Reconhecimento Automático de Tabelas em Documentos

DIANA SOFIA PICADO FONSECA
(Licenciada)

Relatório de Estágio para obtenção do grau de mestre em Matemática Aplicada para a Indústria na Área de Especialização de Tratamento de Dados

Orientadores:
Doutor José Alberto de Sousa Rodrigues
Doutor Luís Manuel Ferreira da Silva

Júri:
Presidente: Doutora Teresa Maria de Araújo Melo Quinteiro
Vogais:
Doutor Carlos José Brás Geraldes
Doutor José Alberto de Sousa Rodrigues

Setembro de 2024

Reconhecimento Automático de Tabelas em Documentos

DIANA SOFIA PICADO FONSECA
(Licenciada)

Relatório de Estágio para obtenção do grau de mestre em Matemática Aplicada para a Indústria, na Área de Especialização de Tratamento de Dados

Orientadores:
Doutor José Alberto de Sousa Rodrigues, ISEL
Doutor Luís Manuel Ferreira da Silva, ISEL

Júri:
Presidente: Doutora Teresa Maria de Araújo Melo Quinteiro, ISEL

Vogais:
Doutor Carlos José Brás Geraldes, ISEL
Doutor José Alberto de Sousa Rodrigues, ISEL

Setembro de 2024

Agradecimentos

O Estágio foi realizado através de um Protocolo de cooperação com o ISEL, na Closer sob a supervisão local da Engenheira Ana Patrícia Afonso.

Antes de mais, gostaria de expressar a minha gratidão aos meus pais, irmãos, namorado, aos meus gatos e a toda a minha família e amigas pelo apoio incondicional e encorajamento ao longo de todo o meu percurso académico. O facto de não duvidarem das minhas capacidades e persistência foi a maior fonte de motivação e de força que poderia ter tido. Não teria conseguido chegar tão longe sem o vosso amor, apoio, confiança e motivação.

Agradeço a todos os professores que tive a oportunidade de conhecer ao ser aluna do Instituto Superior de Engenharia de Lisboa (ISEL), por me terem proporcionado a oportunidade de adquirir competências nos conteúdos do meu maior interesse e por toda a aprendizagem que pude adquirir, ao longo de todo este percurso. A dedicação direcionada para a aprendizagem de todos alunos foi muito importante para o meu desenvolvimento académico e profissional.

Um agradecimento especial ao Professor Doutor Luís Manuel Ferreira da Silva e ao Professor Doutor José Alberto de Sousa Rodrigues por toda a assistência, orientação e apoio prestados ao longo de todo o meu Mestrado.

Agradeço profundamente à Closer, mais especificamente aos membros da Equipa da Evalyze, que me proporcionaram uma experiência muito dinâmica, onde me foi prestado um apoio incansável que garantiu que a minha experiência fosse enriquecedora. Obrigada a todos, especialmente à Engenheira Ana Patrícia Afonso, por toda a orientação e apoio prestados.

O apoio de todos os meus colegas de turma pela entreajuda e pela partilha de ideias e de momentos tornaram esta experiência muito agradável.

Estou profundamente grata por toda esta oportunidade que contribuiu para a minha aquisição de conhecimentos e para o meu sucesso. A confiança no meu trabalho, o encorajamento e a discussão de ideias tornaram a realização deste trabalho de Mestrado possível.

Declaração de integridade

Declaro que esta(e) dissertação / trabalho de projeto / relatório de estágio é o resultado da minha investigação pessoal e independente. O seu conteúdo é original e todas as fontes listadas nas referências bibliográficas foram consultadas e estão devidamente mencionadas no texto. Mais declaro que todas as referências científicas e técnicas relevantes para o desenvolvimento do trabalho estão devidamente citadas e constam das referências bibliográficas.

O autor

Lisboa, 31 de outubro de 2024

Resumo

A extração de dados em formato digital tem vindo a ser uma necessidade comum a muitas empresas e é necessária para o funcionamento adequado das operações em muitos setores, devido ao aumento significativo do volume de dados em formato digital, verificado nos últimos anos.

Os métodos manuais de inserção de informações de dados tabulares resultam em erros frequentes e, muitas vezes, a rapidez com que a tarefa manual da extração dos dados tabulares é realizada não permite atender aos requisitos da maior parte das empresas. Com isto, a Closer teve a necessidade de recorrer a um *software* da Microsoft, o Azure Forms Recognizer.

O *software* tem custos elevados, para a escala em que é necessária a utilização deste por parte dos clientes e é ainda necessária a implementação de restrições que permitem agrupar os dados da tabela que são erradamente separados e para que seja possível extrair os valores de que os clientes normalmente precisam. Esta tese tem como objetivo o desenvolvimento de uma base de implementação de um algoritmo de inteligência artificial (IA) que supere a adversidade dos custos elevados, onde será realizada a deteção, o reconhecimento da estrutura e a extração das informações para um documento estruturado.

Esta metodologia explora a utilização de modelos de deteção de objetos, nomeadamente, o modelo YOLOv8, tanto na tarefa de deteção das tabelas como no reconhecimento de estruturas que auxiliam na obtenção das células das tabelas. Este modelo é reconhecido por realizar deteções com alta precisão e por ser muito rápido no processamento de grandes volumes de imagens.

Para a implementação da metodologia, foram realizados os treinos de vários modelos YOLO, com um conjunto de imagens que contêm tabelas de diferentes fornecedores e foram aplicadas restrições aos resultados das redes neuronais para que fosse possível atender aos requisitos.

Palavras chave

YOLOv8; OCR; Inteligência Artificial; Extração Automática de Dados; Imagens de Tabelas

Abstract

The extraction of data in digital format has become a common necessity for many companies and is essential for the proper functioning of operations across various sectors, due to the significant increase in the volume of digital data observed in recent years.

Manual methods for inputting tabular data information result in frequent errors, and often the speed at which the manual task of extracting tabular data is performed fails to meet the requirements of most companies. Consequently, Closer had to resort to Microsoft's software, Azure Forms Recognizer.

The software incurs high costs for the scale at which it needs to be used by clients, and additional constraints must be implemented to group table data that is erroneously separated, allowing for the extraction of the values typically required by clients. This thesis aims to develop an implementation base for an artificial intelligence (AI) algorithm that overcomes the challenges of high costs, where the detection, structure recognition, and extraction of information into a structured document will be carried out.

This methodology explores the use of object detection models, particularly the YOLOv8 model, both in the task of table detection and in recognizing structures that assist in obtaining the table cells. This model is known for achieving high precision in detections and for its speed in processing large volumes of images.

For the implementation of the methodology, several YOLO models were trained using a set of images containing tables from different suppliers, and constraints were applied to the results of the neural networks to meet the necessary requirements.

Key words

YOLOv8; OCR; Artificial Inteligence; Automatic Data Extraction; Table Images

Índice

Índice de figuras

Simbologia e abreviaturas

Símbologia

Latinas

$E(X)$	Valor esperado (ou valor médio) da variável X
f_c	Resistência à compressão do betão
b	viés
w	peso
y'	Previsão da rede neuronal
k	Filtro de convolução
s	Passo
p	Padding
mc	Número máximo de canais
n	Número de blocos de gargalo
P_1	Ponto original
x_1	coordenada de x original do ponto P_1
y_1	coordenada y original do ponto P_1
x_c	coordenada de x do centro de rotação
y_c	coordenada de y do centro de rotação
x_{novo}	Coordenada x do ponto após a translação para que o centro da rotação coincida com a origem
y_{novo}	Coordenada y do ponto após a translação para que o centro da rotação coincida com a origem
$x_{1rodado}$	Coordenada x do ponto P_1 após a rotação
$y_{1rodado}$	Coordenada y do ponto P_1 após a rotação

Gregas

η	Taxa de Aprendizagem
σ	Função de Ativação Sigmoide
ϵ	Constante pequena em algoritmos de otimização
μ_{lote}	Média do Lote de normalização
σ^2_{Lote}	Variância do Lote de normalização
α	valor de ângulo em graus da inclinação retornado pelo OCR
$\alpha_{(radianos)}$	valor de ângulo em radianos da inclinação retornado pelo OCR

Abreviaturas

PDF	Formato de Documento Portátil
RobusTabNet	Rede Robusta de Deteção de Tabelas e Reconhecimento de Estrutura
Faster R-CNN	Rede Neuronal Convolucional para Deteção de Objetos mais Rápida
TAO	Organização de Tabelas
ROI	Região de Interesse
MLP	Perceptron Multicamada
mAP50	Média de Precisão a 50% de Interseção sobre a União
ResNet	Rede Neuronal Residual
Darknet	Plataforma de redes neuronais com uso no desenvolvimento dos modelos YOLO
ipynb	IPython Notebook
YAML	"YAML Ain't Markup Language"(YAML Não É Uma Linguagem de Marcação)
CornerNet	Rede Neuronal Convolucional para deteção de objetos ao identificar os cantos
GPU	*Graphics Processing Unit* (Unidade de Processamento Gráfico)
GB	Gigabyte (Unidade de Armazenamento de Dados)
JSON	JavaScript Object Notation (Formato de Troca de Dados)
JPG/JPEG	Joint Photographic Experts Group (Formato de Imagem)
IoU	Interseção sobre a União
SiLU	Sigmoid Linear Unit (Função de Ativação de Unidade Linear Sigmoide)
OCR	Optical Character Recognition (Reconhecimento Óptico de Caracteres)
NMS	Non-Maximum Suppression (Supressão de Máximos Não-Máximos)
FPN	Feature Pyramid Network (Rede de Pirâmides de Características)
VP	Verdadeiros positivos
VN	Verdadeiros negativos
FP	Falsos positivos
FN	Falsos negativos
YOLO	You Only Look Once (Modelo de Deteção de Objetos)
CNN	Rede Neuronal Convolucional
IA	Inteligência Artificial
SPPF	*Spatial Pyramid Pooling-Fast*
COCO	Common Objects in Context
ID	Identificador
TXT	Text File (Formato de Ficheiro de Texto)
EAN	European Article Number (Número Europeu de Artigo)
RPN	Region Proposal Network (Rede de Propostas de Região)
XML	eXtensible Markup Language (Linguagem de Marcação Extensível)

Capítulo 1

Introdução

Na atual era digital, o aumento acelerado do volume de dados presentes em documentos digitais exige o desenvolvimento de estratégias que permitem a extração destes dados de uma forma rápida e rigorosa. Grande parte dos dados estão apresentados em tabelas, pelo que a extração automática de informações presentes nas mesmas é muito útil.

A importância dos documentos digitais na sociedade atual tem vindo a crescer e transformou a forma como se gerem as informações e os dados. O facto dos documentos que possuem dados serem em formato digital tem um impacto muito significativo no quotidiano, tanto a nível de trabalho como nas diversas atividades académicas e pessoais. A transição de documentos físicos para documentos digitais deu origem a muitos desafios e oportunidades.

Neste cenário, o desenvolvimento de métodos automáticos com capacidade de extração de informações com precisão e eficiência a partir de documentos digitais, nomeadamente de tabelas, é essencial em muitas aplicações. No entanto, a diversidade de formatos e a complexidade das tabelas torna a identificação automática numa tarefa desafiadora.

Com este trabalho pretende-se obter uma solução eficiente capaz de identificar e de extrair as tabelas de documentos digitais não uniformemente formatados e otimizar atividades de diversos setores, o que permite um avanço significativo na forma de lidar com as informações num contexto digital.

A diversidade de formatos e da complexidade das tabelas em documentos digitais dificulta a sua identificação automática. Este Estágio visa o estabelecimento de uma metodologia, implementada em *Python*, que permita ultrapassar estes desafios.

Serão realizadas a exploração e a comparação de diferentes arquiteturas de modelos de aprendizagem automática e discutidas as suas qualidades e vulnerabilidades, na tarefa de identificação automática de tabelas e será desenvolvido um outro método que tem a capacidade de reconhecer e classificar as células, linhas e colunas. A recorrência intensiva a bibliotecas do *Python*, como a `Open-CV`, `scikit-learn`, `Tensorflow`, `matplotlib`, `numpy`, `pandas` e `tesseract` permite a aplicação de diferentes técnicas de processamento de imagem, de inteligência artificial e de análise de dados, que em conjunto, são otimizadas para lidar com a variabilidade e complexidade encontradas em documentos digitais e que, deste modo, permitem alcançar o objetivo proposto. O algoritmo a desenvolver irá permitir reduzir erros humanos

cometidos na extração de dados de tabelas e tornar esta tarefa automática, o que contribui de forma significativa para as mais diversas funções, com inúmeras vantagens como a economia de tempo, a redução de erros, o aumento da produtividade, o processamento de grandes volumes de dados, a formatação consistente, minimização de custos, devido à redução de trabalho manual, a integração noutros sistemas e a atualização em tempo real.

O algoritmo pretende ter em conta a generalidade dos problemas relacionados com a deteção e extração de tabelas em documentos em formato `PDF`, nomeadamente a variedade de formatos, a ausência de marcadores nítidos e a presença de imagens incorporadas, a baixa qualidade dos documentos digitalizados, as distorções e imperfeições consequentes de erros de digitalização.

Para a realização da tarefa de identificação automática de tabelas em documentos `PDF`, é necessário explorar e comparar diversas arquiteturas de modelos de aprendizagem automática, desenvolver algoritmos avançados para a identificação e classificação de tabelas em documentos, assim como aprimorar competências em processamento de imagem e análise de dados, com a utilização de ferramentas de inteligência artificial, para alcançar o objetivo, vão realizar-se as seguintes etapas:

- Explorar Arquiteturas Neuronais de Modelos de Aprendizagem, onde será efetuada a comparação entre diferentes arquiteturas de modelos de aprendizagem automática, com recurso à utilização de *software* livre.
- Identificar as qualidades e vulnerabilidades de cada um dos modelos obtidos na tarefa de reconhecimento automático de tabelas em documentos.
- Desenvolver um algoritmo de identificação de tabelas, reconhecer e classificar as suas células, linhas e colunas, de modo a facilitar a extração e a análise dos dados.
- Desenvolver competências em processamento de imagem, análise de dados e inteligência artificial, com recurso intensivo de bibliotecas do *Python*, como `open-cv`, `scikit-learn`, `TensorFlow`, `matplotlib`, `numpy`, `pandas`, `tesseract`.

Todos os códigos realizados estão presentes no repositório do GitHub [12].

1.1 Estrutura da Tese

A Tese de Mestrado encontra-se organizada nos seguintes capítulos:

- **Capítulo 1:** A **Introdução** apresenta os objetivos do trabalho, bem como a metodologia adotada para atingir os objetivos propostos;
- **Capítulo 2:** A **Apresentação da Empresa** possui uma breve descrição da Closer Consulting. Este capítulo apresenta o enquadramento da Closer Consulting no mercado, os principais serviços e produtos oferecidos, bem como a relevância da empresa no contexto da Inteligência Artificial e da visão computacional;
- **Capítulo 3:** No **Estado da Arte** são descritos os conceitos teóricos principais, as referências teóricas e a revisão e análise de trabalhos relacionados com a aprendizagem automática, a visão computacional e as técnicas de deteção de objetos, com maior foco no modelo YOLOv8 e na aplicação de OCR. São também discutidas metodologias e abordagens que permitem extrair as informações de documentos.
- **Capítulo 4:** A **Fundamentação Teórica** é constituída pela apresentação de conceitos teóricos de redes neuronais convolucionais e do modelo YOLOv8 e da sua estrutura.
- **Capítulo 5:** Na **Metodologia** são descritas as etapas que constituem o projeto de uma forma detalhada, nomeadamente, o pré-processamento dos dados, a preparação e o treino do modelo de deteção de tabelas, a aplicação do OCR, os treinos das redes neuronais de deteção da altura das linhas e de deteção dos cabeçalhos e a lógica utilizada para inserir e estruturar os resultados em ficheiros do tipo `JSON`, onde são discutidos os processos de desenvolvimento e a análise dos resultados obtidos com os modelos de deteção de objetos treinados.
- **Capítulo 6:** Nos **Resultados** são apresentados exemplos de deteção, reconhecimento e extração das tabelas nas imagens dos documentos e é estabelecida a comparação destes resultados com os que foram retornados pelo Azure Forms Recognizer.
- **Capítulo 7:** Na **Discussão**, são discutidos os objetivos concretizados e apresentadas algumas sugestões de trabalhos futuros que podem ser realizados para completar e melhorar o desempenho do algoritmo.
- **Capítulo 8:** Na **Conclusão** são apresentados os principais contributos do trabalho desenvolvido, o resumo principal dos conhecimentos adquiridos e as aplicações práticas, as reflexões pessoais e a conclusão final do trabalho realizado.

Capítulo 2

Apresentação da Empresa

2.1 A Closer Consulting

A empresa Closer Consulting é especializada em Ciência de Dados e tem como focos principais a Inteligência Empresarial, a Engenharia de Dados e a Inteligência Artificial. A empresa possui experiência numa grande variedade de setores, nomeadamente o de serviços financeiros, de telecomunicações, de centros de contacto e de retalho. A Closer Consulting é composta por uma Equipa multidisciplinar constituída por físicos, matemáticos, especialistas em computação, engenheiros e graduados na área de negócios. O compromisso com a pesquisa e o desenvolvimento constantes permitiu que a Closer Consulting colaborasse com diversas universidades em trabalhos académicos e participasse ativamente em conferências internacionais, que promovem a Missão de "Desafiar a Complexidade". [1]

2.2 Missão

A missão de "Desafiar a Complexidade"é o que move a Closer Consulting no desenvolvimento de soluções que fornecem apoio a organizações, onde são explorados os potenciais da Ciência de Dados e da Inteligência Artificial. O ambiente da empresa possui pessoas muito criativas e extremamente dedicadas, com acesso a recursos e a oportunidades que proporcionam as condições necessárias para a realização dos projetos e para a inovação constante.

2.3 Operações e Gestão

A Closer Consulting utiliza uma metodologia completa em Ciência de Dados, que inclui técnicas, métodos e tecnologias de ponta, como a Analítica Descritiva/Preditiva/Prescritiva, a Aprendizagem Automática e a Inteligência Artificial Generativa, nos processos organizacionais. A empresa torna a Ciência de Dados numa ferramenta prática e acessível, habilita os profissionais orientados para negócios a utilizar as suas perceções para promover o crescimento e a inovação.

2.4 Valores da Empresa

Os valores da Closer Consulting incluem a empatia, a integridade, a responsabilidade, o respeito, a abertura, as relações de proximidade, a cordialidade, o brio profissional, a frugalidade e a criatividade. Estes valores refletem o compromisso da empresa em tratar bem as pessoas, em inovar constantemente, apostando na eficiência das suas operações.

2.5 Departamento Solutions

A unidade de negócios da Closer Consulting, denominada Solutions, tem como foco principal a criação de soluções "chave na mão" e o desenvolvimento de aplicações específicas para satisfazer as necessidades dos clientes e a resolver problemas que estes possam ter. A unidade Solutions tem como foco transformar as soluções em plataformas ou produtos licenciáveis, com destaque nos produtos Presentime e Evalyze, que são o exemplo da inovação e da adaptação tecnológica para os desafios enfrentados pelo mercado. Existirá um maior foco na Evalyze, uma vez que é o produto mais relevante.

2.6 Departamento Solutions - Evalyze

O Departamento Solutions tem uma plataforma Evalyze, de gestão de operações, que utiliza algoritmos de Inteligência Artificial para automatizar e otimizar a distribuição, a priorização e a monitorização de atividades, que resulta num aumento da produtividade em 30%, nas operações de *back-office* implementadas. [11]

2.7 Visão Estratégica e Impacto

O Departamento Solutions tem a ambição expandir internacionalmente a sua base de clientes, com a penetração em novos mercados e o aumento do seu impacto global, através da melhoria contínua e da inovação das suas soluções, baseadas em Inteligência Artificial.

2.8 Direções Futuras da Empresa

O foco estratégico futuro do Departamento Solutions inclui a expansão internacional e a inovação contínua das soluções que implementa para fortalecer o seu impacto a nível global. A empresa compromete-se com a eficiência operacional, a transformação tecnológica e a garantia da resolução de problemas.

2.9 O Estágio na Empresa

Durante o meu Estágio na Closer Consulting, na unidade Solutions, Evalyze, tive a oportunidade de trabalhar remotamente com a minha Orientadora Ana Afonso, com o Vitor Pereira e com o Miguel Fortuna. As rotinas diárias incluíam reuniões para a discussão dos progressos e a definição dos objetivos. No final do Estágio, será realizada uma apresentação do projeto que foi desenvolvido. Esta experiência foi muito enriquecedora, permitiu ampliar o meu conhecimento técnico e ter uma melhor perceção de como as operações são realizadas pela Equipa, de forma eficiente e organizada. Para a concretização do projeto proposto, foi necessário realizar pesquisas aprofundadas e desenvolver um algoritmo de Inteligência Artificial complexo, dentro de um contexto profissional.

Capítulo 3

Estado da Arte

Hoje em dia, a automatização na identificação e extração de informações relevantes em documentos é algo cada vez mais necessário, não apenas devido ao aumento constante do volume de dados, mas também devido ao facto de que esta tarefa tenha de ser realizada num período de tempo reduzido. Este procedimento, quando realizado de forma manual, poderá ser demorado, uma vez que os indivíduos precisam de examinar os documentos individualmente e de inserir manualmente todas as informações relevantes.

Na impossibilidade de fazer uma descrição exaustiva do Estado da Arte, será dada ênfase à descrição das ideias e técnicas aprofundadas nos três documentos que foram mais relevantes para o desenvolvimento deste trabalho. Estas técnicas implicaram o uso recorrente de redes neuronais convolucionais pré-treinadas.

3.1 Redes Neuronais e Redes Neuronais Convolucionais

A Aprendizagem Automática é um ramo da Inteligência Artificial que consiste no desenvolvimento de algoritmos com capacidade de ensinar computadores a realizar determinadas tarefas, a partir de dados. Para tal, é necessária a implementação de modelos que podem melhorar o seu próprio desempenho, numa tarefa específica com a experiência acumulada ao longo do seu treino. É uma área que se encontra em constante desenvolvimento, extremamente importante para a análise e classificação de dados relativos às mais diversas áreas.

3.1.1 Perceptron Simples

Consideremos o exemplo do perceptron simples, um dos primeiros tipos de redes neuronais artificiais, introduzido por Frank Rosenblatt, no *Cornell Aeronautical Laboratory*, em 1958. Consiste numa única camada de neurónios, que realiza uma classificação linear. No perceptron simples, cada neurónio recebe os dados de entrada, multiplica-os pelos pesos associados a esse neurónio, soma as multiplicações obtidas e introduz esta soma numa função de ativação, que permite obter uma classificação. [6]

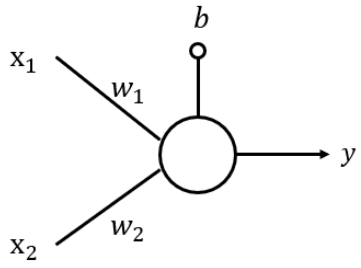


Figura 3.1 Representação de um Perceptron Simples.

Tendo em conta esta estrutura de perceptron simples, a previsão y fica:

$$\begin{aligned} a &= \sigma(w_1 x_1 + w_2 x_2 + b) \\ &= \sigma\left(\begin{pmatrix} x_1 & x_2 \end{pmatrix}\begin{pmatrix} w_1 \\ w_2 \end{pmatrix} + b\right) \\ &= \sigma(XW + b) = \sigma(z) \\ &= \frac{1}{1+e^{-z}} \\ &= a(z) \end{aligned}$$

onde w_1 e w_2 são os pesos e b corresponde ao viés, que consiste num termo adicional no perceptrão que permite ajustar o limiar de decisão, e que a desloca para que se adapte melhor aos dados, mesmo quando todas as entradas são zero. $z = w_1 x_1 + w_2 x_2 + b$ corresponde ao argumento da função de ativação σ, que pode ser, por exemplo, a função *sigmoide*, dada por $\sigma(z) = \frac{1}{1+e^{-z}}$, assumindo os valores reais entre 0 e 1.

O Perceptron Simples é utilizado em muitos problemas de classificação binária, como por exemplo, a deteção de *e-mails* de *spam* onde o *output*, a, pode ser 0, no caso de não ser *e-mail* de *spam* e 1 no caso de ser *e-mail* de *spam*.

Para realizar esta classificação é considerado um conjunto de caraterísticas para cada um dos *e-mails*, que poderá estar relacionado com o número de hiperligações ou o número de palavras que usualmente estão presentes em *e-mails* de *spam*.

Cada *e-mail* possui um conjunto de características $[x_1, x_2, ..., x_3, x_4]$, onde x_1 corresponde ao número de hiperligações presentes no conteúdo do *e-mail*, x_2 corresponde ao número de vezes que aparece a palavra "grátis", x_3 o número de vezes que ocorre a palavra "oferta", x_4 o número de vezes que ocorre a palavra "receba". O Perceptron Simples recebe este vetor como entrada, e cada característica é ponderada por um peso w, que corresponde à importância dessa caraterística do *e-mail* na decisão de ser *spam* ou não.

A operação do Perceptron Simples pode ser descrita nas seguintes etapas:

- Os pesos w_i são inicializados;
- Determina-se a soma de todos os produtos entre os pesos w_i e as caraterísticas de entrada x_i, com a soma do viés b no final, neste caso é da forma:

$$\sum_{i=1}^{4} x_i \cdot w_i + b;$$

- Aplica-se uma função de ativação σ que permite obter as classificações;
- Os pesos são ajustados, de forma a melhorar as previsões do modelo.

Com N dados de entrada que consistem nas caraterísticas e tendo em conta que A corresponde à saída, o perceptron simples seria da forma:

$$A = \sigma\left(\begin{bmatrix} x_{1,1} & x_{1,2} \\ x_{2,1} & x_{2,2} \\ \vdots & \vdots \\ x_{N,1} & x_{N,2} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} + \begin{bmatrix} b \\ b \\ \vdots \\ b \end{bmatrix}\right) = \begin{bmatrix} \sigma(w_1 x_{1,1} + w_2 x_{2,1} + b) \\ \sigma(w_1 x_{1,2} + w_2 x_{2,2} + b) \\ \vdots \\ \sigma(w_1 x_{1,N} + w_2 x_{2,N} + b) \end{bmatrix}$$

Em Aprendizagem Automática, a função de custo (Função de perda ou função de erro) é uma função que pretende quantificar os erros do modelo.

Considerados os dados $(x_{i,1}, x_{i,2}),\ i = \{1, \ldots, N\}$, com a classificação $y_i, i = \{1, \ldots, N\}$, pretende-se maximizar a função de verossimilhança relativamente às previsões obtidas a_i, $i = \{1, \ldots, N\}$, que é dada por

$$L = \prod_{i=1}^{N} a_i^{y_i} \times (1 - a_i)^{1-y_i}$$

que se pretende maximizar.

3.1.2 Perceptrons Muticamada

Os perceptrons multicamada (*Multi Layer Perceptrons*, MLPs) são fundamentais para entender o funcionamento básico das redes neuronais, artificiais, por consistirem em modelos computacionais inspirados no funcionamento do cérebro humano, que são treinados e projetados para a realização de tarefas específicas de aprendizagem automática. As redes neuronais artificias MLP consistem em múltiplas camadas de neurónios densamente conectados, o que significa que cada um dos neurónios de uma camada está conectado com todos os neurónios da camada seguinte, com o objetivo de determinar os pesos w de forma a que os valores de a sejam próximos dos valores de y.

Em redes neuronais, quando os neurónios de uma camada se encontram conectados a todos os da camada seguinte, dizemos que se trata de uma camada densa.

Neste caso, os pesos w e os vieses b serão determinados por minimização da função custo $\mathcal{L}$.

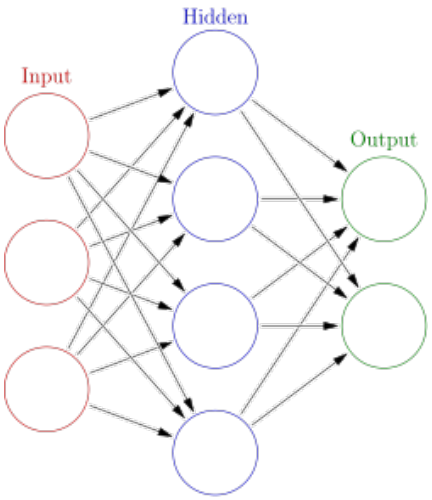


Figura 3.2 Representação de um Perceptron Multicamada. Retirada de [5].

Estes neurónios processam os dados de entrada e geram resultados, com base em funções de ativação. As redes neuronais aprendem e ajustam os seus pesos, de modo a cumprir objetivos específicos, através de um processo conhecido como *backpropagation*, ou retropropagação. Este processo otimiza os pesos e vieses de forma a minimizar o erro das previsões obtidas através da rede neuronal. O ajuste é feito pela determinação das variáveis (w, b) por métodos do tipo de perda através do tipo gradiente descendente, tendo em conta que a função de perda verifica algumas propriedades, nomeadamente a convexidade (que permite evitar mínimos locais), no qual são consideradas as derivadas parciais da função de perda, *Loss function*, $\mathcal{L}$, em relação a cada um dos pesos, w, e dos vieses, b:

$$\frac{\partial \mathcal{L}}{\partial w} \quad \text{e} \quad \frac{\partial \mathcal{L}}{\partial b}$$

Para a atualização dos pesos, e dos vieses, usa-se uma forma do tipo gradiente descendente,

com a taxa de aprendizagem η:

$$w_{novo} = w_{atual} - \eta \frac{\partial \mathcal{L}}{\partial w}$$

No decorrer do processo retropropagação, os pesos da rede neuronal são ajustados de forma a reduzir o erro de previsão. O ajuste é realizado de acordo com a taxa de aprendizagem e o passo de atualização é realizado na direção oposta ao gradiente, onde η determina o tamanho do passo considerado, tendo em conta o valor do erro calculado.

A informação do erro é utilizada de forma a otimizar os pesos da camada em que se encontra. Este procedimento iterativo repete-se em todas as camadas da rede neuronal, partindo da saída da rede neuronal, com avanço inverso até à entrada.

3.1.3 Redes Neuronais Convolucionais

As MLPs possuem limitações significativas, especialmente em tarefas que exigem o processamento de imagens. Estas não são eficientes na captura de relações espaciais e temporais presentes nas imagens e noutros tipos de dados de grandes dimensões, devido ao facto de serem redes neuronais com camadas densamente conectadas. Deste modo, é importante considerar as redes neuronais convolucionais, CNNs. As CNNs usam o conceito de convolução, com utilização de filtros que têm a capacidade de detetar as características locais das imagens, com respeito à relação espacial que existe entre os pixels presentes nas mesmas. Em termos matemáticos [30], a operação de convolução de uma CNN pode ser expressa como:

Sejam $A \in \mathbb{R}^{n \times m}$ a matriz de entrada e $E \in \mathbb{R}^{3 \times 3}$ o filtro de convolução. O produto de convolução $A * E \in \mathbb{R}^{n \times m}$ é definido por:

$$(A * E)_{r,s} = \sum_{i=0}^{2} \sum_{j=0}^{2} a_{r-1+i,s-1+j} \cdot e_{3-i,3-j}$$

Por exemplo, o elemento na sétima linha e décima coluna após a convolução é dado por:

$$(A * E)_{7,10} = \sum_{i=0}^{2} \sum_{j=0}^{2} a_{6+i,9+j} \cdot e_{3-i,3-j}$$

A saída de um neurónio de convolução é a aplicação da função de ativação σ ao resultado da convolução adicionado ao viés b:

$$C_{r,s} = \sigma((A * E)_{r,s} + b)$$

Uma camada de convolução com múltiplos filtros é representada por:

$$(\sigma_1(A * E_1 + B_1), \ldots, \sigma_\ell(A * E_\ell + B_\ell))$$

Para uma entrada com múltiplos canais, a convolução é generalizada para:

$$\sigma(A * \bar{E} + \bar{B}) = (\sigma_1(A_1 * E_1 + B_1), \ldots, \sigma_p(A_p * E_p + B_p))$$

Com múltiplos canais de entrada e múltiplos filtros, a camada de convolução é definida como:

$$(\sigma_1(A * \bar{E}_1 + \bar{B}_1), \ldots, \sigma_\ell(A * \bar{E}_\ell + \bar{B}_\ell))$$

As CNNs utilizam camadas de agrupamento (*pooling*), que concentram a informação de caraterísticas idênticas num só pixel, isto resulta na redução da dimensionalidade dos dados, sem a perda das características essenciais dos dados de entrada. É também possível aplicar o desligamento aleatório (*Dropout*), que reduz o sobreajuste, na fase de treino do modelo da CNN, pois permite determinar se o neurónio vai ser ou não ativado, com a alteração do valor de peso para 0. No decorrer de uma iteração, só serão considerados os neurónios ativos. Nas iterações seguintes, serão efetuadas novas escolhas aleatórias dos neurónios a desativar. No entanto, quando se realizam previsões com o modelo da rede neuronal, todos os neurónios contribuem para as efetuar. Assim, as CNNs conseguem ser computacionalmente mais eficientes do que as MLPs, em tarefas que requerem a análise de imagens.

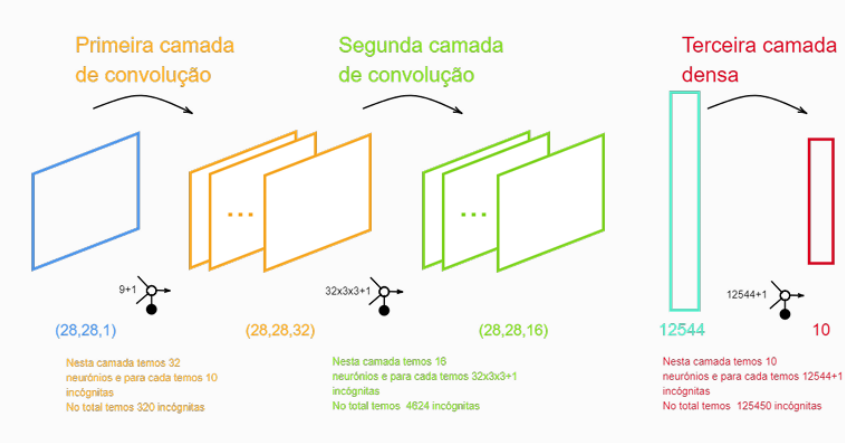


Figura 3.3 Exemplo de uma Arquitetura de Rede Neuronal Convolucional. Retirada de [30].

