de 77% para todos os experimentos realizados com 32 cores, demonstrando uma boa capacidade de generalização para a aplicação desenvolvida. Resultados ainda melhores foram obtidos com um número menor de cores (dimensionalidade reduzida), como por exemplo no caso de 4 cores em que todos os resultados se mantiveram acima dos 60% (Figura 17).

Alguns experimentos não tiveram resultados tão bons, isso provavelmente se deve ao fato de que os mesmos (plantações, animais e frutas) possuíam muita variação de fundo e resolução, além do arranjo dos objetos que se buscava identificar, como por exemplo: uma foto da classe banana pode ter uma única banana e um fundo branco, já em outra pode-se ter vários cachos de banana com uma floresta de fundo, o que acabaria gerando representações bem diferentes para cada uma dessas imagens mencionadas na abordagem utilizada.

Vale a pena destacar os experimentos realizados com apenas 4 cores que obtiveram resultados com precisão acima de 85% na maior parte dos casos, caindo abaixo desse valor apenas na categoria de frutas, onde os resultados obtidos se encontram entre 75% e 60% aproximadamente, não atingindo nenhum valor abaixo dos 65% em nenhum dos experimentos realizados, o que pode ser considerado um ótimo resultado.

Conclui-se com essa pesquisa que a métrica da distância de compressão é capaz de

produzir ótimos resultados desde que se esteja fazendo uso de um modelo de representação adequado para o tipo de dado a se tratar. O modelo de representação genérico baseado na contagem de ocorrência de cores apresentado e utilizado nesse trabalho foi capaz de gerar árvores filogenéticas bem organizadas para mais de um conjunto de dados, permanecendo acima dos 60% de precisão em todos experimentos realizados com apenas 4 cores na etapa de quantização.

Ainda assim, mesmo com os ótimos resultados proporcionado nos casos em que o número de cores é drasticamente reduzido, possivelmente pelo modelo de representação proposto considerar a frequência de cores ao invés de padrões, em alguns casos o modelo não obteve resultados adequados para qualquer situação e pode não funcionar de forma adequada em alguns casos, ideias são discutidas como trabalhos futuros na próxima seção.

5.1 Trabalhos Futuros

Como ideias para trabalhos futuros tem-se possibilidades de exploração do modelo de representação das imagens, além da extensão da aplicação para classificação e também para outros tipos de dados, como por exemplo: vídeos.

5.1.1 Possíveis modificações no modelo de representação das imagens

O modelo de representação proposto carrega como informação representativa de uma imagem, a sua entropia baseada na probabilidade das cores presente na imagem, o que já demonstra ótimos resultados, porém outras abordagens poderiam talvez gerar resultados ainda melhores, como por exemplo o uso de um algoritmo para detecção de padrões como o LZW(WELCH, 1984) na etapa de extração de características em uma representação linear da imagem produzida por uma curva preenchedora de espaço como a curva de Hilbert(TARVER, 2014), com a propriedade de preservação de localidade na representação linear da informação multidimensional.

5.1.2 Extensão para Classificação

Com a árvore gerada pelo algoritmo Neighbor Joining, é possível fazer uso da mesma como modelo de conhecimento para um agente que tem o objetivo de inferir rótulos para novas instâncias de imagem. Um algoritmo como o *k nearest neighbors* (KNN) possivelmente seria capaz de proporcionar bons resultados, mas o mesmo depende de cluster pré-definidos, o que poderia ser feito com a aplicação do algoritmo *fast newman* (FN).

Considera-se a não utilização de um algoritmo de clusterização, e a implementação de um novo algoritmo baseado no KNN para varrer a árvore em busca do elemento mais próximo à uma nova instância a ser classificada, classificando essa instância de acordo com os vizinhos desse elemento mais próximo na árvore, utilizando a própria árvore como

modelo de conhecimento, ao invés de confiar o modelo a algum algoritmo de clusterização que poderia introduzir maior imprecisão.

A figura 11 ilustra essa ideia da extensão para classificação, o que foi almejado como um dos objetivos iniciais, porém não realizado durante a pesquisa. De toda forma, testes poderiam ser realizados com as duas abordagens mencionadas para modelos de conhecimento: seja o modelo simplesmente uma árvore filogenética ou *clusters* definidos por algum algoritmo de agrupamento.

5.1.3 Extensão para outros tipos de dados

O modelo de representação apresentado foi desenvolvido especificamente voltado para imagens, proporcionando resultados claramente superiores quando comparado a total dependência ao algoritmo encapsulado pelo compressor, logo o desenvolvimento de outros modelos de representação para outros tipos de dados como texto, áudio e vídeo poderiam também proporcionar representações mais adequadas e conseqüentemente, resultados superiores em tarefas de RP.

Vídeos, por exemplo, são uma coleção de imagens. Uma técnica de compressão de vídeo poderia ser utilizada para eliminar a redundância entre *frames* (imagens) e essa coleção mais enxuta poderia ser representada como uma combinação das representações das imagens de acordo com o modelo proposto na pesquisa para representação das mesmas.

