UNIVERSIDADE D
COIMBRA

José Pedro Tomás Lopes

STONKINATOR:
GERADOR AUTOMÁTICO DE IMAGENS MEMÉTICAS

Dissertação no âmbito do Mestrado em Design e Multimédia orientada pelo Professor Doutor Pedro José Mendes Martins e co-orientada pelo Professor Doutor João Miguel Andrade Proença da Cunha apresentada ao Departamento de Engenharia Informática da Faculdade de Ciências e Tecnologia da Universidade de Coimbra.

Setembro de 2022

Universidade de Coimbra
Faculdade de Ciências e Tecnologia
Dissertação de Mestrado em Design e Multimédia

Stonkinator: Gerador Automático de Imagens Meméticas

José Pedro Tomás Lopes
2017249536

setembro de 2022

Resumo

Um *meme* é uma unidade de transmissão que carrega uma ideia ou comportamento entre seres humanos. Atualmente, esse termo é associado a imagens ou vídeos encontrados na internet, muitas das vezes com uma mensagem humorística. Numa sociedade cada vez mais ativa e global, os *memes* surgiram como meio de entretenimento e comunicação, e ao longo dos anos foram criadas diversas imagens que trazem consigo mensagens e ideias diferentes.

Aliando este crescimento relativamente à utilização de *memes*, com a forma veloz com que comunicamos através das redes sociais e outras aplicações, é proposto um projeto que explora a temática da Criatividade Computacional, e que tem como objetivo a criação de um sistema capaz de gerar *memes* num formato específico, para a utilização em redes sociais e conversas por escrito informais, de forma ágil e intuitiva para o utilizador, tomando partido de conceitos e técnicas de análise e mistura de imagens. Para isso foi feita uma análise de trabalhos relacionados e de conceitos e algoritmos estado da arte que permitiram o desenvolvimento deste sistema. Nesta dissertação são ainda enumeradas e descritas as diferentes fases do plano e processo de trabalho. De maneira a avaliar o sistema em termos de utilidade, foi realizado um estudo composto por vinte e três participantes para perceber se os *memes* resultantes do sistema seriam partilhados nas redes sociais e em conversas por escrito informais, e ainda quais as preferências dos utilizadores em relação aos métodos de *blending* implementados no sistema.

Palavras-chave: Visão por Computador, Mistura de Imagens, Análise de Imagem, Processamento de Linguagem Natural, Memes

Abstract

A *meme* is a transmission unit that carries an idea or behaviour among human beings. Nowadays this term comes associated with the idea of an image or video from the internet, usually with humorous purposes. In an increasingly active and global society, these *memes* came as a means of entertainment and communication, and throughout the years, many images were created to encode different meanings.

Combining this growth in the use of *memes*, with the fast way we communicate through social networks and other applications, the proposed project explores the theme of Computational Creativity, and aims to create a system capable of generating *memes* in a specific format, to be used in social networks and informal written conversations. The system also aims to be accessible to most people, providing an agile and intuitive way for the user to create *memes* by taking advantage of concepts and techniques of analysis and mixing of images and text. For this, an analysis of related works and of state-of-the-art concepts and algorithms that allowed the development of this system was carried out. This dissertation also lists and describes the different phases of the work plan and process. In order to evaluate the system in terms of usefulness, a study was carried out with twenty three participants to understand if the memes resulting from the system would be shared on social networks and in informal written conversations, and also what are the users preferences in relation to the methods of blending implemented in the system.

Keywords: Computer Vision, Image Blending, Image Analysis, Natural Language Processing, Memes

Aos professores Pedro Martins e João Cunha, por me terem orientado e ajudado ao longo do ano.

À minha família, por me terem proporcionado a oportunidade e condições para continuar a estudar na área que gosto.

À Telma, por nunca me deixar desmotivar e me ter sempre apoiado.

Às minhas amigas, Joana, Marília, Mariana, Raquel, Mafalda, Cátia, Inês e Eva, por me terem acompanhado ao longo da Licenciatura e Mestrado e por estarem presentes nos momentos mais difíceis.

Aos meus amigos mais próximos, Diogo, Bernardo, Kiko, Pedro e António, pela presença prolongada ao meu lado e por me proporcionarem sempre bons momentos de distração e entretenimento.

ÍNDICE

LISTA DE FIGURAS

1 INTRODUÇÃO

Um *meme* é uma unidade de transmissão que carrega uma ideia ou comportamento entre seres humanos, atravessando barreiras geracionais e culturais. Este termo surgiu pela primeira vez no livro *The Selfish Gene* (Dawkins, 1976). Atualmente, o termo *meme* tem uma associação a imagens e vídeos encontrados na internet, normalmente com teor humorístico, cujo objetivo é transmitir uma ideia ou mensagem, sendo capaz de transcender barreiras culturais e por vezes linguísticas (Shifman, 2013).

Hoje em dia existem ferramentas que facilitam a criação de um *meme* por parte do utilizador, por exemplo, ao disponibilizar imagens comummente utilizadas com as posições dos textos já definidas.[1] Existem outras soluções onde, por exemplo, o sistema gera uma frase com base numa imagem escolhida pelo utilizador,[2] ou o utilizador apenas escreve uma palavra como *input* e o sistema faz uma pesquisa de imagens e da semântica da palavra introduzida, de modo a encontrar uma imagem e uma frase para gerar um *meme* relacionado com as mesmas (Gonsalves, 2021).[3]

Porém, estes sistemas têm uma componente de aleatoriedade, não permitindo ao utilizador prever em que situação pode usar o *meme*, ou necessita de muitas ações por parte do utilizador para criar uma imagem com a mensagem pretendida. O projeto que se propõe assenta no desenvolvimento de um sistema capaz de gerar *memes* que esperam ir ao encontro das expectativas do utilizador, mantendo uma certa aleatoriedade que crie variedade nos resultados obtidos. Isto permite ao utilizador escolher uma imagem que melhor se adeque ao contexto a usar, não estando limitado apenas a uma opção.

1.1 MOTIVAÇÃO

Numa sociedade cada vez mais ativa e global, os *memes* surgiram como meio de entretenimento e comunicação nas redes sociais e em aplicações de mensagens instantâneas. Ao longo dos anos foram criados diversos *memes* que trazem consigo mensagens diferentes, variam dependendo do conteúdo da imagem, do contexto em que foram criados e das situações em que se usam.

Assim, a principal motivação para o desenvolvimento deste trabalho deriva da vontade de criar um sistema simples de utilizar, capaz

1 https://imgflip.com/memegenerator

2 https://imgflip.com/ai-meme

3 https://colab.research.google.com/github/robgon-art/ai-memer/blob/main/AI_Memer.ipynb

de gerar *memes* com o mínimo de entradas possíveis, que possam vir a ser utilizados durante conversas escritas, em aplicações de mensagens instantâneas ou nas redes sociais.

Para além de ser uma temática atual, este tema está relacionado com a área da Criatividade Computacional, localizada na intersecção dos campos da inteligência artificial, psicologia cognitiva, filosofia e artes. Esta área pretende simular ou replicar a criatividade utilizando um computador, e está simultaneamente de acordo com a formação adquirida ao longo da Licenciatura e do Mestrado em Design e Multimédia (LDM e MDM) da Faculdade de Ciências e Tecnologia da Universidade de Coimbra (FCTUC).

1.2 ENQUADRAMENTO

Os *memes* são uma forma de comunicação presente na atualidade, não exigindo muito rigor na sua criação; estes podem conter um estilo fotorrealista ou não fotorrealista, como por exemplo uma pintura ou um desenho. Deste modo, foi decidido que o processo de composição do *meme* iria passar pela análise e *blending* de imagens, ou seja, pela combinação de elementos ou zonas de duas imagens de forma a criar uma nova, para desenvolver um sistema capaz de gerar *memes* o mais autonomamente possível.

a) Stonks meme

b) Shef meme

c) Tehc meme

Figura 1.1: Exemplo de *memes* no formato "Stonks"
Fonte: (Adam & Don, 2020)

Como o resultado obtido continuava a ser demasiado abrangente, optou-se por restringir as imagens resultantes em *memes* semelhantes ao "stonks" – um *meme* derivado da palavra "stocks", usado de forma humorística em contextos de má gestão financeira. Desse *meme* surgiram outros como o "shef" e o "tehc", provenientes das palavras "chef" e "tech", ambos relacionados com situações caricatas relativas à culinária e à tecnologia, respetivamente (Adam & F, 2019).

1.3 ESTRUTURA DO DOCUMENTO

Esta secção descreve a estrutura da dissertação, abordando de forma resumida os assuntos que vão ser falados.

Depois da introdução, é apresentado o segundo capítulo, Estado da Arte, abordando a investigação teórica de conceitos e técnicas relacionadas com o *blending*, análise e segmentação de imagens. Este capítulo começa por expor a teoria do *conceptual blending*, passando de seguida para o estudo de algoritmos e sistemas para a análise e composição de imagens. Em relação aos métodos de análise e segmentação de imagem, são descritas algumas ferramentas que permitem realizar a deteção e segmentação de elementos de uma imagem e referidas algumas bibliotecas usadas para facilitar esse processo. Quanto ao estudo dos métodos de *blending* de imagens, as técnicas apresentadas são organizadas por ordem cronológica, de modo a mostrar a evolução de técnicas de *blending* e as diferentes ferramentas que usam para obter melhores resultados. Por fim, existe ainda uma secção com o objetivo de apresentar os recursos usados para a análise de palavras e de texto. Nesta secção também são analisados alguns conceitos chave aplicados no desenvolvimento dos algoritmos e algumas ferramentas fundamentais na elaboração dos sistemas.

O terceiro capítulo, Trabalhos Relacionados, apresenta quatro trabalhos que considero importantes para o projeto de tese. Estes trabalhos serviram de inspiração para algumas abordagens do processo de trabalho e trouxeram ideias para o desenvolvimento do mesmo.

O quarto capítulo, Metodologia, engloba os aspetos formais do trabalho, apresentando os objetivos a cumprir, o processo e abordagem para o seu desenvolvimento, o plano de trabalho e, ainda, os desafios esperados durante a elaboração do projeto.

O quinto capítulo, Gerador de Memes, apresenta e descreve o funcionamento dos módulos que compõem o sistema. Neste capítulo também são apresentados outros métodos testados que acabaram por não ser usados.

No sexto capítulo, Testagem, é apresentada a formalização da testagem e são discutidos os resultados obtidos

Por fim, o sétimo capítulo, Conclusão, apresenta as conclusões retiradas do trabalho realizado.

2

ESTADO DA ARTE

No capítulo apresentado é feita uma análise de conceitos, algoritmos e bibliotecas relevantes para o desenvolvimento do trabalho ou que estiveram presentes no processo do mesmo.

2.1 TEORIA DO BLENDING

Nesta secção é exposta a teoria do *conceptual blending*, que explica como o ser humano desenvolve e interpreta metáforas. Esta explicação ajuda a compreender o relacionamento entre as imagem apresentadas nos *memes* do formato *stonks* e a frase ou ação descrita nas imagens.

2.1.1 *Conceptual Blending*

A teoria do *conceptual blending* é uma teoria cognitiva, desenvolvida por Gilles Fauconnier e Mark Turner que vai ao encontro da explicação sobre como o ser humano desenvolve e interpreta metáforas cognitivamente (Fauconnier & Turner, 2002). Segundo esta teoria, elementos e relações vitais de diferentes cenários são integrados num processo subconsciente, estando presentes nos pensamentos e linguagem do quotidiano. Outra componente da teoria do *conceptual blending* foca-se na natureza recursiva do processo de *blending* e na habilidade de construir ou desconstruir a metáfora, para a produzir ou interpretar. Esta teoria apoia-se na noção de espaços mentais (Figura 2.1), que correspondem a estruturas compostas por um determinado número de conceitos e de interconexões. Os espaços mentais podem ser dinamicamente construídos durante um discurso (e.g., uma conversa) mas também podem preexistir no conhecimento do agente (Steinbrück, 2013).

Consideremos a frase "o cirurgião é um talhante", na qual podemos detetar a integração de dois espaços mentais: o espaço mental para "cirurgião", que inclui o próprio cirurgião e outros conceitos relevantes como o seu bisturi, o paciente, e a sala de cirurgia; e o espaço mental para o "talhante" que inclui o talhante, a faca, a sala de abate, etc. O *conceptual blending* precisa de dois ou mais espaços mentais como *input* (espaços de entrada). O mapeamento destes espaços mentais pode ser feito através de uma analogia estrutural. No exemplo do cirurgião-como-talhante, existe mapeamento entre: a faca do talhante e o bisturi do cirurgião, o animal e o paciente, a sala de abate e a sala de cirurgia. A seleção de elementos forma os elementos dos espaços de entrada que são então projetados para o espaço genérico.

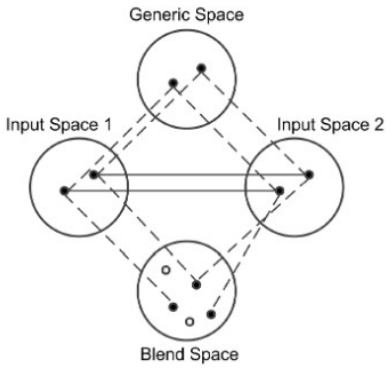


Figura 2.1: *Conceptual Blending*
Fonte: (Pereira & Cardoso, 2002)

Finalmente, o espaço de integração é construído ao elaborar melhor o espaço genérico, por exemplo, ao completar padrões.