A evolução de Perceptrons Multicamada para Redes Neuronais Convolucionais representa um avanço muito importante no campo da aprendizagem profunda, especialmente em contextos que requerem a análise e o processamento de imagens.

3.2 Análise dos Métodos Explorados

Globalmente, existe alguma diversidade de métodos que realizam o reconhecimento de tabelas em documentos, nos quais são abordados amplos conjuntos de desafios que dificultam a identificação precisa das tabelas e a extração e interpretação efetiva de seus conteúdos. Estes desafios são significativos devido à variedade de *layouts* de documentos, estilos de tabela e a presença de elementos gráficos complexos. Entre as abordagens desenvolvidas para superar esses obstáculos, destacamos três métodos inovadores: "Deteção Robusta de Tabelas e Reconhecimento de Estruturas em Imagens de Documentos Heterogéneos", que combina a deteção de tabelas e o reconhecimento de estruturas através de técnicas avançadas e tem a capacidade de lidar com formatos de tabelas muito variados; "Método de Deteção de Tabelas YOLO-baseada", *You Only Look Once*, remove objetos ruidosos que muitas vezes dificultam a identificação das tabelas, através de restrições de *layout* e aplica um modelo de deteção de objetos para localizar as tabelas de forma mais eficiente e precisa; e "Organização de Tabelas (TAO)", focada na estruturação e organização dos dados das tabelas, que possibilitam uma extração mais eficaz.

Cada um destes métodos possui uma perspetiva diferente para a resolução do problema, com o recurso a restrições de *layout*, modelos de aprendizagem automática e modelos de redes neuronais, que melhoram a precisão e a eficiência do processo de deteção e extração de tabelas. A análise detalhada destas abordagens, que será apresentada nas próximas subsecções, permitirá destacar as suas diferentes caraterísticas e permitirá uma compreensão profunda das suas potenciais aplicações e limitações. Este estudo visa não apenas expor os métodos que atualmente possibilitam o reconhecimento de tabelas em documentos, mas também ilustrar a variedade de formas possíveis de realizar esta tarefa, com o realce das forças e das fraquezas com que cada um destes métodos se depara.

3.2.1 Deteção Robusta de Tabelas e Reconhecimento de Estruturas em Imagens de Documentos Heterogéneos

O documento "Robust Table Detection and Structure Recognition from Heterogeneous Document Images" [22], elaborado por Chixiang Ma, Weihong Lin, Lei Sun e Quiang Huo, vinculados ao *Department of EEIS, University of Science and Technology of China, Hefei, China*, e à *Microsoft Research Asia, Beijing, China*, contém avanços importantes no contexto da deteção e reconhecimento automático de tabelas presentes em documentos, através de estratégias de aprendizagem profunda, que permitem ultrapassar dificuldades que muitas vezes surgem na resolução de problemas mais complexos desta categoria, como a existência de diferentes formatos de tabelas, de estruturas complexas e distorções/variações na qualidade da imagem.

A utilização da combinação de redes neuronais convolucionais e modelos de redes neuronais pré-treinadas permitiu a deteção e compreensão da forma como as tabelas se encontram organizadas internamente, a nível da forma como as linhas e colunas se encontram dispostas

e a relação que estas estabelecem entre si, mesmo em tabelas com formatos heterogéneos.

A identificação e interpretação de tabelas inerentes a este método permite identificar e analisar tabelas de grande complexidade, o que costuma ser mais desafiador em soluções mais tradicionais. O modelo avançado de aprendizagem profunda, ao qual se recorreu para a deteção e o que foi utilizado para a identificação de componentes de tabelas foram considerados eficazes, relativamente a outros métodos. Este modelo é muito útil quando existem tipos de documentos muito diversificados, por possuir uma capacidade de adaptação acentuada, por ter a capacidade de reconhecer tabelas com espaços em branco, com algumas curvas e estruturas distorcidas, o que revela uma melhoria significativa em relação às técnicas convencionais. A combinação das caraterísticas apresentadas anteriormente torna este método relevante, na área de análise e processamento de dados em documentos.

Ao longo do artigo, a etapa da deteção de tabelas em documentos envolveu a execução de etapas detalhadas, nas quais foi necessário recorrer às redes neuronais pré-treinadas *CornerNet* e *ResNet-18*.

A *CornerNet* é um modelo de rede neuronal convolucional pré-treinada, ao qual usualmente se recorre para tarefas de deteção de objetos. [18] Esta, ao invés de começar por detetar objetos propondo as regiões candidatas e depois classificando-as, foca-se diretamente na deteção dos cantos dos objetos (neste caso, nos cantos superior esquerdo e inferior direito de uma tabela). A *CornerNet* aprende a prever pares de cantos que pertencem a um mesmo objeto, através de um conjunto de características comuns, no qual os cantos são detetados em simultâneo. Posteriormente, estes cantos são considerados para gerar as caixas delimitadoras dos objetos. Deste modo, a rede neuronal pré-treinada permite a deteção eficiente e precisa de objetos de interesse, neste caso, tabelas em imagens ou documentos.

O pré-treino da rede neuronal convolucional *CornerNet* é efetuado com o conjunto de dados COCO (*Common Objects in Context*), que possui cerca de 338 000 imagens, entre as quais pouco mais de 200 000 são etiquetadas com 80 classes diferentes.

A *ResNet* é uma rede neuronal convolucional considerada extremamente eficiente. Ao longo do artigo, foi utilizado um modelo de rede neuronal convolucional pré-treinada resultante da adaptação da *ResNet*, que se designa *ResNet-18*, que possui apenas 18 camadas, com o intuito de ser mais leve e rápida, sem a perda significativa da capacidade de aprender as características de interesse. Este modelo recorre a blocos residuais, que incluem conexões que evitam que o valor do gradiente (o gradiente indica o quanto os valores dos parâmetros da rede neuronal devem mudar na etapa de treino do modelo) fique muito reduzido, com a transição entre camadas. Esta característica permite que a aprendizagem do modelo seja eficiente, mesmo quando é considerado um número muito elevado de camadas. Esta característica faz com que a rede neuronal seja adequada para tarefas em que os recursos computacionais são limitados.

O pré-treino da *ResNet-18* é frequentemente realizado com o conjunto de dados *ImageNet*, um conjunto de dados que contém mais de 14 milhões de imagens, classificadas em mais de 20 000 categorias diferentes. As imagens deste conjunto de dados possuem uma grande

variedade de objetos [9] e o facto de o pré-treino da *ResNet-18* ser realizado com o *ImageNet* permite que esta conheça características visuais diversificadas e faz com que esta seja adaptável a diversas tarefas computacionais.

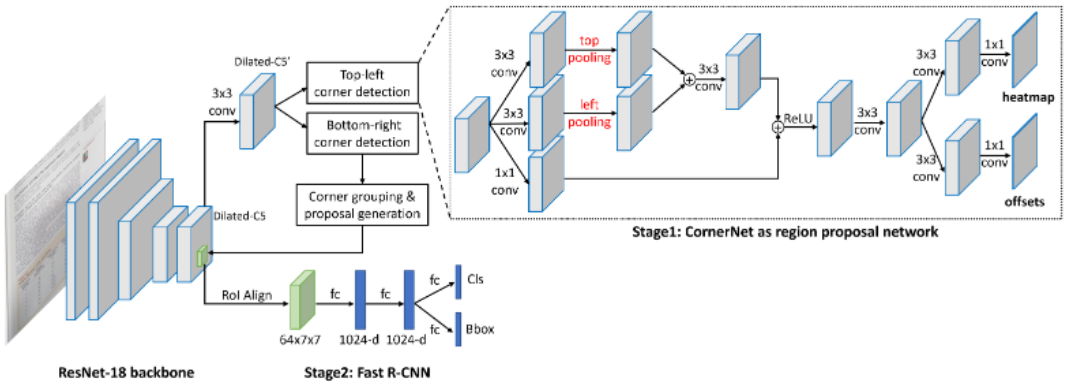


Figura 3.4 Arquitetura do modelo de deteção de tabelas baseado no *CornerNet-FRCN*. Retirada de [22].

A arquitetura apresentada na Figura 3.4 resulta da combinação entre uma rede neuronal baseada na *CornerNet* e uma *Fast R-CNN* (baseada na *ResNet-18*), que refina as delimitações obtidas pelo modelo baseado na *CornerNet*.

Para a deteção dos cantos das tabelas, o modelo sofreu uma bifurcação em dois ramos distintos, nos quais se realizou a deteção dos cantos superiores esquerdos e dos cantos inferiores direitos das tabelas, separadamente. De forma a garantir uma identificação mais precisa, os autores aplicaram operações de convolução e de agrupamento específicas. Com a localização dos cantos obtida, os cantos correspondentes são agrupados, de forma a criar as propostas iniciais para a localização das tabelas nas imagens.

De seguida, as propostas preliminares de tabelas são refinadas, através de uma *Fast R-CNN*. O alinhamento e normalização das caraterísticas extraídas para a análise são realizados com a técnica *RoI Align*, que consiste num método de alinhamento de regiões de interesse (*RoIs*) para uma deteção mais precisa de objetos em imagens. De seguida, as caraterísticas normalizadas passam por duas camadas densamente conectadas, nas quais as informações são compactadas e os elementos essenciais à identificação e categorização passam a ser realçados.

Por fim, a rede neuronal convolucional classifica a tabela e ajusta as coordenadas das caixas delimitadoras pela aplicação dos dados processados, que resulta num contorno delineado na localização real da tabela.

O reconhecimento das estruturas das tabelas foi realizado com um procedimento de divisão e de fusão, no qual se recorreu a dois tipos de redes neuronais convolucionais diferentes. Os autores começam por recorrer a uma *spatial CNN*, para obter as fronteiras horizontais e verticais das células, no interior das tabelas anteriormente delimitadas.

A arquitetura de rede neuronal avançada esquematizada na Figura 3.5 contém duas fases, uma com base na *ResNet-18* e na *Feature Pyramid Network* (FPN), para o reconheci-

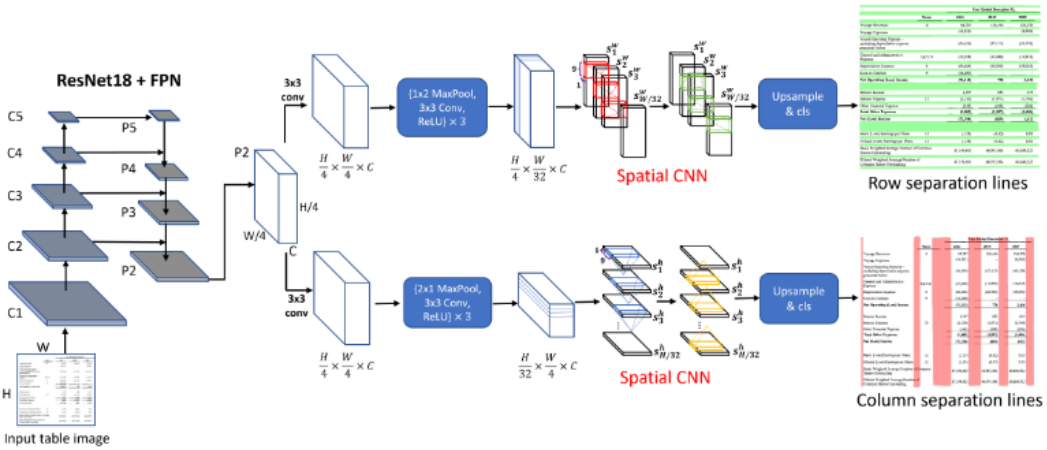


Figura 3.5 Arquitetura do modelo de deteção de tabelas baseado na *Spatial CNN*. Retirada de [22].

mento de estruturas de tabelas. A *ResNet-18*, com as 18 camadas convolucionais que possui, processa a imagem de entrada e a *Feature Pyramid Network* complementa esta etapa com a introdução de uma estrutura piramidal, que estabelece uma hierarquia que permite distinguir características importantes na imagem. A *Spatial CNN*, por sua vez, processa as caraterísticas extraídas e identifica padrões espaciais que permitem identificar as linhas horizontais que separam as células nas tabelas. Simultaneamente a este processo, acontece um processo paralelo para as linhas verticais, no qual a *Spatial CNN* identifica padrões verticais. Ambos os processos resultam da ampliação para definir as linhas que delimitam as células, de forma precisa.

As linhas horizontais e verticais identificadas neste procedimento são fundamentais para conhecer a estrutura da tabela, através da indicação da forma como as células da tabela se encontram dispostas e como os dados se encontram organizados na tabela.

A combinação da *ResNet-18* ajustada com a FPN permite uma extração precisa de características. Em conjunto com a *Spatial CNN*, permite obter a distribuição das células das tabelas encontradas em imagens, o que facilitará a transcrição dos dados tabulares de documentos.

Nesta etapa, a tabela é dividida numa elevada quantidade de células, o que possibilita uma melhor compreensão da estrutura da tabela.

De seguida, é utilizada uma *Grid CNN*, que permite fundir as células após a identificação das mesmas. Esta rede neuronal convolucional em grelha [21] reconhece as divisões de células incorretas e funde-as. A utilização desta última CNN é importante em tabelas onde existem células que se estendem por múltiplas linhas ou colunas, respeitando a estrutura original da tabela.

A estrutura da rede neuronal apresentada na Figura 3.7 é dividida em três etapas. A *Cell Generation* identifica as células individuais na tabela da imagem, o que permite isolar cada uma delas como uma unidade independente, para que esta possa ser posteriormente analisada.

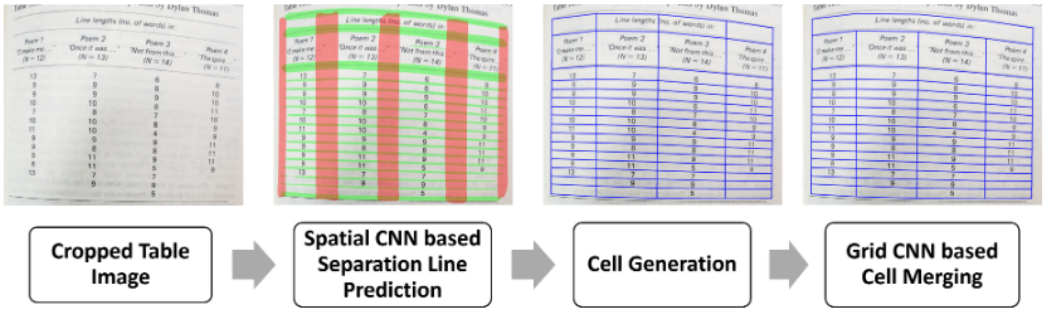


Figura 3.6 Esquema da abordagem de reconhecimento da estrutura de tabelas. Retirada de [22]

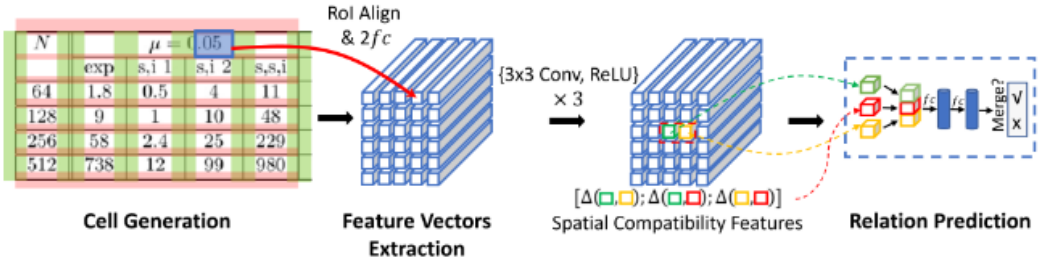


Figura 3.7 Esquema do módulo de fusão de células, com base no modelo *Grid* CNN. Retirada de [22]

De seguida, é realizada *Feature Vectors Extraction*, Extração de Vetores de Caraterísticas. A técnica *RoI Align* é utilizada para extrair caraterísticas uniformes em cada célula e estas são, de seguida, processadas por camadas densamente conectadas, de modo a normalizar caraterísticas e destacar o conteúdo e contexto das mesmas, o que possibilita a compreensão da informação contida nas células.

A etapa seguinte, *Relation Prediction*, define a disposição espacial e a conexão entre as células, com camadas convolucionais e ativações do tipo *ReLU* para que o modelo possa aprender e analisar a compatibilidade espacial, permitindo assim, a organização das relações que as células estabelecem. Esta é uma etapa de extrema importância na reconstrução da estrutura lógica e hierárquica de tabelas.

Assim, o método de Deteção Robusta de Tabelas e Reconhecimento de Estruturas em Imagens de Documentos Heterogéneos descrito no artigo consiste nas seguintes etapas, apresentadas na Figura 3.8:

1. **Deteção de Tabela:** A imagem de entrada é processada para que sejam detetadas e identificadas tabelas;
2. **Tabelas Recortadas:** As tabelas detetadas são recortadas da imagem, para que possam ser analisadas com maior detalhe;
3. **Reconhecimento da Estrutura da Tabela:** As tabelas recortadas são processadas para que sejam identificadas as suas estruturas internas, com a representação das linhas e colunas delimitadas a vermelho;

4. **Tabelas:** As tabelas com as suas estruturas reconhecidas ficam prontas para a extração, de forma a possibilitar a análise de dados.

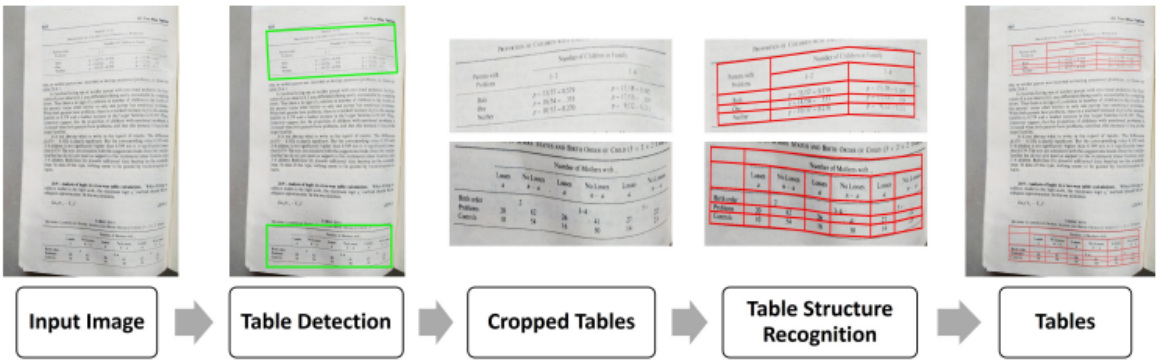


Figura 3.8 Procedimento *RobusTabNet* descrito no Artigo. Retirada de [22]

Para além da reflexão e apresentação das etapas em que consiste este método, é importante referir a função de perda, uma vez que esta é crucial no processo de otimização dos modelos de aprendizagem profunda que foram anteriormente apresentados. A função de perda, também conhecida como *Loss function* influencia diretamente a forma como o modelo aprende, com base nos dados de entrada fornecidos. No artigo, os autores abordam duas funções de perda para otimizar o desempenho dos modelos desenvolvidos. A primeira função de perda considerada serviu para otimizar o modelo e garantir uma deteção precisa de tabelas, que inclui as coordenadas que correspondem aos cantos das tabelas e ao refinamento rigoroso das caixas delimitadoras e a outra função de perda permitiu a otimização do modelo usado para reconhecer as linhas, colunas e a fusão correta das células das tabelas delimitadas, de modo a respeitar a estrutura das tabelas originais presentes nos documentos. O ajuste adequado das funções de perda é fundamental para assegurar que a aprendizagem ao longo do treino dos modelos seja eficiente, o que contribui de forma direta para a obtenção de bons resultados.

Os autores do artigo realizam a avaliação do desempenho dos modelos de deteção de tabelas de forma rigorosa, com o auxílio de algumas métricas, como a precisão, o *recall*, o *F1-Score* e a interseção sobre a união (*IoU*), também conhecida como Índice de *Jaccard*. Esta última é calculada pelo quociente entre a área da interseção entre a região prevista, região delimitada pela caixas delimitadoras, e a região real pela área da união dessas regiões:

$$IoU = \frac{\text{Área de interseção}}{\text{Área de União}}$$

O valor de IoU varia entre 0 e 1, onde 0 indica que não houve sobreposição e 1 indica a sobreposição completa. Em geral, para a análise da medida IoU, é definido um *threshold*, tal que os valores acima desse limiar são considerados corretos.

Os autores também analisam outras métricas, como a precisão, que avalia a proporção de identificações corretas de tabelas entre todas as identificações positivas efetuadas pelo

modelo e o *recall*, que mede a proporção de tabelas corretamente identificadas relativamente ao total de tabelas presentes nos documentos:

$$\text{Precisão} = \frac{\text{TP}}{\text{TP} + \text{FP}} \qquad Recall = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Onde TP corresponde ao número de casos em que o modelo classificou os casos positivos corretamente; FP corresponde ao número de casos que o modelo classificou incorretamente como positivos; FN corresponde ao número de casos que o modelo classificou incorretamente como negativos. Estas métricas permitem avaliar a capacidade do modelo para detetar as tabelas de forma correta.

Por fim, o *F1-Score* é utilizado como uma métrica que resulta da precisão e do *recall*, que permite avaliar o desempenho do modelo. Neste artigo, o *F1-Score* é calculado tendo em conta diferentes limiares da medida IoU (0.6, 0.7, 0.8 e 0.9), que permite uma avaliação mais precisa da capacidade de adaptação do modelo a diferentes níveis de rigor, na tarefa de deteção de tabelas.

3.2.2 Método de Deteção de Tabelas YOLO-baseada

No artigo "A YOLO-based Table Detection Method" [14], por Yilun Huang, Qinqin Yan, Yibo Li, Yifan Chen, Xiong Wang, Liangcai Gao e Zhi Tang, Institute of Computer Science & Technology, Peking University, Beijing, China, State Key Laboratory of Digital Publishing Technology, Founder Group Co., LTD, Beijing, China, pretende-se desenvolver e avaliar um método eficiente que permita detetar tabelas em documentos, utilizando uma versão adaptada do modelo `YOLOv3` de aprendizagem profunda.

Os autores escolheram o modelo `YOLOv3` (*You Only Look Once*, versão `3`) por ser notavelmente vantajoso a nível de eficiência computacional e da capacidade de detetar objetos, que referem ser robusto e eficaz para detetar tabelas em documentos.

O modelo tem a capacidade de realizar o processamento de imagens em tempo real, com a manutenção de alta precisão. É um modelo ideal quando é necessário detetar objetos em pouco tempo, uma vez que simplifica a sua identificação numa única etapa, com a análise e previsão de caixas delimitadoras e de classes em imagens completas efetuadas em simultâneo. O `YOLOv3` consegue captar caraterísticas espaciais e dentro de contextos específicos de forma eficiente, para a tarefa de identificação de tabelas com *layouts* diversos.

O `YOLOv3`, como muitos outros modelos avançados de aprendizagem profunda, encontra-se disponível pré-treinado. O pré-treino realizado com um grande conjunto de dados (como o `ImageNet`), de forma a ser utilizado com uma grande variedade de características de objetos aprendidas, o que permite economizar recursos computacionais.

No artigo são descritos dois métodos de pós-processamento utilizados:

1. **Remoção de Margens em Branco**: consiste na eliminação das margens em branco das regiões previstas. Como a maioria dos documentos em `PDF` tem um fundo branco, as regiões previstas podem ser maiores do que as reais. As margens em branco nessas regiões previstas são removidas para aumentar a precisão e o valor de IoU (interseção sobre a união) do modelo.

2. **Filtragem de Objetos de Página Ruidosos**: Esta técnica é usada para filtrar objetos de página, como cabeçalhos e rodapés, que são erroneamente rotulados como tabelas. São aplicadas regras heurísticas para identificar e remover essas previsões falsas positivas, melhorando assim a precisão do modelo, entre elas:

    - **Distância Mínima da Página**: Um objeto é descartado se a distância mínima entre ele e o topo ou fundo da página for menor que `5`% da altura da página.
    - **Tamanho do Objeto**: Objetos com área menor que `500` pixels são removidos, pois são considerados pequenos demais para serem tabelas.
    - **Proporção Largura-Altura**: Objetos cuja razão entre largura e altura ou altura e largura é maior que `12` são descartados, pois essa proporção não é típica de tabelas.

Este procedimento é muito importante para refinar os resultados da identificação de tabelas e melhorar o modelo `YOLOv3`, com estas adaptações. Estas regras ajudam a aumentar

significativamente a precisão do modelo, com a remoção de falsos positivos, como cabeçalhos, rodapés e linhas separadoras, que o modelo pode confundir com tabelas.

O processo de otimização das caixas delimitadoras definidas para a identificação de tabelas é feito através do algoritmo de *clustering k-means*. O *k-means* é um algoritmo de *clustering* que agrupa dados com base nas suas características, com a tentativa de minimizar as diferenças dentro de cada grupo. Deste modo, os autores recorrem a este algoritmo para analisar o conjunto de dados de treino e encontrar as dimensões mais comuns das tabelas. O *clustering k-means* agrupa as tabelas por tamanho e proporção, permitindo identificar os tamanhos de caixas delimitadoras que melhor se adequam às características das tabelas no conjunto de dados. Estas caixas delimitadoras otimizadas são então utilizadas pelo modelo YOLOv3, com a melhoria da capacidade de detetar as tabelas nos documentos com elevada precisão. De uma forma geral, este método garante que o modelo esteja mais alinhado com as características reais das tabelas encontradas em documentos.

A metodologia de avaliação do modelo abordada neste artigo é detalhada e rigorosa. Primeiramente, o modelo é testado em dois conjuntos de dados da competição *ICDAR*: a "ICDAR 2013 Table Competition"e a "ICDAR 2017 Page Object Detection (POD) Competition". A avaliação é realizada tendo em conta as métricas de precisão, *recall* e *F1-Score*, em conjunto com um limiar específico de Interseção sobre União (IoU), para determinar a precisão da deteção das tabelas. Os resultados mostram que o modelo adaptado YOLOv3, com as otimizações e métodos de pós-processamento propostos, alcança um ótimo desempenho e supera muitos outros modelos já existentes, o que o torna uma solução robusta e eficiente na identificação de tabelas.

Especificamente, o modelo demonstra alta precisão e *recall*, o que indica uma capacidade de detetar tabelas corretamente, através da minimização de falsos positivos e do não reconhecimento de tabelas nos locais onde estas se encontram. O artigo "A YOLO-based Table Detection Method"[14], representa uma contribuição significativa na deteção de tabelas em documentos, com a introdução de uma abordagem diferente e eficiente, com potencial para aplicações diversas e pesquisas futuras, na deteção de objetos em documentos diversos.

3.2.3 Organização de Tabelas (TAO)

O "*TAble Organization* (TAO)" [27], desenvolvido por Martha O. Perez-Arriaga, Trilce Estrada e Soraya Abad-Mota, *Departments of Computer Science, and Electrical and Computer* da *Engineering University of New Mexico*, nos Estados Unidos, é um método que consiste numa abordagem eficaz para a deteção de tabelas em documentos PDF. Apesar deste método não recorrer a redes neuronais, fornece um contributo significativo para o entendimento de uma forma diferente de detetar tabelas, que resulta de abordagens avançadas de processamento de documentos e heurísticas estruturais, que permitem uma análise e uma organização física e lógica das páginas do documento.

O método recebe um documento em formato PDF como entrada e é dotado de um processo de conversão para XML, com o auxílio da ferramenta PDFMiner. De seguida, é efetuada uma análise pormenorizada da hierarquia dos elementos de cada página, nomeadamente *layouts*, espaços, caracteres e marcadores de figuras e, deste modo, são identificadas as caixas de texto. Posteriormente, é calculada a distância de Manhattan entre as caixas de texto, dada por

$$d_M((x_i, y_i), (x_j, y_j)) = |x_i - x_j| + |y_i - y_j|.$$

Se o valor de distância d_M obtido entre as coordenadas dos cantos das caixas de texto for inferior a um determinado valor, *threshold*, as caixas são interpretadas como se estivessem dispostas em coluna, caso a proximidade seja verificada horizontalmente, ou em linha, se for verticalmente.

Posteriormente, as caixas de texto entre as quais se verifica a proximidade horizontal ou vertical são consolidadas e, deste modo, são construídas as candidatas a tabelas, o que permite detetá-las, mesmo quando a delimitação das células não é nítida, o que é reconhecido como uma vantagem do TAO, relativamente a outros métodos conhecidos.

Quando se efetua a extração de tabelas, o TAO reconhece e guarda as tabelas identificadas, já consolidadas. O TAO acrescenta as informações extraídas com dados adicionais, tendo em conta as informações relacionadas com a fonte e as coordenadas que realmente correspondem às células da tabela, que fornece uma representação mais completa e real das tabelas extraídas de documentos PDF. É gerado um ficheiro em formato JSON com as informações da tabela e dados extraídos.

Além disso, o TAO possui uma combinação entre o método dos k-vizinhos mais próximos e algumas heurísticas de *layout*. Este método permite a adaptação a diferentes formatações de documentos PDF e fornece uma solução flexível, com a deteção e a organização robustas de tabelas em documentos PDF.

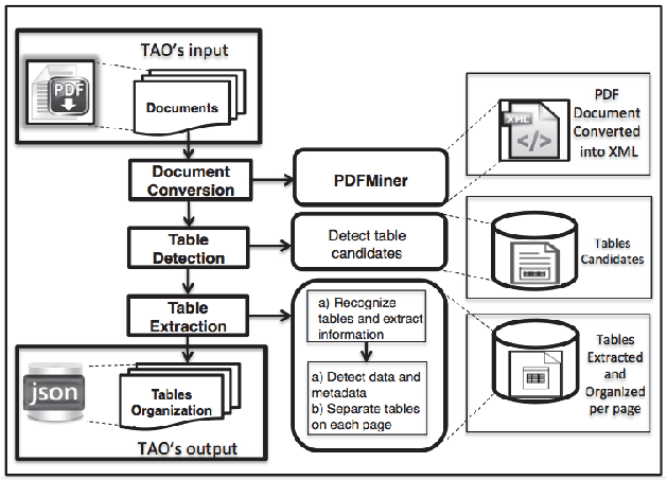


Figura 3.9 Procedimento *TAble Organization* descrito no Artigo. Retirada de [27]

Os resultados da avaliação indicam a eficácia do TAO na deteção e reconhecimento de tabelas, com medidas de precisão e de *recall* elevadas. O desempenho deste sistema é comparado em diferentes conjuntos de dados, o que demonstra a sua adaptabilidade a vários formatos e *layouts* de documentos. Os resultados obtidos com o TAO são comparados ao sistema *TableSeer*.

O *TableSeer* é um método avançado desenvolvido para a identificação e análise de tabelas em documentos digitais. Este, através de bibliotecas digitais, indexa e classifica tabelas e recorre a heurísticas baseadas no tamanho da fonte, padrões fixos e espaços em branco. O *TableSeer* também permite detetar a estrutura das tabelas e dos dados, de uma maneira eficiente que permite gerir e entender informações tabulares. Permite ainda determinar os tipos de dados, de *layout*, e a posição das tabelas nos documentos, o que o torna uma ferramenta muito útil para a extração e organização de dados tabulares.

Table Detection			
Method	***Precision***	***Recall***	***F1 measure***
TAO	0.903	0.886	0.895
TableSeer	0.962	0.453	0.616
Table Recognition			
Method	***Precision***	***Recall***	***F1 measure***
TAO	0.884	0.894	0.889
TableSeer	0.436	0.465	0.450

Figura 3.10 Métricas de avaliação do método *TAble Organization*. Retirada de [27]

A comparação do desempenho dos métodos TAO e *TableSeer* nas tarefas de identificação e reconhecimento de tabelas, presente na tabela da Figura 3.10, reflete que o método TAO teve uma precisão ligeiramente inferior, mas um *recall* e um *F1-Score* muito superiores, em comparação com o *TableSeer*, na tarefa de identificação de tabelas. No reconhecimento das tabelas, o método TAO superou significativamente o *TableSeer*, em todas as métricas de avaliação apresentadas. Deste modo, os autores consideraram o método TAO mais eficaz na identificação e reconhecimento das tabelas nos documentos.

3.2.4 Comparação dos Modelos Apresentados

Os três artigos apresentados, referem abordagens muito diferentes para detetar tabelas em documentos.

O *RobusTabNet* destaca-se por possuir uma abordagem híbrida, que permite não só a identificação de tabelas, como também o reconhecimento das suas estruturas e, deste modo, oferece uma solução completa para tabelas em documentos muito heterogéneos, uma vez que obteve bons resultados em conjuntos de imagens muito diversos.

Por outro lado, o método que utiliza o modelo `YOLOv3` destaca-se por ser mais rápido e eficiente, com adaptações específicas para tabelas e para os métodos de pós-processamento, que envolvem a remoção de ruído e de margens.

O método TAO, por sua vez, efetua a extração de tabelas através da combinação de aprendizagem automática com heurísticas de *layout*, com foco na riqueza das informações extraídas das tabelas.

A análise destes três métodos reflete a diversidade de técnicas e abordagens que permitem obter bons resultados na identificação de tabelas com *layouts* muito diversificados. Os métodos *RobusTabNet* e TAO possuem uma abordagem mais completa, com a capacidade de reconhecer a estrutura das tabelas. No entanto, a abordagem do modelo YOLO-baseado permite obter resultados de um modo mais eficiente e o facto dos modelos YOLO serem projetados para a realização de tarefas de deteção de objetos pode facilitar no reconhecimento não apenas das tabelas, como da sua estrutura.