Referencias Bibliográficas

- AXELSSON, S. Using normalized compression distance for classifying file fragments. In: IEEE. **2010 International Conference on Availability, Reliability and Security**. [S.l.], 2010. p. 641–646. Citado na página 8.
- CAMBRIDGE, A. L. **The ORL Database of Faces**. 1994. Disponível em: <<https://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html>>. Citado 2 vezes nas páginas 17 e 18.
- CANARIA, U. de Las Palmas de G. **Explicación del sistema de compresión JPEG**. Universidad de Las Palmas de Gran Canaria, 2014. Disponível em: <<https://www.youtube.com/watch?v=Yj6a8Kefcmk&t=1574s>>. Citado na página 21.
- CESAR, B. K. M. **Estudo e extensão da metodologia DAMICORE para tarefas de classificação**. Tese (Doutorado) — Universidade de São Paulo, 2016. Citado 8 vezes nas páginas 1, 4, 5, 7, 11, 30, 39 e 43.
- CILIBRASI, R.; VITÁNYI, P. M. Clustering by compression. **IEEE Transactions on Information theory**, IEEE, v. 51, n. 4, p. 1523–1545, 2005. Citado na página 7.
- HUDSON, N. J. et al. Compression distance can discriminate animals by genetic profile, build relationship matrices and estimate breeding values. **Genetics Selection Evolution**, BioMed Central, v. 47, n. 1, p. 78, 2015. Citado na página 8.
- HUFFMAN, D. A. A method for the construction of minimum-redundancy codes. **Proceedings of the IRE**, IEEE, v. 40, n. 9, p. 1098–1101, 1952. Citado na página 6.
- LAN, Y.; HARVEY, R. Image classification using compression distance. In: CITESEER. **Proceedings of the 2nd International Conference on Vision, Video and Graphics**. [S.l.], 2005. p. 173–180. Citado na página 8.
- LI, F. F. **How we teach computers to understand pictures - TEDTalks**. 2015. [Online; accessed 8-junho-2019]. Disponível em: <<https://www.youtube.com/watch?v=40riCqvRoMs>>. Citado na página 4.
- LOPES, G. R.; DELBEM, A. C. Classificação ao de tráfego de ataques em redes de computadores através de técnicas de mineração de dados. Citado 2 vezes nas páginas 5 e 43.
- MORO, L. F. d. S. et al. Caracterização de alunos em ambientes de ensino online: Estendendo o uso da damicore para minerar dados educacionais. In: **Anais dos Workshops do Congresso Brasileiro de Informática na Educação**. [S.l.: s.n.], 2014. v. 3, n. 1, p. 631. Citado 2 vezes nas páginas 5 e 43.
- NORVIG, P.; RUSSELL, S. **Inteligência artificial: Tradução da 3a Edição**. Elsevier Editora Ltda., 2014. ISBN 9788535251418. Disponível em: <<https://books.google.com.br/books?id=BsNeAwAAQBAJ>>. Citado na página 3.

- OJALA, T.; PIETIKAINEN, M.; HARWOOD, D. Performance evaluation of texture measures with classification based on kullback discrimination of distributions. In: IEEE. **Proceedings of 12th International Conference on Pattern Recognition**. [S.l.], 1994. v. 1, p. 582–585. Citado na página 12.
- OLIVA, A.; TORRALBA, A. Modeling the shape of the scene: A holistic representation of the spatial envelope. **International journal of computer vision**, Springer, v. 42, n. 3, p. 145–175, 2001. Citado na página 12.
- PINTO, R. S.; DELBEM, A. C.; MONACO, F. J. Caracterização do perfil de carga a partir de programas binários. SBC, 2017. Citado 2 vezes nas páginas 5 e 43.
- REIS, S. F. dos; ZUBEN, F. J. V. **The Neighbor-Joining (NJ) Method Uma descrição passo-a-passo**. 2007. Disponível em: <ftp://ftp.dca.fee.unicamp.br/pub/docs/vonzuben/bp590_1s04/neighbor_joining.pdf>. Citado na página 8.
- SANCHES, A.; CARDOSO, J. M.; DELBEM, A. C. Identifying merge-beneficial software kernels for hardware implementation. In: IEEE. **2011 International Conference on Reconfigurable Computing and FPGAs**. [S.l.], 2011. p. 74–79. Citado 4 vezes nas páginas 1, 4, 5 e 43.
- SHANNON, C. E. A mathematical theory of communication. **Bell system technical journal**, Wiley Online Library, v. 27, n. 3, p. 379–423, 1948. Citado 4 vezes nas páginas 5, 9, 11 e 26.
- SHARMA, M. Compression using huffman coding. **IJCSNS International Journal of Computer Science and Network Security**, v. 10, n. 5, p. 133–141, 2010. Citado 2 vezes nas páginas 7 e 11.
- SILVA, J. J.; SALVINI, R. Análise de sentimentos de conteúdos textuais de redes sociais por meio de modelos de compressão de dados. In: SBC. **Anais da VI Escola Regional de Informática de Goiás**. [S.l.], 2018. p. 107–120. Citado 2 vezes nas páginas 5 e 43.
- TARVER, T. Hilbert’s space-filling curve. **Asian Journal of Mathematics and Applications**, v. 2014, p. 2307–7743, 06 2014. Citado na página 44.
- TORRES, P. H. L. Utilizando Árvores filogenéticas para a identificação de similaridades em pinturas digitalizadas. 2018. Citado 5 vezes nas páginas 1, 5, 7, 12 e 36.
- VISION, G. C. **How Computer Vision Works**. Youtube, 2015. Disponível em: <https://www.youtube.com/watch?v=OcycT1Jwsns>. Citado na página 3.
- WEIJER, J. V. D.; SCHMID, C.; VERBEEK, J. Learning color names from real-world images. In: IEEE. **2007 IEEE Conference on Computer Vision and Pattern Recognition**. [S.l.], 2007. p. 1–8. Citado na página 12.
- WELCH, T. A. A technique for high-performance data compression. **Computer**, IEEE, n. 6, p. 8–19, 1984. Citado na página 44.
- WILDING, J. **Perception: From Sense to Object**. Taylor & Francis, 2017. (Psychology Library Editions: Perception). ISBN 9781138206571. Disponível em: <https://books.google.com.br/books?id=-0YlvgAACAAJ>. Citado 2 vezes nas páginas 3 e 21.