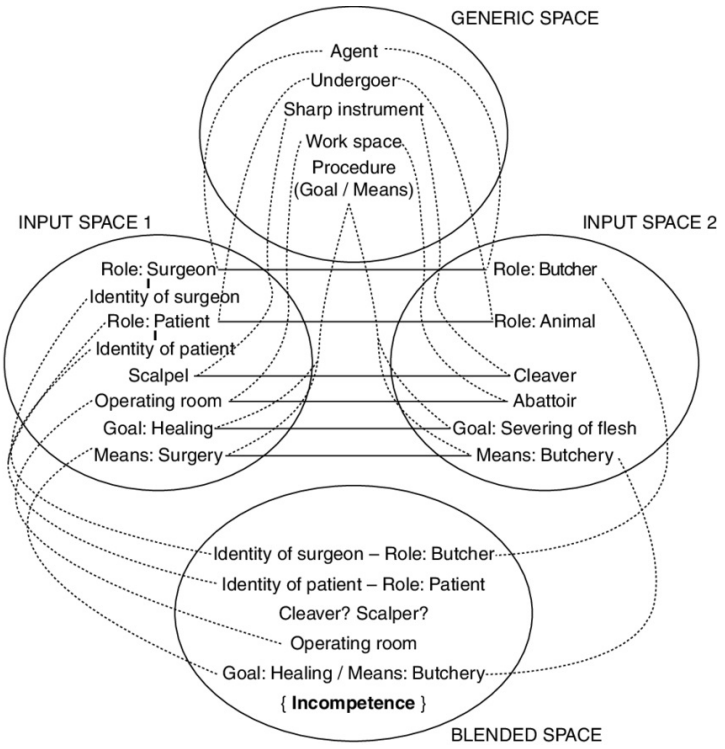


Figura 2.2: Exemplo de espaço mental para
a frase "o cirurgião é um talhante"
Fonte: (Birdsell, 2014)

Li et al. (2012) demonstraram que existem dois tipos de artefactos criativos que podem ser produzidos através do *conceptual blending*: misturas que têm um uso na comunicação, como a frase do cirurgião como talhante, e misturas que representam novos conceitos auto-sustentáveis. Estes novos conceitos estão bastante presentes na ficção como, por exemplo, os dragões. Também podem ser criados outros conceitos que vão para além da geração de criaturas: por

exemplo, no *Star Wars*, um "sabre de luz" representa uma mistura dos conceitos "espada" e "emissor de laser" (Steinbrück, 2013).

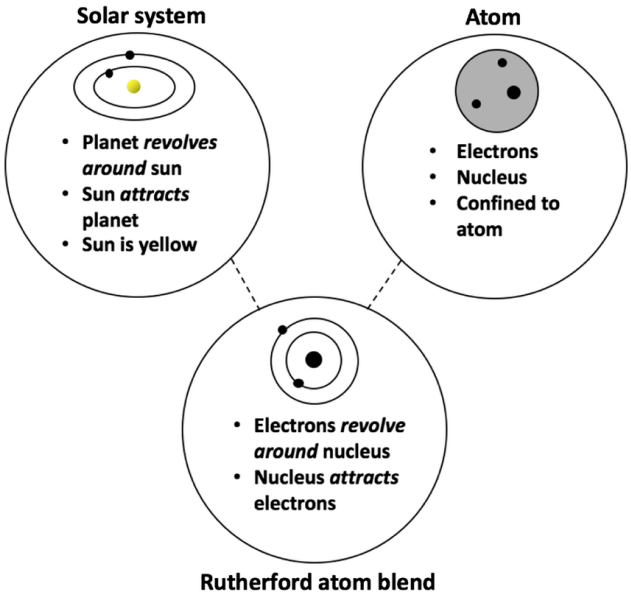


Figura 2.3: Exemplo de *conceptual blending* usado em física
Fonte: (Hoehn & Finkelstein, 2018)

2.2 ANÁLISE DE IMAGENS

Esta secção começa por apresentar um algoritmo fundamental para a deteção de faces, e vai evoluindo para ferramentas mais complexas que tiram partido de modelos neuronais para identificar elementos numa imagem.

2.2.1 *Algoritmo de Deteção Viola-Jones*

Viola & Jones (2001) apresentaram um método, conhecido por *Viola-Jones Algorithm*, que seleciona regiões importantes na imagem e compara-as com determinadas características das feições humanas. O processo de deteção de rostos é feito através de uma árvore de decisões, chamada *classifier cascade* (ou cascata de classificadores). Se o resultado de um classificador for positivo, é desencadeada a avaliação do classificador seguinte. Se esse resultado também for positivo é avaliado um terceiro classificador e assim por diante. Caso exista um resultado negativo, em qualquer momento, essa cascata é interrompida e é iniciada uma nova árvore de decisões referentes a outra zona da imagem. Este método atinge uma maior eficácia na deteção de faces, ao aumentar o desempenho e reduzir o tempo de computação (Viola & Jones, 2001).

Por exemplo, na Figura 2.4, podemos ver a comparação entre a imagem escolhida e as duas primeiras regiões correspondentes às ca-

racterísticas de uma face. A primeira característica tira partido do facto de, geralmente, a região dos olhos ser mais escura que a região que atravessa a parte superior das bochechas, e a segunda característica compara a intensidade entre as regiões oculares e a sua ponte, no topo do nariz (Wang, 2014).

Figura 2.4: Exemplo de características detetadas pelo algoritmo Viola-Jones
Fonte: (Viola & Jones, 2001)

2.2.2 *Mediapipe*

O *framework* Mediapipe é um *framework open-source* desenvolvido pela Google, que tira partido de aprendizagem automática para analisar e devolver pontos chave de uma imagem ou série de imagens, cujo objetivo é ser uma solução fácil de implementar em projetos de aprendizagem automática. As soluções do *framework* podem ser implementadas em Python, JavaScript, Android e mais, e incluem deteção de rosto, poses, mãos, segmentação de *selfies* e cabelo em imagens, entre outros (Lugaresi et al., 2019).

2.2.3 *HOG Descriptor*

O HOG (Histogram of Oriented Gradients) é um *feature descriptor* (ou descritor de características) usado em visão por computador para processamento de imagem, com o propósito de detetar objetos. Um *feature descriptor* é a representação de uma imagem ou um pedaço de imagem que simplifica a imagem original ao extrair informação útil e deixando de lado qualquer informação irrelevante. Este método ganhou visibilidade depois da publicação do artigo "Histograms of Oriented Gradients for Human Detection" (Dalal & Triggs, 2005). Este era um método considerado estado da arte, antes da utilização do *deep learning*, normalmente usado para a deteção de pedestres. O HOG descriptor foca-se na estrutura e forma de um objeto. É melhor do que qualquer outro *edge descriptor* porque tira partido da magnitude e ângulos de gradiente para analisar as características. Para as

regiões da imagem, são gerados histogramas usando a amplitude e orientação dos gradientes da imagem (Ahmad, 2019) (Mallick, 2016) (Tyag, 2021).

2.2.4 *ImageAI*

ImageAI (Moses & Olafenwa, 18) é uma biblioteca Python de visão por computador que dá a capacidade de desenvolvedores implementarem funções estado da arte de inteligência artificial nas suas aplicações e sistemas facilmente. Esta biblioteca oferece funções como:

- Reconhecimento de imagem e deteção de objetos, reconhecendo até mil objetos diferentes, e detetando e localizando oitenta objetos comuns do quotidiano numa imagem, usando modelos pré treinados;
- Análise e deteção de vídeo, que deteta, localiza e identifica oitenta objetos em vídeos e recolhe dados analíticos de cada frame, segundo e minuto. Esta função funciona com ficheiros de vídeo e câmaras de vídeo e de tráfego.
- Reconhecimento através de treino personalizado, permitindo treinar novos modelos de reconhecimento de imagem através de conjuntos de imagens personalizados, para casos específicos.

2.2.5 *Convolutional Neural Network (CNN)*

Uma Convolutional Neural Network (CNN) é um tipo de rede neuronal usada no reconhecimento e processamento de imagem, otimizada para o tratamento de dados sob a forma de pixéis. Desse modo, CNNs são arquiteturas fundamentais e básicas para a tarefa de segmentação de imagem em visão por computador (Odemakinde, 2022).

A arquitetura de CNN é composta por três camadas principais, como visto na Figura 2.5:

- Camada convolucional: Esta camada ajuda a tornar abstrato a imagem input, criando um mapa de características.
- Camada de *pooling*: Esta camada ajuda a reduzir o mapeamento de características ao resumir a presença de características em regiões do mapa de características.
- Camada totalmente conectada: Camadas totalmente conectadas que ligam todos os neurónios de uma camada a todos os neurónios de outra camada.

Combinar as camadas de uma CNN permite à rede neuronal desenhada aprender a identificar e reconhecer o objeto de interesse numa imagem. CNNs simples são criadas para classificar uma imagem e

detetar objetos com um único elemento na imagem, mas numa situação mais complexa, com múltiplos objetos numa imagem, uma arquitetura de CNN simples não é a mais ideal (Odemakinde, 2022).

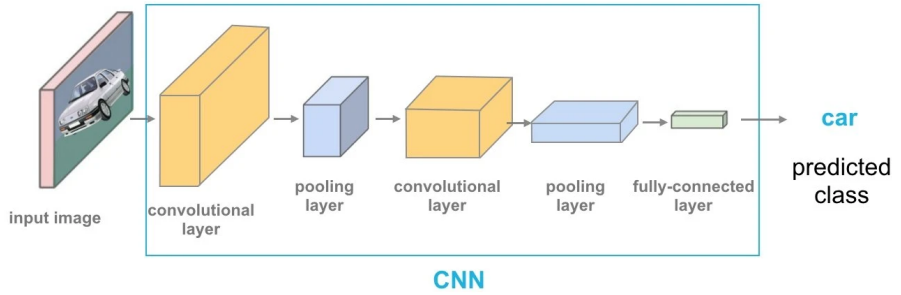


Figura 2.5: Esquema do funcionamento de um CNN
Fonte: (Odemakinde, 2022)

2.2.6 *Segmentação de Instâncias através de Mask-RCNN*

Deteção de objetos é o processo combinatório da localização de objetos e classificação de imagens. Isto significa que um algoritmo de deteção de objectos, dado um conjunto de identificadores/classificadores, começa por atribuir uma identificação/classe à imagem, assim como uma pontuação do nível de certeza. Ao mesmo tempo, o algoritmo é responsável por encontrar a posição do objeto nessa mesma imagem (coordenadas/tamanhos).

O algoritmo de segmentação de imagens é mais avançado do que o algoritmo de deteção de objetos ao gerar uma máscara, pixel a pixel, de acordo com a forma e área ocupada por cada objeto na imagem. Por outras palavras, isola a área coberta por cada objeto ao representar as suas formas através de máscaras com diferentes cores (Figura 2.6).

Desenvolvido por investigadores do Facebook em 2017, o Mask RCNN é uma rede neuronal usada para a segmentação de instâncias, ou seja, executa a segmentação de cada objeto individualmente. Esta rede foi criada a partir de dois modelos de aprendizagem:

- Faster R-CNN: É um modelo de deteção de objetos que devolve a posição de um objeto na imagem, assim como a pontuação do nível de confiança relativamente à previsão.
- FCN: É um modelo de rede convolucional completa que apenas usa camadas convolucionais.

A estrutura de Faster R-CNN incorporada na rede neuronal usa o modelo FCN para aplicar filtros convolucionais nas imagens input, de modo a aprender novas características. Depois, o modelo passa as características por uma Region Proposal Network (rede de proposta de regiões) para gerar potenciais regiões onde possa estar um objeto,

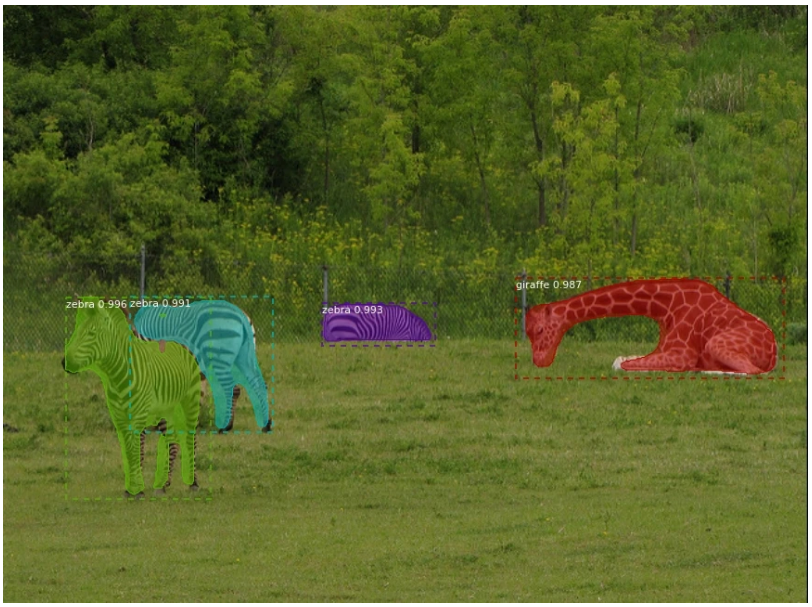


Figura 2.6: Exemplo de segmentação de imagens feita através de Mask-RCNN
Fonte: (Odemakinde, 2022)

assim como a sua posição. Para confirmar que cada potencial objeto está a ser processado, as características são filtradas através de camadas de *max-pooling*. Por fim, de modo a devolver os resultados desta aprendizagem, as potenciais características dos objetos são passadas por uma camada totalmente conectada, que as classifica em diferentes categorias e identifica as coordenadas para os retângulos delimitadores (He et al., 2017) (Maindola, 2021) (Odemakinde, 2022).

2.3 IMAGE BLENDING

Nesta secção são apresentados conceitos base do *blending* de imagens, assim como algoritmos e sistemas desenvolvidos para uma maior qualidade no resultado da mistura de imagens.

2.3.1 *Gaussian Laplacian Pyramid Blending*

Uma pirâmide de imagens é uma estrutura de dados que ajuda a representar a informação de uma imagem. Contém uma série de imagens duplicadas, equivalentes à imagem original, decompostas com um filtro passa-banda ou passa-baixo. Cada imagem duplicada tem uma resolução diferente, sendo disposta por ordem crescente de resolução, do topo ao fundo, formando uma pirâmide, como observado na Figura 2.7.