3.3 Evolução dos Modelos YOLO

A deteção de objetos é uma tarefa muito importante em visão computacional. Enquanto a classificação de imagens consiste apenas na identificação de uma classe dominante na imagem, um problema que se resolve com uma rede neuronal convolucional, a deteção de objetos tem como objetivo não apenas classificar múltiplos objetos de interesse numa imagem, como também prever as localização e delimitação dos mesmos. Deste modo, a tarefa de deteção de objetos possui dois desafios muito significativos:

- Localização de um ou mais objetos dentro de uma imagem;
- Classificação correta do objeto de interesse no interior da região prevista.

Nos últimos anos, têm sido desenvolvidas e exploradas diferentes formas de resolver problemas de deteção de objetos, que têm resultado no aumento da precisão e da velocidade da realização desta tarefa.

Os modelos YOLO (*You Only Look Once*) têm tido muito destaque na evolução das técnicas de deteção de objetos. Com o início do modelo YOLO, criou-se uma perspetiva nova, na qual a deteção de objetos passou a ser um problema que apresenta um contraste com os métodos tradicionais que utilizam classificadores regionais para identificar objetos. Esta abordagem simples destacou-se por ser rápida e eficaz, porque permite que uma rede neuronal processe uma imagem inteira com uma única passagem, o que permite a obtenção de resultados em tempo real. A popularidade do modelo YOLO tem crescido rapidamente. Estes modelos são adotados em muitas aplicações que exigem uma previsão rápida e eficiente. Com o avanço da tecnologia, surgiram versões sucessivas do modelo YOLO, onde cada uma tem melhorias consideráveis na precisão, rapidez e capacidade de lidar com diferentes tipos de objetos.

A arquitetura do modelo YOLO tem sido aperfeiçoada desde a versão original até às mais recentes, como a YOLOv4 e YOLOv5, que possuem novas técnicas que melhoram a deteção de objetos mais pequenos e que otimizam a utilização de recursos computacionais. A evolução dos modelos YOLO [42] demonstra o interesse atual de alcançar uma deteção de objetos mais precisa e eficiente, com a manutenção da rapidez do modelo, o que permite obter previsões em tempo real.

- **YOLOv1 (2016):** Os principais autores do YOLOv1 foram Joseph Redmon, Santosh Divvala, Ross Girshick e Ali Farhadi. Joseph Redmon e Ali Farhadi estudavam na Universidade de Washington e Ross Girshick colaborava com a Microsoft Research e foi um dos pioneiros da R-CNN, que influenciou na criação do modelo YOLOv1. Este modelo é o primeiro com a capacidade de realizar a tarefa de deteção de objetos como um problema de regressão direta. O modelo revolucionou a tarefa de deteção de objetos, devido à rapidez e à capacidade de processar as imagens numa única passagem pela rede neuronal, que contrariamente ao modelo R-CNN não começa por identificar as regiões propostas; [10]

- **YOLOv2 (2016):** Os principais autores do modelo YOLOv2 foram novamente Joseph Redmon e Ali Farhadi, ambos na Universidade de Washington, com contribuições do Allen Institute for AI (AI2). Introduziu a normalização por lotes, que suporta imagens de alta resolução, o uso de âncoras para melhorar a precisão (inspiradas pela *Faster R-CNN*) e a deteção de objetos menores; [31]

- **YOLOv3 (2018):** Mais uma vez, os autores principais são Joseph Redmon e Ali Farhadi, com a mesma afiliação com a AI2. Com uma arquitetura mais profunda com conexões residuais, passou a prever objetos em três escalas diferentes para as previsões, com a utilização da espinha dorsal `Darknet-53`, que permite uma maior eficiência e precisão nas previsões [4]. Apresenta um equilibrio adequado entre a rapidez e precisão nas suas previsões; [41]

  O principal autor dos modelos de deteção de objetos YOLO, Joseph Redmon, abandonou a pesquisa relativa à visão computacional em 2020, devido a questões éticas relativas ao uso indevido da Inteligência Artificial. Após a saída de Joseph Redmon, o desenvolvimento das versões posteriores do YOLO foram realizados por outros autores e organizações.

- **YOLOv4 (2020):** O autor principal do modelo YOLOv4 foi Alexey Bochkovskiy, que contribuiu para a implementação do YOLO em `Darknet` liderou no desenvolvimento dos modelos YOLO, após o abandono de Joseph Redmon. O modelo foi desenvolvido em conjunto com Chien-Yao Wang e Hong-Yuan Mark Liao, afiliados ao instituto de pesquisa Academia Sinica. Melhorou a nível do desempenho, com conexões adicionais numa arquitetura `CSPDarknet53` e otimizações para *hardware*, com um desempenho eficiente e rápido em dispositivos com recursos limitados; [28]

- **YOLOv5 (2020):** A empresa responsável pela implementação do modelo foi a Ultralytics, que desenvolve *software* em visão computacional. O autor principal foi Glenn Jocher, o fundador da empresa. O modelo teve como objetivo a melhoria da facilidade da utilização e a rapidez do treino do modelo, que permite a integração com o `PyTorch`; [40]

  Surgiram questões relativas à designação do modelo YOLOv5, por este ter sido desenvolvido por autores diferentes dos originais. No entanto, devido à popularidade e à facilidade da implementação do modelo, o nome YOLOv5 foi o adotado.

- **YOLOv6 (2022):** O modelo YOLOv6 foi desenvolvido pela empresa chinesa Meituan. A abordagem implementada neste modelo inicide predominantemente na eficiência para produção, na qual, houve a otimização da sua arquitetura, para que as suas previsões se tornassem rápidas e leves. A espinha dorsal da rede neuronal é o `RepVGG`, por ser um modelo simples e eficiente. O modelo também não recorre a âncoras para obter as previsões, o que aumentou muito significativamente a sua rapidez. Adequado para a realização de previsões em tempo real; [8]

- **YOLOv7 (2022):** Os autores principais são Chien-Yao Wang, Alexey Bochkovskiy e Hong-Yuan Mark Liao, que voltaram a colaborar no desenvolvimento do modelo YOLOv7. Neste desenvolvimento, foram introduzidas alterações, nomeadamente, a alteração da componente da cabeça da rede neuronal, que passou a separar a classificação da regressão das caixas delimitadoras. Melhorou a aprendizagem, através da captura de informações em várias escalas. É eficiente quando o conjunto de dados é limitado e adicionou a previsão da pose humana; [15]

- **YOLOv8 (2023):** O modelo YOLOv8 possui melhorias significativas tanto a nível da precisão, como da velocidade e da eficiência e foi implementado pela Ultralytics. A versão 8 do modelo introduz uma nova arquitetura, que facilita o ajuste dos hiperparâmetros e o treino em conjuntos de dados diferentes. O modelo YOLOv8 destaca-se por ter a capacidade de realizar deteções de uma forma mais precisa, em imagens de vários tamanhos e otimizações que permitem uma integração mais simples com estruturas de desenvolvimento, como o `PyTorch`. [17] [29]

O pré-treino dos modelos YOLO foi realizado com o conjunto de dados de grande escala COCO (*Common Objects in Context*)[37], que possui cerca de 338 000 imagens, entre as quais pouco mais de 200 000 são etiquetadas com 80 classes diferentes, que incluem objetos básicos como bicicletas, carros e animais. [3] As anotações do conjunto COCO incluem caixas delimitadoras óticas e máscaras de segmentação, as quais indicam os pixels da imagem que pertencem a um objeto ou classe.

O conjunto de dados COCO oferece métricas de avaliação uniformizadas para a deteção de objetos, o que permite uma comparação mais intuitiva do desempenho de diferentes modelos. O COCO encontra-se dividido em três conjuntos: treino, validação e teste, o que permite a utilização deste conjunto de dados para diversas tarefas, de uma forma mais fácil.

Capítulo 4

Fundamentação Teórica

4.1 Introdução

O modelo de aprendizagem profunda de última geração, YOLOv8, tem como objetivo a realização da deteção de objetos em tempo real, em diversas tarefas de visão computacional.

A arquitetura do modelo é muito sofisticada e revolucionária, por ter tornado a deteção de objetos numa tarefa eficiente e precisa, mesmo em tempo real. [26]

Este tipo de modelos de aprendizagem profunda, como os modelos YOLO tornaram-se extremamente importantes em diferentes áreas, como a gestão de tráfego, a vigilância por vídeo, a medicina, a condução automática de veículos e muitas outras. A capacidade que o modelo possui na deteção de objetos em tempo real auxilia na tomada de decisão.

As caraterísticas mais notáveis do modelo YOLOv8 são a rapidez e a precisão com que este identifica e localiza objetos em imagens e em vídeos. [38]

Neste capítulo serão apresentados conceitos matemáticos subjacentes ao modelo YOLO, com a inclusão das redes neuronais convolucionais e a apresentação das funções de perda e das técnicas de otimização. Será também referida a forma como o modelo YOLO foi utilizado na deteção e no reconhecimento de estruturas em tabelas.

4.2 Componentes Principais das Redes Neuronais Convolucionais

As redes neuronais convolucionais, também conhecidas como CNNs, consistem em redes neuronais que se destinam ao processamento de dados com a estrutura em grelha, como é o caso das imagens. As camadas convolucionais servem para detetar padrões e permitem que o modelo aprenda caraterísticas em imagens. As CNNs são muito utilizadas no reconhecimento e deteção de objetos. As redes neuronais convolucionais são constituídas por um ou vários blocos de camadas convolucionais e depois possuem camadas densamente conectadas, de acordo com a Figura 4.1.

Nas redes neuronais convolucionais são realizadas operações como as que são descritas nas subsecções que se seguem.

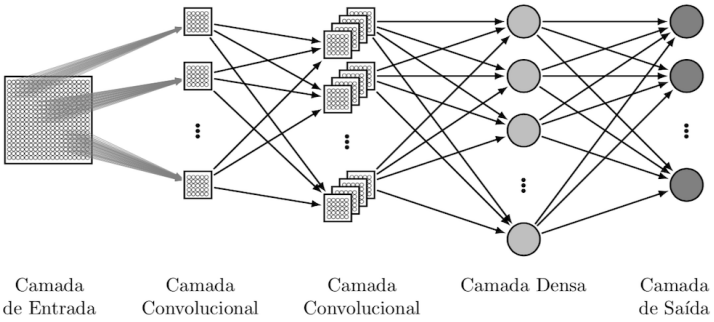


Figura 4.1 Exemplo de Rede Neuronal Convolucional. Retirada de [32].

4.2.1 Filtro (*Kernel*)

Os **filtros** consiste num *array* bidimensional e são muitas vezes interpretados como detetores de caraterísticas, o valor no filtro pode ser adaptado no decorrer do treino do modelo. O filtro percorre a imagem e realiza operações entre a entrada e o valor produzido na saída, sendo esta conhecida como o mapa de caraterísticas. Este consiste na saída da camada convolucional de uma rede neuronal, onde são guardadas caraterísticas detetadas, tais como bordas, texturas ou formas nas imagens de entrada;

Sejam $A \in \mathbb{R}^{n\times m}$ a matriz de entrada e E um filtro de convolução de dimensões de, por exemplo, 3×3, tal que $E \in \mathbb{R}^{3\times 3}$. O produto de convolução $A * E \in \mathbb{R}^{n\times m}$ é definido por:

$$(A * E)_{r,s} = \sum_{i=0}^{2}\sum_{j=0}^{2} a_{r-1+i,s-1+j} \cdot e_{3-i,3-j}$$

Para a imagem I e o filtro de dimensões 3×3, o mapa de caraterísticas de I fica da forma:

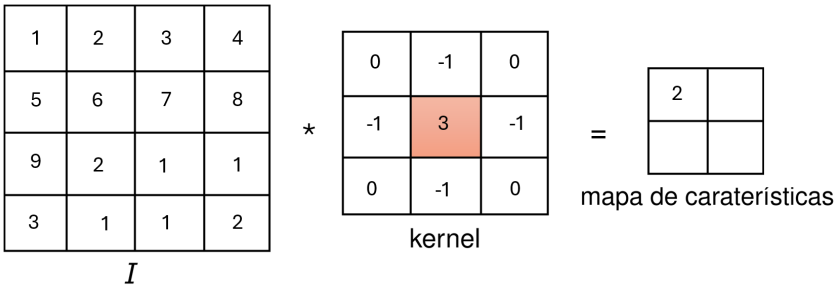


Figura 4.2 Convolução em CNNs.

O valor 2 presente no mapa de caraterísticas foi determinado da forma:

$$0 \times 1 + (-1 \times 2) + 0 \times 3 + (-1 \times 5) + (3 \times 6) + (-1 \times 7) + (0 \times 9) + (-1 \times 2) + (0 \times 1) = 2$$

O **número de filtros** está relacionado com a profundidade da produção. No caso de haver dois filtros diferentes, estes resultariam em dois mapas de caraterísticas diferentes, o que resulta numa profundidade de dois.

4.2.2 Passo (*Stride*)

O **passo** consiste no número de pixels que o filtro avança ao aplicar a convolução numa imagem ou noutro tipo de dados. Um passo maior torna a redução do tamanho da imagem mais rápida, enquanto a utilização de um passo menor realiza a convolução com maior número de sobreposições e existe uma maior preservação dos detalhes da imagem.

4.2.3 Preenchimento (*Padding*)

O **preenchimento** consiste numa operação que adiciona valores na borda da imagem. Esta operação é frequentemente utilizada quando não é possível aplicar o filtro na imagem de entrada. Existem vários tipos de *padding*.

- *Padding* de Zeros: É um tipo de *padding* muito utilizado em casos nos quais o filtro não se encaixa na imagem. Desta forma, todos os elementos que se encontram fora da imagem de entrada ficam iguais a zero. Assim, ocorre o descarte da última camada de convolução, caso as dimensões não se alinhem.

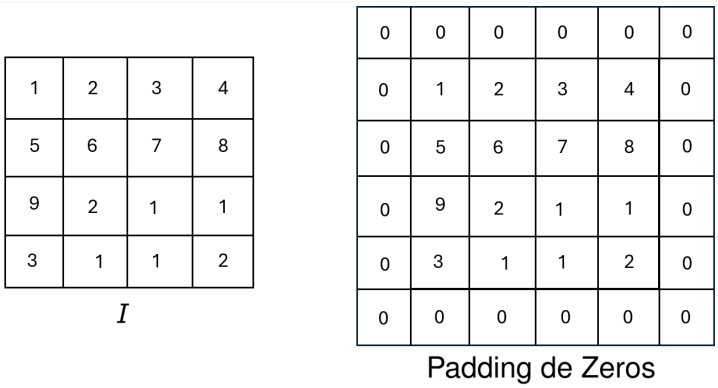


Figura 4.3 *Padding* de Zeros.

- *Padding* Igual: A camada de produção permanece com o mesmo tamanho da camada de entrada.

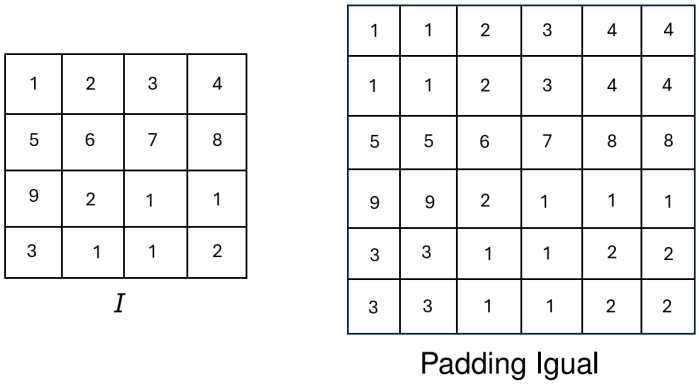


Figura 4.4 *Padding* Igual.

4.3 Arquitetura do modelo YOLOv8

O conhecimento da arquitetura e dos recursos do modelo YOLOv8 [43] é muito importante para a obtenção de conhecimentos relativos às aplicações e às capacidades que o modelo possui na realização de diversas tarefas de visão computacional. Em geral, a estrutura da rede neuronal YOLOv8 é dividida em três partes: [16]

- **Espinha dorsal:** Arquitetura de aprendizagem profunda, que atua como um extrator de caraterísticas;
- **Pescoço:** Combina as caraterísticas das camadas que constituem a componente da espinha dorsal;
- **Cabeça:** Prevê as caixas delimitadoras e as classes, consiste na saída produzida pelo modelo de deteção de objetos.

A estrutura do modelo YOLOv8 encontra-se na Figura 4.5.

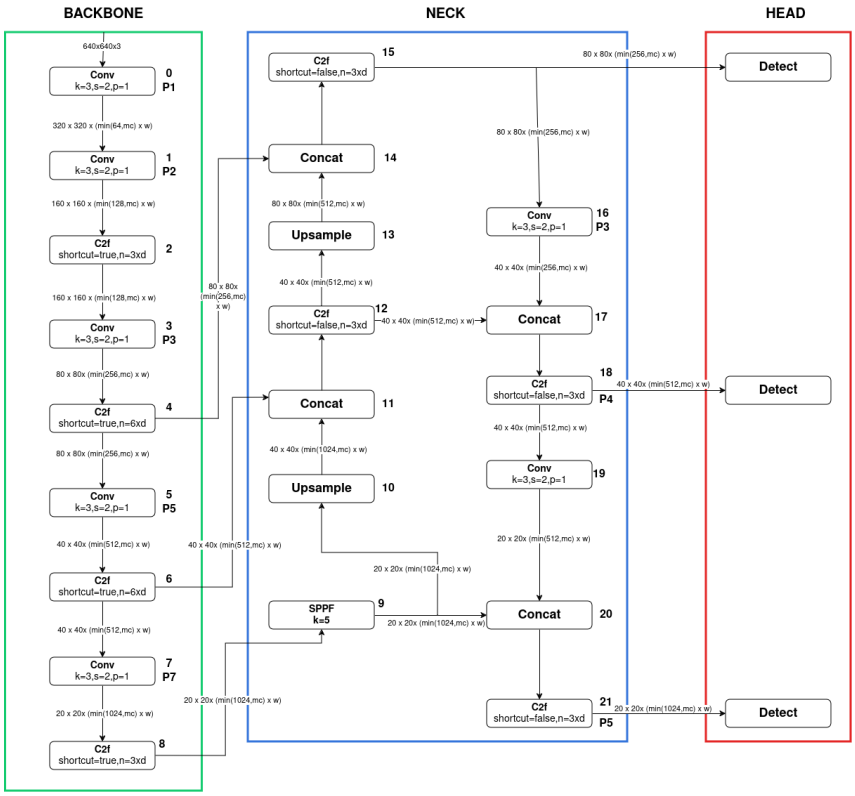
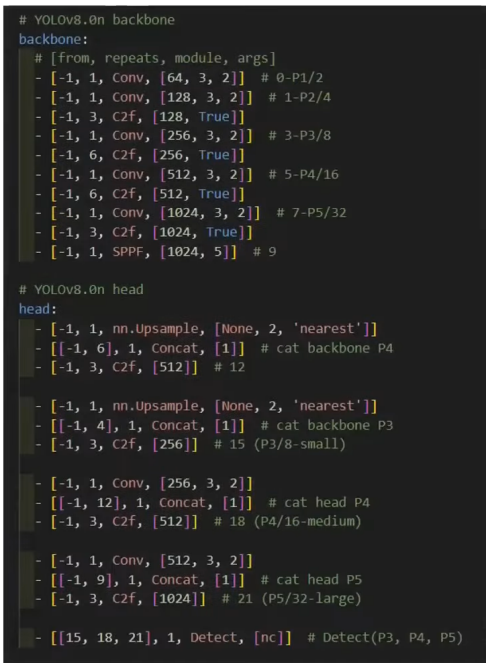


Figura 4.5 Arquitetura do Modelo YOLOv8 treinado. Retirada de [33].

O código em *Python* com o qual a espinha dorsal e o pescoço do modelo YOLOv8 são implementados é o que se encontra representado na Figura 4.6 abaixo.



```yaml
# YOLOv8.0n backbone
backbone:
  # [from, repeats, module, args]
  - [-1, 1, Conv, [64, 3, 2]]  # 0-P1/2
  - [-1, 1, Conv, [128, 3, 2]]  # 1-P2/4
  - [-1, 3, C2f, [128, True]]
  - [-1, 1, Conv, [256, 3, 2]]  # 3-P3/8
  - [-1, 6, C2f, [256, True]]
  - [-1, 1, Conv, [512, 3, 2]]  # 5-P4/16
  - [-1, 6, C2f, [512, True]]
  - [-1, 1, Conv, [1024, 3, 2]]  # 7-P5/32
  - [-1, 3, C2f, [1024, True]]
  - [-1, 1, SPPF, [1024, 5]]  # 9

# YOLOv8.0n head
head:
  - [-1, 1, nn.Upsample, [None, 2, 'nearest']]
  - [[-1, 6], 1, Concat, [1]]  # cat backbone P4
  - [-1, 3, C2f, [512]]  # 12

  - [-1, 1, nn.Upsample, [None, 2, 'nearest']]
  - [[-1, 4], 1, Concat, [1]]  # cat backbone P3
  - [-1, 3, C2f, [256]]  # 15 (P3/8-small)

  - [-1, 1, Conv, [256, 3, 2]]
  - [[-1, 12], 1, Concat, [1]]  # cat head P4
  - [-1, 3, C2f, [512]]  # 18 (P4/16-medium)

  - [-1, 1, Conv, [512, 3, 2]]
  - [[-1, 9], 1, Concat, [1]]  # cat head P5
  - [-1, 3, C2f, [1024]]  # 21 (P5/32-large)

  - [[15, 18, 21], 1, Detect, [nc]]  # Detect(P3, P4, P5)
```

Figura 4.6 Código da Construção da Espinha dorsal e da cabeça do Modelo YOLOv8. Retirada de [16].

Os valores de parâmetro para o múltiplo de profundidade, o múltiplo de largura e o número de canais variam de acordo com o modelo utilizado. Existem vários modelos:

- `n`: o modelo de dimensão mais pequeno, que permite uma previsão mais rápida, mas com menores valores de exatidão (*accuracy*);
- `s`: modelo pequeno que possui um bom equilíbrio entre o tempo de previsão e o valor da medida exatidão;
- `m`: o modelo médio, com uma velocidade de previsão menos rápida que as versões acima apresentadas, prevê objetos com valores maiores de exatidão (*accuracy*);
- `l`: modelo grande, que permite obter valores de exatidão superiores aos de todos os modelos acima referidos, mas realiza as previsões da forma mais lenta;
- `xl`: modelo extra grande, que possui o maior valor de exatidão de entre todos os modelos existentes e é adequado para tarefas em que é necessária uma quantidade significativa de recursos de *hardware* para funcionarem corretamente.

Os parâmetros para cada um destes modelos são definidos da forma apresentada na figura que se segue.

Onde o múltiplo de profundidade (d) corresponde ao número de blocos da rede neuronal, o múltiplo de largura (w) e o número máximo de canais (mc) determinam o canal de *output*. A entrada do modelo `YOLOv8` possui três canais.

Os blocos convolucionais presentes na arquitetura do modelo YOLOv8 são constituídos de acordo com a Figura 4.8.

Variante do Modelo	Múltiplo de profundidade (d)	Múltiplo de largura (w)	Número máximo de canais (mc)
n	0,33	0,25	1024
s	0,33	0,5	1024
m	0,67	0,75	768
l	1,00	1,00	512
xl	1,00	1,25	512

Figura 4.7 Valores de parâmetros para os diferentes modelos YOLOv8.

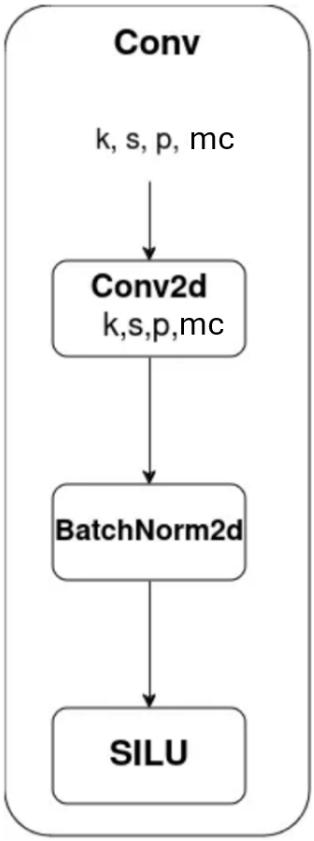


Figura 4.8 Blocos Convolucionais na Arquitetura YOLOv8. Retirada de [33].

Os blocos convolucionais, representados por **Conv** são os mais simples de toda a arquitetura e são compostos por uma camada `Conv2d`, que é uma camada convolucional, uma camada `BatchNorm2d` e a função de ativação do tipo `SiLU`.

A camada de `BatchNorm2d` é uma camada de normalização que normaliza em lotes. As camadas de normalização são muito utilizadas em modelos de aprendizagem profunda, porque estabilizam o treino do modelo e tornam a convergência dos modelos mais rápida.

Em primeiro lugar, são determinados os valores da média μ e da variância σ^2 de cada um dos lotes, através das expressões:

$$\mu_{Lote} = \frac{1}{m}\sum_{i=1}^{m} x_i$$

$$\sigma_{Lote}^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{Lote})^2$$

A normalização em lotes ajusta a saída da camada anterior através da subtração da média do lote e da divisão pela raiz da variância, com a adição de uma constante ϵ, de valor pequeno, ao desvio padrão do lote. A saída normalizada é da forma:

$$\hat{x}_i = \frac{x_i - \mu_{Lote}}{\sqrt{\sigma_{Lote}^2 + \epsilon}}$$

Em CNNs, a camada `BatchNorm2d` realiza a normalização em lotes de entradas bidimensionais, que são geradas pelas camadas convolucionais. Esta camada permite que os valores que percorrem a rede neuronal não se encontrem em diferentes escalas, nem demasiado grandes nem demasiado pequenos, de forma a evitar problemas no decorrer do treino do modelo de aprendizagem profunda.

A função de ativação `SiLU`, *Sigmoid Linear Unit*, também conhecida como *swish* e designada por função de ativação sigmoide ponderada é uma função de ativação recente que tem tido resultados promissores. A `SiLU` tem propriedades das funções lineares e não lineares, o que permite que os valores positivos sejam preservados, enquanto os valores negativos possuem uma não linearidade. Esta caraterística contribui para melhorar o desempenho do modelo, no decorrer da aprendizagem.

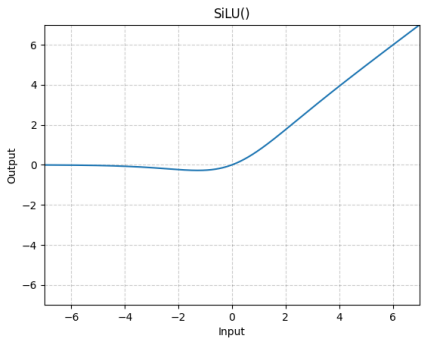


Figura 4.9 Função de Ativação `SiLU`. Retirada de [34].

A função de ativação `SiLU` é dada por:

$$SiLU = x \cdot \sigma(x)$$

onde a função $\sigma(x)$ é a sigmoide logística, dada por

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

Para o desenvolvimento da nova metodologia de deteção e reconhecimento das tabelas em documentos, foi utilizado o modelo `YOLOv8x`, por ter a capacidade de oferecer maiores valores de *accuracy*, no qual os valores dos parâmetros múltiplo de profundidade, múltiplo de largura e número de canais são iguais a 1.00, 1.25 e 512, respetivamente.

O primeiro bloco convolucional do modelo YOLOv8 é representado pela Figura 4.10.

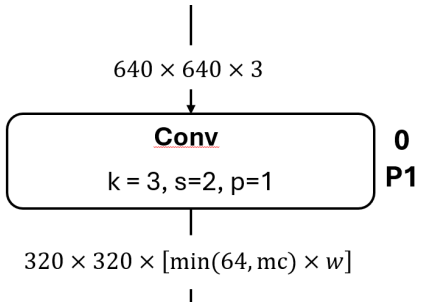


Figura 4.10 Bloco Convolucional do modelo YOLOv8.

A espinha dorsal do modelo YOLOv8 é constituída por duas camadas convolucionais de filtro k igual a 3, passo s igual a 2 e *padding* p igual a 1.

O modelo `YOLOv8` está preparado para receber imagens de entrada de dimensões 640 por 640 e estas imagens devem possuir 3 canais. Após a passagem pela primeira camada convolucional da espinha dorsal, ocorre a redução das dimensões espaciais da imagem. O número de canais após a passagem pela primeira camada fica igual a $\min(64, mc) \times w$.

Segundo a Figura 4.6, o canal de saída na base do primeiro bloco convolucional é igual a 64, o que significa que a operação convolucional da rede YOLOv8 gera 64 mapas de caraterísticas, que representam padrões e caraterísticas extraídas da imagem de entrada no decorrer da primeira convolução realizada pela rede neuronal.

Tendo também em conta que no modelo `YOLOv8x`, os valores de parâmetro são $d = 1.00$, $w = 1.25$ e $mc = 512$, o número de canais de saída do primeiro bloco convolucional fica $min(64, mc) \times w = min(64, 512) \times 1.25 = 80$. Deste modo, o canal de saída da primeira camada de convolução, para a variante do YOLOv8 utilizada é 80. É possível analisar os canais de saída de cada bloco convolucional de forma análoga a esta primeira camada.

A saída da segunda camada convolucional da espinha dorsal é a entrada no bloco C2f, que possui os parâmetros `shortcut` e `n`.

- O bloco *bottleneck*, ou de gargalo, é composto por um bloco convolucional que possui uma conexão de atalho. Esta conexão permite que o modelo de aprendizagem profunda ignore algumas camadas, o que facilita no treino do modelo e melhora a aprendizagem do mesmo, para além de prevenir problemas associados à alteração dos valores de gradientes para `0`. Quando o parâmetro `shortcut` assume valor `True`, utiliza-se o atalho no bloco de gargalo, como representado na Figura 4.11. Caso o valor `shortcut` assuma valor `False`, a entrada é realizada com dois blocos convolucionais seguidos, de acordo com a Figura 4.12.

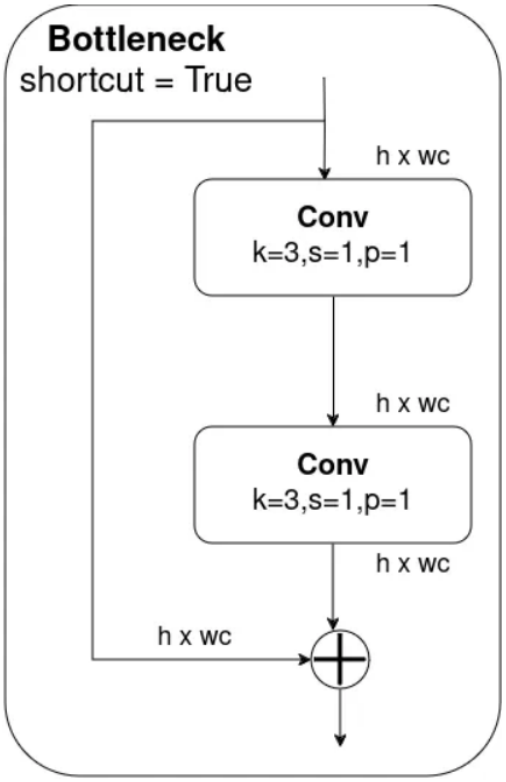


Figura 4.11 C2f com parâmetro `shortcut = True`. Retirada de [33].

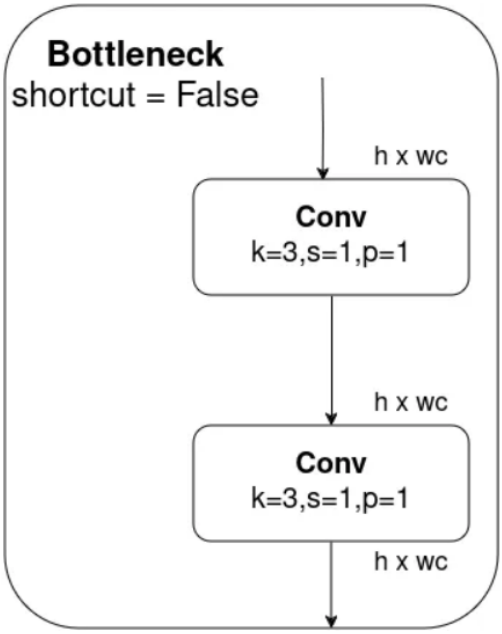


Figura 4.12 C2f com parâmetro `shortcut = False`. Retirada de [33].

- O valor `n` corresponde ao número de blocos de gargalo que vão ser utilizados. Este valor é determinado por $3 \times d = 3 \times 1.00 = 3$

Ainda na espinha dorsal do código, existem novos blocos de camadas convolucionais e de C2f. O último bloco C2f está conectado com SPPF. O bloco SPPF, representado na Figura 4.13, do inglês *Spatial Pyramid Pooling Fast* é constituído por camadas de `MaxPooling2d` e uma camada convolucional de filtro de valor 3, um passo igual a 1 e *padding* 1.

O bloco SPPF consiste na junção de um bloco convolucional com três camadas `MaxPool2d`, cujas saídas são concatenadas e processadas novamente. A utilização do *Spatial Pyramid Pooling* permite que as redes neuronais tratem imagens de diferentes tamanhos, com a captura de informações em múltiplas escalas, o que possibilita o reconhecimento de objetos. O SPPF utiliza apenas um filtro para o *pooling*, o que reduz o tempo de execução do código e a carga computacional. [20]

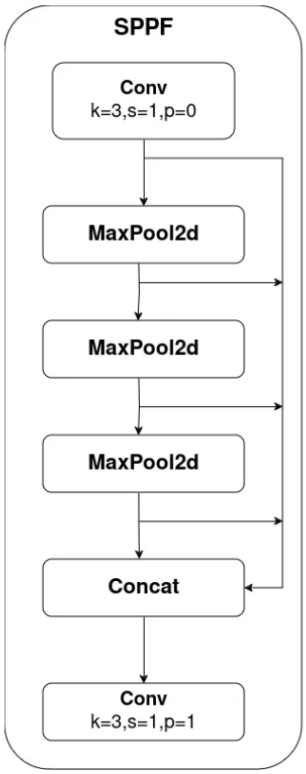


Figura 4.13 *Spatial Pyramid Pooling Fast*. Retirada de [33].