Para usar o algoritmo de *blending* multirresolução são necessários dois tipos de pirâmides, as gaussianas e as laplacianas, residindo a diferença nos dados guardados ao aplicar os filtros. Concretamente,

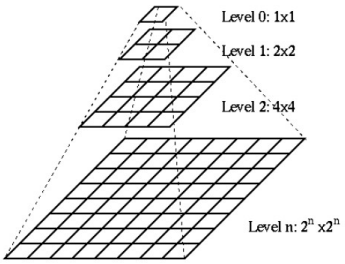


Figura 2.7: Representação da Pirâmide de Imagens
Fonte: (Drakos et al., 2013)

a pirâmide gaussiana é composta pelo conjunto de imagens esbatidas através de um filtro gaussiano, enquanto a pirâmide laplaciana contém os dados referentes ao detalhe perdido pelas imagens ao aplicar o filtro de esbatimento.

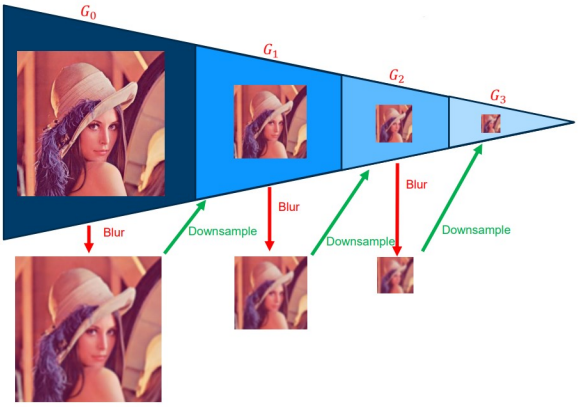


Figura 2.8: Pirâmide gaussiana
Fonte: (Dyrdal, Dyrdal)

Figura 2.9: Ao subtrair a imagem esbatida, com o filtro gaussiano, à imagem original, obtêm-se os detalhes dessa imagem, ou seja, o laplaciano
Fonte: (Dyrdal, Dyrdal)

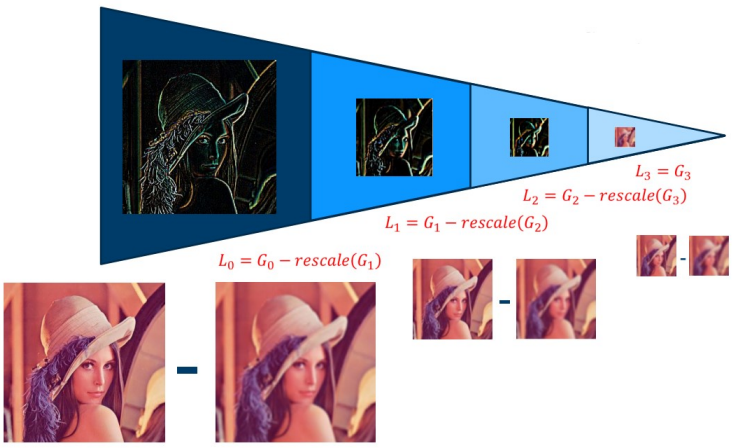


Figura 2.10: Pirâmide laplaciana
Fonte: (Dyrdal, Dyrdal)

Observando as Figuras 2.8 e 2.10, podemos associar cada imagem da pirâmide gaussiana a G_k e da pirâmide laplaciana L_k com k a indicar o nível da pirâmide. Para além disso, relativamente à Figura 2.10, definimos *rescale(image)* como uma função que altera a resolução da imagem image de volta à sua resolução anterior. Com base nestas variáveis, e de acordo com a Figura 2.9, é possível deduzir que se L_k = G_k - *rescale*(G_k *-1)* então G_k = L_k + *rescale*(G_k *-1)*.

Atendendo à informação apresentada anteriormente, é possível introduzir um algoritmo de *blending* de imagens. Para executar este algoritmo serão necessárias três imagens: fonte, alvo e máscara. Posteriormente, é necessário gerar a pirâmide gaussiana para as três imagens e a pirâmide laplaciana para a fonte e o alvo. Com estas pirâmides geradas, multiplica-se a pirâmide laplaciana da fonte e do alvo pela pirâmide gaussiana da máscara e somam-se os resultados, obtendo uma nova pirâmide que será reconstruída até à resolução das imagens originais, como se observa na Figura 2.11 (Radke, 2014).

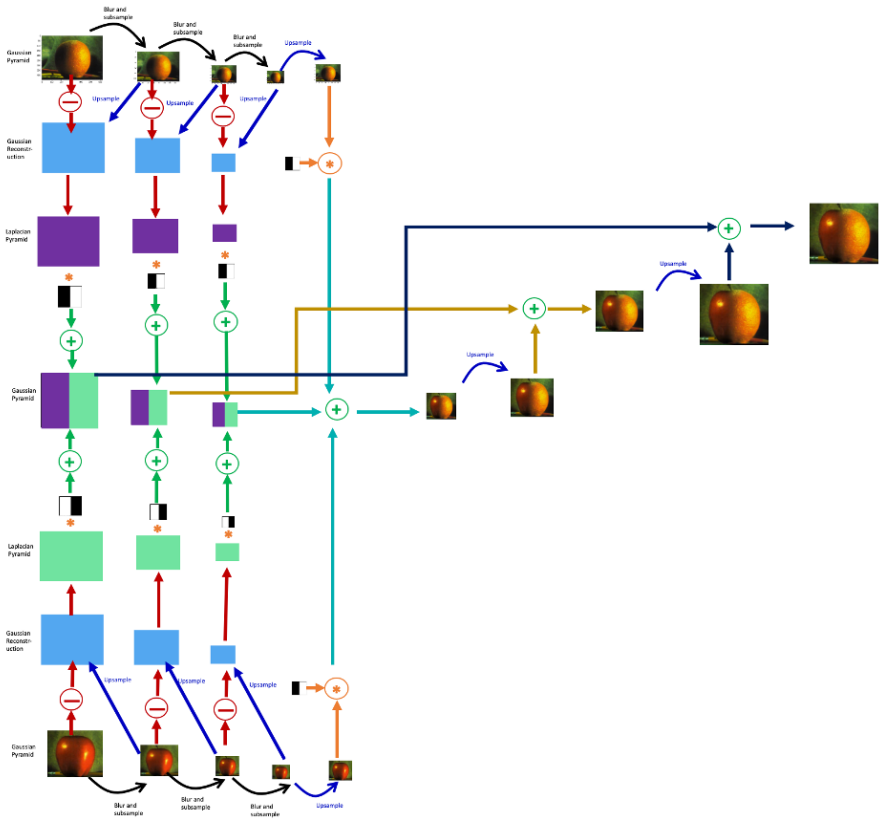


Figura 2.11: Representação do algoritmo de *blending* usando Pirâmides Laplacianas
Fonte: (Zhao, 2020)

2.3.2 *Poisson Blending*

A técnica de Poisson Blending foi proposta por Pérez et al. (2003) com o objetivo de obter alterações locais em regiões de uma imagem, manualmente selecionadas, o mais discretamente possível. Tem sido habitualmente usada para esconder artefactos ou detalhes que denunciem uma alteração da imagem original em aplicações/programas que prolonguem o tamanho dessa mesma imagem (Szeliski et al., 2011).

A ferramenta matemática usada como base para esta técnica é a equação de Poisson (com condições de limite de Dirichlet), que especifica o laplaciano de uma função desconhecida sobre o domínio de interesse, assim como os valores dessa função desconhecida ao longo dos limites do domínio (Pérez et al., 2003).

A ideia é que, atendendo à Figura 2.12, ao realizar a composição de imagens, as bordas dentro de Ω sejam iguais às de S, e as cores dentro de $\partial\Omega$ sejam iguais às de Ω (Radke, 2014).

Com este sistema também é possível recriar as técnicas de clonagem usadas em ferramentas já existentes (Figura 2.13). Isto permite ao utilizador remover ou adicionar objetos facilmente, de forma a não

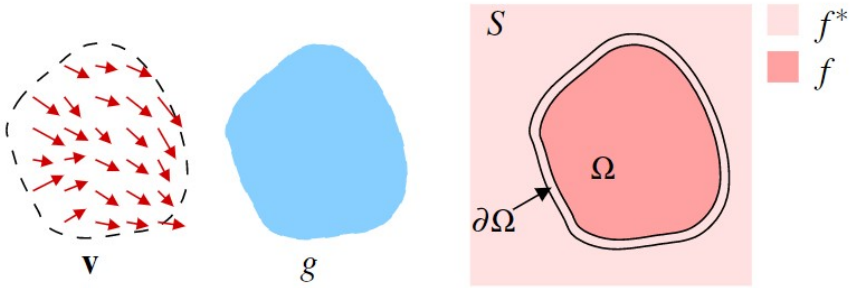


Figura 2.12: f: função de destino, f^*: função de fonte, S: fonte, Ω: destino, $\partial\Omega$: domínio de interesse
Fonte: (Pérez et al., 2003)

existirem artefactos na imagem que denunciem essa alteração. Ao misturar o gradiente da imagem fonte com o gradiente do destino, é possível também adicionar objetos transparentes de forma convincente. Para além disso, esta técnica permite a adição de objetos com contornos complexos, inclusive aberturas, de forma automática e sem ser necessário o recorte dessas secções.

O mesmo sistema pode ser usado para modificar a aparência de uma imagem num domínio específico, evitando descontinuidades visíveis no domínio de interesse. Em particular, é possível alterar a cor, textura ou iluminação de um objeto de forma simples e sem a necessidade de delinear precisamente os limites do objeto (Pérez et al., 2003).

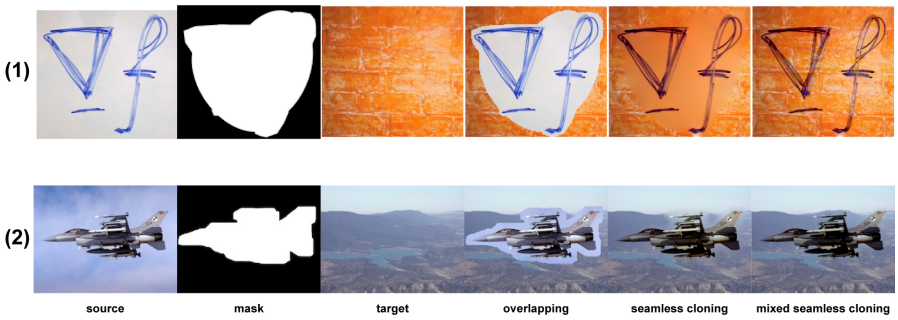


Figura 2.13: Exemplo de resultados do Poisson Blending
Fonte: (rinsa318, 2019)

2.3.3 *Generative Adversarial Networks*

Modelos Generativos Profundos (Deep Generative Models, DGMs) como, por exemplo, Restricted Boltzmann Machines (RBMs), Deep Belief Networks (DBNs), Deep Boltzmann Machines (DBMs), Denoising Autoencoder (DAE) e Generative Stochastic Network (GSN), começaram a chamar atenção por capturarem distribuições ricas e ocultas de áudio, imagens, vídeo, e sintetizarem novas amostras. Estes

modelos generativos são baseados no algoritmo Markov Chain Monte Carlo (MCMC) (Saxena & Cao, 2021). A integração Monte Carlo desenvolve amostras médias, baseadas noutras amostras, cujo resultado é próximo das expectativas. O algoritmo Markov Chain Monte Carlo retém esses valores médios ao corrê-los através de uma cadeia Markov durante muito tempo (Gilks et al., 1995). Isto significa que estes métodos vão ter resultados demorados e nem sempre precisos por serem baseados em probabilidades (Saxena & Cao, 2021).

De modo a combater os problemas mencionados anteriormente, Goodfellow et al. (2014) propõe uma metodologia de treino diferente para modelos generativos: as Redes Generativas Adversariais (Generative Adversarial Networks, GANs) (Saxena & Cao, 2021). As GANs podem ser caraterizadas como um par de redes em competição uma com a outra (Figura 2.14). Uma analogia comum para para o funcionamento das GANs é pensar numa rede como um criador de arte e a outra como um avaliador de arte. O criador, denominado de gerador, G, cria objetos com o objetivo de desenvolver imagens realistas. O avaliador, conhecido por discriminador, D, recebe as imagens criadas por G e imagens reais, sendo o seu objetivo distinguir a artificial da real. Ambos são treinados simultaneamente e em competição um com o outro (Creswell et al., 2018).

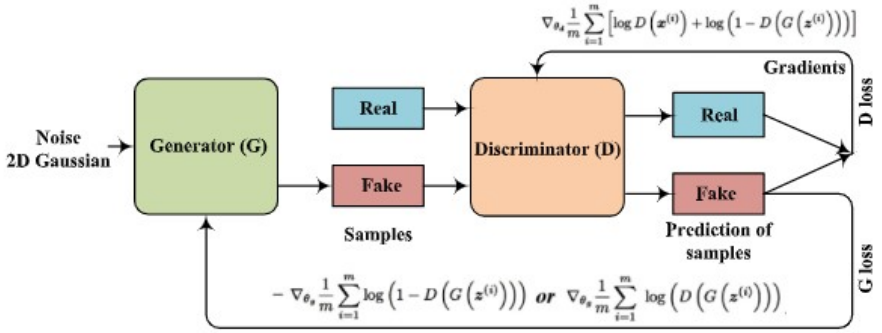


Figura 2.14: Arquitetura base de uma GAN
Fonte: (Saxena & Cao, 2021)

2.3.4 *Deep Image Blending*

Deep Image Blending é um algoritmo de duas fases, proposto por Zhang et al. (2020), com o objetivo de criar um sistema que permita a mistura de imagens evitando perda de estilo e conteúdo e permitindo o uso de uma máscara sem limites perfeitamente delineados. Na primeira fase, o algoritmo gera um limite homogéneo para a região de origem e, seguidamente, aperfeiçoa essa região com estilos e texturas idênticas às da imagem alvo. Neste algoritmo, é proposta uma perda diferencial que força o propósito equivalente ao objetivo original da equação de Poisson, podendo ser facilmente combinado com

outras funções de reconstrução objetivas. O algoritmo proposto funciona corretamente com imagens alvo do mundo real, assim como com imagens não fotorrealistas, como pinturas, ao utilizar técnicas que evitam a perda de estilo e conteúdo. A região de mistura é reconstruída ao atualizar os pixels iterativamente (Figura 2.15), usando um algoritmo de optimização (L-BGS solver). Adicionalmente, o algoritmo resolve a composição de imagens usando apenas uma imagem de origem, uma imagem máscara que delineie aproximadamente a região a misturar e uma imagem alvo (Zhang et al., 2020).

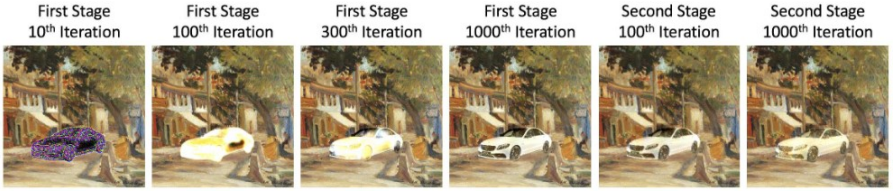


Figura 2.15: Reconstrução de imagens iterativa nas diferentes fases (Zhang et al., 2020)

2.4 PROCESSAMENTO DE LINGUAGEM NATURAL

Nesta secção são analisadas as diferentes bibliotecas usadas pelo primeiro módulo do sistema para analisar a frase introduzida pelo utilizador e obter palavras relacionadas com a mesma.

2.4.1 *Wordnet*

WordNet (Princeton University, 2010) é uma base de dados lexical da língua inglesa. Nomes, verbos, adjetivos e advérbios são agrupados em conjuntos de sinónimos cognitivos, denominados *synsets*, cada um expressando conceitos distintos.

Apesar do Wordnet se assemelhar a um dicionário de sinónimos, existem algumas distinções importantes. Em primeiro, o Wordnet interliga não só a forma da palavra, mas também os sentidos específicos da mesma. O resultado disso é a inexistência de ambiguidade semântica entre duas palavras que sejam próximas uma à outra na base de dados. Em segundo, o Wordnet classifica as relações semânticas entre palavras, enquanto que num dicionário de sinónimos o agrupamento de palavras não segue nenhum padrão específico para alem da semelhança do significado (Princeton University, 2010).

2.4.2 *NLTK*

Natural Language Toolkit (NLTK) (Bird et al., 2009) é um framework para o desenvolvimento de programas, em Python, que trabalha com dados relativos à linguagem humana. Esta ferramenta inclui recursos

lexicais e corporais, como o Wordnet, assim como um vasto conjunto de funções para extrair e devolver informações relativas ao contexto e significado das palavras.

Recorrendo ao Wordnet, em conjunto com o NLTK, é possível obter sinónimos e antónimos de palavras, e usando o package Corpus do NLTK é possível ler e identificar o contexto semântico das palavras introduzidas. Para além dos sinónimos e antónimos, o Wordnet também relaciona semanticamente as palavras através de hiperónimos, holónimos, hipónimos e merónimos (Bird et al., 2009) (Princeton University, 2010) (Gübür, 2021).

2.4.3 *RAKE*

Rapid Automatic Keyword Extraction (RAKE) (Rose et al., 2010) é um algoritmo de extração de palavras-chave, usado para determinar e classificar expressões-chave num corpo de texto através de um sistema de pontuação. A composição deste algoritmo pode ser dividida em três partes. A primeira converte o texto introduzido para caixa-baixa e transforma-o numa lista de palavras. A segunda parte tira partido dos sinais de pontuação e de uma *stop words list*, uma lista de palavras normalmente usada em algoritmos de processamento de linguagem natural com o objetivo de selecionar as palavras mais comuns como preposições, pronomes e conjunções coordenativas (Anindya, 2020), de modo a dividir a lista criada anteriormente em expressões. Na última parte é feita a classificação das expressões obtidas através de uma fórmula que utiliza valores de frequência de palavras no corpo de texto e a relação das palavras com as expressões (Agarwal, 2020); (Saxena, 2020).[1]

1 https://thinkinfi.com/automatic-keyword-extraction-using-rake-in-python/

3

TRABALHOS RELACIONADOS

3.1 CONCEPTUAL BLENDING FOR THE VISUAL DOMAIN

A dissertação "Conceptual Blending For The Visual Domain"pretende definir um sistema que formalize o processo do *conceptual blending* e o aplique ao domínio visual. Para isso, Steinbrück (2013) divide a arquitetura do sistema em cinco módulos. Os primeiros dois módulos são dedicados à aquisição de conhecimento e permitem uma construção dinâmica de conhecimento base. O primeiro módulo deteta características visuais numa imagem através do uso de diferentes algoritmos de visão por computador, como o Grabcut (Rother et al., 2004). O módulo seguinte recolhe conhecimento semântico sobre o conceito apresentado numa imagem, com ajuda de ontologias de senso comum. Os restantes módulos lidam com o processo de composição da imagem. O terceiro módulo implementa as regras para a seleção da melhor combinação de pares e o quarto módulo seleciona as partes de cada imagem a estarem envolvidas no processo de *blending*. Por fim, o último módulo executa o processo de combinação de imagens e gera o resultado final (Figura 3.1). No âmbito deste projeto, interessam-me maioritariamente os módulos que analisam características visuais e fazem o *blend* de imagens, mais concretamente, o primeiro e último módulo (Steinbrück, 2013).

a) Mistura de uma carapaça de caracol e um escudo da polícia

b) Mistura de uma roda de bicicleta e de um globo

Figura 3.1: Exemplo de imagens geradas pelo trabalho de Steinbrück (2013)
Fonte: (Steinbrück, 2013)

3.2 VISMANTIC: MEANING-MAKING WITH IMAGES

Vismantic é um sistema semi-automático desenvolvido por Xiao & Linkola (2015) que gera propostas de composições visuais de modo a expressar o significado de determinada expressão ou frase. Utiliza o conhecimento conceptual para encontrar diversas representações visuais de conceitos abstratos, com a capacidade de juntar duas imagens de três formas diferentes: justaposição, fusão e substituição. O sistema recebe uma frase ou expressão e identifica um "sujeito" e uma "mensagem transmitida". De seguida, o sistema em si é dividido em três módulos: o primeiro encontra e filtra imagens de modo a serem representativas do "sujeito" e "mensagem"; o segundo módulo faz um pré-processamento para realizar outra filtragem, analisando as imagens de modo a prever e a retirar as que tenham um resultado final indesejado no módulo seguinte; por fim, o terceiro módulo faz a junção entre a imagem "sujeito" e "mensagem", aplicando as técnicas de justaposição, fusão e substituição, obtendo o resultado final (Xiao & Linkola, 2015). Apesar do sistema apresentar potencial para bons resultados, como os da Figura 3.2, o sistema apresentado necessita de um maior nível de automação para obter melhores resultados, como se pode verificar pelos resultados da Figura 3.3. Não obstante, a forma como as imagens são obtidas e filtradas neste trabalho são uma solução que poderá ser útil no desenvolvimento do segundo módulo do projeto de tese.

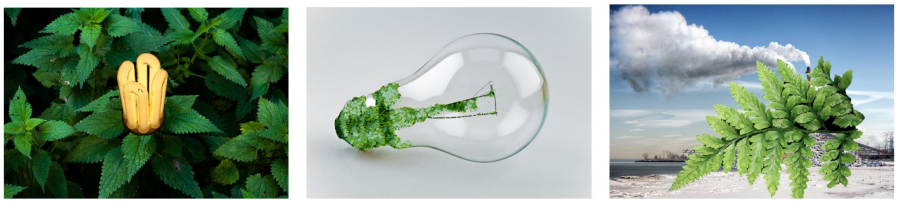


Figura 3.2: Exemplo de imagens geradas pelo trabalho de Xiao & Linkola (2015) para a ideia "electricity is green (sustainable)"
Fonte: (Xiao & Linkola, 2015)

Figura 3.3: Exemplo de resultados falhados gerados pelo trabalho de Xiao & Linkola (2015)
Fonte: (Xiao & Linkola, 2015)

Neste trabalho, Wu et al. (2019) focaram-se em resolver o problema de misturas entre imagens de alta resolução. Apesar de já existirem métodos clássicos com bons resultados no que toca à resolução, como o Poisson Blending, estes resultados tendem a ser irrealistas e podem conter diversos tipos de artefactos não pretendidos. Para resolver esse problema, os autores propõem uma framework que designam de "Gaussian-Poisson Generative Adversarial Network (GP-GAN)" (Figura 3.4).

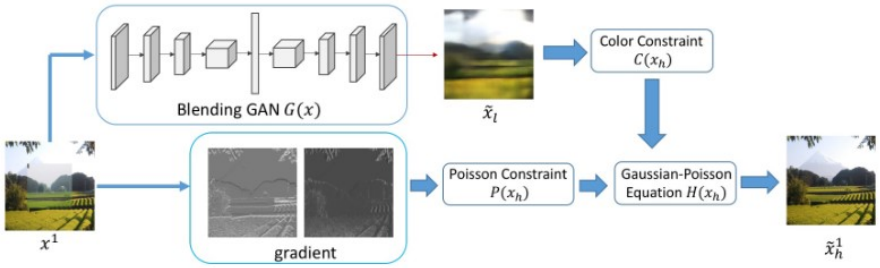


Figura 3.4: *Framework* do sistema GP-GAN
Fonte: (Wu et al., 2019)

Este sistema junta métodos baseados em gradientes com GANs, de modo a tirar partido do maior número de vantagens dos dois métodos. A diferença em relação às outras abordagens baseadas em gradientes reside na utilização das GANs para gerar uma imagem realista de baixa resolução que serve de restrição de cor, resultando numa composição de imagens mais natural. Este método pretende atingir transições suaves nas bordas das imagens fonte e alvo, assim como reduzir as diferenças de iluminação e de cor. Deste modo é possível gerar uma imagem com um *blending* bem executado, ainda que não exista muito rigor na criação da máscara a ser usada. Este projeto tem relevância para o meu trabalho, uma vez que permite realizar o *blending* automático de imagens com uma qualidade superior à existente em técnicas mais tradicionais (Wu et al., 2019) (Figura 3.5).

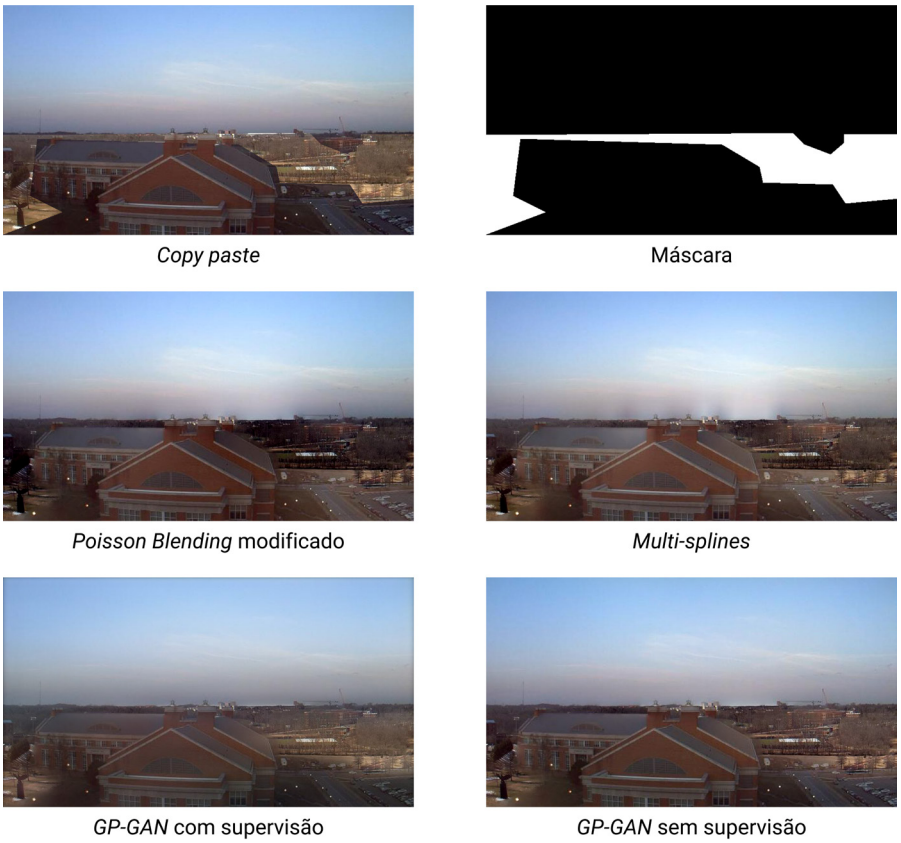
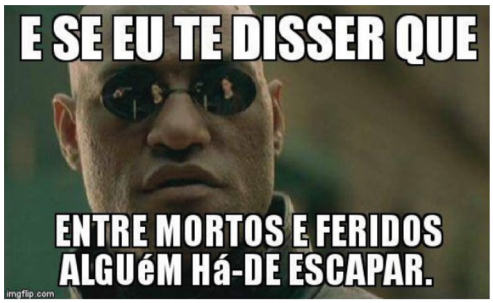


Figura 3.5: Comparação de resultados usando diferentes técnicas de *blending* de imagem
Fonte: (Wu et al., 2019)

3.4 ONE DOES NOT SIMPLY PRODUCE FUNNY MEMES! – EXPLORATIONS ON THE AUTOMATIC GENERATION OF INTERNET HUMOR

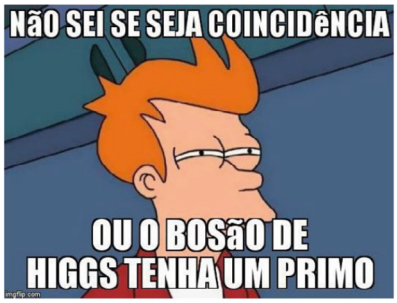
Neste artigo Oliveira et al. (2016) apresentam um sistema para a geração automática de *memes* da internet, baseados em imagens-macro, ou seja, potenciais combinações entre uma imagem e um texto, com o intuito de serem espalhadas em redes sociais. Os *memes* são produzidos a partir de cabeçalhos de notícias, para os quais, de acordo com características linguísticas, são associadas determinadas imagens-macro e o texto é adaptado (Figura 3.6). Os autores testaram o sistema com o objetivo de avaliar os resultados do sistema em termos de coerência, adequabilidade, surpresa e humor, e compará-los com *memes* criados por ser humanos. Apesar de terem obtido avaliações positivas e neutras, os autores concluíram que, no geral, os *memes* gerados automaticamente ainda estão a um nível inferior dos gerados por humanos.