As **camadas de *pooling*** são usadas para reduzir as dimensões do mapa de características (altura e largura), através de uma subamostragem, com o objetivo de diminuir o custo computacional e de memória. O processo de *pooling* consiste na determinação da média ou do valor máximo de uma área retangular pequena do mapa de características. É necessário definir o tamanho da área, do passo e o tempo de preenchimento (*padding*), como acontece nas camadas convolucionais.

As camadas *Pooling* máximo a 2 dimensões consistem nos casos em que se determina o valor máximo da região retangular pequena do mapa de caraterísticas que for definida, segundo a expressão:

$$g(a,b) = max_{0\leq p<k;0\leq q<k}\{f(a\cdot s+p, b\cdot s+q)\}$$

onde:

- g corresponde ao mapa de caraterísticas no final do *max pooling*.
- f corresponde ao mapa de caraterísticas de entrada;

- p e q são os índices no interior do filtro de *pooling*;
- a e b correspondem aos índices do mapa de caraterísticas da saída;
- k corresponde à dimensão do lado do filtro;
- s é o passo, que define os pixels em que o filtro se irá mover ao final de cada vez que percorre.

De acordo com o exemplo da Figura 4.14.

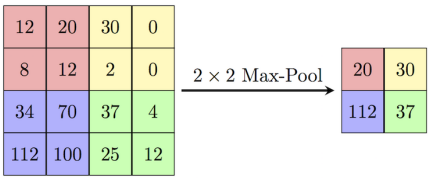


Figura 4.14 Exemplo de *Pooling* máximo 2×2. Retirada de [2].

A camada de *upsample* na componente do pescoço do modelo YOLOv8 duplica o tamanho do mapa de características, de forma a não alterar o número de canais de saída.

O bloco de concatenação tem como objetivo a junção dos canais de saída dos blocos combinados, que mantêm a mesma resolução.

É utilizada uma camada *upsample* para aumentar o mapa de caraterísticas e garantir que a saída do SPPF tem a mesma resolução que a saída do C2f (*Concat* 20). Assim, o número de canais passa a ser igual a 1024.

No final da estrutura, representada do lado direito da Figura 4.5, que consiste na componente da cabeça, o bloco de deteção de cima permite detetar objetos pequenos. Este recebe o mapa de caraterísticas resultante do bloco C2f, com dimensões 80×80 e um número de canais igual a

$$min(256, 512) \times 1.25 = 256 \times 1.25 = 320.$$

Este mapa de caraterísticas vai também dar entrada no bloco convolucional número 16, com filtro igual a 3, passo igual a 2 e *padding* 1. Por outro lado, o bloco de deteção do meio deteta objetos maiores que o bloco de deteção representado acima e menores do que o bloco de deteção representado abaixo. Recebe, como entrada, o mapa de caraterísticas que sai do C2f, com dimensões 40×40 e um número de canais igual a $min(512, 512) \times 1.25 = 640$. Por fim, o último bloco de deteção recebe o mapa de caraterísticas de saída do bloco C2f de dimensão 20×20 e número de canais igual a $min(1024, 512) \times w = 512 \times 1.25 = 640$ e permite detetar objetos maiores do que os detetados pelos outros dois blocos. A entrada de C2f é igual à da concatenação entre o SPPF e a camada convolucional que deu entrada na penúltima camada de deteção.

4.3.1 Função de Perda

No modelo YOLOv8, a precisão das previsões da deteção dos objetos é otimizada através da função de perda com a combinação de diferentes componentes que permitem ajustar a confiança da deteção, das previsões das caixas delimitadoras e da classificação dos objetos.

O modelo YOLO divide a imagem que dá entrada numa grelha com $S \times S$ células, na qual cada célula é responsável por detetar os objetos que se encontram no seu interior. A célula responsável pela previsão de um objeto é a que possui o centro do objeto. Deste modo, o YOLO adquire uma maior facilidade em localizar e classificar objetos de diferentes tamanhos e posições.

Tendo em conta que a imagem de entrada no modelo YOLO possui dimensões 640×640 pixels, caso esta seja dividida numa grelha de 8×8, cada célula na grelha terá de ter dimensões 80×80 pixels. Se o centro de um objeto de interesse estiver localizado no interior de uma das células, de acordo com o exemplo presente na Figura 4.15, a célula onde se encontra essa região central do objeto de interesse é a responsável pela previsão da caixa delimitadora que contorna o objeto de interesse e o classifica corretamente com a classe `gaivota`. Na figura abaixo representada, o objeto gaivota possui mais do que uma célula, no entanto, a célula que é responsável pela previsão é a do centro da gaivota, que se encontra delimitada a vermelho.

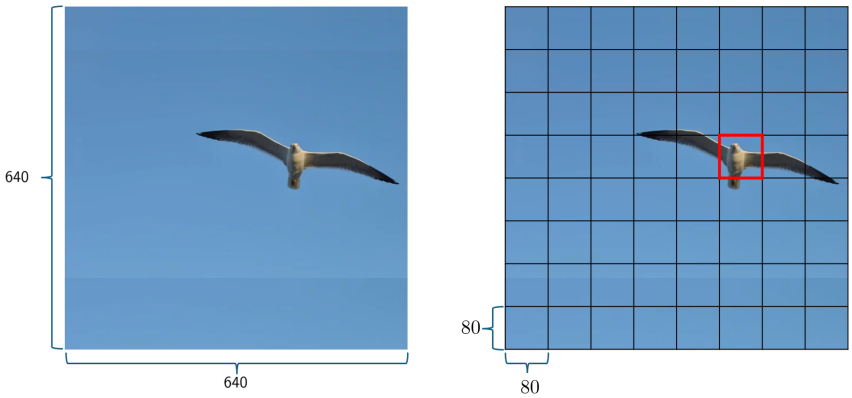


Figura 4.15 Imagem dividida em Grelha de 8×8. Imagem da Gaivota retirada de [7].

Em cada célula, são obtidas B previsões de caixas delimitadoras através de B previsores. O previsor fornece as informações geométricas das caixas delimitadoras, nomeadamente, a altura ($\hat{h}$), a largura ($\hat{w}$) e a posição da caixa delimitadora e ainda um valor de confiança associado à previsão. O previsor que assume a responsabilidade de prever a caixa delimitadora é o que gera a previsão da caixa delimitadora com maior valor de interseção sobre a união (IoU) entre a caixa delimitadora prevista e a caixa delimitadora real do objeto.

A função de perda é composta por:

- **Erro de localização:** penaliza as diferenças entre as coordenadas das caixas delimitadoras reais dos objetos e as coordenadas das caixas delimitadoras previstas;
- **Erro de confiança:** avalia o nível de confiança de que a caixa delimitadora prevista contém um objeto de interesse;

- **Erro de classificação:** verifica se o modelo classifica corretamente o objeto, tendo em conta a célula que possui o centro do objeto, entre as classes disponíveis.

A função de perda é dada por

$$\lambda_{coord}\sum_{a=0}^{S^2}\sum_{b=0}^{B}1_{ab}^{obj}\left[(x_a-\hat{x}_a)^2+(y_a-\hat{y}_a)^2\right]+\lambda_{coord}\sum_{a=0}^{S^2}\sum_{b=0}^{B}1_{ab}^{obj}\left[\left(\sqrt{w_a}-\sqrt{\hat{w}_a}\right)^2+\left(\sqrt{h_a}-\sqrt{\hat{h}_a}\right)^2\right]+$$

$$+\sum_{a=0}^{S^2}\sum_{b=0}^{B}1_{ab}^{obj}\left(C_a-\hat{C}_a\right)^2+\lambda_{noobj}\sum_{a=0}^{S^2}\sum_{b=0}^{B}1_{ab}^{noobj}\left(C_a-\hat{C}_a\right)^2+\sum_{a=0}^{S^2}1_{a}^{obj}\sum_{c\in\text{classes}}\left(p_a(c)-\hat{p}_a(c)\right)^2$$

onde: 1_a^{obj} indica se o objeto aparece na célula a e 1_{ab}^{obj} indica que o b-ésimo previsor da caixa delimitadora na célula a é o "responsável" por essa previsão. O valor λ_{noobj} é um hiperparâmetro que pondera a perda associada às previsões de confiança nas células que não contêm objetos. O valor deste hiperparâmetro é introduzido para dar menos peso às previsões incorretas dos objetos em células, nas quais não existe objeto e evita que as células sem objetos de interesse dominem o processo de treino do modelo YOLO. O valor de λ_{noobj} é usualmente ajustado para valores menores, de forma a reduzir o impacto dessas perdas. O $\hat{p}_a(c)$ corresponde à probabilidade prevista de que a classe c esteja presente no objeto da célula a. O $p_a(c)$ corresponde à probabilidade real (*ground truth*) da classe c para o objeto naquela célula. A função de perda penaliza a diferença entre a classe prevista e a classe verdadeira. Já o valor $\hat{C}_a$ corresponde à confiança prevista para a caixa delimitadora na célula a. Esta confiança indica a probabilidade de uma caixa delimitadora conter um objeto de interesse e a precisão da localização da caixa prevista, relativamente à localização da caixa verdadeira do objeto. O valor da confiança resulta do produto entre a probabilidade de um objeto estar presente na caixa e o valor de *IoU* (Interseção sobre a União) entre a caixa prevista e a caixa real, ou seja $P[\text{Objeto}] \times IoU(\text{caixa delimitadora prevista}; \text{caixa delimitadora real})$. Por fim, o valor C_a corresponde ao valor de confiança real para a célula a e indica se existe ou não um objeto nessa célula e a interseção sobre a união da caixa real com a caixa prevista. Se a célula contém um objeto, $C_a = 1$ e se a célula não contiver objeto, $C_a = 0$. [13]

A função de perda apenas penaliza o erro associado à classificação do objeto no caso deste se encontrar na célula da grelha e penaliza o erro nas coordenadas da caixa delimitadora, apenas se o previsor for o responsável pela caixa real, ou seja, se tiver um valor de IoU superior a todos os previsores daquela célula da grelha, que divide a imagem.

4.3.2 Transferência de Aprendizagem do modelo YOLOv8

A **transferência de aprendizagem** consiste numa técnica de aprendizagem automática, na qual se reaproveita um modelo pré-treinado para a realização de uma nova tarefa relacionada com um conjunto de dados de menor dimensão, ou com um conjunto de dados mais específico para a tarefa a realizar.

Com esta técnica não existe a necessidade de treinar um modelo de raiz, o que em tarefas de maior complexidade pode exigir uma grande quantidade de dados e um período de treino muito prolongado. A transferência de aprendizagem aproveita o conhecimento que o modelo já possuía para tarefas mais gerais, por este ser treinado com um conjunto de dados mais diversificado, e volta a ser treinado com um conjunto de dados mais específico, que utiliza o conhecimento base num contexto diferente.

Para o desenvolvimento da metodologia de deteção e reconhecimento automático de tabelas em documentos, foi necessária a utilização da técnica de transferência de aprendizagem, com a adaptação do modelo YOLOv8 desenvolvido para realizar a deteção de diferentes objetos, para a tarefa de deteção das tabelas em imagens. O aproveitamento da aprendizagem anteriormente obtida pelo modelo YOLOv8 permite otimizar o treino, por serem necessárias menos épocas para a obtenção de resultados satisfatórios, mesmo em casos onde os recursos computacionais são limitados.

4.4 Métricas de Desempenho do Modelo de Deteção de Objetos

Para analisar a aprendizagem do modelo ao longo do treino e o desempenho deste na deteção de objetos, são consideradas métricas de avaliação do desempenho do modelo [39], nomeadamente:

- **Exatidão** consiste proporção de previsões corretas feitas por um modelo em relação ao total de previsões, incluindo verdadeiros positivos e negativos, dada por

$$\text{Exatidão} = \frac{\text{VP} + \text{VN}}{\text{VP} + \text{FP} + \text{VN} + \text{FN}}$$

- **Precisão** mede a proporção de previsões positivas corretas em relação ao total de previsões positivas obtidas pelo modelo.

$$\text{Precisão} = \frac{\text{VP}}{\text{VP} + \text{FP}}$$

- **Sensibilidade** mede a proporção de casos positivos corretamente identificados pelo modelo em relação ao total de casos positivos verdadeiros.

$$\text{Sensibilidade} = \frac{\text{VP}}{\text{VP} + \text{FN}}$$

- **Medida F1** combina a precisão e a sensibilidade, dada por:

$$F1 = 2 \times \frac{\text{precisão} \times \text{sensibilidade}}{\text{precisão} + \text{sensibilidade}}$$

O valor varia entre 0 e 1, onde um valor igual a 1 indica o equilíbrio perfeito entre a precisão e a sensibilidade, útil em casos em que ocorre o desequilíbrio entre duas classes.

Onde:

- **VP** corresponde aos Verdadeiros Positivos, quando o modelo identifica corretamente a presença de um objeto, ou seja, o objeto está na imagem e foi corretamente detetado e localizado;
- **VN**, Verdadeiros Negativos, ocorrem quando o modelo identifica corretamente que não há objeto, ou seja, não existe objeto de interesse na imagem e o modelo não sinalizou incorretamente nada;
- **FP**, Falsos Positivos, ocorrem quando o modelo deteta incorretamente um objeto que não está presente na imagem;
- **FN**, Falsos Negativos, ocorrem quando o modelo falha na deteção de um objeto que está realmente presente na imagem, ou seja, o objeto está lá, mas o modelo não o detetou.

A medida de **interseção sobre a união**, IoU, consiste numa métrica que é utilizada em deteção de objetos e para avaliar a precisão das previsões de um modelo, através da comparação da área da caixa delimitadora prevista com a área da caixa delimitadora verdadeira, dada por

$$IoU = \frac{\text{Área da Interseção}}{\text{Área da União}}$$

Onde:

- Área da Interseção: área comum entre a caixa delimitadora prevista e a caixa delimitadora verdadeira do objeto;
- Área da União: área total da cobertura das duas caixas, ou seja, a soma das áreas das caixas delimitadoras prevista e real, com a subtração da interseção entre as duas.

Um valor de IoU igual a 1 significa que a sobreposição entre as caixas delimitadoras prevista e real foram perfeitas, enquanto que um valor de IoU igual a 0 significa que não há sobreposição entre as caixas delimitadoras prevista e verdadeira.

Capítulo 5

Metodologia

5.1 Introdução

A metodologia para a deteção e reconhecimento automático de tabelas em documentos PDF foi desenvolvida de forma a extrair uma tabela completa num documento estruturado em formato JSON.

Para alcançar o objetivo proposto, foi necessário seguir um conjunto de etapas, que incluem:

1. **Descarregamento dos Dados**
2. **Metodologia de Deteção de Tabelas:**
   (a) Pré-Processamento dos Dados
   (b) Treino de um modelo de Deteção de Objetos YOLOv8
   (c) Análise dos Resultados Obtidos
   (d) Previsão da região onde se encontra a tabela na imagem com o Modelo Treinado
3. **Metodologia de Reconhecimento da Estrutura das Tabelas Detetadas:**
   (a) Pré-Processamento dos Dados
   (b) Treino de modelos YOLO
      - Deteção dos Cabeçalhos;
      - Deteção da Altura das Linhas;
   (c) Análise de Resultados.
4. **Junção da Informação dos Cabeçalhos Detetados e das Alturas das Linhas Detetadas com as deteções de texto com o OCR**;
5. **Restrições de *Layout***;
6. **Extração da Informação das Tabelas para o Formato** JSON.

Deste modo, foi necessário realizar o processamento dos dados, o treino de modelos de deteção de objetos, a avaliação do desempenho dos modelos treinados e a definição de restrições de *layout* para a obtenção da estrutura da tabela.

O primeiro passo foi a análise das características e da diversidade das estruturas das tabelas presentes nos documentos `PDF` descarregados. De seguida, cada uma das páginas dos documentos `PDF` descarregados foi convertida numa imagem no formato `JPEG`, depois estas imagens foram importadas para o *Roboflow*, que permitiu realizar as anotações e redimensionar as imagens, para que fosse possível realizar o treino dos modelos de rede neuronal utilizados.

O treino dos modelos de redes neuronais foi uma das fases mais importantes para atingir o funcionamento do algoritmo implementado. Recorreu-se a modelos de rede neuronal YOLOv8, por terem sido considerados mais eficazes que outros em fases preliminares do projeto, face aos recursos computacionais disponíveis.

Após o treino dos modelos de redes neuronais, foi necessário aplicar restrições de *layout* que se adequassem às tabelas de diferentes tipos, presentes nos dados, de forma a garantir que a etapa do reconhecimento da estrutura das tabelas funcione de forma automática e rigorosa. Serão abordadas as especificações da máquina virtual utilizada para o treino do modelo, as etapas de pré-processamento das imagens e dos dados, o treino dos modelos e a escolha das redes neuronais e das operações adicionais realizadas.

Para o desenvolvimento da metodologia de Deteção de Tabelas foi desenvolvido um modelo *Faster R-CNN*, com o qual não foi possível alcançar os resultados pretendidos, devido ao facto de se tratar de uma adaptação da implementação de uma rede neuronal `ResNet-50`, pré-treinada para a classificação das imagens inteiras. O facto de adaptar um modelo treinado para a classificação de imagens completas para um modelo de deteção e classificação de objetos, *Faster R-CNN* através da remoção das duas últimas camadas convolucionais, a adição de uma RPN (Rede Neuronal de Proposta de Região) e de um gerador de âncoras comprometeu a eficiência do treino do modelo, uma vez que este ficou muito pesado computacionalmente.

A eficiência do treino do modelo é muito importante para que seja possível concretizar os objetivos propostos. Após uma avaliação detalhada das limitações enfrentadas, foram exploradas alternativas que pudessem oferecer um equilíbrio mais favorável entre o tempo de treino e o desempenho do modelo. Neste contexto, foi selecionado o modelo YOLO, amplamente reconhecido pela sua rapidez e eficiência em tarefas de deteção de objetos em tempo real. Esta metodologia demonstrou ser mais robusta e eficaz para este problema, face aos recursos computacionais disponíveis.

5.2 Contexto e Fonte dos Dados

O conjunto de dados utilizado para o desenvolvimento do algoritmo de reconhecimento automático de tabelas em documentos foi descarregado da plataforma Evalyze, que possui múltiplos formatos de tabelas de diversos fornecedores, de acordo com a Figura 5.1. Os dados consistem em transações registadas, que constituem um total de 3900 páginas de documentos PDF. Estes documentos representam guias de remessa que contêm um conjunto de produtos provenientes de várias empresas.

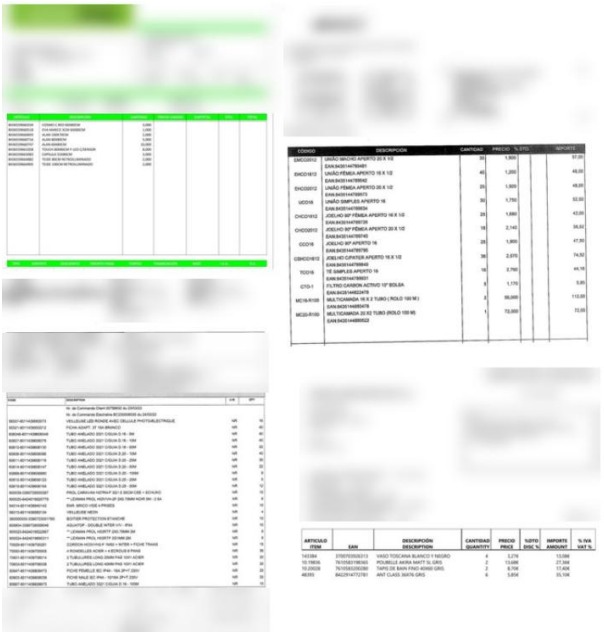


Figura 5.1 Exemplos de Tabelas nos Dados.

5.3 Composição e Estrutura dos Dados

Os documentos PDF que constituem o conjunto de dados representam guias de remessa de diversas empresas. Estes documentos possuem informações como os dados do remetente e do destinatário, a descrição dos produtos, a quantidade movimentada de cada produto e os seus valores, que variam consoante o fornecedor. A metodologia desenvolvida possibilita a deteção e o reconhecimento da estrutura de tabelas, de uma forma ajustável aos diferentes fornecedores.

5.3.1 Limitações do Conjunto de Dados

O conjunto de dados apresenta algumas caraterísticas que podem resultar em dificuldades na deteção automática das tabelas e no reconhecimento da sua estrutura. Estas características consistem no facto de os dados dependerem do contexto empresarial e de existir a possibilidade dos registos não estarem completos. Em alguns documentos ocorreram rotações, a fraca qualidade de alguns documentos, que resulta na difícil perceção do que se

encontra escrito, a ocorrência de erros de digitalização e, por fim, a grande diversidade dos fornecedores que faz com que a nomenclatura das colunas varie de fornecedor para fornecedor.

5.4 Recursos Computacionais Utilizados

5.4.1 Máquina Virtual

A Máquina Virtual disponibilizada pela empresa é uma instância do tipo Standard_NC8as_T4_v3. Esta instância está equipada com as seguintes especificações:

- **Processador:** 8 núcleos
- **Memória RAM:** 56 GB
- **Armazenamento em disco:** 352 GB
- **Arquitetura do Sistema:** Sistema operativo de 64 bits, processador baseado em x64
- **Acesso à Rede:** Integrado com recursos de nuvem do Microsoft Azure.

As configurações específicas podem ser encontradas em [24], que contém a documentação oficial do Microsoft Azure. Esta máquina virtual foi adquirida pela empresa no decorrer do estágio e tem um custo de US$ $0,94$ por hora. Foi apenas utilizada para o treino dos modelos YOLO, porque requeriam GPU e, sem a máquina virtual fornecida, não seria possível alcançar os resultados pretendidos.

5.4.2 Computador da Empresa

O computador pessoal utilizado na empresa possui as seguintes especificações técnicas:

- **Processador:** 11th Gen Intel(R) Core(TM) i5-1135G7, que opera a 2.40 GHz até 2.42 GHz
- **Memória RAM:** $16,0$ GB ($15,7$ GB utilizável)
- **Armazenamento em Disco:** 475 GB
- **Arquitetura do Sistema:** Sistema operativo de 64 bits, processador baseado em x64

As especificações do equipamento apresentadas são as que foram utilizadas para as tarefas diárias, no decorrer do estágio na empresa.

5.5 Treino do Modelo de Deteção de Tabelas

Para o desenvolvimento de um algoritmo de deteção de tabelas em documentos de forma automática, é necessário realizar etapas de pré-processamento dos dados que permitam o treino de um modelo de deteção de objetos YOLOv8.

O pré-processamento de dados foi realizado em várias etapas para garantir que o formato e a apresentação dos dados fossem adequados para o treino do modelo da rede neuronal e para que as imagens pudessem ser previstas pelo modelo treinado.

Estas etapas, presentes no ficheiro `preprocessamento.py` do repositório GitHub [12]:

- Conversão das páginas do `PDF` em imagens;
- Remoção de imagens que não possuem tabelas do conjunto de dados com que a rede neuronal vai ser treinada;
- Redimensionamento, recorte e rotação de imagens do conjunto de dados;
- Importação das imagens para o *Roboflow*;
- Anotação das regiões onde se encontram as tabelas nas imagens;
- Descarregamento dos dados no formato YOLOv8.

5.5.1 Extração e Manipulação das Imagens

Inicialmente, os documentos `PDF` descarregados são convertidos em imagens `JPEG`, com o auxílio da biblioteca `pdf2image`. Esta etapa de conversão do formato dos dados vai permitir a construção de uma rede neuronal treinada para a localização das tabelas contidas nas páginas, onde cada página do documento `PDF` é tratada como uma imagem.

5.5.2 Preparação das Diretorias

Para organizar as imagens resultantes do programa em *Python* anteriormente referido, foi estabelecido um sistema de diretorias, que utiliza uma função que cria diretorias, caso estas não existam previamente, o que permite guardar os dados no decorrer do seu pré-processamento.

5.5.3 Ajustes e Padronização das Imagens

Após a conversão das páginas de `PDF` para `JPEG`, algumas imagens precisaram de sofrer rotação, devido à forma como os documentos foram digitalizados. Estes erros de digitalização poderiam dificultar a aprendizagem do modelo. Neste sentido, com o auxílio de um programa em *Python*, algumas destas, como as que estão representadas na Figura 5.2 foram rodadas em 90, 180 e 270 graus, para que ficassem na vertical. Foi também necessário o recorte de um lado das imagens, Figura 5.3 e a eliminação de imagens que não contêm tabelas.

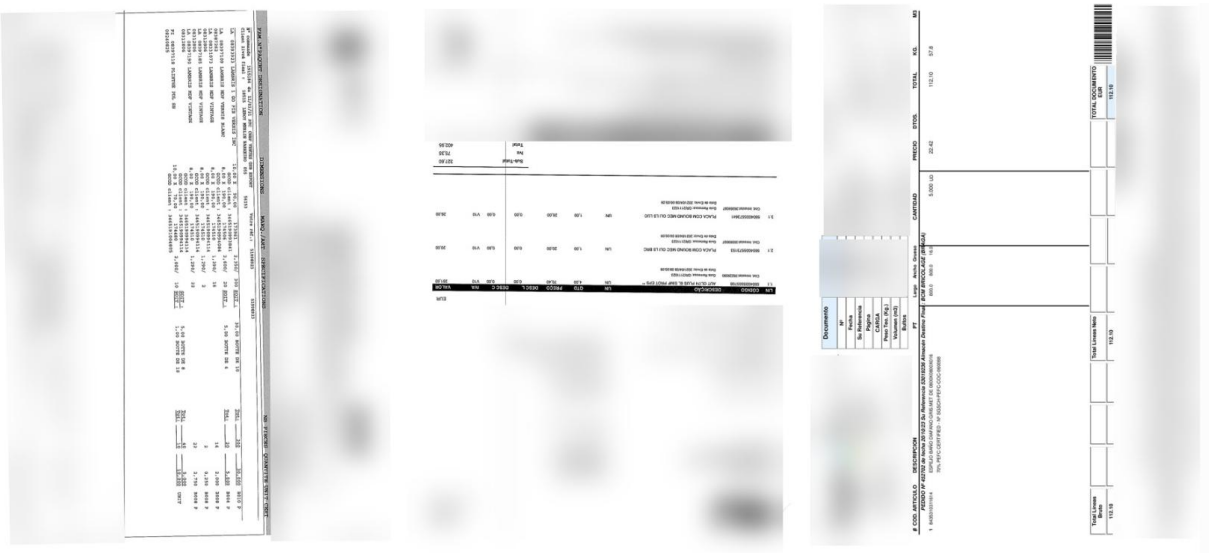


Figura 5.2 Exemplos de Imagens que Sofreram Rotação

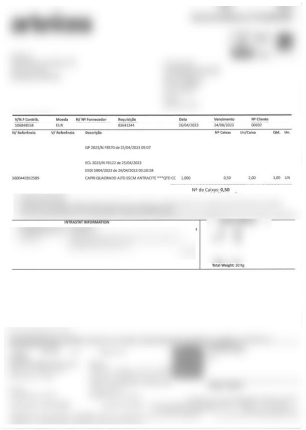


Figura 5.3 Fornecedor no qual foi necessário o Recorte da Imagem.

5.5.4 Anotação da Região de Interesse nas Imagens

Após as técnicas de pré-processamento dos dados referidas anteriormente, as imagens foram importadas para o *Roboflow*, que consiste numa plataforma que auxilia no processo de construção do conjunto de dados, com o qual se treinam as redes neuronais e na qual também é possível treinar modelos de visão computacional, por oferecer ferramentas que permitem importar, anotar e realizar pré-processamento de imagens, com opção de *data augmentation*, através de diversas operações, como a rotação e a translação, alterações nas dimensões das imagens, a anotação dos objetos de interesse e a exportação dos dados anotados em vários formatos, compatíveis com diferentes implementações de modelos de aprendizagem automática.

No tratamento dos dados fornecidos, os documentos foram inicialmente convertidos de `PDF` para `JPEG`, com o auxílio do programa em *Python* anteriormente descrito. De seguida, as imagens resultantes foram importadas para o *Roboflow*, nas quais foram realizadas manu-

almente as anotações precisas das caixas que delimitam a região que corresponde à tabela principal, na qual se encontram a lista de produtos e as respetivas quantidades. As anotações permitem que o modelo de deteção de objetos aprenda a localização das tabelas nas imagens.

Após a realização de todas as anotações necessárias, as imagens foram convertidas para dimensões de 640×640 pixels, que são as adequadas para o modelo YOLOv8, com a opção de redimensionamento *Fit with (white edges) in* disponível no *Roboflow*, que ajusta as dimensões das imagens originais para as especificadas, de forma a garantir que a proporção original se mantenha. Assim, a imagem permanece visível no seu todo, com as proporções originais. Por exemplo, se a imagem original tiver dimensões 1040×2080 e a opção de redimensionar a imagem para 640×640, a maior dimensão (2080) será reduzida para 640 pixels, e a outra será ajustada proporcionalmente, o que resulta numa imagem de 320×640 pixels, na qual as bordas da largura restante serão preenchidas com branco para que a imagem fique centrada e com dimensões 640×640 pixels.

Após a anotação e o redimensionamento das imagens para o formato adequado ao modelo YOLOv8, realizou-se a divisão do conjunto de imagens e anotações em subconjuntos de treino, teste e validação, no qual 70% das imagens pertencem ao conjunto de treino $(2413$ imagens), 20% pertencem ao conjunto de validação $(687$ imagens) e 10% pertencem ao conjunto de teste $(339$ imagens). Procedeu-se à exportação das imagens anotadas. A escolha do formato de exportação destes dados deve estar relacionada com o tipo de modelo de rede neuronal que se irá treinar.

Os formatos para a exportação de dados que o *Roboflow* possui permitem que a adaptação das imagens anotadas seja mais eficiente, de forma a dar entrada em vários modelos de aprendizagem automática, sem a necessidade de realizar alterações muito significativas no formato dos dados, no decorrer da etapa de pré-processamento.

Visto que o objetivo é treinar o modelo YOLOv8, os dados foram exportados do *Roboflow*, no formato YOLOv8, no qual as coordenadas são da forma $[x, y, w, h]$, onde x e y correspondem às coordenadas do centro, w corresponde à largura e h à altura da caixa delimitadora das tabelas, de acordo com a Figura 5.4.

5.5.5 Estrutura das Anotações YOLOv8

No formato YOLOv8, cada anotação é representada num ficheiro `TXT`, que corresponde a cada uma das imagens, em `JPEG`, no qual cada linha do ficheiro `TXT` descreve um objeto de interesse. Os elementos de uma linha de anotação incluem:

- **Classe do Objeto:** Um número inteiro que representa a classe do objeto (neste caso, '0' para a classe 'TABLE');
- **Coordenadas do Centro do Objeto:** As coordenadas x e y do centro da caixa delimitadora, normalizadas em relação às dimensões da imagem (resultam em valores compreendidos entre 0 e 1);

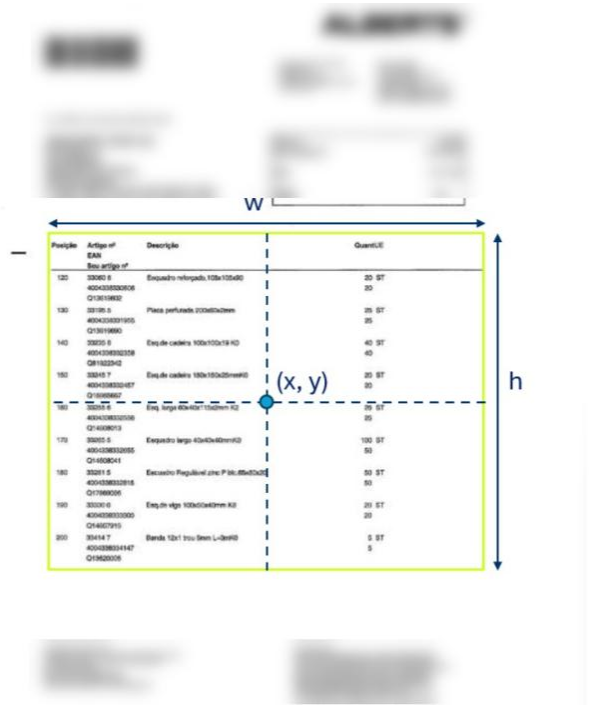


Figura 5.4 Caixas Delimitadoras das tabelas no formato YOLOv8.

- **Dimensões da Caixa Delimitadora:** Largura e altura da caixa delimitadora, também normalizadas em relação às dimensões da imagem.

A forma como as anotações e as imagens se encontram organizadas neste formato facilita o processamento e o acesso aos dados, durante o treino dos modelos, devido às seguintes caraterísticas:

- **Nomes Correspondentes:** O nome do ficheiro da anotação `TXT` é exatamente igual ao nome do ficheiro da imagem `JPEG`, nos quais apenas diferem as extensões. Por exemplo, se uma imagem tiver o nome `documento1.JPEG`, a anotação correspondente será `documento1.TXT`;
- **Localização das Anotações:** Dentro de cada ficheiro `TXT`, as anotações são listadas e as caixas delimitadoras e as classes dos objetos identificados na imagem correspondente são descritas.

Esta organização de dados é vantajosa devido à:

- **Correspondência Direta:** A correspondência entre as imagens e os ficheiros de anotações simplifica a utilização dos dados, especialmente perante grandes volumes de dados, garante que cada imagem possa ser facilmente associada à sua anotação respetiva;
- **Facilidade de Acesso e Processamento:** Manter as anotações em ficheiros separados, mas com nomes correspondentes às imagens, permite que os processos de treino e de validação de modelos de visão computacional sejam mais organizados e eficientes.