Relativamente aos métodos e ferramentas usadas, os cabeçalhos de notícias portuguesas são obtidos automaticamente, através do Google News RSS Feed.



a) Resultado para a notícia
"Acidente faz nove feridos e condiciona trânsito no IC2"

a) Resultado para a notícia
"Casa mais cara do mundo foi vendida
em Paris por 275 milhões de euros"



a) Resultado para a notícia
"É coincidência ou o bosão de Higgs tem um primo?"

Figura 3.6: Exemplo de resultados gerados pelo trabalho de Oliveira et al. (2016)
Fonte: (Oliveira et al., 2016)

Durante a segunda fase são utilizadas ferramentas de recursos linguísticos. Para selecionar um *meme* de acordo com um cabeçalho, o texto é identificado através do "part-of-speech" (POS) e lematizado. Para esta parte são usados um identificador baseado em OpenNLP com modelos portugueses e o lematizador LemPORT para palavras em português. Para identificar o sentimento presente nas palavras do cabeçalho é usado o SentiLex, onde são anotadas as polaridades das palavras em português. Quando é necessário usar flexões para produzir o texto final, é utilizado o LABEL-LEX, um léxico morfológico para português. Para trabalhar os verbos que precisam de ser nominalizados recorre-se à ferramenta Nomlex-PT, um léxico de nominalização para português.

A identificação da palavra mais relevante no cabeçalho é simplificada através da seleção do verbo, nome ou adjetivo menos usado, de acordo com as listas de frequência do projeto AC/DC. A palavra selecionada tem de estar incluída nessas listas. Também podem ser usados os provérbios disponibilizados no âmbito do projeto Natura. A semelhança semântica entre o provérbio e o cabeçalho é feita através da semelhança média de nomes, verbos e adjetivos contidos, utilizando o método PMI-IR na Wikipédia portuguesa.

Na última fase o texto do *meme* é adicionado à imagem-macro com a ajuda da API Imgflip. O resultado obtido é postado na rede social Twitter utilizando o Twitter4J (Oliveira et al., 2016).

4 METODOLOGIA

O objetivo deste trabalho é desenvolver uma ferramenta de uso acessível para o utilizador, capaz de gerar imagens a partir de uma entrada textual. Desse modo, a secção 4.1 apresenta os objetivos que o sistema pretende cumprir, e na secção 4.2 é descrito o plano e processo de trabalho.

Para o desenvolvimento do projeto foi seguida a metodologia Design Science Research, por se focar na procura de soluções para um problema inicial e por permitir resolver os diferentes desafios de cada módulo de forma iterativa (Venable et al., 2017).

4.1 OBJETIVOS

O sistema desenvolvido pretende cumprir os seguintes objetivos:

- Analisar uma frase ou expressão e obter palavras relacionadas com a mesma;
- Recolher e filtrar imagens autonomamente, através de uma entrada;
- Analisar imagens e identificar elementos distintos na mesma;
- Criar máscaras para os elementos identificados nas imagens;
- Aplicar o *blending* de imagens;
- Permitir aos utilizadores criar imagens para usar em conversas e redes sociais.

4.2 PLANO E PROCESSO DE TRABALHO

O plano de trabalho delineado para cumprir com os objetivos anteriores foi dividido em cinco fases. Esta secção pretende enumerar e descrever essas diferentes fases do processo de trabalho.

- Fase 1:
  - Definição das regras do sistema;
  - Pesquisa bibliográfica;
  - Análise do estado da arte;
  - Recolha de técnicas de *blending* de imagens;
  - Recolha de técnicas de processamento e análise de imagens;

  - Recolha de técnicas de análise e manipulação de texto;
  - Seleção de algumas técnicas.
- Fase 2:
  - Testes com as técnicas escolhidas anteriormente;
  - Análise e comparação de resultados;
  - Escolha das técnicas a serem utilizadas nos diferentes módulos.
- Fase 3:
  - Desenvolvimento dos módulos definidos;
  - Análise e aprimoramento de resultados.
- Fase 4:
  - Junção dos módulos do sistema.
  - Implementação do sistema num *website*
- Fase 5:
  - Testagem e análise dos resultados

A primeira fase foi uma fase de recolha e investigação, cujo objetivo final foi selecionar as diferentes técnicas relativas a cada módulo a serem testadas na fase seguinte, e de definição das regras do sistema a ser desenvolvido. Para isso, foi feita uma pesquisa bibliográfica e foram recolhidos trabalhos relacionados com os conceitos que este projeto pretende explorar. Foram analisados trabalhos e artigos que abordam obtenção, análise, processamento e *blending* de imagens, como o trabalho "Vismantic: Meaning-making with images"(Xiao & Linkola, 2015), e a análise e manipulação de texto, como o trabalho "One does not simply produce funny memes!–explorations on the automatic generation of internet humor"(Oliveira et al., 2016). Desta análise bibliográfica foram recolhidos e testados diferentes bibliotecas e algoritmos.

O desenvolvimento deste projeto começou com a definição das regras que o sistema iria seguir para desenvolver cada imagem. Assim, foi definido que os *memes* resultantes estão restritos a um único formato, chamado Stonks. Neste formato, os *memes* são imagens de reação com uma pessoa e um fundo relacionado a uma ação descrita no topo da imagem, onde a personagem retratada é apresentada como estando injustificadamente orgulhosa da ação descrita (Adam & Don, 2020).

Estes *memes* possuem uma estrutura específica, sendo compostos por quatro elementos:

- Uma pessoa com a cabeça da personagem *Mememan*;[1]
- Uma imagem de fundo
- Uma frase a descrever uma ação ou contexto
- Uma palavra relacionada com a frase escrita

Exceptuando o posicionamento da descrição da ação, não há regras sobre a posição desses elementos na imagem, deixando essa escolha ao criador da imagem. A cabeça da personagem *Mememan* está presente em todos os *memes* do formato Stonks, sendo esta combinada com diferentes corpos, relacionados com assunto dos *memes*. Além disso, todos os outros elementos também devem ter uma ligação com o assunto da frase escrita. Esta estrutura específica pode ser traduzida num conjunto de regras que definem a criação dos *memes* pelo sistema, mais concretamente:

- O *meme* deve conter uma imagem de fundo
- O *meme* deve conter uma figura humana
- A figura humana deve ter a cabeça da personagem *Mememan*
- A cabeça da personagem deve estar orientada de acordo com a orientação da figura humana
- No topo do *meme* deve estar escrita a frase que descreve uma ação ou contexto
- O *meme* deve conter uma palavra relacionada com a frase escrita

Com as regras do sistema definidas, a exploração das bibliotecas e ferramentas ao dispor para o desenvolvimento deste trabalho começou pela procura de um método automático de obtenção de imagens, visto que estas seriam um material importante a ser usado na exploração de algoritmos de análise e *blending* de imagens. Desse modo, o início da investigação começou por bibliotecas que permitissem obter imagens de qualquer repositório ou *website*, existindo assim a flexibilidade de poder escolher a fonte de imagens com melhor qualidade. Para tal, foi usada a biblioteca BeautifulSoup,[2] que permite obter quaisquer dados presentes num ficheiro HTML ou XMS, guardando assim uma lista de todos os endereços relativos às imagens que compõem a página. Como passo final, tirando partido da biblioteca Urllib,[3] foi possível criar um ciclo que executa um pedido para

1 https://knowyourmeme.com/memes/meme-man
2 https://www.crummy.com/software/BeautifulSoup/
3 https://docs.python.org/3/library/urllib.html

guardar cada imagem presente na lista criada anteriormente. Este método, apesar das vantagens apresentadas, traz desvantagens significativas, como a fraca qualidade das imagens se o repositório utilizar miniaturas e a limitação do número de imagens, visto que muitos *websites* utilizam JavaScript e outras ferramentas para o carregamento dinâmico das páginas.

Estas desvantagens levaram à procura de outra abordagem. Neste caso, a biblioteca seguinte a ser usada foi a "Google-Images-Search",[4] que permite procurar e guardar imagens diretamente do repositório da Google, assim como realizar uma filtragem por imagens de diferentes tamanhos, cores e estilos. Para utilizar esta biblioteca foi necessário criar um projeto CX da Google, gerar uma chave de API para o projeto, e criar um motor de busca personalizado. Após a implementação desta biblioteca e a testagem da mesma, verificou-se que existia um número limite de pedidos que podia ser feito, fazendo assim esta biblioteca inviável para o uso do sistema.

Por fim, após a análise do trabalho "Vismantic: Meaning-making with images"(Xiao & Linkola, 2015), foi testada a abordagem utilizada pelo mesmo. Esta abordagem consiste na utilização da biblioteca FlickrAPI,[5] semelhante à descrita anteriormente mas sem limitações quanto ao número de pedidos e sem a necessidade de criar uma chave de API ou outras chaves semelhantes, visto que todas as chaves e informações necessárias para a implementação deste método estão disponíveis online.

Com o algoritmo de obtenção de imagens desenvolvido, o segundo conceito a ser explorado no processo do trabalho foi o *blending* de imagens, devido à sua importância no contexto do trabalho e à sua complexidade no que toca à implementação do mesmo. De acordo com o Estado da Arte e os projetos apresentados na secção Trabalhos Relacionados, o sistema "Gp-gan: Towards realistic high-resolution image blending"(Wu et al., 2019) apresenta os melhores resultados e aparenta ter uma implementação fácil por ter o código disponível publicamente, juntamente com instruções para a instalação e utilização. Contudo, ao implementar o código, foram encontrados múltiplos obstáculos, estando o mais relevante relacionado com a instalação de requisitos, onde foram obtidos inúmeros erros, e apesar de alguns destes terem sido solucionados, outros ainda persistiram. Uma das tentativas de resolução passou pela utilização do Google Colab,[6] que permite correr o código numa máquina da Google com muitos dos requisitos pedidos já instalados, porém continuaram a ser devolvidos os mesmos erros. Desta forma, e não encontrando outras soluções para resolver o problema, este sistema acabou por não ser usado no trabalho.

4 https://github.com/arrrlo/Google-Images-Search
5 https://stuvel.eu/software/flickrapi/
6 https://colab.research.google.com/

Foram então implementados outros algoritmos apresentados no Estado da Arte, nomeadamente o Poisson Blending e o Gaussian Laplacian Blending, como descrito na Secção 5. Estes algoritmos produzem resultados diferentes e foram ambos incluídos no sistema para a realização de testes, onde se irá verificar o método de preferência dos utilizadores.

Relativamente à análise de imagens, o método inicialmente implementado foi a deteção de faces segundo o algoritmo Viola-Jones (Viola & Jones, 2001), mas este método por vezes devolvia falsos positivos. Assim, foi utilizada a biblioteca ImageAI[7] para detetar a presença de pessoas na imagem. Esta abordagem acabou por aumentar o tempo necessário para analisar todas as imagens obtidas, por isso ambos os métodos acabaram por ser usados. Primeiro é realizada uma filtragem mais rápida e menos rigorosa, utilizando o algoritmo Viola-Jones, e depois é utilizada a biblioteca ImageAI para detetar pessoas nas imagens com resultado positivo do método anterior. Esta utilização dos dois métodos permitiu melhorar a eficiência da análise de imagens por parte do sistema a ser desenvolvido.

Para a segmentação das imagens, a primeira técnica testada para obter um elemento da imagem isolado consistia na deteção de pessoas utilizando a biblioteca ImageAI e na remoção do fundo com algoritmo Grabcut[8] na região detetada pela biblioteca. Este método acabou por não ser utilizado por obter resultados inconsistentes e não desejados, como os da Figura 4.1, apesar de devolver alguns resultados aceitáveis, como as Figuras 4.2 e 4.3.

a) Imagem original

b) Deteção feita por ImageAI

c) Remoção de fundo através do algoritmo Grabcut

Figura 4.1: Exemplo de segmentação

7 http://www.imageai.org/
8 https://docs.opencv.org/3.4/d8/d83/tutorial_py_grabcut.html

a) Imagem original

b) Deteção feita por ImageAI

c) Remoção de fundo através do algoritmo Grabcut

Figura 4.2: Exemplo de segmentação

a) Imagem original

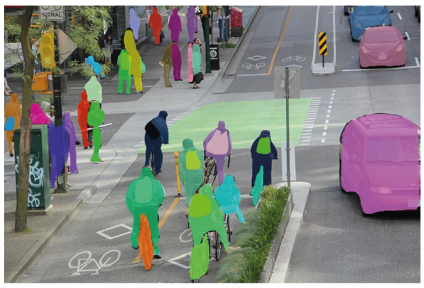
b) Deteção feita por ImageAI

c) Remoção de fundo através do algoritmo Grabcut

Figura 4.3: Exemplo de segmentação

Foi testada ainda a biblioteca PixelLib[9], que à semelhança da ImageAI utiliza redes neuronais para a identificação de elementos na imagem. A principal diferença reside na possibilidade de realizar a segmentação de imagens utilizando a biblioteca PixelLib, algo que não é possível utilizando ImageAI. Como observado na Figura 4.4, apesar do resultado apresentado no website da biblioteca ser igual ao da Figura 4.4b, o resultado obtido está apresentado na Figura 4.4c. Este erro verificou-se em todas as imagens e em todas as diferentes redes neuronais que a biblioteca pode utilizar.

a) Imagem original



b) Resultado apresentado

c) Resultado obtido

Figura 4.4: Exemplo de segmentação

Desse modo foi implementado o método apresentado no Capítulo 5, que utiliza a biblioteca OpenCV e a rede neuronal Mask RCNN, para realizar a segmentação de elementos da imagem.