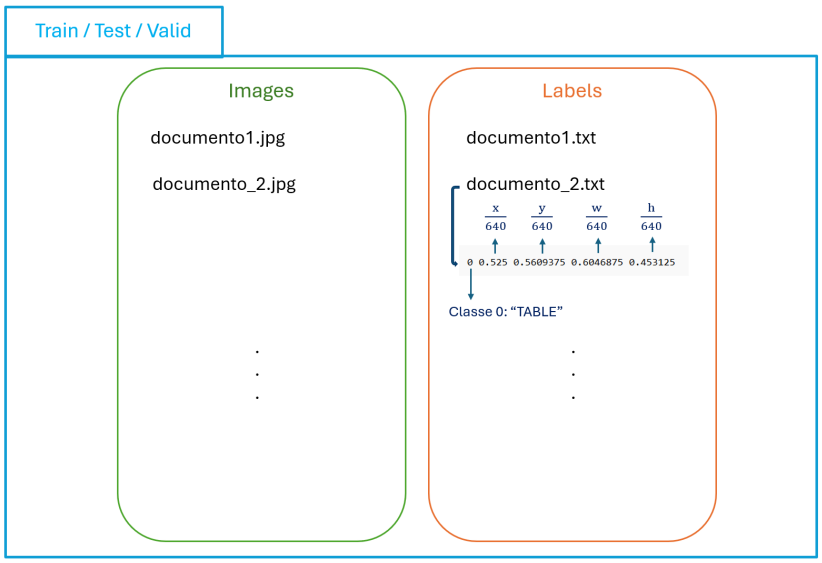


Figura 5.5 Correspondência entre as imagens e as anotações.

O formato deste conjunto de dados permite a identificação das tabelas em documentos, através da análise e do processamento automático de dados.

Os recursos do computador da empresa não permitem a realização do treino do modelo de deteção de objetos YOLOv8 com o conjunto de dados, por limitação na configuração da GPU. Assim, foi necessário pedir à Evalyze uma máquina virtual com GPU, que permitisse o treino do modelo da rede neuronal.

A máquina virtual tinha um custo associado elevado e, por isso, o número de imagens utilizado para o treino do modelo da rede neuronal foi reduzido a 40% do número de imagens original, presente na pasta `all_images_yolov8__`, que consiste na divisão do conjunto de treino em 70% treino, 20% validação e 10% teste. Assim, o novo conjunto de dados mais pequeno foi construído com o código `novo_dataset_mais_pequeno.py` [12], da seguinte forma:

1. **Conjunto de Treino:** Escolha aleatória de 40% das imagens do conjunto de treino da pasta `all_images_yolov8__`, com as caixas delimitadoras das tabelas correspondentes a estas imagens;

2. **Conjunto de Validação:** Escolha aleatória de 40% das imagens do conjunto de validação e das caixas delimitadoras destas imagens;

3. **Conjunto de Teste:** Escolha aleatória de 40% das imagens do conjunto de teste, em conjunto com as caixas delimitadoras das tabelas presentes nas imagens correspondentes.

Foi realizada uma estratégia de *fine tunning* de uma rede neuronal YOLOv8 com um conjunto de dados que corresponde a imagens de documentos PDF com tabelas de produtos de clientes, com dimensões 640 por 640 com a operação de *fit within (White edges)*, que mantém as proporções da imagem original, que resultou da conversão de uma página de PDF para JPEG.

O conjunto com o qual o modelo foi treinado possui 1026 imagens no conjunto de treino, 292 no conjunto de validação e 143 no conjunto de teste.

As imagens foram anotadas manualmente e os dados das imagens e das respetivas anotações foram descarregados no formato YOLOv8.

As imagens e as anotações possuem o mesmo nome de ficheiro e os ficheiros com as anotações em `TXT` contêm o nome da classe e as coordenadas centrais que correspondem ao centro do objeto, a altura e a largura do mesmo.

5.5.6 Treino do Modelo de Deteção de Tabelas - YOLOv8 por 100 épocas

O modelo `YOLOv8x` foi treinado por 2 horas e 40 minutos, com o conjunto de treino e de validação por 100 épocas, `treino_det_tabelas_100_epocas_yolov8x.ipynb` [12]. O treino do modelo foi realizado localmente, num *software* que utiliza GPU. Após várias tentativas, devido ao grande volume de dados, os requisitos computacionais para o treino do modelo excederam a capacidade do ambiente local.

Foi necessário importar os dados que foram descarregados do *Roboflow* e importar um ficheiro do tipo `YAML`, `data_small_colab.yaml` [12], que contém a diretoria das pastas nas quais se encontra o conjunto de treino e de validação a utilizar para a realização do treino do modelo.

O *output* do treino do modelo consiste numa pasta `runs` na qual são guardados os parâmetros do melhor modelo obtido `best.pt` e do último modelo treinado `last.pt` e os gráficos com os valores de medidas de desempenho do modelo, ao longo das épocas do treino.

- **Valor das Funções de Perda ao longo das épocas de treino do modelo**

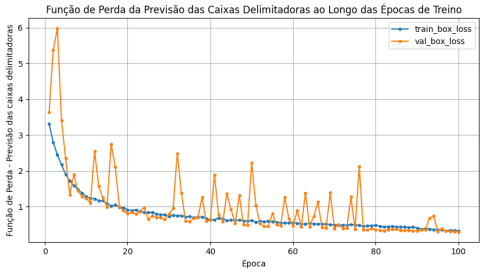


Figura 5.6 Função de Perda associada às previsões das caixas delimitadoras.

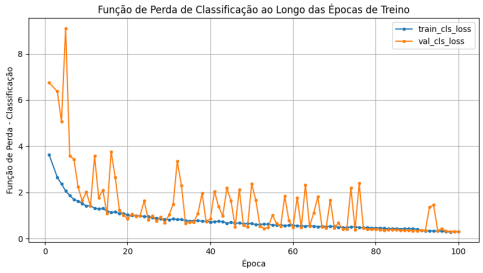


Figura 5.7 Função de Perda associada à classificação dos objetos no interior das caixas delimitadoras.

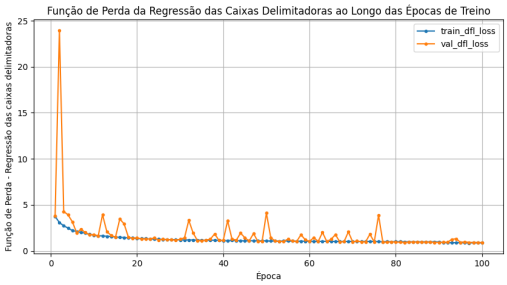


Figura 5.8 Função de Perda associada à regressão das caixas delimitadoras.

A função de perda da previsão de caixas delimitadoras (Figura 5.6), da regressão da classificação do objeto que se encontra no interior da caixa delimitadora (Figura 5.7) e da regressão das caixas delimitadoras (Figura 5.8) apresentam um comportamento semelhante, no qual o conjunto de treino (linha azul) diminui de uma forma consistente, indicando que o modelo está a aprender bem ao longo do seu treino, tendo ocorrido um decréscimo acentuado dos valores de perda nas primeiras épocas de treino do modelo.

Quanto ao conjunto de validação (linha cor de laranja), também ocorreu um decréscimo rápido dos valores da perda nas primeiras épocas de treino do modelo e o gráfico mostra uma diminuição dos valores de perda que indica que o modelo se está a ajustar aos dados de validação, mas com oscilações consequentes do cálculo numérico do gradiente da função de perda, no decorrer do processo de minimização. A instabilidade relativa às curvas do conjunto de validação, que se verifica ao longo das épocas de treino do modelo, pode indicar que o conjunto de validação possui imagens que são significativamente diferentes em relação às imagens do conjunto de treino. O modelo pode estar a ter dificuldades em generalizar para estes fornecedores, o que pode ser a causa das oscilações verificadas nos valores de perda de validação.

- **Curvas de F1-Score e de Precisão-recall**

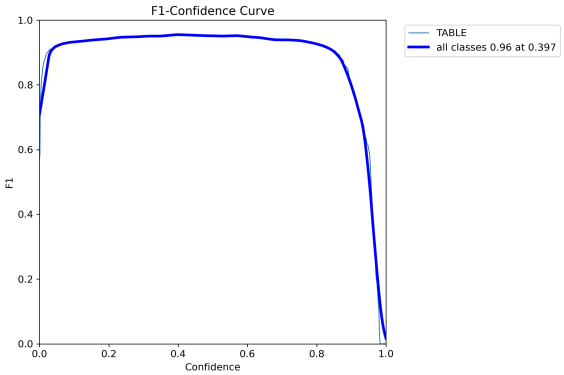


Figura 5.9 Curva de F1-Score para a Deteção de Tabelas.

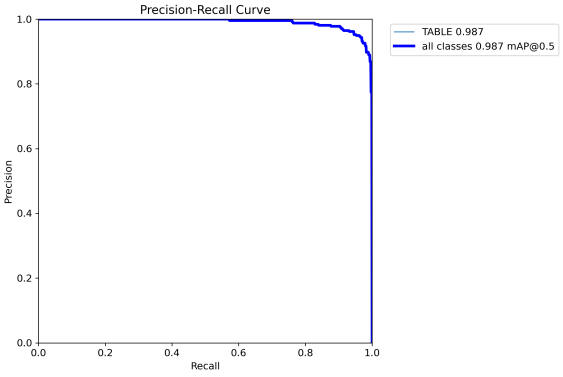


Figura 5.10 Curva Precisão-Recall para a Análise do Desempenho do Modelo para Cada Classe.

A curva de *F1-Score* e Confiança, presente na Figura 5.9 indica que o *F1-Score* permanece com valores baixos e começa a aumentar drasticamente, à medida que a confiança ultrapassa o valor de aproximadamente 0.3. Atinge rapidamente um patamar próximo de 1.0, o que indica que o modelo, quando se considera um limiar de confiança adequado, é muito preciso na deteção de tabelas.

A curva Precisão-*Recall* representada pela Figura 5.10 indica que a precisão é muito próxima de 1.0 para a maior parte dos valores de *recall*, exceto no canto superior direito, onde o valor de *recall* é igual a 1.0, e a precisão diminui ligeiramente, ou seja, segundo o gráfico o modelo mantém uma alta precisão, mesmo quando tenta capturar todos os casos positivos (valor de *recall* igual a 1.0), com um comprometimento muito pequeno na precisão, no valor de *recall* de 1.0.

5.5.7 Treino do Modelo de Deteção de Tabelas - YOLOv8 por 250 épocas

O modelo `YOLOv8x` foi treinado por 6 horas e 18 minutos, com o conjunto de treino e de validação por 250 épocas. O treino do modelo exigiu uma quantidade significativa de recursos computacionais, que incluiu o processamento com GPU. [24] Estes requisitos excederam a capacidade do ambiente local, devido ao elevado tempo de execução. Por este motivo, foi necessária a utilização de uma máquina virtual equipada com GPU que permitisse a realização do treino do modelo YOLO por 250 épocas.

O procedimento realizado para o treino do modelo `YOLOv8x` por 250 épocas, `treino_dos_modelos_com_maquina_virtual.ipynb` [12], foi realizado de forma análoga ao de 100 épocas, com a importação dos dados e do ficheiro com as diretorias do tipo `YAML`, `data_small_dataset_2.` [12].

Assim como no modelo treinado para 100 épocas, o *output* consiste numa pasta `runs` na qual são guardados os parâmetros do melhor modelo obtido `best.pt` e do último modelo treinado `last.pt` e os gráficos com os valores de medidas de desempenho do modelo, ao longo das épocas do treino do modelo que contém os seguintes gráficos:

- **Valor das Funções de Perda ao longo das épocas de treino do modelo**

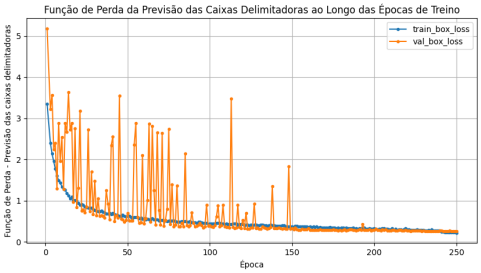


Figura 5.11 Função de Perda associada às previsões das caixas delimitadoras.

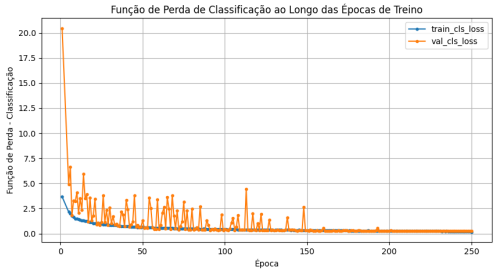


Figura 5.12 Função de Perda associada à classificação dos objetos no interior das caixas delimitadoras.

O valor da função de perda presente no gráfico da Figura 5.11 começa por ser elevado, mas reduz significativamente ao longo das épocas de treino do modelo. No conjunto de treino, os valores da função de perda estabilizam-se. No entanto, no conjunto de validação, os valores da função de perda apresentam flutuações significativas até à época 150 do treino do modelo. A análise deste gráfico indica que o modelo tem uma boa capacidade de aprendizagem da previsão das caixas delimitadoras.

Pela observação dos valores que a função de perda assume ao longo das épocas de treino do modelo presente na Figura 5.12, existe um decréscimo rápido destes valores no conjunto de treino e este estabiliza. No conjunto de validação observam-se flutuações consideráveis que aparentam diminuir com o número de épocas treinadas.

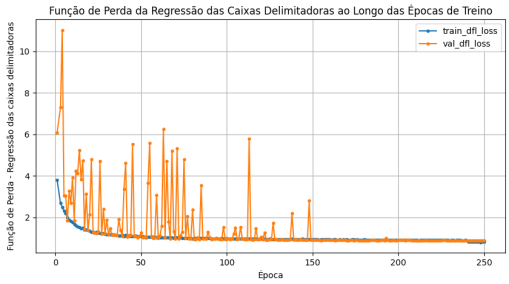


Figura 5.13 Função de Perda associada à regressão das caixas delimitadoras.

Pela observação das curvas relativas aos valores da função de perda associada à regressão das caixas delimitadoras do conjunto de treino e do conjunto de validação ao longo do treino do modelo, presentes no gráfico 5.13, os valores da função de perda são elevados nas primeiras épocas do treino do modelo e existe um decréscimo significativo nas primeiras 150 épocas. O conjunto de validação apresenta oscilações muito significativas ao longo das épocas, maiores do que as que se encontram na função de perda relativa à classificação dos objetos no interior das caixas delimitadoras. Estas características sugerem que o modelo está a aprender a ajustar as caixas delimitadoras. As oscilações podem dever-se ao facto de o modelo estar a ser submetido a formatos de tabelas no conjunto de validação que não estavam presentes no conjunto de treino. No entanto, a partir da época 150, as oscilações deixam de ser muito significativas.

Pela observação dos gráficos 5.11, 5.12 e 5.13, o modelo parece estar a aprender de uma forma eficaz, uma vez que os valores das funções de perda decrescem com o aumento do número de épocas de treino do modelo.

As oscilações significativas verificadas nos valores das funções de perda no conjunto de validação podem estar relacionadas com a grande diversidade de formatos de tabelas entre os diferentes fornecedores. Embora as imagens de um mesmo fornecedor mantenham um formato similar, existem diferenças muito significativas a nível da estrutura, da configuração e da apresentação entre os fornecedores.

Existem quatro fornecedores no conjunto de validação que não constam no conjunto de treino. Adicionalmente, existem fornecedores que, apesar de estarem presentes no conjunto de treino, possuem um número muito limitado de imagens, o que pode não ser suficiente para que o modelo aprenda de uma forma abrangente.

Estas características que os conjuntos de treino e de validação utilizados para o treino do modelo apresentam podem justificar a instabilidade observada nos valores de perda no conjunto de validação nas primeiras 150 épocas de treino do modelo.

- **Curvas de F1-Score e de Precisão-recall**

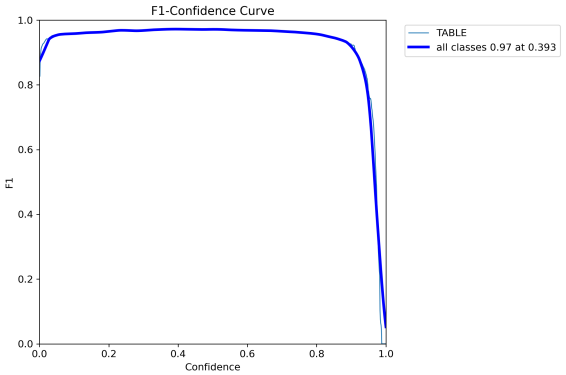


Figura 5.14 Curva de F1-Score para a Deteção de Tabelas.

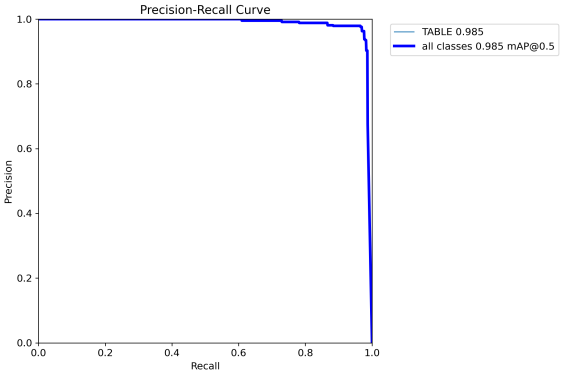


Figura 5.15 Curva Precisão-Recall para a Análise do Desempenho do Modelo para Cada Classe.

O gráfico de *F1-Score* (Figura 5.14) revela um valor de *F1-Score* elevado. Este é um indicador de que o modelo possui um bom equilíbrio entre a precisão e o *recall*. Esta caraterística de um modelo é relevante em casos em que as deteções de verdadeiros positivos e de verdadeiros negativos são importantes.

A curva verificada no gráfico de precisão-*recall* da Figura 5.15 representa a relação entre a precisão e o *recall* para a classe "`TABLE`". Esta curva encontra-se muito próxima do canto superior direito e o valor de mAP é de 0.985, o que significa que o modelo YOLO treinado possui uma boa capacidade de prever e detetar as tabelas nas imagens. A curva geral encontra-se sobreposta à coluna da classe "`TABLE`" por esta ser a única considerada pelo modelo, pelo que as curvas são iguais.

5.5.8 Treino do Modelo de Deteção de Tabelas - YOLOv9, 250 épocas

O procedimento realizado para o treino do modelo `YOLOv9c`, cujo código se encontra no ficheiro `yolov9_table_detection_.ipynb` e o ficheiro `YAML` corresponde a `data_small_dataset_2.yaml`, em [12], foi análogo ao que se aplicou nos modelos `YOLOv8x` e a pasta que resulta do treino do modelo possui as mesmas informações que o modelo `YOLOv8x`, que contêm os seguintes gráficos:

- **Valor das Funções de Perda ao longo das épocas de treino do modelo**

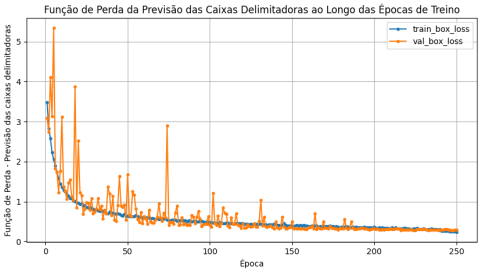


Figura 5.16 Função de Perda associada às previsões das caixas delimitadoras.

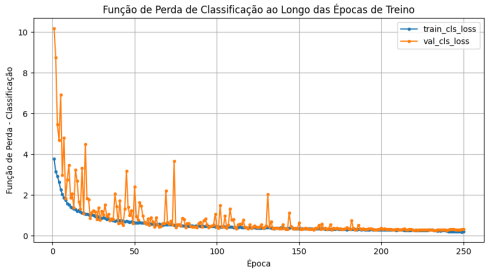


Figura 5.17 Função de Perda associada à classificação dos objetos no interior das caixas delimitadoras.

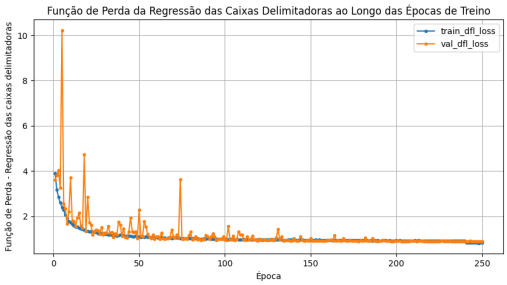


Figura 5.18 Função de Perda associada à regressão das caixas delimitadoras.

Os gráficos das Figuras 5.16, 5.17 e 5.18 das funções de perda relativas à previsão das caixas delimitadoras, classificação do objeto no interior da caixa delimitadora e à regressão das caixas delimitadoras, respetivamente, apresentam quedas progressivas destes valores no conjunto de dados de treino (curva azul), o que indica que o modelo está a aprender no conjunto de treino. Quanto ao conjunto de dados de validação (laranja), o modelo apresenta instabilidade nas curvas, o que reflete que o modelo tem dificuldade em generalizar a capacidade de deteção para o conjunto de validação, que pode ser devido ao facto de as imagens do conjunto de dados de validação ser muito diferente das do conjunto de dados de treino.

- **Curvas de F1-Score e de Precisão-recall**

O gráfico de F1-*Score*-Confiança 5.19 mostra que o modelo atinge um ótimo equilíbrio entre precisão e sensibilidade num nível de confiança moderado, de aproximada-

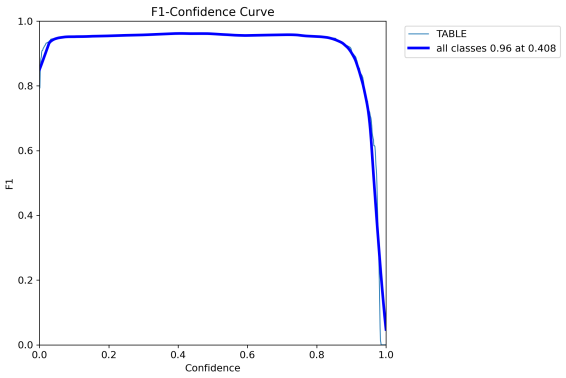


Figura 5.19 Curva de F1-Score para a Deteção de Tabelas.

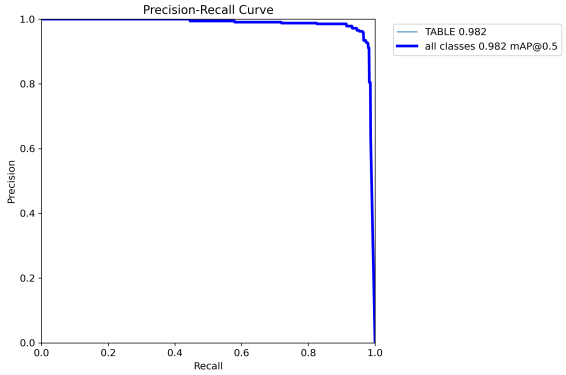


Figura 5.20 Curva Precisão-*Recall* para a Análise do Desempenho do Modelo para Cada Classe.

mente 0.4. No entanto, quando o nível de confiança passa a ser mais elevado, o valor de *F1-Score* começa a diminuir e indica que o modelo se torna muito conservador, o que prejudica o equilíbrio entre a precisão e o *recall*.

A Curva Precisão-*Recall* 5.20 indica que o modelo consegue manter valores elevados de precisão e de *recall* em simultâneo na maior parte dos limiares de decisão, de acordo com o que é indicado pelo valor de 0.982 para mAP0.5, o que demonstra um desempenho bem equilibrado na deteção e na classificação de objetos.

5.5.9 Análise dos Resultados Obtidos nas Previsões das Regiões das Tabelas nas Imagens do Conjunto de Teste

Após o treino do modelo YOLO para detetar tabelas nas imagens e da análise das métricas de desempenho ao longo do treino do modelo, no conjunto de treino e de validação, foi necessário avaliar o desempenho deste modelo treinado no conjunto de dados de teste, para que fosse possível entender se este tem uma boa capacidade de generalização para novos dados. Para possibilitar a análise, foram realizados os seguintes gráficos para cada um dos modelos YOLO de Deteção de Tabelas treinados:

- Gráfico circular com as proporções das previsões de cada uma das imagens do conjunto de treino com valores de IoU em três categorias:
  - **Verde:** $0.9 \leq IoU \leq 1$;
  - **Amarelo:** $0.6 \leq IoU < 0.9$;
  - **Vermelho:** $0 \leq IoU < 0.6$.
- Gráfico de barras com a média dos valores de IoU obtidos com as previsões do modelo no conjunto de treino, para cada um dos fornecedores, em três categorias:
  - **Verde:** $0.9 \leq IoU \leq 1$;
  - **Amarelo:** $0.6 \leq IoU < 0.9$;
  - **Vermelho:** $0 \leq IoU < 0.6$.

A análise destes gráficos permite verificar se a aprendizagem adquirida pelo modelo YOLO no final das épocas de treino permite obter bons resultados nas previsões dos novos dados, presentes no conjunto de teste.

A análise permite ainda distinguir os fornecedores que obtiveram resultados satisfatórios dos que não alcançaram os resultados pretendidos. Deste modo, é possível identificar os fornecedores com os quais se definiria um novo conjunto de dados de treino, com o qual se poderia enriquecer a aprendizagem do modelo. Para além disso, é possível ter uma noção prévia da forma como o modelo se irá comportar na previsão de imagens dos diferentes fornecedores.

Para além da construção destes gráficos, foi determinado o tempo médio da previsão das imagens do conjunto de teste, para cada um dos modelos de deteção de objetos YOLO treinados.

- **Modelo YOLOv8 treinado por** 100 **épocas**

  Pela observação do gráfico circular, o modelo YOLOv8 treinado por 100 épocas detetou tabelas com uma medida de IoU superior a 0.9 em 308 imagens (86%), 37 imagens (10.3%) com um valor de IoU entre 0.6 e 0.9 e 9 (2.5%) imagens com um valor de IoU inferior a 0.6. O modelo não obteve previsão para a localização de tabelas em 4 das 358 imagens do conjunto de teste (1.1%).

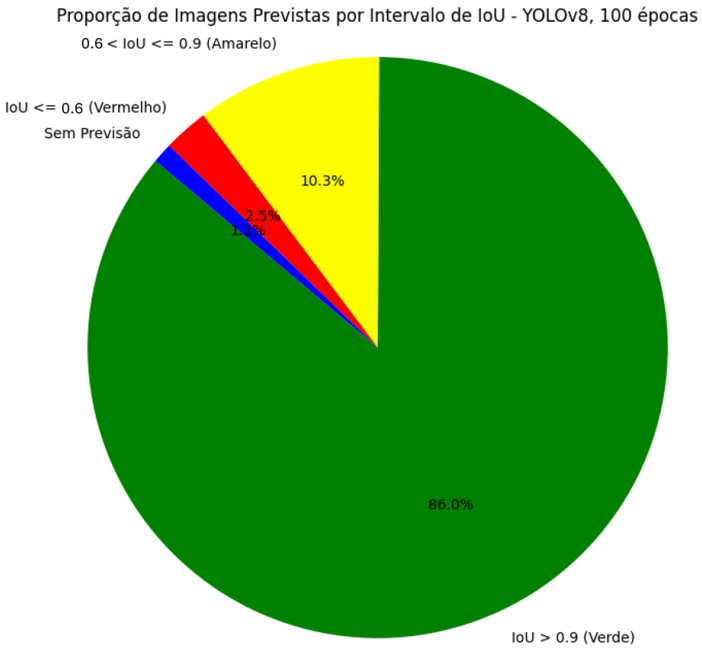


Figura 5.21 Gráfico circular com a categoria de IoU de todas as previsões obtidas pelo modelo.

Pela observação deste gráfico, o modelo parece ter uma boa capacidade de generalização para os novos dados.

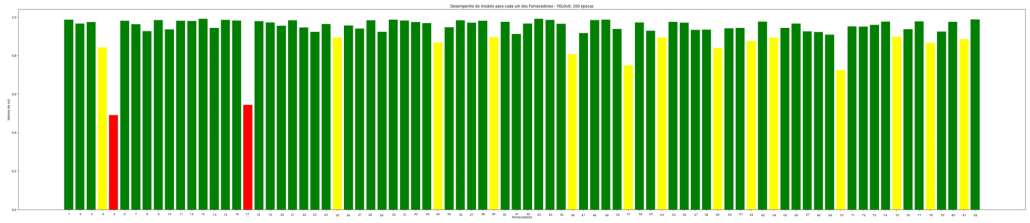


Figura 5.22 Gráfico de barras com a categoria de IoU das previsões obtidas pelo modelo YOLOv8 treinado por 100 épocas para cada um dos fornecedor.

O gráfico de barras relativo às categorias de IoU das previsões do modelo para cada fornecedor demonstra que os fornecedores **5** e **17** obtiveram um valor de IoU baixo, inferior a 0.6. Os restantes fornecedores possuem uma média de IoU superior a 0.6, que indica que o modelo deteta as tabelas de uma forma razoável.

Tempo médio de previsão para uma imagem no conjunto de teste: 2.11 segundos.

- **Modelo YOLOv8 treinado por 250 épocas**

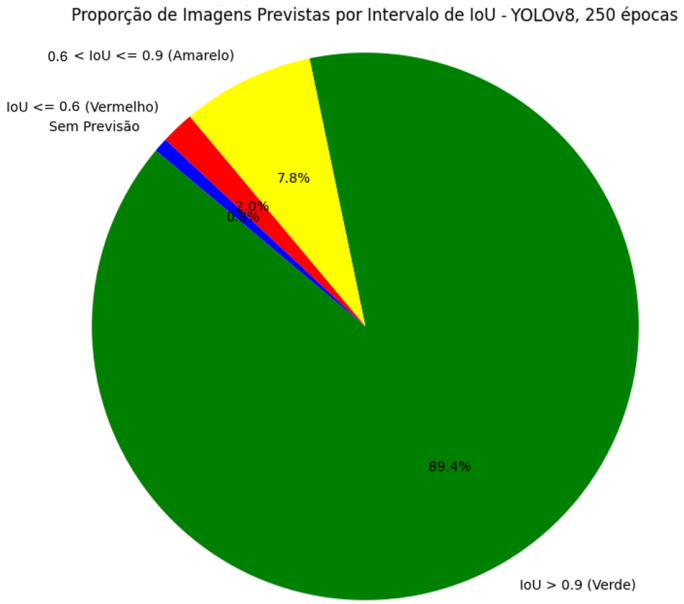


Figura 5.23 Gráfico circular com a categoria de IoU de todas as previsões obtidas pelo modelo.

Pela análise do gráfico circular obtido para o treino do modelo YOLOv8 por 250 épocas, o modelo teve previsões com valor de IoU superior a 0.9 em 320 imagens, 20 imagens com previsão de valor de IoU entre 0.6 e 0.9 e 7 imagens com valor de IoU inferior a 0.6. O modelo não foi capaz de detetar tabelas em 3 imagens.

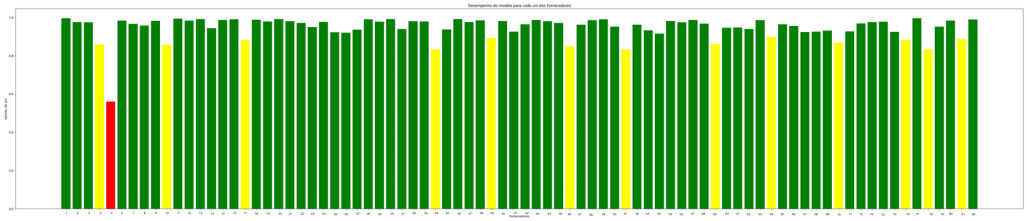


Figura 5.24 Gráfico de barras com a categoria de IoU das previsões obtidas pelo modelo para cada um dos fornecedor.

A partir do gráfico de barras, é possível observar que o valor médio de IoU foi inferior a 0.6, apenas no fornecedor **5**. Com o treino do modelo por mais épocas, verificou-se uma melhoria significativa na deteção de tabelas nas imagens do fornecedor **17**.

Tempo médio de previsão para uma imagem no conjunto de teste: 2.42 segundos.

- **Modelo YOLOv9 treinado por** 250 **épocas**

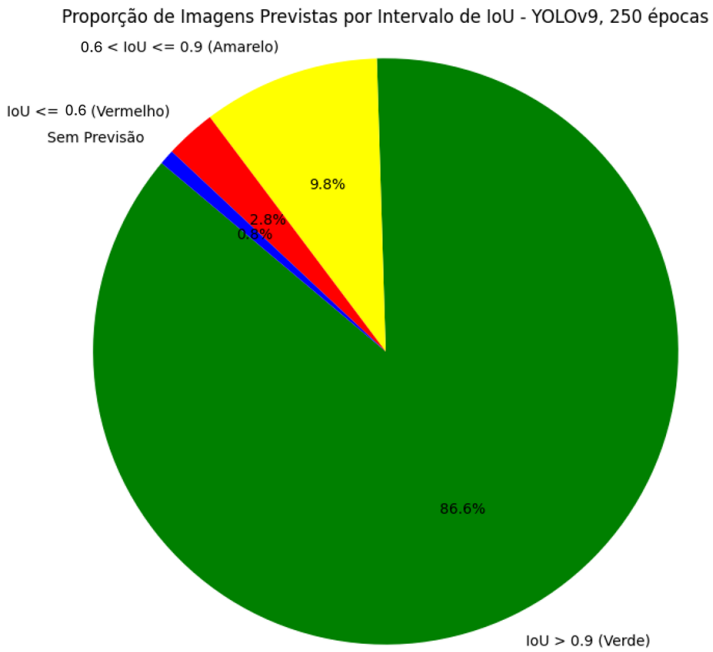


Figura 5.25 Gráfico circular com a categoria de IoU de todas as previsões obtidas pelo modelo YOLOv9 treinado por 250 épocas.

O gráfico circular com as categorias de IoU em que cada uma das previsões se insere, independentes do fornecedor, indica que o número de imagens, na qual o modelo não detetou tabelas foi igual a 3, o número de imagens com previsão de IoU superior a 90% foi 310, o de previsões com IoU entre 0.6 e 0.9 foi 35 e o número de imagens com previsão de valor de IoU inferior a 0.6 foi de 10 imagens.

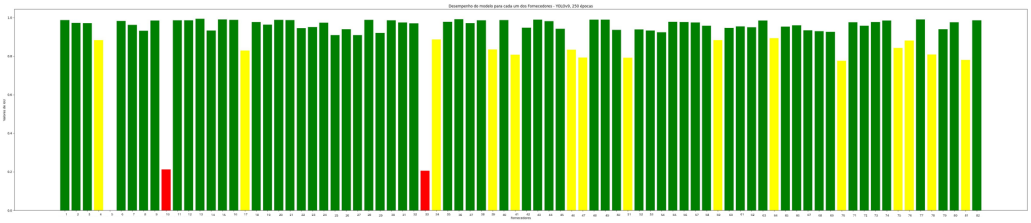


Figura 5.26 Gráfico de barras com a categoria de IoU das previsões obtidas pelo modelo para cada um dos fornecedor.

O gráfico de barras indica que o modelo não detetou tabelas no fornecedor **5** e que os valores médios de IoU foram muito baixos nos fornecedores **10** e **33**.

Tempo médio de previsão para uma imagem no conjunto de teste: 1.23 segundos.

5.5.10 Comparação dos Modelos de Deteção de Objetos YOLO treinados

Pela análise dos gráficos, é possível observar que o modelo com melhor desempenho no conjunto de dados de teste foi o YOLOv8 treinado por 250 épocas. Este modelo não conseguiu obter previsões para apenas 3 das imagens. Foi possível obter a deteção de tabelas com valor de IoU superior a 0.9 em 89.4% das imagens do conjunto de teste e houve apenas um fornecedor com um valor médio de IoU abaixo de 0.6.

O modelo YOLOv8 treinado por 100 épocas também é eficaz na deteção das tabelas e possui um tempo de previsão ligeiramente menor, comparativamente ao da rede neuronal YOLOv8 treinada por 250 épocas.

O modelo YOLOv9 é o que apresenta resultados menos satisfatórios, que refletem uma generalização menos eficaz em novos dados. Embora os resultados deste modelo não tenham sido satisfatórios para os fornecedores anteriormente referidos, as deteções realizadas por este são as mais rápidas, de entre os modelos treinados.

5.6 Treino dos Modelos de Deteção dos Cabeçalhos e da Altura das Linhas

5.6.1 Pré-processamento dos Dados

Para treinar as redes neuronais `YOLOv8x` que permitem o reconhecimento da estrutura das tabelas foi necessário construir dois novos conjunto de dados.

O primeiro passo para a construção do conjunto de dados foi a obtenção das imagens que apenas contêm a tabela. Estas imagens correspondem às do conjunto de teste, após a remoção de toda a parte que se encontra fora da caixa delimitadora da tabela, no conjunto de dados utilizado para treinar a rede neuronal de deteção das tabelas, de acordo com as Figuras 5.27, 5.28 e 5.29.

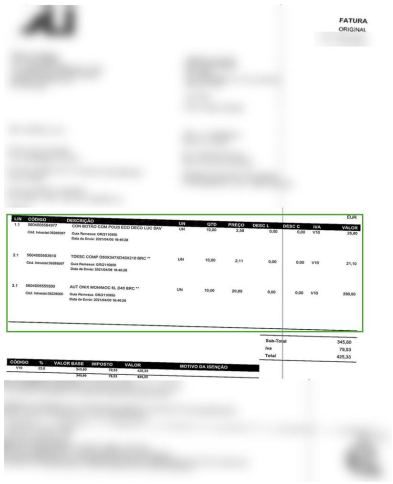


Figura 5.27 Imagem do fornecedor de dimensões 2400×2400 com a anotação da caixa delimitadora da região da tabela.

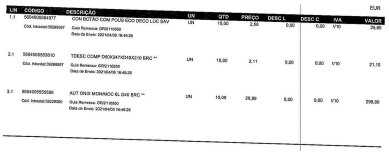


Figura 5.28 Imagem do Interior da caixa delimitadora da Tabela.

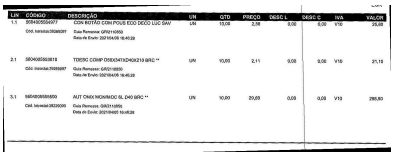


Figura 5.29 Rotação da Imagem da Tabela com o valor do ângulo retornado pelo OCR do Azure.

Após a obtenção das imagens que apenas possuem a região da tabela, copiaram-se os ficheiros `TXT` com as informações relativas ao OCR respetivo. Estas informações contêm o valor do ângulo que as imagens recortadas devem rodar. De seguida, as imagens recortadas foram rodadas com o valor de ângulo do OCR. Estas foram importadas para o *Roboflow* e anotadas manualmente, de seguida, os dados foram descarregados e as imagens convertidas para escala de cinza, para que ficassem com apenas um canal, para que o tempo de treino do modelo fosse reduzido, sem prejudicar os resultados.

O **OCR**, **Reconhecimento Ótico de Caracteres**, consiste numa tecnologia com a capacidade de reconhecer e extrair as informações de texto, através de imagens ou de documentos digitalizados. Este texto é extraído pelo OCR, o que torna esta técnica muito importante para reduzir, ou mesmo eliminar, a necessidade da inserção manual de dados, permite a extração automática das informações a partir dos documentos ou imagens de entrada. [35]

5.6.2 Treino do Modelo para o Reconhecimento dos Cabeçalhos

As imagens do *Roboflow* foram anotadas manualmente com a classe

- `0: cabecalho`.

Estas imagens foram convertidas para uma escala de cinza e ficaram com apenas um canal de cor para tornar o treino da rede neuronal mais rápido e exequível, devido à limitação dos recursos computacionais. O treino do modelo, apresentado no ficheiro `treino_dos_modelos_com_maquina_virtual.ipynb`, com o conjunto de dados com diretorias presentes no ficheiro `YAML` `data_cabecalho_1000_imgs.yaml` [12], foi realizado por 200 épocas e demorou 3 horas e 9 minutos, nas quais o *output* foi o seguinte:

- **Valor das Funções de Perda ao longo das épocas de treino do modelo**

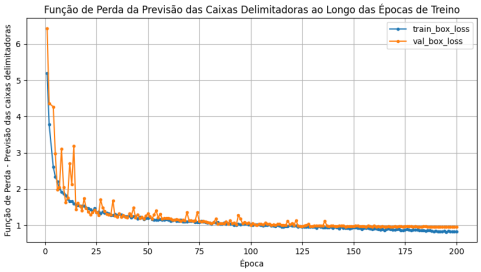


Figura 5.30 Função de Perda associada às previsões das caixas delimitadoras.

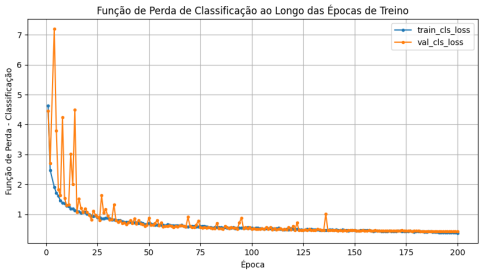


Figura 5.31 Função de Perda associada à classificação dos objetos no interior das caixas delimitadoras.

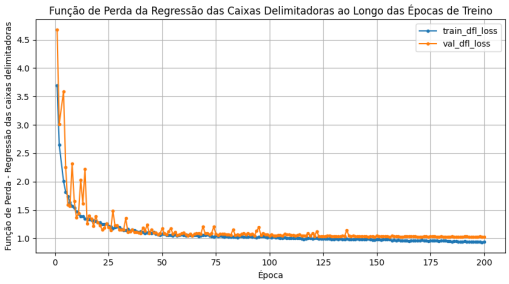


Figura 5.32 Função de Perda associada à regressão das caixas delimitadoras.

Os gráficos das funções de perda associadas à previsão das caixas delimitadoras, à classificação dos objetos no interior das caixas delimitadoras e à regressão das caixas delimitadoras dos objetos, presentes nas Figuras 5.30, 5.31 e 5.32, respetivamente, indicam um bom ajuste do modelo. Embora nas primeiras épocas os valores das funções de perda possuam oscilações muito significativas no conjunto de validação, ao fim de 150 épocas do treino do modelo, estas acabam por se tornar menores, o que sugere uma estabilização das funções de perda, em valores muito baixos. A estabilização verificada entre as 150 e as 200 épocas do modelo indicam que o treino poderia ter sido definido com apenas 150 épocas, porque os valores de perda já se encontram estabilizados e o

modelo não aprendeu mais. Assim, a observação dos gráficos das funções de perda sugere que o modelo possui uma boa capacidade de previsão de cabeçalhos em imagens de tabelas.

- **Curvas de F1-Score e de Precisão-recall**

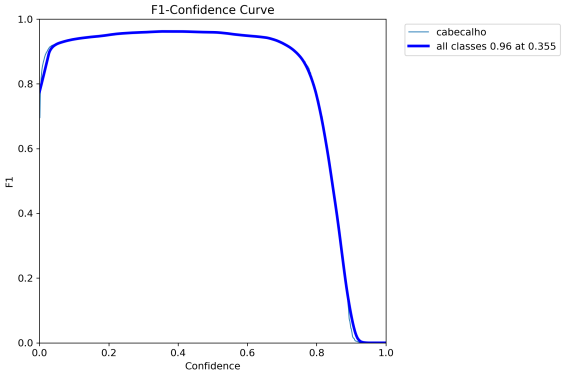


Figura 5.33 Curva de F1-Score para a Deteção de Objetos.

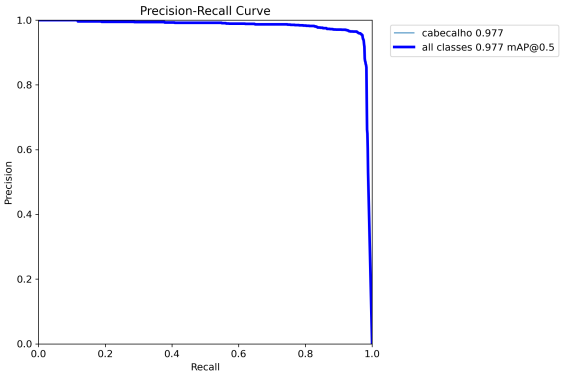


Figura 5.34 Curva Precisão-Recall para a Análise do Desempenho do Modelo para Cada Classe.

A curva de F1-Score, representada na Figura 5.33 indica que o valor da medida F1-Score varia tendo em conta diferentes níveis de confiança. O F1-Score possui um valor estável e elevado num grande intervalo de valores de confiança, o que pode significar que o modelo possui equilíbrio significativo entre a precisão e o recall. O decréscimo acentuado do valor da medida depois de um valor de confiança de cerca de 0.8 pode ser representativo de que o modelo se torne muito conservador e que possa, a partir desse nível de confiança, ignorar muitas deteções verdadeiras, de forma a evitar previsões de falsos positivos.

5.6.3 Treino do Modelo para o Reconhecimento da Altura das Linhas

As imagens do *Roboflow* foram anotadas manualmente com a classe `0: altura_linha`.

Estas imagens foram convertidas para uma escala de cinza e ficaram com apenas um canal de cor para tornar o treino da rede neuronal mais rápido e exequível, devido à limitação dos recursos computacionais. O treino do modelo, realizado em `treino_dos_modelos_com_maquina_virtual.ipyn` [12], com o conjunto de dados das diretorias presentes no ficheiro `data_altura_linha_1000_.yaml` [12], por 250 épocas e demorou 3 horas e 46 minutos, nas quais o *output* foi o seguinte:

- **Valor das Funções de Perda ao longo das épocas de treino do modelo**

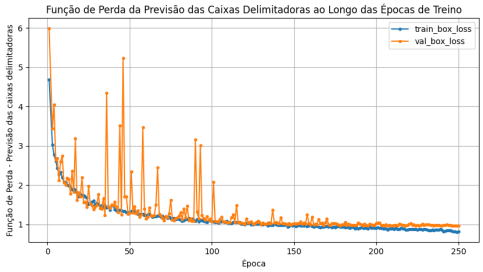


Figura 5.35 Função de Perda associada às previsões das caixas delimitadoras.

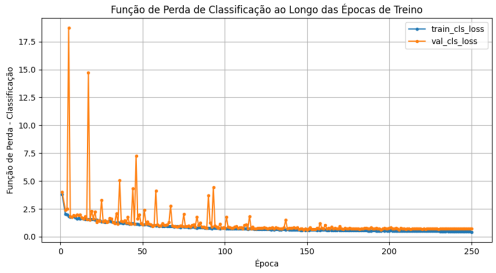


Figura 5.36 Função de Perda associada à classificação dos objetos no interior das caixas delimitadoras.

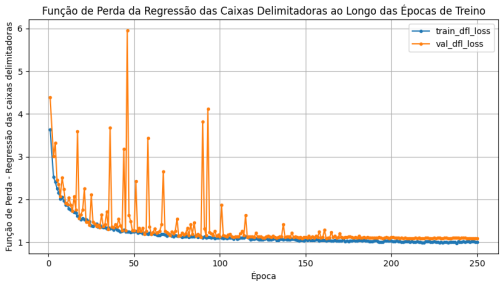


Figura 5.37 Função de Perda associada à regressão das caixas delimitadoras.

Nas Figuras 5.35, 5.36 e 5.37 observa-se o decréscimo rápido dos valores das funções de perda nas primeiras épocas, os valores das funções de perda relativas ao conjunto de teste decrescem rapidamente, de uma forma consistente. No entanto, a observação dos valores da função de perda relativos ao conjunto de validação possuem muitas oscilações antes da época 200. As oscilações dos valores da função de perda podem, mais uma vez, ser devidas ao facto do conjunto de validação possuir imagens de fornecedores com tabelas de formatos diferentes dos do conjunto de treino. A estabilidade que se verifica nas últimas épocas de treino e o facto da função de perda se ter estabilizado em valores baixos indica que o modelo se ajustou bem ao conjunto de validação. É ainda possível observar que o modelo poderia ter sido treinado por apenas 200 épocas porque o modelo já se encontrava bem ajustado.

- **Curvas de F1-Score e de Precisão-recall**

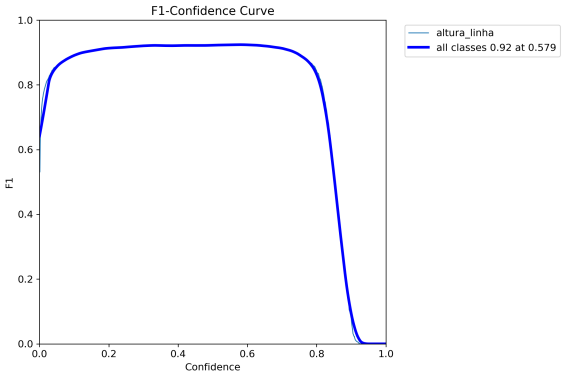


Figura 5.38 Curva de F1-Score para a Deteção de Objetos.

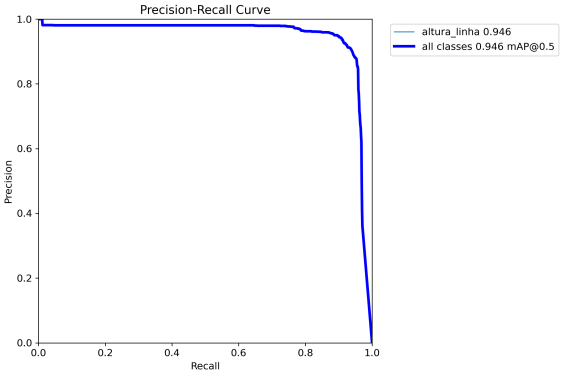


Figura 5.39 Curva Precisão-Recall para a Análise do Desempenho do Modelo para Cada Classe.

O gráfico 5.38 indica que o modelo possui um valor de F1-Score muito elevado (acima de $0,8$) para a maioria dos níveis de confiança. Após um certo nível de confiança, o valor de F1-Score decresce de forma acentuada. Segundo a análise deste gráfico, o modelo aparenta ser muito confiável até um determinado valor de confiança, acima do qual as deteções parecem tornar-se menos confiáveis.

A curva de Precisão-recall, presente na Figura 5.39, indica que o modelo possui valores elevados de precisão e de recall, pelo que o modelo é eficaz na deteção dos objetos da classe de interesse e mantém uma taxa baixa de falsos positivos. Este gráfico também sugere que a performance do modelo não decresce significativamente até atingir valores de recall muito elevados.

Em resumo, os gráficos de F1-Score e de Precision-recall indicam que o modelo YOLOv8x treinado para a deteção da altura das linhas da tabela é robusto e permanece com níveis de F1-Score e de precisão-recall elevados para uma grande diversidade de limites de confiança. O comportamento que este modelo possui é desejável em aplicações práticas nas quais é necessário minimizar tanto os falsos positivos quanto os falsos negativos.

5.7 Nova Metodologia para a Deteção, Reconhecimento e Extração de Tabelas em Documentos PDF

A metodologia para a deteção automática de tabelas em documentos PDF foi desenvolvida de forma a extrair uma tabela completa para um documento estruturado em formato JSON.

Para alcançar o objetivo proposto, foi necessário seguir um conjunto de etapas, que inclui o processamento dos dados, o treino de modelos de redes neuronais pré-treinadas de deteção de objetos, a avaliação do desempenho dos modelos treinados e a definição de restrições de *layout* para a obtenção da estrutura da tabela.

O primeiro passo foi a análise das características e da diversidade das estruturas das tabelas presentes nos documentos PDF descarregados. De seguida, cada página dos documentos PDF descarregados foi convertida numa imagem no formato JPEG, depois estas imagens foram importadas para o *Roboflow*, que permitiu realizar as anotações e redimensionar as imagens, para que fosse possível realizar o treino dos modelos de rede neuronal utilizados.

O treino dos modelos de redes neuronais foi uma das fases mais importantes para atingir o funcionamento do algoritmo implementado. Recorreu-se a modelos de rede neuronal YOLOv8, por terem sido considerados mais eficazes que outros em fases preliminares do projeto, face aos recursos computacionais disponíveis.

Após o treino dos modelos de redes neuronais, foi necessário aplicar restrições de *layout* que se adequassem às tabelas de diferentes tipos, presentes nos dados, de forma a garantir que a etapa do reconhecimento da estrutura das tabelas funcione de forma automática e rigorosa.

Serão abordadas as especificações da máquina virtual utilizada para o treino do modelo, as etapas de pré-processamento das imagens e dos dados, o treino dos modelos e a escolha das redes neuronais e das operações adicionais realizadas.

Este capítulo descreve a construção da Metodologia de Deteção e de Reconhecimento de Tabelas em Imagens, de uma forma detalhada. Foi necessária a aplicação de operações de processamento de imagem, como o redimensionamento, o recorte e a conversão de formatos, a realização de previsões com os modelos YOLO treinados, a utilização de um OCR e de modelos de deteção de objetos e a aplicação de transformações de coordenadas de caixas delimitadoras. As etapas da metodologia incluem o pré-processamento da imagem, a deteção de objetos com o auxílio dos modelos treinados e o ajuste das coordenadas de texto e das delimitações da estrutura da tabela. A metodologia permite a manutenção do desempenho dos modelos treinados e é constituída por quatro etapas fundamentais:

1. Deteção das Tabelas;
2. Reconhecimento da Altura das Linhas e dos Cabeçalhos;
3. Aplicação de Restrições de *Layout*;
4. Extração das Informações da Tabela para um ficheiro JSON estruturado.

5.7.1 Deteção das Tabelas

O procedimento de Deteção de Tabelas foi desenvolvido de acordo com os seguintes passos:

1. Inserir o caminho da imagem. A imagem da diretoria inserida deve corresponder a uma página de um ficheiro `PDF` já convertida para o formato `JPG`;
2. Carregar a imagem e redimensioná-la para 2400×2400, com respeito às proporções originais da imagem;
3. Converter a imagem redimensionada para 2400 por 2400 para formato de `bytes`, porque o OCR do `Azure` recebe a imagem no formato de `bytes`;
4. Aplicar o OCR. Guardar os textos detetados, as respetivas caixas delimitadoras e o ângulo de rotação.
5. Aplicar a rotação na imagem redimensionada com o valor do ângulo obtido com o OCR em graus;
6. Redimensionamento da imagem rodada para 640 por 640 com borda branca. As proporções da imagem são mantidas e é acrescentada uma borda branca para que a imagem fique quadrada;
7. Redimensionar a imagem rodada para 2400 por 2400 novamente, porque são as dimensões de imagem especificadas pela Microsoft para a utilização do serviço `Azure OCR`;
8. Definição do modelo de Deteção de Tabelas, que corresponde ao que foi escolhido anteriormente: modelo YOLOv8 treinado por 250 épocas;
9. Obter a previsão da imagem rodada com dimensões 640 por 640 com o modelo. As coordenadas da caixa delimitadora prevista para a tabela já se encontram convertidas para imagens de dimensão 2400 por 2400 pixels.
    - Se não houver deteção de tabelas por parte do modelo, o algoritmo termina.
    - Se o modelo detetar tabelas na imagem, o algoritmo continua.

5.7.2 Reconhecimento da Estrutura das Tabelas

Após a deteção da tabela, foi necessário desenvolver duas metodologias distintas, que afetam significativamente os resultados na etapa de extração das informações das tabelas:

- **Aplicação do OCR uma só vez**, `metodologia_ocr_aplicado_1_vez.py` [12];
- **Aplicação do OCR duas vezes** `metodologia_ocr_aplicado_2_vezes.py` [12]

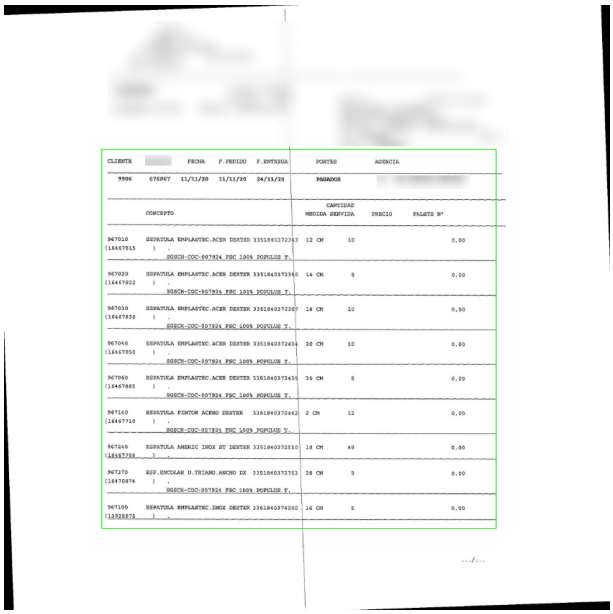


Figura 5.40 Imagem Anotada com a Tabela Prevista.

O OCR utilizado [25] para o reconhecimento das informações das tabelas dos dados pertence ao serviço *Azure AI Document Intelligence* [23], que utiliza muitas técnicas e modelos de aprendizagem automática, que permitem a extração de textos manuscritos ou impressos de fontes muito diversificadas, seja de documentos complexos como relatórios e artigos, ou através de imagens menos complexas como as de etiquetas de produtos ou de sinais de rua.

O algoritmo foi implementado de acordo com as etapas que se seguem, que permitem o reconhecimento eficaz da estrutura das tabelas de diferentes formatos. Os passos que são exclusivos das metodologias encontram-se destacados.

Pré-processamento

1. Obter a imagem recortada, que resulta da remoção de toda a parte da imagem rodada de dimensões 2400 por 2400 que se encontra fora da caixa delimitadora resultante da previsão, como na Figura 5.41;

CLIENTE	[illegible]	FECHA	F.PEDIDO	F.ENTREGA	PORTES	AGENCIA
9906	676867	11/11/20	11/11/20	24/11/20	PAGADOS	[illegible]

	CONCEPTO	CANTIDAD MEDIDA	SERVIDA	PRECIO	PALETS Nº	
967010 (16467815) .	ESPATULA EMPLASTEC.ACER DEXTER 3351840372343 SGSCH-COC-007924 FSC 100% POPULUS T.	12 CM	10			0,00
967020 (16467822) .	ESPATULA EMPLASTEC.ACER DEXTER 3351840372350 SGSCH-COC-007924 FSC 100% POPULUS T.	14 CM	5			0,00
967030 (16467836) .	ESPATULA EMPLASTEC.ACER DEXTER 3351840372367 SGSCH-COC-007924 FSC 100% POPULUS T.	16 CM	10			0,00
967040 (16467850) .	ESPATULA EMPLASTEC.ACER DEXTER 3351840372404 SGSCH-COC-007924 FSC 100% POPULUS T.	20 CM	10			0,00
967060 (16467885) .	ESPATULA EMPLASTEC.ACER DEXTER 3351840372435 SGSCH-COC-007924 FSC 100% POPULUS T.	30 CM	5			0,00
967140 (16467710) .	ESPATULA PINTOR ACERO DEXTER 3351840372442 SGSCH-COC-007924 FSC 100% POPULUS T.	2 CM	12			0,00
967240 (16467766) .	ESPATULA AMERIC INOX ST DEXTER 3351840372510	10 CM	48			0,00
967370 (16470874) .	ESP.ENCOLAR D.TRIANG.ANCHO DX 3351840372763 SGSCH-COC-007924 FSC 100% POPULUS T.	28 CM	9			0,00
967100 (13925975) .	ESPATULA EMPLASTEC.INOX DEXTER 3351840374200	16 CM	5			0,00

Figura 5.41 Imagem da Tabela Prevista Recortada.

2. Aplicar a rotação das caixas delimitadoras de texto detetadas com o OCR na etapa 4 da Deteção das Tabelas (Etapa exclusiva para o caso em que se aplica o OCR uma única vez) é realizada a todos os pontos da caixa delimitadora. Esta rotação é realizada da seguinte forma: Dado o ponto $P_1 = (x_1, y_1)$, que se pretende rodar por um ângulo em graus, representado por α, em torno do centro (x_c, y_c).

   - O valor de ângulo α de graus para radianos;
   - Aplicar a translação ao ponto P_1 que faz com que o centro da rotação corresponda à origem:

$$x_{\text{novo}} = x_1 - x_c, \tag{5.1}$$

$$y_{\text{novo}} = y_1 - y_c; \tag{5.2}$$

   - Rotação do ponto com as fórmulas trigonométricas:

$$\begin{bmatrix} x_{\text{rodado}} \\ y_{\text{rodado}} \end{bmatrix} = \begin{bmatrix} \cos(\alpha_{(\text{radianos})}) & -\sin(\alpha_{(\text{radianos})}) \\ \sin(\alpha_{(\text{radianos})}) & \cos(\alpha_{(\text{radianos})}) \end{bmatrix} \begin{bmatrix} x_{\text{novo}} \\ y_{\text{novo}} \end{bmatrix}$$

   - Nova translação do ponto, para que o ponto rodado retorne à posição original do centro

$$\begin{bmatrix} x1_{\text{rodado}} \\ y1_{\text{rodado}} \end{bmatrix} = \begin{bmatrix} x_{\text{rodado}} \\ y_{\text{rodado}} \end{bmatrix} + \begin{bmatrix} x_c \\ y_c \end{bmatrix}$$

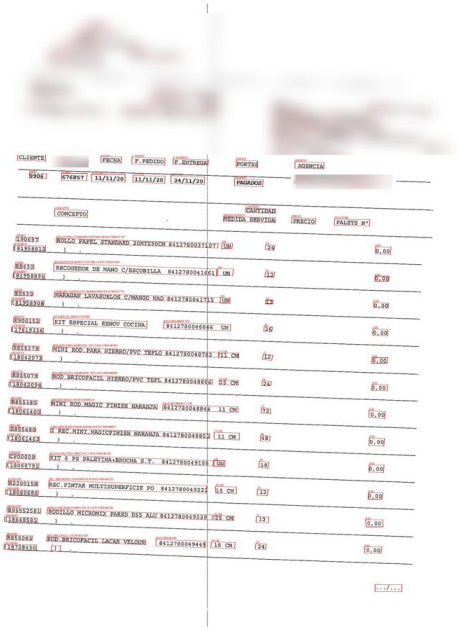


Figura 5.42 Caixas delimitadoras do texto detetado na imagem, com o OCR.

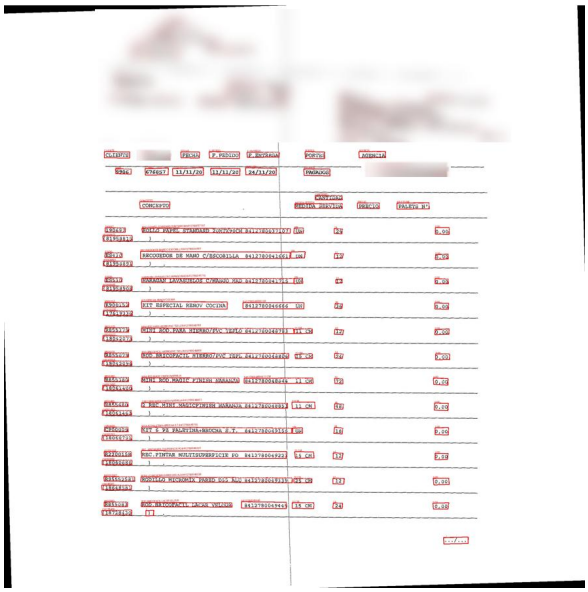


Figura 5.43 Caixas delimitadoras do texto detetado na imagem, com o OCR rodadas.

3. Aplicar a restrição na qual apenas se consideram os textos detetados pelo OCR no interior da caixa delimitadora da tabela prevista (Etapa exclusiva para o caso em que se aplica o OCR uma única vez)

4. Converter a imagem recortada de `PIL` para `bytes`(Etapa exclusiva para o caso em que se aplica o OCR duas vezes);

5. Aplicar novamente o OCR (Etapa exclusiva para o caso em que se aplica o OCR duas vezes). São guardadas as coordenadas das caixas delimitadoras dos textos detetados na imagem recortada, os textos detetados e o valor do ângulo (que não será usado).

6. Transladar as coordenadas das caixas delimitadoras do texto detetado pelo OCR. (Etapa exclusiva para o caso em que se aplica o OCR duas vezes). As caixas delimitadoras de texto obtidas na imagem detetada são transladadas de forma a que a sua posição seja correspondente na imagem rodada não recortada de dimensões 2400 por 2400, ou seja, as coordenadas de x de todas as caixas delimitadoras de texto detetadas são somadas com o valor de x_1 e as coordenadas de y das caixas delimitadoras de texto detetadas são somadas com o valor y_1, onde x_1 e y_1 correspondem ao canto superior esquerdo da caixa delimitadora prevista para a localização da tabela. Esta operação é realizada de acordo com a Figura 5.45.

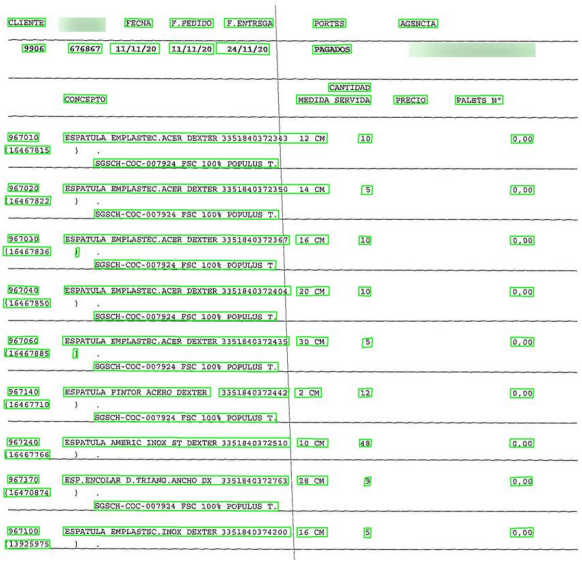


Figura 5.44 Coordenadas das caixas delimitadoras dos textos detetados pelo OCR na imagem recortada.

Figura 5.45 Caixas delimitadoras dos textos detetados pelo OCR na imagem recortada transladadas e representadas na imagem antes de ser recortada.

7. Converter as coordenadas das caixas delimitadoras do texto detetado para coordenadas retangulares.

8. Redimensionar a imagem recortada, obtida através do recorte da imagem de dimensões 2400 por 2400, que apenas contém a região prevista para a localização da tabela para um tamanho de 640 no lado maior e um tamanho que mantém as proporções originais no lado menor;

Reconhecimento da delimitação das colunas da tabela e da delimitação da região onde se encontram os cabeçalhos:

1. Definir o modelo de deteção de cabeçalhos;

2. Realizar a previsão da localização dos cabeçalhos nas imagens recortadas e redimensionadas, de acordo com a Figura 5.46;

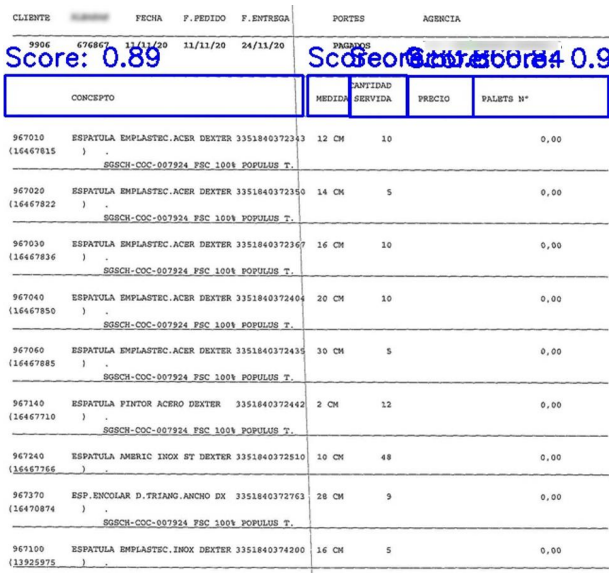


Figura 5.46 Previsão dos Cabeçalhos obtida com o Modelo YOLOv8 treinado por 200 épocas.