Por fim as últimas ferramentas a explorar foram relativas à análise e manipulação de texto. Para começar, foi necessário encontrar uma forma de separar a frase escrita em expressões-chave e eliminar palavras redundantes, como preposições, pronomes e conjunções coordenativas. Para isso foi utilizada a biblioteca RAKE, descrita no Capítulo 5. Com a divisão da frase em expressões-chave, foi explorada a biblioteca PyDictionary[10] que permite obter significados e sinónimos de palavras. Apesar dessas funções, não era possível obter o significado de acordo com o contexto da frase, obtendo muitas vezes erros nas palavras obtidas. Para substituir esta biblioteca foi testada a biblioteca NLTK, que acabou por ser utilizada no sistema final e por isso está explicada no Capítulo 5.

9 https://pixellib.readthedocs.io/en/latest/index.html
10 https://github.com/geekpradd/PyDictionary

Durante a segunda fase foram realizados mais testes para averiguar se as ferramentas escolhidas anteriormente seriam adequadas para o desenvolvimento dos módulos propostos na fase seguinte. Como já tinham sido realizados alguns testes na fase anterior, houve um maior foco na testagem das ferramentas de análise e manipulação de texto, mais concretamente com o objetivo de verificar se era possível extrair uma palavra relacionada com a frase introduzida. O desenvolvimento destes testes foi demorado, visto que foi necessário criar uma função capaz de escolher palavras relevantes na frase que pudessem resumir o tema da mesma, e outra para obter palavras relacionadas com as obtidas pela função anterior. Durante esta parte do trabalho foi necessário explorar as várias funções disponibilizadas pela biblioteca e conjugá-las da melhor forma para obter os melhores resultados, visto que não existe um critério exato, que resulte para todas as frases, para extrair uma palavra capaz de resumir uma frase.

Nesta fase também se verificou a necessidade de implementar um algoritmo que devolvesse a orientação da figura humana presente na imagem para implementar nos módulos propostos na fase seguinte. Para isso foram analisadas os resultados obtidos pela função de obtenção de imagens, pelo algoritmo de deteção de faces Viola-Jones, e pela biblioteca ImageAI, e foi procurada uma relação entre a posição dos corpos detetados pela biblioteca ImageAI e a posição das faces detetadas pelo algoritmo Viola-Jones. Desta análise, e de acordo com as imagens obtidas até ao momento, foi concluído que quando a face está mais à direita do corpo significa que a cabeça da pessoa está orientada para o lado esquerdo, e vice-versa, como pode ser observado no exemplo da Figura 4.5.

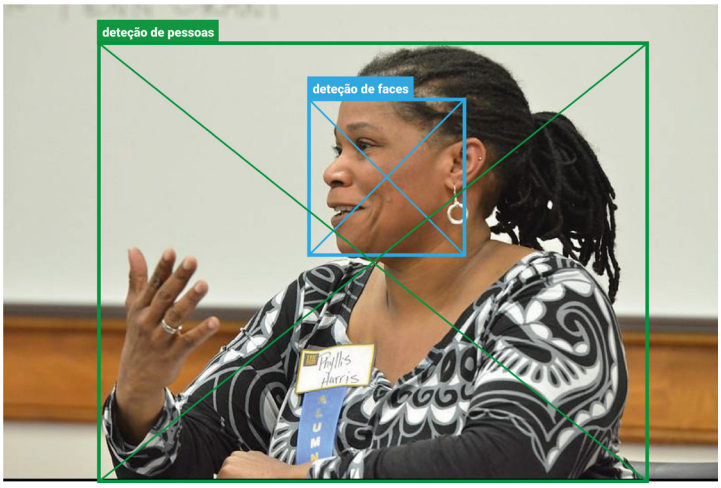


Figura 4.5: Exemplo de relação entre as posições das deteções da pessoa e da face

Na terceira fase foram desenvolvidos diferentes módulos individualmente. Criaram-se 4 módulos:

- Módulo 1: analisa o input e devolve uma lista de palavras relacionadas.
- Módulo 2: recolhe imagens relacionadas com o *input*.
- Módulo 3: realiza uma análise de características visuais e filtra e divide as imagens recolhidas. Também seleciona as imagens a usar, altera-as para realizar o *blending*, e cria uma máscara de recorte para os elementos a serem extraídos.
- Módulo 4: executa o *blending* e cola os elementos que compõem a imagem final, mais concretamente a cabeça do *Mememan*, uma palavra obtida e a frase de input.

Esta fase beneficiou especialmente da metodologia DSR que permitiu ir resolvendo problemas que foram surgindo ao longo do desenvolvimento dos módulos. À medida que os módulos iam sendo criados, e por sua vez iam sendo feitos mais testes, verificou-se que por vezes as imagens obtidas pelo segundo módulo eram muito restritas a um elemento da frase. Este comportamento devia-se ao facto de apenas serem obtidas palavras relacionadas com a palavra extraída pelo primeiro módulo. Então a função do segundo módulo foi chamada também para fazer *download* de imagens para a palavra com maior relevância selecionada pelo primeiro módulo. Mais tarde passaram também a ser obtidas imagens para todos nomes presentes na frase introduzida, para aumentar a variedade de resultados.

Foi também introduzida outra biblioteca, chamada Mediapipe, para obter a orientação das cabeças das pessoas nas imagens, visto que ao

realizar mais testes com o método anterior se verificou que por vezes o resultado obtido não estava correto.

Durante a quarta fase foi desenvolvido o sistema final, juntando os diferentes módulos e implementando-o num *website*. Esta fase foi importante para permitir o acesso ao sistema por parte de outros utilizadores, permitindo também a utilização de qualquer dispositivo com um *web browser*, e está explicada no Capítulo 5.

Por fim, a quinta fase, descrita no Capítulo 6, foi a fase de testes finais com utilizadores, onde se obteve *feedback* sobre diferentes partes do sistema. Durante esta fase do trabalho também foi escrito um artigo que foi submetido na conferência EAI ArtsIT 2022.[11]

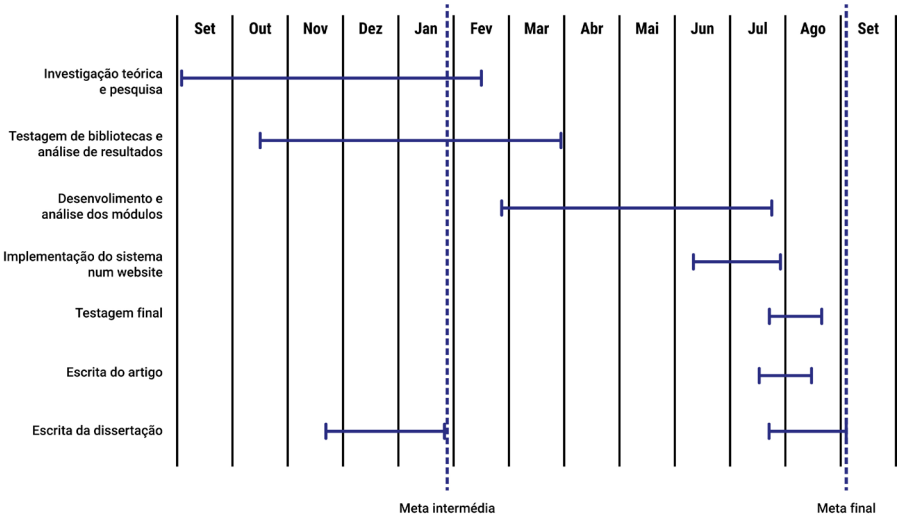


Figura 4.6: Diagrama de Gantt

11 https://artsit.eai-conferences.org/2022/

5

STONKINATOR: GERADOR DE MEMES

5.1 SISTEMA FINAL

O sistema apresentado nesta dissertação, o Stonkinator, é um sistema capaz de gerar *memes* a partir de um texto escrito pelo utilizador. Este capítulo apresenta o fluxo do sistema através dos seguintes módulos (Figura 5.1), que foram implementados em Python e utilizam diversas bibliotecas para manipulação de texto e obtenção, análise e *blending* de imagens :

- Módulo 1: Manipulador de texto
- Módulo 2: Obtenção de imagens
- Módulo 3: Análise de imagens
- Módulo 4: Mistura de imagens

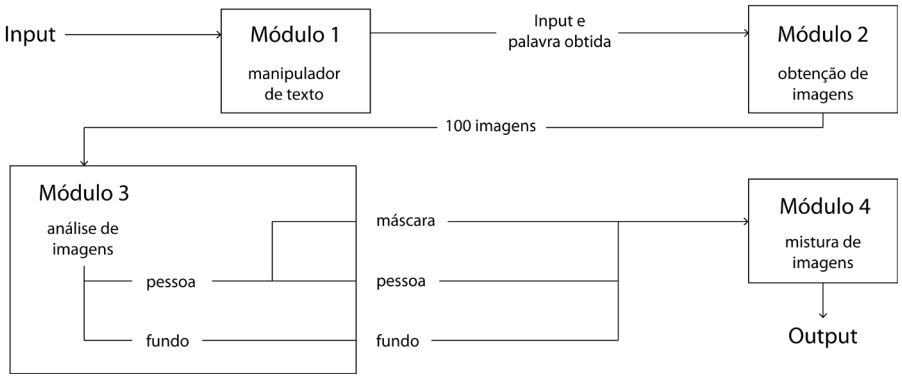


Figura 5.1: *Framework* do Stonkinator

Em primeiro lugar, este capítulo explicará o tratamento de texto por parte do primeiro módulo. Em seguida, são descritos o segundo e terceiro módulos, explicando como funcionam e as ferramentas utilizadas para obter e analisar as imagens. Por fim, é apresentado o processo por detrás do último módulo, responsável pela mistura e composição do produto final, e é explicada a implementação do sistema num *website*.

5.1.1 *Manipulador de Texto*

O primeiro módulo recebe como input uma frase escrita pelo utilizador, em inglês, e foca-se em obter uma palavra relacionada com o assunto da frase. Para isso, o sistema começa por utilizar a biblioteca

NLTK, juntamente com o Wordnet[1] e o NLTK Corpus[2], para classificar as palavras da frase de acordo com o seu POS[3] (part-of-speech). É também utilizada a biblioteca "python-rake"[4] (ou apenas RAKE) para dividir a frase em palavras e expressões e atribuir uma pontuação de acordo com o relevo da expressão, como podemos ver no exemplo da Tabela 5.1. De seguida é selecionada a palavra ou expressão com pontuação mais elevada e, no caso de ser escolhida uma expressão com mais do que uma palavra, o sistema prioriza nomes, advérbios, adjetivos e verbos, por esta ordem. Com esta palavra extraída, o algoritmo corre diferentes funções NLTK para criar uma lista de hiperónimos e sinónimos da palavra. Para isso é obtido o significado da palavra extraída de acordo com o contexto da frase introduzida, e é feita a procura de hiperónimos e sinónimos segundo esse significado obtido. Por último, é comparada a similaridade de significado entre a palavra extraída da frase e a lista de sinónimos e hiperónimos, de modo a ordenar a lista das palavras mais semelhantes às menos semelhantes.

Tabela 5.1: Exemplo de *output* da biblioteca RAKE

Palavras-chave	Pontuação
put	1.0
bowl	1.0
uncooked rice	4.0
soaked mobile phone	9.0

(a) Palavras-chave e pontuações para a frase "When I put a soaked mobile phone into a bowl of uncooked rice"

Palavras-chave	Pontuação
put oregano	4.0
microwaved frozen pizza	9.0

(b) Palavras-chave e pontuações para a frase "When you put oregano on a microwaved frozen pizza"

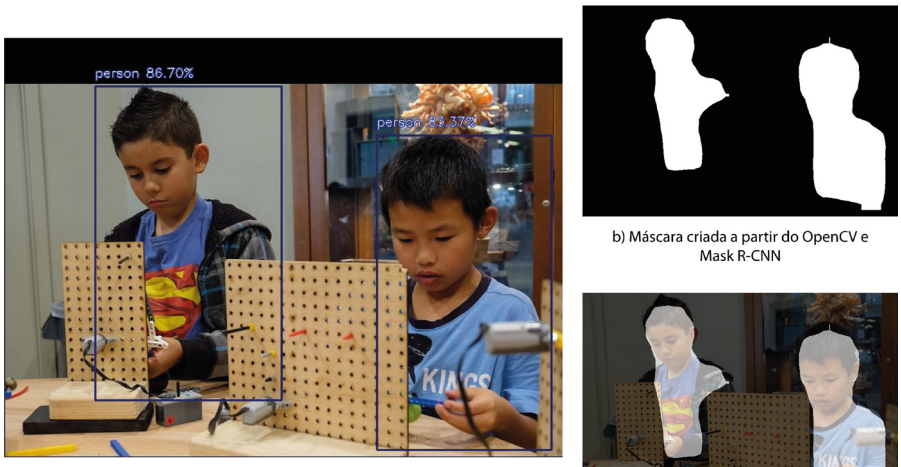
5.1.2 *Obtenção e Análise de Imagens*

O segundo módulo recebe a primeira palavra da lista criada pelo módulo anterior. De seguida, o sistema utiliza o FlickrAPI[5] para pes-

1 https://wordnet.princeton.edu/download
2 https://www.nltk.org/api/nltk.corpus.html
3 https://www.nltk.org/api/nltk.tag.html
4 https://github.com/aneesha/RAKE
5 https://stuvel.eu/software/flickrapi/

quisar imagens na base de dados Flickr e o Urllib[6] para as guardar numa pasta temporária. O número de imagens a serem guardadas são 100 por cada pedido.