3. Converter o formato das caixas delimitadoras resultantes da previsão para o formato de 4 cantos;

4. Guardar os valores de y do canto inferior esquerdo de cada cabeçalho previsto numa lista, para que o limite inferior do cabeçalho no final do algoritmo seja correspondente à média dos valores de x do canto inferior esquerdo dos cabeçalhos previstos;

5. Guardar os valores de y do canto superior esquerdo de cada cabeçalho previsto numa lista, na qual o menor valor de y corresponde ao limite superior dos cabeçalhos, no final do reconhecimento da estrutura final da tabela;

6. Criar as caixas delimitadoras do topo à base da imagem, nas quais cada uma das linhas será posicionada no valor da coordenada de x de cada uma das previsões de cabeçalhos realizada pela rede neuronal, de acordo com a Figura 5.47.

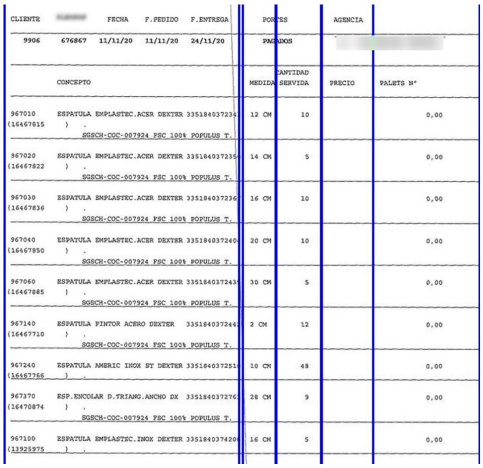


Figura 5.47 Linhas verticais obtidas com as previsões dos cabeçalhos.

7. Aplicar o método *Non-Maximum Suppresion*. [19]

   O método *Non-Maximum Supression* (NMS) é muito utilizado em tarefas de deteção de objetos, por permitir eliminar caixas delimitadoras redundantes e que se encontram sobrepostas, mantendo as caixas delimitadoras com maior valor de confiança. Nesta metodologia, o método NMS foi realizado para a remoção de caixas redundantes que representam as divisões horizontais e verticais da estrutura da tabela, antes da aplicação de restrições de *layout*.

   A aplicação do *Non-Maximum Suppression* foi realizada através da função

   `tf.image.non_max_suppression()` [36], da biblioteca `Tensorflow`, que recebe:

   - as caixas delimitadoras resultantes da conversão das linhas representadas na Figura 5.47 para caixas delimitadoras do topo à base da imagem;
   - os valores de confiança associados às caixas delimitadoras dos cabeçalhos obtidas com a previsão do modelo YOLOv8 treinado para a deteção de cabeçalhos;
   - o número máximo de caixas delimitadoras que o *Non-Maximum Suppression* pode retornar, ao qual foi atribuído o valor `1000` para garantir que não haja eliminação de caixas por número excessivo, mas sim, por redundância e análise da sobreposição das mesmas;
   - o valor de IoU utilizado para remover a caixa com menor confiança foi definido com o valor de `0.000001`. O valor muito baixo deste parâmetro implica que a sobreposição mínima entre duas caixas delimitadoras resulte na eliminação da caixa que possui menor valor de confiança. Com este valor de limiar, apenas são removidas as caixas delimitadoras idênticas, uma vez que o objetivo é eliminar as linhas redundantes nas tabelas.
   - o valor de confiança a partir do qual a caixa delimitadora não é eliminada pelo método. Atribuiu-se o valor `-Inf` a este parâmetro, o que significa que todas as caixas delimitadoras são válidas, para que as caixas previstas não sejam eliminadas por ter um valor de confiança baixo na previsão, mas sim, por serem praticamente idênticas.

   O método possui uma função que retorna os índices das caixas delimitadoras que não foram eliminadas e estas são desenhadas na imagem da tabela, como representado pela Figura 5.48.

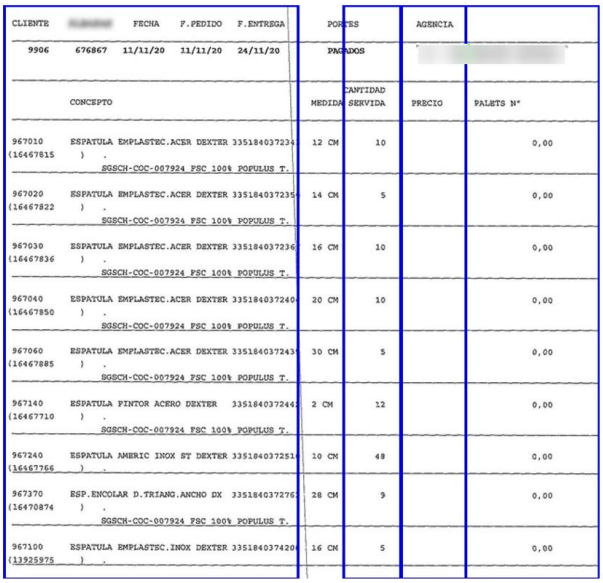


Figura 5.48 Divisão das colunas obtida com as previsões dos cabeçalhos após a aplicação do método *Non-Maximum Suppression*.

Mesmo com a aplicação do método NMS, existem ainda algumas linhas verticais redundantes para a separação das colunas das tabelas, pelo que foi necessário aplicar restrições de *layout*.

8. Criar uma lista com as coordenadas de x após a utilização do método *Non-Maximum Suppression*;

9. Ordenar a lista e, se distância entre as coordenadas de x for inferior a 20, considerando apenas o valor maior de x do par. É garantido que o último valor da lista de x também é adicionado;

10. Forçar que a primeira e última colunas sejam sempre delimitadas porque, em alguns casos, a delimitação da primeira ou da última coluna da tabela não era detetada pelo modelo.

    - Acrescentar o valor 0 no início da lista se o valor da primeira coordenada de x for maior do que 20;
    - Acrescentar o valor 640 no final da lista de coordenadas x se o valor da primeira coordenada de x for menor do que 620;

11. Guardar a lista das coordenadas de x após a aplicação das restrições de *layout* e realizar as linhas verticais de delimitação das colunas das células.

Reconhecimento da delimitação das linhas da tabela

1. Definir o modelo de deteção da altura das linhas;
2. Obter a previsão da altura das linhas com o modelo, de acordo com a Figura 5.49.

Figura 5.49 Previsão da Altura das Linhas obtida com o Modelo YOLOv8 treinado por 250 épocas.

3. Transformar as coordenadas previstas para coordenadas de cantos, que facilita na criação das linhas horizontais, representadas na Figura 5.50;

Figura 5.50 Linhas horizontais obtidas com as previsões da altura das linhas.

4. Criar uma lista com as coordenadas das previsões em formato de quatro cantos;
5. Realizar o método *Non-Maximum Suppression* como anteriormente para a delimitação das colunas definida pela previsão dos cabeçalhos;

Figura 5.51 Delimitação das linhas obtida com as previsões da altura das linhas após a aplicação do método *Non-Maximum Suppression*.

6. Criar uma lista com as coordenadas de y que delimitam as linhas, após o nms. Nesta lista são acrescentados os limites inferior e superior dos cabeçalhos detetados anteriormente. Assim, as delimitações das linhas da tabela ficam de acordo com a Figura 5.52. Após

Figura 5.52 Linhas horizontais obtidas com as previsões da altura das linhas após nms com as delimitações do limite superior e inferior dos cabeçalhos.

a realização do nms permanecem ainda algumas linhas redundantes para a divisão das linhas das tabelas, o que tornou necessária a realização de uma restrição de *layout*.

7. Se a distância vertical entre os valores de y for inferior a 6 pixels, considerar apenas o maior valor entre os dois. O último valor da lista é garantidamente adicionado;

8. Ocorreram casos em que a última linha não tinha delimitação, o que tornou necessário o acrescento de uma linha na parte de baixo da tabela, ou seja, se o maior valor de y for menor do que a altura da imagem – 40 pixels, deve-se acrescentar à lista de valores de y o valor da altura da imagem, o que garante que a última linha é considerada na delimitação da tabela.

Junção das Linhas Horizontais e Verticais

1. Criar uma lista com as linhas horizontais e verticais obtidas após as restrições de *layout*, de acordo com a Figura 5.53.

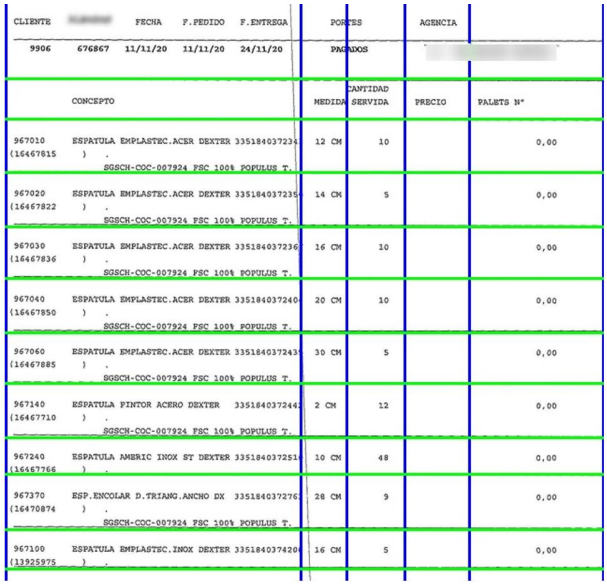


Figura 5.53 Linhas horizontais e verticais obtidas com as restrições aplicadas aos resultados dos modelos YOLO treinados.

2. Redimensionar as coordenadas de x e de y presentes na lista com os valores das coordenadas proporcionalmente para as dimensões originais da imagem recortada antes de ter sido convertida em 640 por 640, com o auxílio das fórmulas:

$$x_{\text{redimensionado}} = x \cdot \frac{\text{largura da imagem recortada}}{\text{largura da imagem redimensionada com lado maior de 640 pixels}}$$

$$y_{\text{redimensionado}} = y \times \frac{\text{altura da imagem recortada}}{\text{altura da imagem redimensionada com lado maior de 640 pixels}}$$

3. Somar às coordenadas de x, o valor x_1 da previsão da localização da tabela e a todas as coordenadas de y o valor de y_1 da previsão da localização da tabela (somar o canto superior esquerdo da previsão). O resultado da translação dos valores de x e de y resultou na delimitação da Figura 5.54.

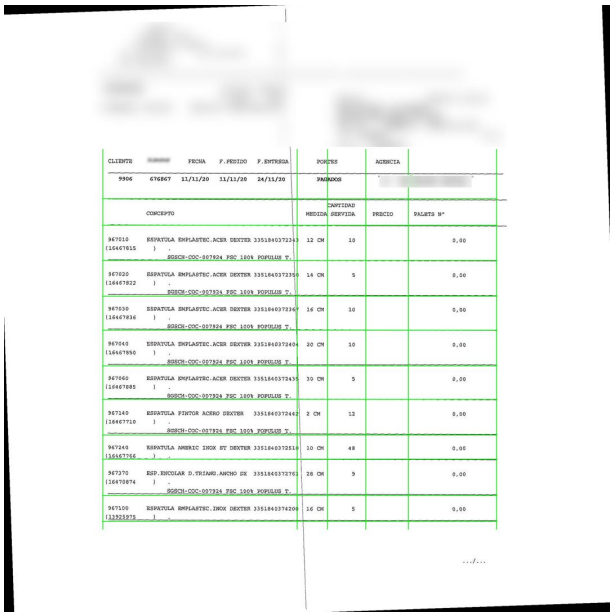


Figura 5.54 Linhas horizontais e verticais obtidas com as restrições aplicadas aos resultados dos modelos YOLO treinados transladadas para a localização da tabela detetada.

4. Considerar os textos e as coordenadas das caixas delimitadoras dos textos detetados pelo OCR transladadas, obtidas anteriormente (Figura 5.45);

5. Juntar as informações das coordenadas das delimitações do texto obtidas pelo OCR transladadas com os valores das coordenadas de x que delimitam as linhas verticais e com as coordenadas de y que delimitam as linhas horizontais, de acordo com a Figura 5.54.

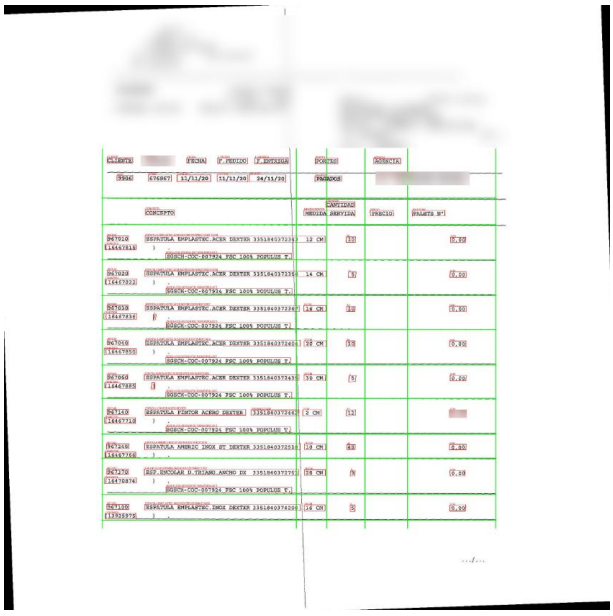


Figura 5.55 Linhas horizontais e verticais obtidas com as restrições aplicadas aos resultados dos modelos YOLO treinados transladadas para a localização da tabela detetada com anotações de OCR transladadas.

Após as restrições aplicadas, ocorrem ainda os seguintes constrangimentos:

- existem delimitações nas quais não existem cabeçalhos adequados, por haver uma linha que divide um cabeçalho sem conteúdo;
- existem delimitações na base de algumas imagens que não possuem informações, o que faz com que haja delimitação desnecessária.

Assim, foi necessário aplicar os resultados do OCR nas restrições de *layout* necessárias para a correção destes problemas.

Restrições de *Layout* com Informações do OCR, das Linhas e das Colunas

1. Remover a coordenada de x que delimita uma coluna no caso de não haver texto detetado pelo OCR no cabeçalho onde essa delimitação se encontra. A remoção é efetuada com a função `remove_bboxes_without_ocr`;
2. Utilizar a função `remove_unneeded_y_coords`, que remove o último valor de y na lista de y finais ordenada caso não existam delimitações de OCR entre o penúltimo e último valores de y.

5.7.3 Resultados Obtidos

Com a metodologia desenvolvida, foi possível delimitar automaticamente a estrutura das tabelas com resultados satisfatórios em 87% das tabelas presentes nas imagens do conjunto de teste.

- **Gráfico circular com a Proporção de Tabelas corretamente divididas no conjunto de teste**
  - **Verde:** Tabelas Corretamente Divididas;
  - **Amarelo:** Tabelas com Divisão Incorreta.
- **Gráfico de barras da percentagem de delimitações adequadas para cada um dos fornecedores**
  - **Verde:** Proporção de tabelas corretamente delimitadas num fornecedor superior a 90%;
  - **Amarelo:** Proporção de tabelas com divisão incorreta num fornecedor inferiores a 90%.

O gráfico circular da Figura 5.56 indica que a proporção de tabelas com divisão correta é de 87% e que a proporção de tabelas com divisão incorreta é de 13%. As divisões incorretas

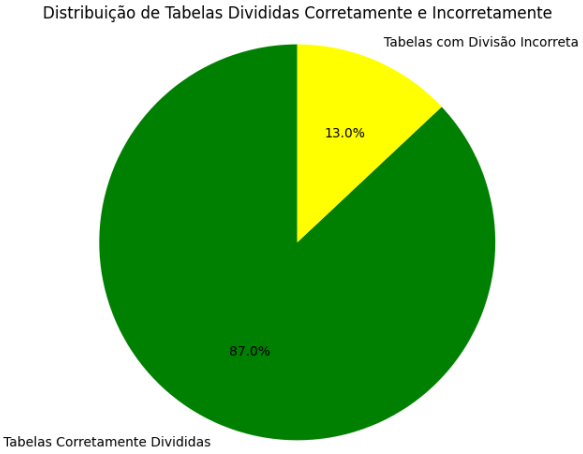


Figura 5.56 Proporção de Tabelas Corretamente Delimitadas no Conjunto de Treino.

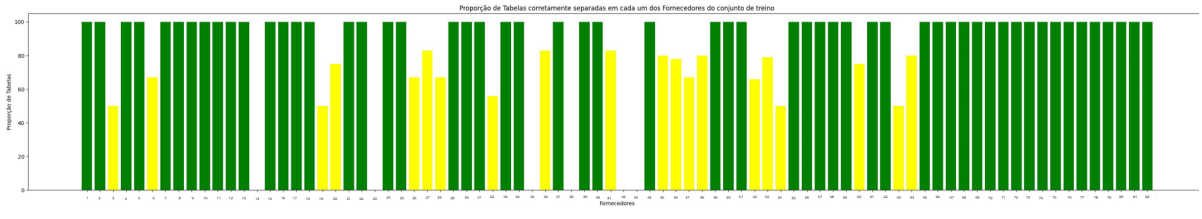


Figura 5.57 Proporção de Tabelas Corretamente Delimitadas por Fornecedor do Conjunto de Treino.

correspondem às divisões que possuem a disposição das linhas horizontais e verticais finais nos locais errados, ou sem linhas ou com linhas a mais na divisão entre as células da tabela. Estas proporções refletem um bom desempenho por parte da metodologia implementada para o reconhecimento das tabelas, embora exista ainda margem para melhorar estes resultados.

O gráfico de barras representado na Figura 5.57 indica a proporção de tabelas corretamente divididas por cada um dos fornecedores. A análise deste gráfico indica que a maioria dos fornecedores possui um desempenho adequado. No entanto, o desempenho da metodologia de reconhecimento da estrutura da tabela possui alguma variação do seu desempenho entre os fornecedores, pois alguns deles possuem proporção de divisões incorretas significativas.

A discrepância do desempenho da metodologia de reconhecimento da estrutura da tabela entre os fornecedores indica que podem ser necessários ajustes adicionais para que os modelos de rede neuronal possam detetar os cabeçalhos e a altura das linhas de uma forma mais eficaz para os formatos de tabelas existentes.

5.8 Extração das Informações das Tabelas para um Documento Estruturado JSON

Após a deteção das tabelas e o reconhecimento da sua estrutura, procedeu-se à extração das informações das tabelas para documentos estruturados no formato JSON.

Para a realização do procedimento, foi necessária a junção das informações das coordenadas das células das tabelas com as coordenadas dos textos detetados pelo OCR. Após a junção, foi necessário aplicar restrições que permitissem a separação dos textos e a sua inserção nas células em que estes se encontram.

As restrições aplicadas foram:

1. Obtenção das Coordenadas das Células:
    - Calcular as caixas delimitadoras para cada célula, com os valores de x e de y das delimitações verticais e horizontais obtidas após a etapa do **Reconhecimento da Estrutura das Tabelas**, que organiza as células em linhas e em colunas.
    - Estas caixas delimitadoras são guardadas numa lista e cada célula é identificada por um ID com os valores da linha e da coluna correspondentes na tabela.
2. Processamento das Informações do Texto Detetado pelo OCR:
    - Os textos detetados na imagem e as coordenadas das caixas delimitadoras onde se encontram são guardadas numa lista.
    - Para cada célula, verifica-se se o texto detetado pelo OCR se interseta com a caixa delimitadora da célula.
    - No caso de haver **apenas uma interseção**, o texto é associado à célula com a qual se interseta, num dicionário, no qual a chave corresponde à posição da célula e o valor corresponde ao texto.
3. Tratamento de Múltiplas Caixas Delimitadoras de Texto no Interior de Uma Célula:
    - Quando a caixa delimitadora de uma célula possui mais do que uma caixa delimitadora de texto no seu interior, os textos das caixas são agrupados com a adição de um caractére de espaço no dicionário associado ao ID da célula correspondente.
4. Tratamento das Caixas Delimitadoras que se Intersetam com Múltiplas Células:
    - Quando uma caixa delimitadora de texto detetado pelo OCR se **interseta com mais do que uma célula**, o código divide o texto (em palavras) entre as células envolvidas, com restrições que juntam palavras específicas como uma só, porque se encontram sempre juntas nas tabelas dos dados, de forma que o conteúdo não seja repetido em várias células e para preservar a estrutura correta da tabela.
    - Se o texto se intersetar com mais do que uma célula, este é repartido de acordo com a proporção da interseção e inserido nas respetivas células.

5. Deteção e Construção dos Cabeçalhos na Estrutura do Ficheiro:
    - Se a célula pertencer à primeira linha, é tratada como cabeçalho. Para este caso, existem regras específicas para lidar com a interseção dos textos com as células do cabeçalho, que permitem que o texto seja corretamente inserido, mesmo que uma única caixa de OCR se intersete com mais do que uma célula.
    - Quando uma caixa de texto do cabeçalho se interseta com mais do que uma célula, o texto é separado e distribuído pelas colunas do cabeçalho.
6. Restrição para Evitar Textos Repetidos:
    - São aplicadas restrições para que o mesmo troço de texto não seja adicionado mais do que uma vez nas células.
7. Processamento de Interseções Complexas:
    - Quando uma caixa de texto se interseta com mais do que uma célula, mas a caixa de texto seguinte não se interseta com a célula seguinte, o código divide o texto de forma a evitar que as células seguintes dos cabeçalhos fiquem vazias.
8. Criação dos ficheiros do tipo `JSON` e Excel:
    - Depois da criação do dicionário com a associação dos textos às células, os dados são formatados num ficheiro do tipo `JSON` estruturado, que inclui as coordenadas de cada célula e do texto correspondente.
    - Por fim, cria-se um ficheiro Excel com as informações do ficheiro `JSON`, que contêm a tabela estruturada e o texto inserido nas células correspondentes.

O ficheiro estruturado do tipo `JSON` possui a informação de cada uma das células constituída de acordo com a Figura 5.58.


```
{
    "tables": [
        {
            "text": "Descrição Preço Unit",
            "row": 0,
            "column": 0,
            "left": 465.1826477050781,
            "top": 812.3738479614258,
            "width": 1012.4437499999999,
            "height": 34.636363636363626,
            "top_left": [
                465.1826477050781,
                812.3738479614258
            ],
            "bottom_right": [
                1477.626397705078,
                847.0102115977894
            ]
        },
```


Figura 5.58 Exemplo do Formato de Uma Célula num Ficheiro Estruturado `JSON`.

No ficheiro, cada célula é representada num dicionário, no qual cada chave "`tables`"contém as informações relativas ao conteúdo e à localização da célula, na imagem completa da guia de remessa. Esta estrutura, em dicionários, permite a organização fixa dos dados, o que facilita na extração e na análise destes, independentemente do formato original da tabela de onde são extraídos.

- `text`: contém o texto que a célula possui;
- `row`: ID da linha da célula;
- `column`: ID da coluna da célula;
- `left`: coordenada de x da imagem da página completa do guia de remessa do fornecedor onde se encontra o limite esquerdo da célula;
- `top`: coordenada y da imagem da página completa da guia de remessa do fornecedor onde se encontra o limite superior da célula;
- `width`: largura da célula da tabela;
- `height`: altura da célula da tabela;
- `top_left`: lista com as coordenadas x e y, respetivamente, do canto superior esquerdo da célula, na imagem inteira do guia de remessa;
- `bottom_right`: lista com as coordenadas x e y, respetivamente, do canto inferior direito da célula, na imagem inteira do guia de remessa.

Com esta implementação, é possível obter um ficheiro estruturado do tipo `JSON`, cuja formatação é representada num ficheiro Excel, com as informações da tabela.

5.8.1 Aplicação do OCR uma única vez

O facto de o OCR ser aplicado uma única vez na imagem inteira faz com que nem todos os textos que se encontram no interior da tabela sejam detetados, como representado na Figura 5.59.

Este acontecimento é muito comum nas imagens do conjunto de dados e não permite que a extração da tabela seja realizada sem que esta fique desformatada, como se verifica na Figura 5.60.

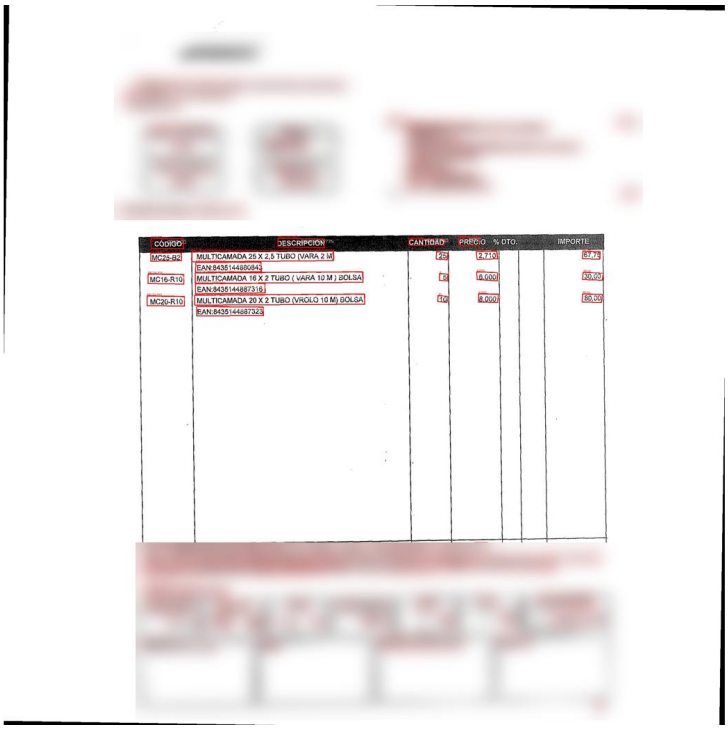


Figura 5.59 Deteção dos Textos com a Aplicação do OCR numa Imagem Inteira.

CÓDIGO	DESCRIPCIÓN	CANTIDAD	PREC
MC25-B2	MULTICAMADA 25 X 2,5 TUBO (VARA 2 M) EAN:8435144880843	25	2,710 67,75
MC16-R10	MULTICAMADA 16 X 2 TUBO (VARA 10 M) BOLSA EAN:843514488731	5	6,000 30,00
MC20-R10	MULTICAMADA 20 X 2 TUBO (VROLO 10 M) BOLSA EAN:843514488732	10	8,000 80,00

Figura 5.60 Tabela Extraída com a Aplicação do OCR Uma Única Vez.

O tempo médio da execução do Algoritmo de Deteção, Reconhecimento e Extração das Tabelas completo, com uma utilização única do OCR do `Azure` foi de 33.56 segundos, para as imagens do conjunto de teste, com um custo de $0,924$ € a cada 1000 imagens, sem o custo de nuvem associado.

5.8.2 Aplicação do OCR duas vezes

Para tentar resolver este inconveniente, foi efetuada uma tentativa da aplicação do OCR na imagem recortada, que apenas contém a tabela. A realização desta alteração torna possível a deteção de todos os textos no interior da mesma, Figura 5.61.

Assim, foi realizada a tentativa de aplicar o OCR duas vezes. A primeira para obter o ângulo de rotação da imagem e a segunda para detetar os textos completos no interior das tabelas.

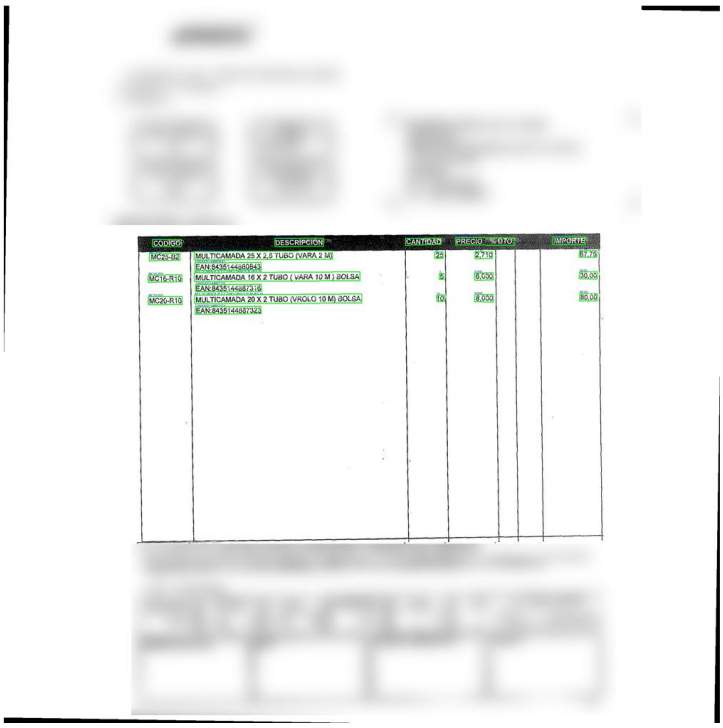


Figura 5.61 Deteção dos Textos com a Aplicação do OCR na Imagem que apenas contém a Tabela.

CODIGO	DESCRIPCIÓN	CANTIDAD	PRECIO	DTO	IMPORTE
MC25-B2	MULTICAMADA 25 X 2,5 TUBO (VARA 2 M) EAN:8435144880843	25	2,710		67,75
MC16-R10	MULTICAMADA 16 X 2 TUBO (VARA 10 M) BOLSA EAN:8435144887316	5	6,000		30,00
MC20-R10	MULTICAMADA 20 X 2 TUBO (VROLO 10 M) BOLSA EAN:8435144887323	10	8,000		80,00

Figura 5.62 Tabela Extraída com a Aplicação do OCR Duas Vezes.

O tempo médio da execução do Algoritmo de Deteção, Reconhecimento e Extração das Tabelas completo, com duas utilizações do OCR do `Azure` foi de 35.43 segundos, para as imagens do conjunto de teste, com um custo de $1,848$ € a cada 1000 imagens, sem o custo de nuvem associado.

Apesar do acréscimo do custo da utilização da metodologia numa imagem, a melhoria dos resultados com a utilização do OCR duas vezes é muito significativa.

5.8.3 Resultados Obtidos

Verificou-se anteriormente que a versão da Metodologia com a utilização do OCR duas vezes foi a que permitiu obter melhores resultados. Deste modo, serão abaixo representados alguns resultados específicos que refletem o bom funcionamento da etapa da Extração das Informações e a representação de alguns constrangimentos para os quais seria necessária a implementação de restrições adicionais, ou a utilização de um OCR que não agrupe as palavras indevidamente.

Os exemplos de extrações das informações das tabelas que se seguem, consistem em casos nos quais a deteção da tabela e o reconhecimento da sua estrutura foram realizados de forma adequada com a metodologia implementada:

- Exemplo A

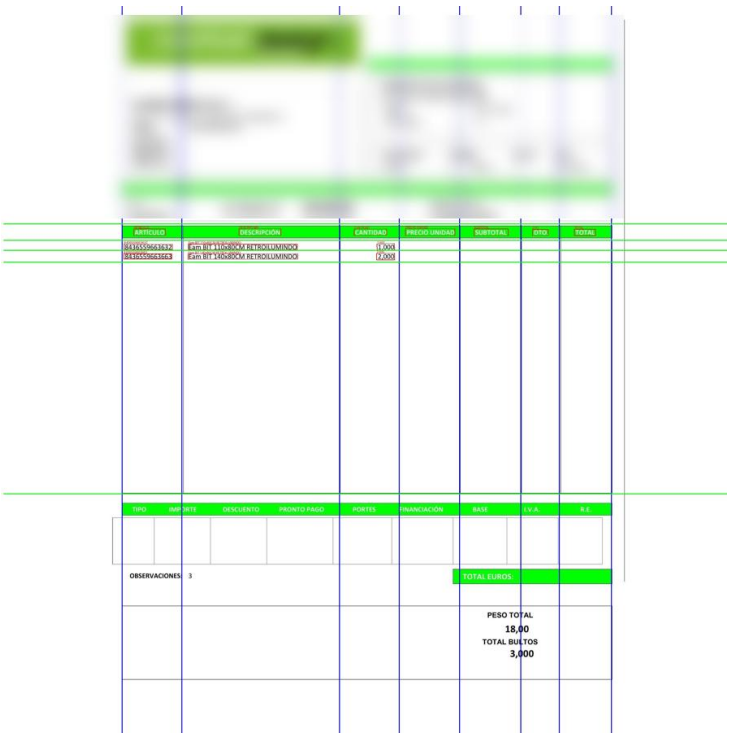


Figura 5.63 Exemplo A.

ARTÍCULO	DESCRIPCIÓN	CANTIDAD	PRECIO UNIDAD	SUBTOTAL	DTO	TOTAL
8436559663632	Eam BIT 110x80CM RETROILUMINDO	1,000				
8436559663663	Eam BIT 140x80CM RETROILUMINDO	2,000				

Figura 5.64 Excel - Exemplo A.

Na Figura 5.63 é possível observar que não ocorreu a sobreposição entre uma caixa delimitadora de texto com mais do que uma caixa delimitadora de célula de tabela. Em todos os casos em que esta situação ocorreu, a extração de tabela funcionou de acordo com o esperado, como é possível observar na Figura 5.64.