Depois a primeira parte do terceiro módulo analisa as imagens da pasta temporária e identifica caras e figuras humanas. Primeiro, o sistema tira partido da função HoG Detector da biblioteca Dlib[7] para identificar faces nas imagens, e sempre que o resultado é positivo coloca a imagem numa pasta "faces", dentro da pasta temporária. Se o resultado da análise for negativo coloca a imagem na pasta "background", para ser usada como fundo. Depois, utilizando a biblioteca ImageAI[8] (Moses & Olafenwa, 18) e um modelo pré-treinado no *dataset* COCO, o sistema procura por pessoas nas imagens da pasta "faces", obtendo um retângulo relativo à posição da pessoa detetada e uma percentagem com uma probabilidade de certeza (Figura 5.2a). Por último, o sistema recorre à função de *face mesh* da biblioteca Mediapipe[9] para detetar a orientação das caras detetadas nas imagens da pasta "faces", com o objetivo de identificar a orientação correta da cabeça do *Mememan* a colocar na segunda parte do terceiro módulo. Enquanto a análise destas imagens é feita o sistema apaga imagens da pasta "faces"onde não sejam detetadas pessoas ou onde a colocação da cabeça por parte do sistema revele um fraco resultado devido à orientação da pessoa.[10]



a) Deteção utilizando a biblioteca ImageAI

b) Máscara criada a partir do OpenCV e Mask R-CNN

c) Sobreposição da máscara sobre a imagem

Figura 5.2: *Output* de diferentes bibliotecas para a seleção e segmentação de imagens

Apesar deste processo utilizar muitas ferramentas para selecionar as imagens finais, este método produz um aumento na eficiência do

6 https://docs.python.org/3/library/urllib.html
7 http://dlib.net/python/index.html
8 http://www.imageai.org/
9 https://mediapipe.dev/
10 por exemplo, se a pessoa estiver a olhar muito para cima ou para baixo

sistema comparando com a utilização exclusiva da biblioteca ImageAI para identificar pessoas, e reduz a margem de erro na análise de imagens, ao realizar uma maior filtragem com um algoritmo mais rápido, neste caso o algoritmo Viola-Jones, mesmo que este devolva falsos positivos, e utilizar um método mais complexo e mais lento como a biblioteca ImageAI, mas com resultados mais precisos, num conjunto de imagens mais pequeno. Para além disso, ao apagar as imagens que não serão utilizadas durante a primeira análise significa que todas as imagens contidas na pasta "faces"são imagens que podem ser usadas para criar o *meme* nas fases seguintes do algoritmo, ou seja, se o utilizador quiser criar uma nova imagem com o mesmo input o sistema vai ser mais rápido que antes, visto que já não precisa de analisar centenas de imagens e apenas tem de escolher uma de cada pasta ("background"e "faces").

Depois da separação e filtragem de imagens, se a pasta "background"ou "faces"estiver vazia, o sistema faz download de mais uma centena de imagens para a palavra seguinte na lista de palavras devolvida pelo primeiro módulo e volta a correr a primeira parte do terceiro módulo. Este ciclo continua até existirem imagens nas duas pastas. Relativamente à seleção das imagens a usar para gerar o *meme*, inicialmente o sistema escolhe uma imagem aleatória da pasta "background"e a imagem com maior probabilidade de conter uma figura humana da pasta "faces", mas se o utilizador escolher criar uma nova imagem com o mesmo input o sistema seleciona uma imagem aleatória de cada pasta para gerar uma imagem diferente.

5.1.3 *Mistura de Imagens*

Com as imagens a ser usadas escolhidas, a segunda parte do terceiro módulo cria uma *máscara* (Figura 5.2b e Figura 5.2c) para a figura humana, através da biblioteca OpenCV[11] e da rede neuronal Mask R-CNN[12], e, utilizando as bibliotecas OpenCV e Python Image Library,[13] altera o tamanho das imagens *máscara* e *pessoa* para corresponder ao tamanho da imagem *background*. Esta alteração é importante porque os algoritmos usados devolvem os melhores resultados com imagens do mesmo tamanho, e, para evitar a deformação do conteúdo da *máscara* e *pessoa*, a alteração de tamanho é feita colando essas imagens numa imagem preta do mesmo tamanho que o *background*.

Com este passo concluído, o quarto módulo recebe o *background* e as imagens *máscara* e *pessoa* com os novos tamanhos, assim como a frase usada como *input* e a primeira palavra da lista devolvida pelo primeiro módulo. De seguida, o sistema executa a mistura da imagem *pessoa* no *background*, através da máscara criada com o módulo

11 https://github.com/opencv/opencv-python
12 https://github.com/matterport/Mask_RCNN
13 https://pillow.readthedocs.io/en/stable/installation.html

anterior. Com o objetivo de testar o melhor método, o sistema devolve três imagens, cada uma com um método de mistura diferente. Para a primeira imagem é feita a mistura mais simples, apenas um recorte da pessoa e colagem no *background* (Figura 5.3a), utilizando o OpenCV. A segunda imagem é feita através do algoritmo Gaussian Laplacian Blending, utilizando cinco níveis (Figura 5.3b). Por fim, para a terceira imagem é aplicado o algoritmo Poisson Blending (rinsa318, 2019), utilizando o método "seamless cloning"(Figura 5.3c).

a) Colagem simples

b) Pirâmides Gaussianas Laplacianas

c) Poisson seamless cloning

Figura 5.3: *Outputs* de diferentes métodos de *blending* com imagens selecionadas pelo Módulo 3

Depois desta primeira mistura, a imagem obtida volta a ser analisada e são obtidas as posições da figura humana presente na imagem e da sua face, e é colada a cabeça da personagem *Mememan*, através das bibliotecas OpenCV e Python Image Library, no local da face da pessoa na imagem, sendo a direção da cabeça da personagem determinada utilizando a ferramenta "face mesh"da biblioteca Mediapipe. A deteção de ambas a figura humana e a face neste passo é importante para evitar a deteção de rostos inexistentes por parte do algoritmo usado. Por fim, a composição final é criada ao adicionar a frase introduzida no topo da imagem e ao colar a palavra obtida utilizando a ferramenta de deteção de MSER (Maximally Stable Extremal Regions) da biblioteca OpenCV (como visto na Figura 5.3a).

5.1.4 *Interface Web*

De modo a testar o sistema Stonkinator, foi criada uma interface *web* através da biblioteca Flask.[14] Esta página *web* serviu como prova de conceito para a realização dos testes, mas existe vontade de criar uma página melhorada para disponibilizar o sistema ao público. Devido a

14 `https://palletsprojects.com/p/flask/`

falhas relativas ao servidor disponibilizado, o *website* foi hospedado num computador privado e foi configurado o redirecionamento de portas no *router* para que se pudessem ligar dispositivos fora da rede local. A página Flask desenvolvida apresenta no centro uma barra de escrita que permite ao utilizador escrever a sua frase ou expressão. Depois, essa frase é enviada para o *script* que contém a ferramenta desenvolvida para ser utilizada como *input* do sistema, e começa o processo descrito acima. Enquanto o utilizador espera pelo resultado final, é apresentado um *slideshow* com as imagens geradas por outros utilizadores.

No final do processo, a página produz um som que indica que o resultado foi entregue e o utilizador é apresentado com três imagens, uma para cada método de *blending* e três botões que executam diferentes funções. O primeiro botão redireciona o utilizador para a página inicial para este poder criar uma nova imagem com um novo input. O segundo botão produz uma nova imagem com o mesmo input, selecionando uma imagem aleatória de cada pasta temporária. Por fim, o terceiro botão guarda o *meme* gerado no sistema para poder ser visto por outros utilizadores na galeria ou no *slideshow* da página inicial.

6

AVALIAÇÃO

Nesta secção são apresentados e discutidos os resultados das experiências com utilizadores.

Foi realizado um estudo com utilizadores para avaliar a qualidade do sistema no que toca à qualidade técnica do *output*,[1] previsibilidade dos resultados, usabilidade, utilidade, e para perceber as preferências dos utilizadores em relação aos métodos de *blending*. Os principais objetivos foram:

- Compreender se o resultado obtido seria aceite pelos utilizadores;
- Compreender se o output era previsível ou estava de acordo com a ideia que queriam transmitir;
- Perceber o quão fácil é utilizar o sistema implementado;
- Averiguar se os utilizadores utilizariam o resultado nas suas redes sociais e conversas de texto informais;
- Identificar quais os métodos de *blending* de preferência;
- Obter *feedback* em relação aos *memes* obtidos.

Para obter esses dados, foram apresentadas algumas tarefas aos participantes e estes foram solicitados a responder a algumas perguntas relacionadas às tarefas que realizaram e aos resultados obtidos. O processo de teste começou com o fornecimento do acesso ao *website* aos participantes e pedindo-lhes para criar um *meme* introduzindo uma frase como *input* (por exemplo: "I like to eat marmalade with cheese"). Depois disso, o website apresentou aos utilizadores um conjunto de três imagens, cada imagem misturada com um método diferente, e em seguida, a segunda tarefa era guardar esse conjunto no sistema caso achassem o *meme* interessante. Concluída esta etapa, a terceira tarefa pedia que os participantes gerassem outro *meme* utilizando a mesma entrada, e mais uma vez, a tarefa seguinte era guardar o conjunto gerado, caso quisessem. No final, foi-lhes dada a opção de responder a um questionário ou continuar a usar o sistema livremente e responder posteriormente. Antes de responder a qualquer pergunta, os participantes foram solicitados a escolher e abrir uma janela com um conjunto de imagens geradas por eles para ser usado posteriormente durante as respostas. O questionário foi elabo-

1 Análise de imagem, segmentação de imagem, mistura de imagens e palavra obtida

rado utilizando o Google Forms[2] para estudar os tópicos já citados e foi composto por 11 tarefas:

- T1: Avaliar a qualidade técnica da saída entre 1 (má qualidade) e 10 (boa qualidade)
- T2: Avaliar a previsibilidade da saída entre 1 (imprevisível) e 10 (previsível)
- T3: Avaliar a usabilidade do sistema entre 1 (confuso) e 10 (intuitivo)
- T4: Descrever o *meme* gerado (esta foi uma pergunta aberta, embora alguns exemplos tenham sido dados aos participantes, por exemplo: engraçado, aborrecido, sem sentido, ...)
- T5: Partilharia o *meme* entre amigos/nas redes sociais
- T6: Utilizaria o *meme* durante uma conversa de texto informal

Para as últimas quatro perguntas, os participantes tiveram que analisar diferentes conjuntos de imagens e identificar o método de *blending* preferido. Para isso, foram escolhidos três *memes* gerados (sendo cada um deles um conjunto de imagens produzidas a partir dos três métodos de *blending*) e pedimos aos participantes para realizarem a seguinte tarefa para cada *meme* (T7-9): Escolher o método de *blending* de preferência entre o conjunto de imagens apresentadas.

Em seguida, a mesma tarefa foi solicitada para o conjunto de imagens escolhido pelo participante durante o teste do sistema (T10). A última tarefa (T11) dizia respeito a uma questão aberta opcional, que pedia comentários e *feedback* sobre o teste e o sistema.

6.1 TESTES COM UTILIZADORES

No total, o Stonkinator foi testado por 23 participantes com idades entre 17 e 25 anos. Para melhor mostrar os resultados, na Tabela 2a, dividimos as respostas em três grupos: "mau"(resposta inferior a 6 em 10); "ok"(resposta de 6 ou 7 em 10); "bom"(resposta superior a 7 em 10). De modo geral, a análise dos resultados dos testes indica que, embora o valor médio das avaliações quanto à qualidade técnica das imagens obtidas seja geralmente bom, o mesmo não pode ser dito sobre a previsibilidade dos *memes* gerados, conforme a Tabela 6.2a. Quanto à usabilidade, o resultado é geralmente bom, com apenas duas pessoas insatisfeitas com o discreto *feedback* obtido ao guardar as imagens. Apenas dois utilizadores relataram que não partilhariam o resultado obtido nas redes sociais, mas cem por cento dos inquiridos usariam os *memes* obtido em conversas de texto informais (Tabela

2 Os participantes foram guiados ao longo das perguntas apresentadas para evitar má interpretação

6.2b). Por fim, em relação aos métodos de *blending*, de acordo com a Tabela 6.2c, podemos verificar que o método com resultados mais atrativos para os utilizadores é o de colagem simples.

Tabela 6.1: Resultados dos testes

Avaliação	mau	ok	bom
Qualidade Técnica	4.35%	30.43%	65.22%
Previsibilidade	68.57%	21.74%	8.69%
Usabilidade	0%	34.78%	65.22%

(a) Resultados da avaliação de diferentes parâmetros

Avaliação	sim	não
Uso em conversas privadas	100%	0%
Partilha nas redes sociais	91.3%	8.7%

(b) Utilização dos resultados

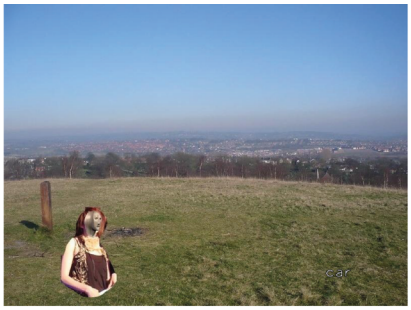
Método	Percentagem
Colagem	69.56%
Laplacian	13.04%
Poisson	17.4%

(c) Preferência no método de *blending*

6.2 DISCUSSÃO GERAL

Uma das maiores críticas em relação à qualidade técnica dos *memes* gerados está relacionada com a segmentação de imagens, onde por vezes não é conseguida a deteção das bordas das figuras humanas na imagem com precisão, sendo, portanto, um ponto de partida para melhorar o sistema. Um fator que pode ser favorável a essas falhas está relacionado à existência de *memes* cujo objetivo é fazer com que essas mesmas falhas sejam exibidas intencionalmente na imagem, embora não seja esse o propósito do sistema. Esse facto também pode explicar o porquê de alguns utilizadores tenderem a preferir métodos de *blending* de mais simples em vez de métodos mais complexos, conforme a Tabela 2c, uma vez que existem vários *memes* são feitos com colagens simples de elementos numa imagem. Na Tabela 6.2a podemos verificar que a previsibilidade da imagem obtida é muito baixa. Como o objetivo do sistema é criar uma imagem capaz de transmitir uma ideia ou mostrar uma reação próxima à pretendida pelo utilizador, podemos concluir que, apesar de algumas vezes o sistema conseguir

gerar uma saída previsível, geralmente as imagens obtidas não são como o utilizador esperava. Alguns comentários no final do questionário referem que este resultado se deve ao facto de, por vezes, o utilizador não ter uma ideia pré-concebida do resultado e procurar uma mensagem adequada ao contexto nas imagens obtidas, mas outros reportam que de facto o resultado obtido estava pouco relacionado com a frase escrita no *input*. Este problema pode ser contornado desenvolvendo uma busca mais específica, utilizando várias palavras ou mesmo expressões presentes na entrada. O sistema também pode ser alterado para dar prioridade a imagens que contenham elementos que correspondam às palavras da entrada ao realizar a análise da imagem. No entanto, este problema não é o mais prioritário, pois, como visto na Tabela 6.2b, os participantes partilhariam o *meme* produzido numa rede social ou numa conversa de texto informal. Apesar das falhas apontadas pelos participantes, verificamos que a maioria deles conseguiu utilizar o sistema sem grande dificuldade, atingindo o objetivo do sistema ser simples e funcionar com "uma entrada", e que mais de noventa por cento do inquiridos partilharia o *meme* produzido numa rede social, sendo que cem por cento a utilizaria numa conversa de texto. Estes resultados demonstram que, mesmo que a saída final apresente falhas, o sistema pode ser utilizado para atingir os resultados para os quais foi proposto. Abaixo, na Figura 6.1, mostramos alguns memes produzidos pelo nosso sistema. Podemos ver duas imagens diferentes geradas com a mesma entrada nas Figuras 6.1a e 6.1b e outras duas imagens criadas com entradas diferentes nas Figuras 6.1c e 6.1d. Estes *memes* foram produzidos pelos participantes durante o período de testes.



a) Meme produzido para o input
"When you get early so you don't miss the bus"

b) Meme produzido para o input
"When you get early so you don't miss the bus"

c) Meme produzido para o input
"I like to eat marmalade with cheese"

d) Meme produzido para o input
"Me after checking 2+2 on the calculator"

Figura 6.1: Exemplos de *memes* gerados pelo sistema

7 CONCLUSÃO

Esta dissertação apresenta uma nova ferramenta, Stonkinator, capaz de criar *memes* num formato específico, "stonks", a partir de um único *input*. Apesar da existência de ferramentas que auxiliam a criação de *memes*, na sociedade cada vez mais ativa e global em que vivemos, faz sentido que exista uma forma ágil e descomplicada de gerar imagens que possam ser inseridas rapidamente numa conversa escrita ou rede social, de modo a transmitir uma ideia ou mensagem. Foram também definidos o conceito de "meme" e o formato de *memes* a serem gerados pelo sistema apresentado.

Nesta dissertação são apresentados trabalhos relacionados, que inspiraram e trouxeram ideias para o desenvolvimento deste trabalho, e é referido o estado da arte, onde são analisadas ferramentas e bibliotecas usadas durante a criação do sistema apresentado. É ainda abordada a metodologia usada, e descrito o plano e o processo de trabalho.

O sistema foi desenvolvido tirando partido de várias bibliotecas Python, incluindo métodos e algoritmos de análise de texto, análise de imagens, segmentação de imagens e mistura de imagens. O sistema foi implementado através de um website que apresenta os resultados utilizando três métodos de *blending* diferentes. Foi realizado um teste com utilizadores com o objetivo de compreender as preferências dos participantes em relação aos métodos de *blending* e a viabilidade do sistema.

Os resultados mostram que o sistema é capaz de criar imagens satisfatórias, sendo as mais atrativas visualmente as geradas com método de colagem simples, e que os utilizadores utilizariam os memes gerados em publicações nas redes sociais ou no contexto de uma conversa de texto privada. Existe a vontade de melhorar o programa proposto no futuro, implementando um melhor algoritmo de segmentação de imagens, além de criar um sistema de seleção de imagens melhorado, para que os resultados obtidos possam atender melhor às ideias que os utilizadores pretendem transmitir. Além disso, as palavras ou expressões usadas nos *memes* do Stonks são marcadas com um pequeno erro de escrita. Existe também o interesse em explorar esse recurso com o objetivo de melhorar a palavra escrita nos *memes* gerados.

Adam & Don (2020). Meme man wurds / stonks edits. https://knowyourmeme.com/memes/meme-man-wurds-stonks-edits. Acedido em: 23 de Agosto de 2022.

Adam & F, Y. (2019). Stonks. https://knowyourmeme.com/memes/stonks. Acedido em: 23 de Agosto de 2022.

Agarwal, K. (2020). Rake: Rapid automatic keyword extraction algorithm. https://medium.datadriveninvestor.com/rake-rapid-automatic-keyword-extraction-algorithm-f4ec17b2886c. Acedido em: 23 de Agosto de 2022.

Ahmad, R. (2019). A take on h.o.g feature descriptor. =https://medium.com/analytics-vidhya/a-take-on-h-o-g-feature-descriptor-e839ebba1e52. Acedido em: 23 de Agosto de 2022.

Anindya (2020). How to remove stop words in python. https://thinkinfi.com/how-to-remove-stop-words-in-python/. Acedido em: 23 de Agosto de 2022.

Bird, S., Loper, E., & Klein, E. (2009). *Natural Language Processing with Python*. O'Reilly Media Inc.

Birdsell, B. (2014). Fauconnier's theory of mental spaces and conceptual blending. In J. Littlemore & J. R. Taylor (Eds.), *The Bloomsbury Companion to Cognitive Linguistics* chapter 2.4, (pp. 72–90).

Creswell, A., White, T., Dumoulin, V., Arulkumaran, K., Sengupta, B., & Bharath, A. A. (2018). Generative adversarial networks: An overview. *IEEE Signal Processing Magazine*, *35*(1), 53–65.

Dalal, N. & Triggs, B. (2005). Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, (pp. 886–893). IEEE.

Dawkins, R. (1976). *The Selfish Gene*. Oxford University Press.

Drakos, N., Moore, R., & Wang, R. (2013). Gaussian-laplacian pyramid image coding. http://fourier.eng.hmc.edu/e161/lectures/canny/node3.html. Acedido através da WaybackMachine, cópia de 23 de Abril de 2021, em 23 de Agosto de 2022 [https://web.archive.org/web/20210423183202/http://fourier.eng.hmc.edu/e161/lectures/canny/node3.html].

Dyrdal, I. Image Processing Lecture 2.3 – Laplace blending [PowerPoint slides]. University of Oslo. https://www.uio.no/studier/e

mner/matnat/its/nedlagte-emner/UNIK4690/v17/forelesninger/lecture_2_3_blending.pdf.

Fauconnier, G. & Turner, M. (2002). The way we think: Conceptual blending and the mind's hidden complexities. Basic Books.

Gilks, W., Richardson, S., & Spiegelhalter, D. (1995). *Markov Chain Monte Carlo in Practice*. Chapman & Hall/CRC Interdisciplinary Statistics. CRC Press.

Gonsalves, R. A. (2021). Ai-memer: Using machine learning to create funny memes. https://towardsdatascience.com/ai-memer-using-machine-learning-to-create-funny-memes-12fc1fe543e4. Acedido em: 23 de Agosto de 2022.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative adversarial nets. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., & Weinberger, K. (Eds.), *Advances in Neural Information Processing Systems*, volume 27.

Gübür, K. (2021). Nltk and python wordnet: Find synonyms and antonyms with python. https://www.holisticseo.digital/python-seo/nltk/wordnet. Acedido em: 23 de Agosto de 2022.

He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask r-cnn. In *Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV)*, (pp. 2980–2988).

Hoehn, J. & Finkelstein, N. (2018). Students' flexible use of ontologies and the value of tentative reasoning: Examples of conceptual understanding in three canonical topics of quantum mechanics. *Physical Review Physics Education Research*, *14*.

Li, B., Zook, A., Davis, N., & Riedl, M. O. (2012). Goal-driven conceptual blending: A computational approach for creativity. In *Proceedings of the 2012 International Conference on Computational Creativity (ICCC 2012)*, (pp. 3–16).

Lugaresi, C., Tang, J., Nash, H., McClanahan, C., Uboweja, E., Hays, M., Zhang, F., Chang, C.-L., Yong, M., Lee, J., Chang, W.-T., Hua, W., Georg, M., & Grundmann, M. (2019). Mediapipe: A framework for perceiving and processing reality. In *Third Workshop on Computer Vision for AR/VR at IEEE Computer Vision and Pattern Recognition (CVPR) 2019*.

Maindola, G. (2021). Instance segmentation using mask-rcnn in opencv python. https://machinelearningknowledge.ai/instance-segmentation-using-mask-rcnn-in-opencv-python/. Acedido em: 23 de Agosto de 2022.

Mallick, S. (2016). Histogram of oriented gradients explained using opencv. https://learnopencv.com/histogram-of-oriented-gradients/. Acedido em: 23 de Agosto de 2022.

Moses & Olafenwa, J. (2018–). Imageai, an open source python library built to empower developers to build applications and systems with self-contained computer vision capabilities. https://github.com/OlafenwaMoses/ImageAI.

Odemakinde, E. (2022). Everything about mask r-cnn: A beginner's guide - viso.ai. https://viso.ai/deep-learning/mask-r-cnn/. Acedido em: 23 de Agosto de 2022.

Oliveira, H. G., Costa, D., & Pinto, A. M. (2016). One does not simply produce funny memes!–explorations on the automatic generation of internet humor. In *Proceedings of the Seventh International Conference on Computational Creativity (ICCC 2016)*.

Pereira, F. C. & Cardoso, A. (2002). The boat-house visual blending experience. In *Proceedings of the Symposium for Creativity in Arts and Science of AISB 2002*.

Pérez, P., Gangnet, M., & Blake, A. (2003). Poisson image editing. *ACM Transactions on Graphics*, *22*(3), 313–318.

Princeton University (2010). About wordnet. https://wordnet.princeton.edu/. Princeton University.

Radke, R. (2014). Cvfx lecture 6: Multiresolution blending and poisson image editing. https://youtu.be/g7zFl8EnbzI.

rinsa318 (2019). poisson-image-editing. https://github.com/rinsa318/poisson-image-editing. Acedido em: 23 de Agosto de 2022.

Rose, S., Engel, D., Cramer, N., & Cowley, W. (2010). Automatic keyword extraction from individual documents. In M. W. Berry & J. Kogan (Eds.), *Text Mining: Applications and Theory* chapter 1, (pp. 1–20).

Rother, C., Kolmogorov, V., & Blake, A. (2004). "grabcut"interactive foreground extraction using iterated graph cuts. *ACM transactions on graphics (TOG)*, *23*(3), 309–314.

Saxena, D. & Cao, J. (2021). Generative adversarial networks (gans): Challenges, solutions, and future directions. *ACM Computing Surveys*, *54*(3).

Saxena, N. (2020). Extracting keyphrases from text: Rake and gensim in python. https://towardsdatascience.com/extracting-keyphrases-from-text-rake-and-gensim-in-python-eefd0fad582f. Acedido em: 23 de Agosto de 2022.

Shifman, L. (2013). *Memes in Digital Culture*. The MIT Press.

Steinbrück, A. (2013). Conceptual blending for the visual domain. Bachelor thesis, University of Amsterdam.

Szeliski, R., Uyttendaele, M., & Steedly, D. (2011). Fast poisson blending using multi-splines. In *Proceedings of the 2011 IEEE International Conference on Computational Photography (ICCP)*, (pp. 1–8). IEEE.

Tyag, M. (2021). Hog(histogram of oriented gradients). `https://towardsdatascience.com/hog-histogram-of-oriented-gradients-67ecd887675f`. Acedido em: 23 de Agosto de 2022.

Venable, J. R., Pries-Heje, J., & Baskerville, R. L. (2017). Choosing a design science research methodology. In *Proceedings of the 28th Australasian Conference on Information Systems.*

Viola, P. & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer vision and Pattern Recognition (CVPR'01)*, volume 1, (pp. I–I). IEEE.

Wang, Y.-Q. (2014). An analysis of the viola-jones face detection algorithm. *Image Processing On Line*, *4*, 128–148.

Wu, H., Zheng, S., Zhang, J., & Huang, K. (2019). Gp-gan: Towards realistic high-resolution image blending. In *Proceedings of the 27th ACM international Conference on Multimedia*, (pp. 2487–2495).

Xiao, P. & Linkola, S. M. (2015). Vismantic: Meaning-making with images. In *Proceedings of the Sixth International Conference on Computational Creativity (ICCC 2015)*. Brigham Young University.

Zhang, L., Wen, T., & Shi, J. (2020). Deep image blending. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, (pp. 231–240).

Zhao, M. (2020). Image blending using laplacian pyramids. `https://becominghuman.ai/image-blending-using-laplacian-pyramids-2f8e9982077f`. Acedido em: 23 de Agosto de 2022.