- Exemplo B

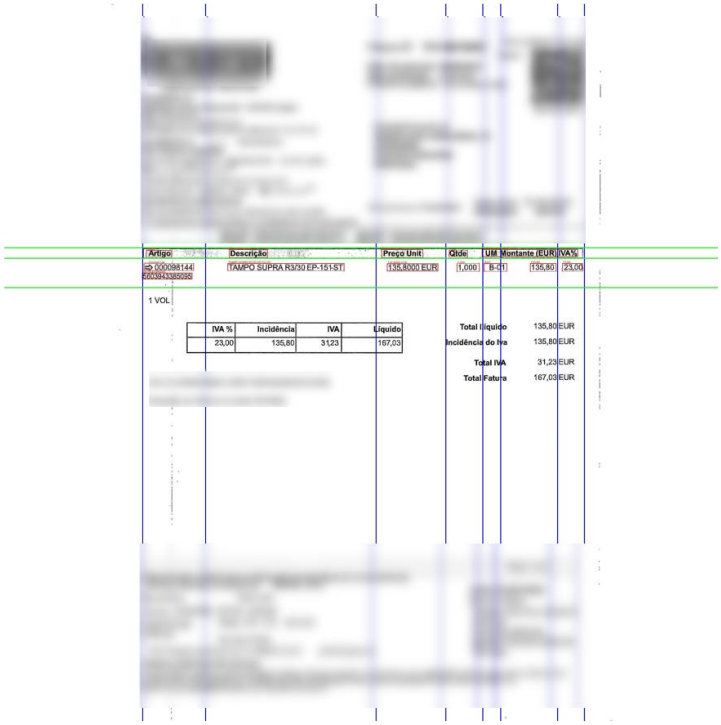


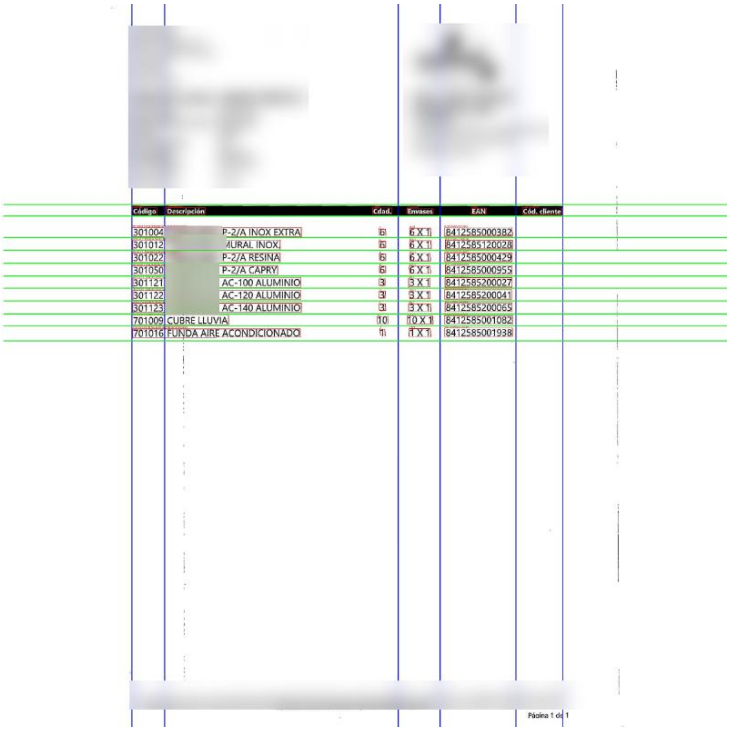
Figura 5.65 Exemplo B.

Artigo	Descrição	Preço Unit	Qtde	UM	Montante (EUR)	IVA%
5603943385095 => 000098144	TAMPO SUPRA R3/30 EP-151ST	135,8000 EUR	1,000	B-01	135,80	23,00

Figura 5.66 Excel - Exemplo B.

Na Figura 5.65, contrariamente ao que aconteceu na Figura 5.63, ocorreu a sobreposição de uma caixa delimitadora de texto com mais do que uma célula da tabela. Com as restrições implementadas na etapa de extração da estrutura da tabela, foi possível separar o texto agrupado indevidamente, de forma a alcançar o resultado desejado na formatação do documento do tipo `JSON`, representada no documento Excel da Figura 5.66.

- Exemplo C



Código	Descripción	Cdad.	Envases	EAN	Cód. cliente
301004	P-2/A INOX EXTRA	6	6 X 1	8412585000382	
301012	MURAL INOX	6	6 X 1	8412585120028	
301022	P-2/A RESINA	6	6 X 1	8412585000429	
301050	P-2/A CAPRY	6	6 X 1	8412585000955	
301121	AC-100 ALUMINIO	3	3 X 1	8412585200027	
301122	AC-120 ALUMINIO	3	3 X 1	8412585200041	
301123	AC-140 ALUMINIO	3	3 X 1	8412585200065	
701009	CUBRE LLUVIA	10	10 X 1	8412585001082	
701016	FUNDA AIRE ACONDICIONADO	1	1 X 1	8412585001938	

Página 1 de 1

Figura 5.67 Exemplo C.

Código	Descripción Cdad.	Envases	EAN	Cód. cliente
301004	TENDEDERO P-2/A INOX EXTRA 6	6 ×1	8412585000382	
301012	TENDEDERO MURAL INOX. 6	6 × 1	8412585120028	
301022	TENDEDERO P-2/A RESINA 6	6 × 1	8412585000429	
301050	TENDEDERO P-2/A CAPRY 6	6 × 1	8412585000955	
301121	TENDEDERO AC-100 ALUMINIO 3	3× 1	8412585200027	
301122	TENDEDERO AC-120 ALUMINIO 3	3 × 1	8412585200041	
301123	TENDEDERO AC-140 ALUMINIO 3	3× 1	8412585200065	
701009	CUBRE LLUVIA 10	10 X 1	8412585001082	
	701016 FUNDA AIRE ACONDICIONADO 1	1× 1	8412585001938	

Figura 5.68 Excel - Exemplo C.

Na Figura 5.67, como aconteceu na Figura 5.65, ocorreu a sobreposição de uma caixa delimitadora de texto com mais do que uma coluna. Neste caso, a metodologia desenvolvida para a extração das informações da tabela teve uma pequena falha, que se observa na última linha do Excel da Figura 5.68, com a representação da formatação final do ficheiro `JSON` para este fornecedor.

- Exemplo D

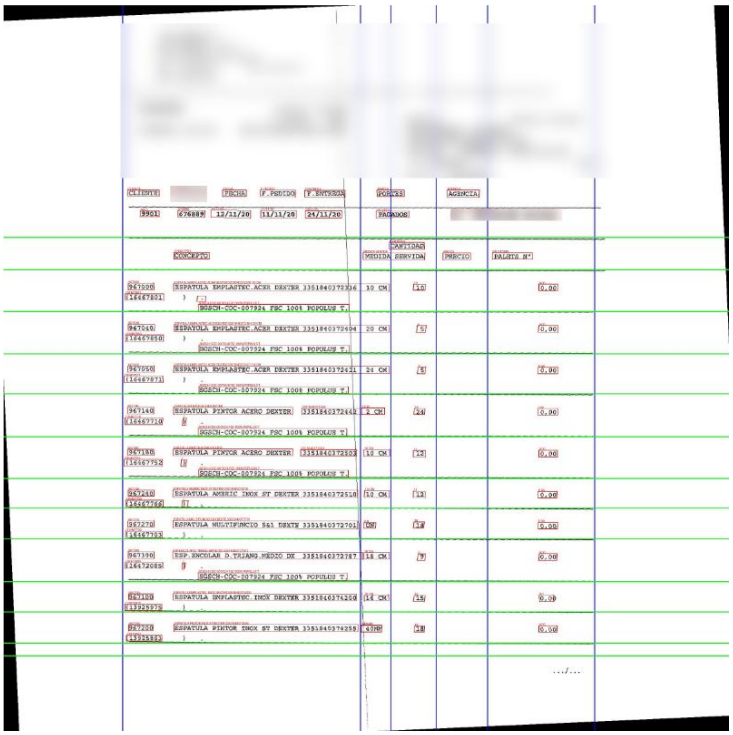


Figura 5.69 Exemplo D.

CONCEPTO	MEDIDA	SERVIDA CANTIDAD	PRECIO	PALETS Nº
(16467801 967000 ESPATULA EMPLASTEC.ACER DEXTER 3351840372336 10 CM . SGSCH-COC-007924 FSC 100% POPULUS T.		10		0,00
(16467850 967040 ESPATULA EMPLASTEC.ACER DEXTER 3351840372404 20 CM SGSCH-COC-007924 FSC 100% POPULUS T.		5		0,00
(16467871 967050 ESPATULA EMPLASTEC.ACER DEXTER 3351840372411 24 CM SGSCH-COC-007924 FSC 100% POPULUS T.		5		0,00
(16467710 967140 ESPATULA PINTOR ACERO DEXTER 1 SGSCH-COC-007924 FSC 100% POPULUS T 3351840372442	2 CM	24		0,00
(16467752 967180 ESPATULA PINTOR ACERO DEXTER 1 SGSCH-COC-007924 FSC 100% POPULUS T. 3351840372503	10 CM	12		0,00
(16467766 967240 ESPATULA AMERIC INOX ST DEXTER 3351840372510 1	10 CM	12		0,00
(16467703 967270 ESPATULA MULTIFUNCIO 5&1 DEXTE 3351840372701	UN	18		0,00
(16472085 967390 ESP.ENCOLAR D. TRIANG.MEDIO DX 3351840372787 | 1 SGSCH-COC-007924 FSC 100% POPULUS T.	ESP.ENCOLAR D. TRIANG.MEDIO DX 3351840372787 | 18 CM	9		0,00
(13925975 967100 ESPATULA EMPLASTEC. INOX DEXTER 3351840374200	16 CM	15		0,0
13925863 967200 ESPATULA PINTOR INOX ST DEXTER 3351840374255	140MM	18		0,00

Figura 5.70 Excel - Exemplo D.

Neste fornecedor, Figura 5.69, ocorreu a mesma situação que se verificou nas Figuras 5.65 e 5.67, com a sobreposição de uma caixa de texto com mais do que uma célula da tabela. Neste caso, a metodologia de extração das informações das tabelas, teve um problema mais significativo na formatação final do ficheiro `JSON`, como é representado pelo Excel da Figura 5.70.

De um modo geral, os resultados da extração das tabelas apresentam um desempenho satisfatório nas células corretamente identificadas e na extração de grande parte dos textos. No entanto, embora a metodologia consiga extrair as informações das tabelas de uma forma adequada na grande maioria dos casos, ainda não tem a capacidade de extrair algumas das tabelas adequadamente, mesmo quando o reconhecimento da estrutura é correto. Podem ser adicionadas restrições para melhorar ainda mais este processo.

O principal desafio consiste na forma como o texto é agrupado pelo OCR na maioria das caixas delimitadoras, o que exige a separação do texto e a perda das coordenadas de início e de fim das palavras, que se encontram no interior da caixa. Este acontecimento gera erros de

sobreposição das informações, que resultam em problemas na inserção das informações nas células.

Apesar das limitações, a implementação permite extrair com sucesso todos os casos em que as palavras do OCR se encontram separadas e numa parte significativa dos casos em que as palavras se encontram indevidamente agrupadas nas deteções do OCR, o que serve de base sólida para melhorias futuras. No caso de se pretender utilizar este OCR, o foco recai sobre o ajuste e a implementação de restrições adicionais, que permitirão corrigir estas falhas e melhorar a precisão global do sistema de extração de informações.

Capítulo 6

Resultados

6.1 Metodologia de Deteção, Reconhecimento e Extração das Tabelas utilizada pela Closer Consulting

A metodologia utilizada pela Closer Consulting, para além da utilização do Azure Forms Recognizer, possui restrições. As informações importantes para os clientes nas tabelas são o Código EAN, a descrição do Produto e a quantidade expedida. O funcionamento do algoritmo é realizado da seguinte forma:

1. Utilização do Azure Forms Recognizer, que deteta e reconhece a estrutura de todas as tabelas presentes na imagem que dá entrada. Após a utilização do Azure Forms Recognizer, são aplicadas muitas restrições, no entanto, devido à impossibilidade de apresentar todas, serão apenas apresentadas algumas;

2. Obtenção da Tabela correta. Existe um conjunto de palavras admissíveis que a tabela principal deve possuir, nomeadamente: EAN, Produto, Quantidade, que consoante os fornecedores, podem ser representadas por diferentes palavras e todas são validadas. Existem várias restrições associadas a este processo e, uma delas, por exemplo, é realizada no caso em que existe mais de uma tabela com as palavras admissíveis consideradas. O que é feito é considerar a tabela que se encontra na região mais central;

3. Reconhecimento da estrutura. Para esta parte, são implementadas várias restrições, por exemplo, quando a tabela é monolinha, isto é, quando os produtos possuem uma única linha com conteúdos associados e multilinha. Nos casos monolinha, ou seja, que têm apenas uma linha dentro de uma célula da tabela, as informações não são agrupadas. Por outro lado, nos casos multilinha, como só deve existir uma quantidade em cada linha da tabela, no caso de a coluna da quantidade possuir um valor e as informações de EAN correspondentes a essa linha. As linhas são agrupadas de forma a que haja um valor quantidade em cada uma delas.

O custo da execução do Azure Forms Recognizer, por 1000 imagens é de $9,364$€, sem o custo de nuvem associado.

6.2 Comparação dos Resultados Obtidos com a Metodologia com os Resultados da Implementação da Closer Consulting

Para efeitos de comparação, foram considerados alguns exemplos de resultados obtidos com o *software* Azure Forms Recognizer, utilizado pela Closer Consulting e de resultados obtidos com a Metodologia desenvolvida ao longo deste projeto, nas mesmas imagens:

- Exemplo 1

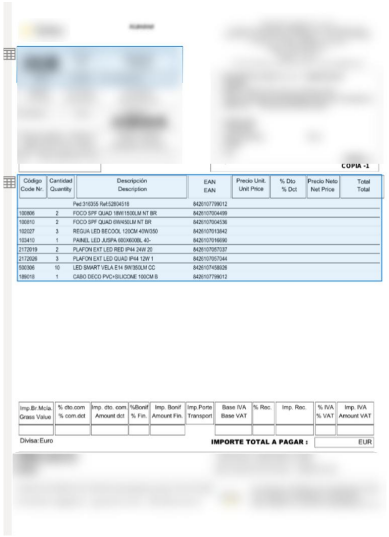


Figura 6.1 Exemplo 1 - Deteção da Tabela pelo Azure Forms Recognizer

Table

Código Code Nr.	Cantidad Quantity	Descripción Description	EAN EAN	Precio Unit. Unit Price	% Dto % Dct	Precio Neto Net Price	Total Total
Ped:316355 Ref:52804518 8426107799012							
100806	2 FOCO SPF	QUAD 18W/1500LM NT BR	8426107004499				
100810	2 FOCO SPF	QUAD 6W/450LM NT BR	8426107004536				
102027	3 REGUA	LED BECOOL 120CM 40W/350	8426107013842				
103410	1 PAINEL	LED JUSPA 600X600BL 40-	8426107016690				
2172019	2 PLAFON	EXT LED RED IP44 24W 20	8426107057037				
2172026	3 PLAFON	EXT LED QUAD IP44 12W 1	8426107057044				
500306	10 LED SMART	VELA E14 5W/350LM CC	8426107458926				
189018	1 CABO	DECO PVC+SILICONE 100CM B	8426107799012				

Figura 6.2 Exemplo 1 - Tabela Extraída pelo Azure Forms Recognizer

Figura 6.3 Exemplo 1 - Deteção da Tabela pela Metodologia Implementada

Code Nr. Código	Cantidad Quantity	Descripción Description	EAN EAN	Precio Unit. Unit Price	% Dto % Dct	Precio Neto Net Price	Total Total
		Ped:316355 Ref:52804518	8426107799012				
100806	2	FOCO SPF QUAD 18W/1500LM NT BR	8426107004499				
100810	2	FOCO SPF QUAD 6W/450LM NT BR	8426107004536				
102027	3	REGUA LED BECOOL 120CM 40W/350	8426107013842				
103410	1	PAINEL LED JUSPA 600X600BL 40-	8426107016690				
2172019	2	PLAFON EXT LED RED IP44 24W 20	8426107057037				
2172026	3	PLAFON EXT LED QUAD IP44 12W 1	8426107057044				
500306	10	LED SMART VELA E14 5W/350LM CC	8426107458926				
189018	1	CABO DECO PVC+SILICONE 100CM B	8426107799012				

Figura 6.4 Exemplo 1 - Tabela Extraída pela Metodologia Implementada

Os resultados obtidos com o Azure Forms Recognizer, representados nas Figuras 6.1 e 6.2 são muito similares aos obtidos com a metodologia desenvolvida, representada nas Figuras 6.3 e 6.4.

- Exemplo 2

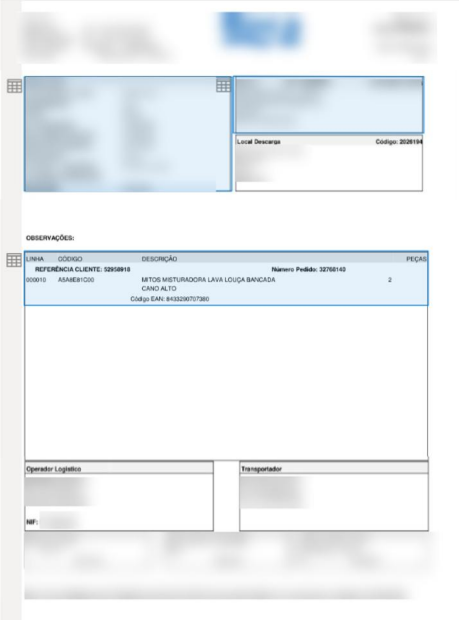


Figura 6.5 Exemplo 2 - Deteção da Tabela pelo Azure Forms Recognizer

Table

LINHA	CÓDIGO	DESCRIÇÃO	PEÇAS
REFERÊNCIA CLIENTE: 52958918 Número Pedido: 32768140			
000010	A5A8E81C00	MITOS MISTURADORA LAVA LOUÇA BANCADA CANO ALTO	2
		Código EAN: 8433290707380	

Figura 6.6 Exemplo 2 - Tabela Extraída pelo Azure Forms Recognizer

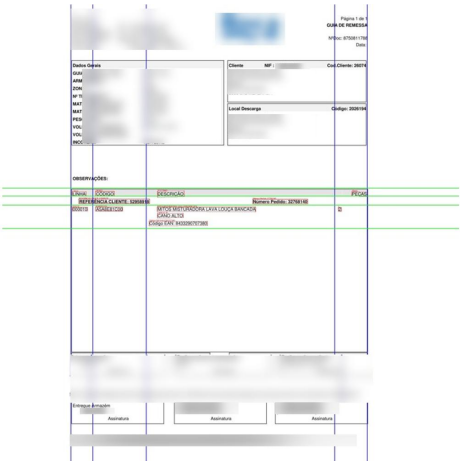


Figura 6.7 Exemplo 2 - Deteção da Tabela pela Metodologia Implementada

LINHA	CÓDIGO	DESCRIÇÃO	PEÇAS
	REFERÊNCIA CLIENTE:	52958918 DESCRIÇÃO Número Pedido: 32768140	
000010	A5A8E81C00	Código EAN: 8433290707380 MITOS MISTURADORA LAVA LOUÇA BANCADA CANO ALTO	2

Figura 6.8 Exemplo 2 - Tabela Extraída pela Metodologia Implementada

Pela observação das Figuras 6.5 e 6.6, dos resultados obtidos pelo Azure Forms Recognizer, e 6.7 e 6.8, com os resultados obtidos pela metodologia desenvolvida, os resultados são idênticos e agrupam as informações dos produtos de forma adequada.

- Exemplo 3

Figura 6.9 Exemplo 3 - Deteção da Tabela pelo Azure Forms Recognizer

Table

Referencia	EAN	Descripción	Medida	Cantidad
4081020300301	4003530012778	CP YELLOX Z1 TR S 25u	3,0x30	5
4081020400601	4003530012952	CP YELLOX Z2 PR S 10u	4,0x60	5
4081020350307	4003530021312	CP YELLOX Z2 TR L 200u	3,5x30	5
4081020350407	4003530021336	CP YELLOX Z2 TR L 150u	3,5x40	5
4081020400457	4003530021411	CP YELLOX Z2 TR L 100u	4,0x45	5
4081020400507	4003530021428	CP YELLOX Z2 TR L 100u	4,0x50	5
4081020350302	4003530021916	CP YELLOX Z2 TR M 75u	3,5x30	5
4201010400121	4003530038877	CR WIROX TX TR S 20u	4,0x12	5
4201010450161	4003530038884	CR WIROX TX TR S 20u	4,5x16	5
4201010500161	4003530038891	CR WIROX TX TR S 16u	5,0x16	5
4081020500902	4003530042898	CP YELLOX Z2 PR M 10u	5,0x90	5

Figura 6.10 Exemplo 3 - Tabela Extraída pelo Azure Forms Recognizer

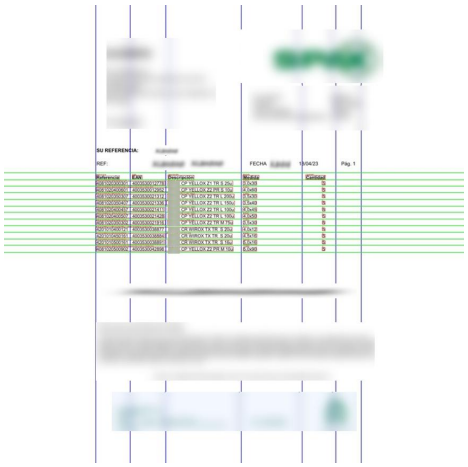


Figura 6.11 Exemplo 3 - Deteção da Tabela pela Metodologia Implementada

Referencia	EAN	Descripción	Medida	Cantidad
4081020300301	4003530012778	CP YELLOX Z1 TR S 25u	3,0×30	5
4081020400601	4003530012952	CP YELLOX Z2 PR S 10u	4,0×60	5
4081020350307	4003530021312	CP YELLOX Z2 TR L 200u	3,5×30	5
	4081020350407	4003530021336 CP YELLOX Z2 TR L 150u	3,5×40	5
4081020400457	4003530021411	CP YELLOX Z2 TR L 100u	4,0×45	5
4081020400507	4003530021428	CP YELLOX Z2 TR L 100u	4,0×50	5
4081020350302	4003530021916	CP YELLOX Z2 TR M 75u	3,5×30	5
4201010400121	4003530038877	CR WIROX TX TR S 20u	4,0×12	5
4201010450161	4003530038884	CR WIROX TX TR S 20u	4,5×16	5
4201010500161	4003530038891	CR WIROX TX TR S 16u	5,0×16	5
4081020500902	4003530042898	CP YELLOX Z2 PR M 10u	5,0×90	5

Figura 6.12 Exemplo 3 - Tabela Extraída pela Metodologia Implementada

Os resultados obtidos com a metodologia implementada neste projeto e com o *software* Azure Forms Recognizer são muito semelhantes. No entanto, existe um valor de "Referência" na Figura 6.12 que se encontra numa célula onde devem corresponder apenas os valores de EAN e o valor de EAN presente na Descrição do produto, o que não aconteceu na tabela extraída pelo *software* utilizado pela Closer Consulting, Figura 6.10.

- Exemplo 4

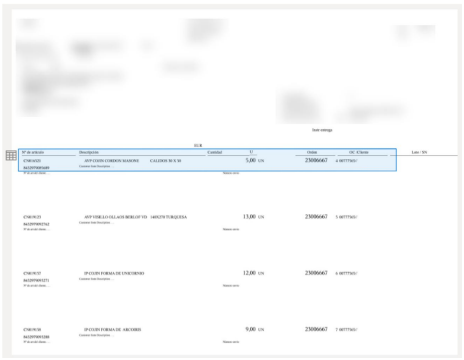


Figura 6.13 Exemplo 4 - Deteção da Tabela pelo Azure Forms Recognizer

Table

Nº de artículo	Descripción		Cantidad	U	Orden	OC /Cliente
CN016521	AVP COJIN CORDON MASONE	CALIDOS 30 X 50		5,00 UN	23006667	4 00777565//
8432979085689 Nº de art del cliente . .	Customer Item Description . . .					
			Número	envio		

Figura 6.14 Exemplo 4 - Tabela Extraída pelo Azure Forms Recognizer

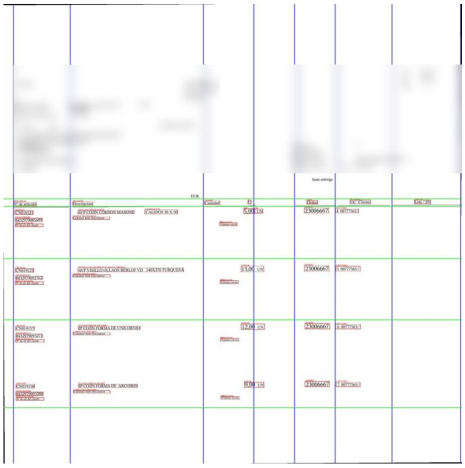


Figura 6.15 Exemplo 4 - Deteção da Tabela pela Metodologia Implementada

Nº de artículo	Descripción	Cantidad U		Orden	OC /Clien Lote / SN
CN016521 8432979085689 Nº de art del cliente . . .	Customer Item Description . . . AVP COJIN CORDON MASONE CALIDOS 30 X 50	Número envio 5,00	UN	23006667	4 00777565//
Nº de art del cliente . . . CN019123 8432979092762	Customer Item Description . . . AVP VISILLO OLLAOS BERLOF VD 140X270 TURQUESA	Número envio 13,00 UN		23006667	5 00777565//
Nº de art del cliente . . . CN019157 8432979093271 8432979093288 CN019158 Nº de art del client	Customer Item Description . . . Customer Item Description . . . IP COJIN FORMA DE UNICORNIO IP COJIN FORMA DE ARCOIRI	Número envio Número envio 12,00 UN 9,00 UN		9,00 UN	12,00 UN 23006667 2300666 6 00777565// 7 00777565//

Figura 6.16 Exemplo 4 - Tabela Extraída pela Metodologia Implementada

Neste caso, os resultados obtidos pelo Azure Forms Recognizer presentes nas Figuras 6.13 e 6.14 aparentam ser menos ajustados à formatação da tabela do que os obtidos com a metodologia desenvolvida, presentes nas Figuras 6.15 e 6.16. Embora a tabela de interesse tenha sido reconhecida na sua totalidade, a divisão que se obteve da sua estrutura não foi a mais adequada nas duas últimas linhas.

- Exemplo 5

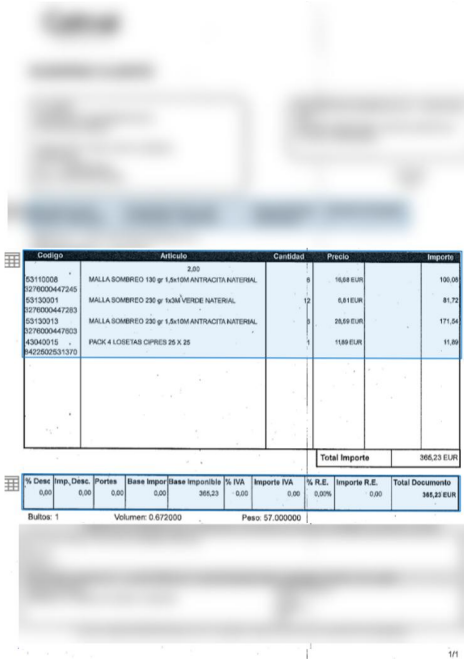


Figura 6.17 Exemplo 5 - Deteção da Tabela pelo Azure Forms Recognizer

Table

Codigo	Articulo	Cantidad	Precio		Importe
	2,00	--...			
53110008 3276000447245	MALLA SOMBREO 130 gr 1,5x10M ANTRACITA NATERIAL	6	16,68 EUR		100,08
53130001	MALLA SOMBREO 230 gr 1x3M VERDE NATERIAL	12	6,81 EUR		81,72
3276000447283					
53130013 3276000447603	MALLA SOMBREO 230 gr 1,5x10M ANTRACITA NATERIAL	6	28,59 EUR		171,54
43040015 . 8422502531370	PACK 4 LOSETAS CIPRES 25 X 25		11,89 EUR		11,89

Figura 6.18 Exemplo 5 - Tabela Extraída pelo Azure Forms Recognizer

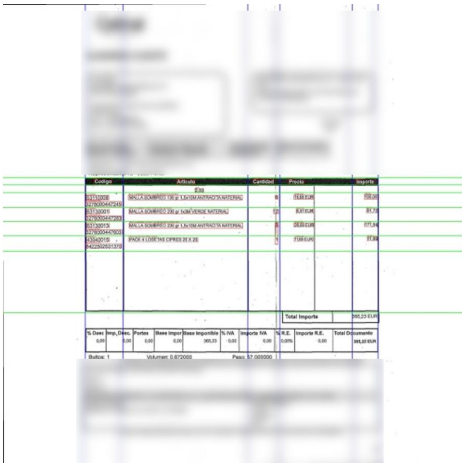


Figura 6.19 Exemplo 5 - Deteção da Tabela pela Metodologia Implementada

Codigo	Articulo	Cantidad	Precio	Importe
	2,00			
3276000447245 53110008	MALLA SOMBREO 130 gr 1,5x10M ANTRACITA NATERIAL	6	16,68 EUR	100,08
3276000447283 53130001	MALLA SOMBREO 230 gr 1x3M VERDE NATERIAL	12	6,81EUR	81,72
3276000447603 53130013	MALLA SOMBREO 230 gr 1,5x10M ANTRACITA NATERIAL	6 CO	28,59 EUR	171,54
8422502531370 43040015	PACK 4 LOSETAS CIPRES 25 × 25	CO	11,89 EUR	11,89

Figura 6.20 Exemplo 5 - Tabela Extraída pela Metodologia Implementada

Neste exemplo, observam-se resultados mais adequados à estrutura da tabela na metodologia de deteção, reconhecimento e extração implementada, relativamente aos resultados obtidos com o Azure Forms Recognizer.

Como foi possível observar, a Metodologia de deteção e reconhecimento automático de tabelas em documentos implementada no projeto obteve resultados semelhantes aos do *software* que a Closer Consulting utiliza de base, onde é necessária a implementação de restrições à sua saída. O custo da deteção, reconhecimento e extração das tabelas em 1000 imagens é significativamente inferior na Metodologia implementada neste projeto, com um valor de $1,848$€, enquanto que o *software* utilizado pela Closer Consulting é de aproximadamente $9,364$€, ambos sem o custo de nuvem associado.

Capítulo 7

Discussão

Neste capítulo vamos proceder à discussão dos resultados à luz dos objetivos definidos na introdução. Discussão sobre as limitações do estudo e do modelo YOLO na análise de tabelas. Indicações de como o modelo pode ser melhorado ou adaptado para novas aplicações.

O estudo realizado foi de encontro aos objetivos estabelecidos no início do Estágio com a Closer, quanto à deteção das tabelas, o reconhecimento das suas estrutura e a extração das informações das tabelas para um documento estruturado `JSON`. No entanto, os resultados poderiam ter sido mais generalizados na tarefa de reconhecimento da estrutura das tabelas para os diferentes fornecedores e a extração das informações poderia ter ficado numa formatação mais precisa.

7.1 Trabalhos Futuros

O trabalho realizado teve a limitação de os Recursos Computacionais serem pagos. Este facto implicou uma redução do número de imagens a utilizar no treino dos modelos. A diminuição do volume de dados pode ter sido a causa de a aprendizagem dos modelos que permitem a obtenção dos resultados na deteção das tabelas e no reconhecimento da sua estrutura não ter sido tão generalizada para alguns fornecedores.

O OCR utilizado é um recurso pago e, por vezes, junta as palavras de diferentes células numa única caixa de texto detetada pelo OCR. Esta caraterística é útil em inúmeras tarefas, porém, no caso da extração das informações das tabelas para um ficheiro estruturado do tipo `JSON`, resultou em algumas dificuldades e tornou necessária a implementação de restrições para separar as informações do OCR, que se encontram agrupadas e para juntar alguns textos que ficam separados, para que estes sejam inseridos nas células corretas.

Duas estratégias que poderiam melhorar os resultados poderiam ser:

- Criação de um OCR gratuito que deteta os textos presentes nas imagens, mas não aglomera as palavras umas com as outras. Deste modo, seria apenas necessário agrupar as palavras que se encontram no interior de cada uma das células;
- Treino do modelo YOLOv8 com um conjunto de imagens maior e com um número mais

equilibrado de ficheiros por cada um dos fornecedores, que permitiria uma melhor generalização da aprendizagem realizada pelo modelo;

- Para a generalização desta metodologia em novos fornecedores, será necessária a criação de um novo conjunto de dados, com o número de imagens idêntico para cada um dos fornecedores, com as imagens dos novos fornecedores, nos quais é necessária a realização do procedimento e treino dos modelos YOLOv8.

Capítulo 8

Conclusão

8.1 Resumo dos Principais Conhecimentos Adquiridos e Aplicações Práticas

Para atingir os requisitos, foram necessários conhecimentos em Inteligência Artificial e no Reconhecimento Ótico de Caracteres (OCR). Foi utilizado o modelo YOLOv8 para detetar os objetos de interesse e foram realizadas adaptações para possibilitar a aplicação do modelo para uma tarefa específica. A forma como as imagens e as anotações são pré-processadas é muito importante para o alcance da eficiência nas deteções e para o funcionamento adequado do modelo. A implementação de restrições no texto extraído permitiu associar corretamente as células das tabelas ao conteúdo a que correspondem. O modelo desenvolvido pode ser aplicado no **processamento automático de guias de remessa**, na **análise de dados tabulares** e no **processamento de grandes volumes de dados em estruturas tabulares**. Com a implementação de melhorias, o modelo poderá ser utilizado em novos fornecedores, ou mesmo, com melhorias mais significativas, em gestão documental e em plataformas automáticas, pois o modelo permite a deteção e o reconhecimento automático de tabelas, o que leva à aceleração de processos manuais, necessário devido ao acréscimo do volume de dados, observado na atualidade.

8.2 Reflexão Pessoal

O trabalho desenvolvido ao longo deste estágio foi desafiante e exigiu a aplicação de conhecimentos relativos a modelos de deteção de objetos, como o YOLOv8, bem como a utilização de um OCR. Foi necessário aplicar restrições às coordenadas extraídas pelo OCR e às coordenadas da saída do modelo YOLO, foi também realizada a extração de dados para ficheiros estruturados no formato `JSON`.

Esta experiência proporcionou um aumento significativo dos meus conhecimentos em aprendizagem automática e em visão computacional. A inteligência artificial é uma área com aplicações em setores muito variados e encontra-se em desenvolvimento constante, com melhorias contínuas nos modelos e nas técnicas disponíveis. Foi possível adquirir conhecimentos

mais aprofundados relativos à importância de otimizar e adaptar os modelos existentes para problemas específicos, que resulta em novas possibilidades para a aplicação prática.

8.3 Conclusão Final

Foi desenvolvido um modelo que permite a deteção de tabelas, das suas estruturas e do reconhecimento das informações das tabelas, através de imagens, nas quais são aplicadas previsões e restrições de coordenadas. Embora os resultados obtidos estejam de acordo com o que se pretendia, existe uma margem de melhorias que consiste na aplicação de mais restrições entre as coordenadas de texto extraídas pelo OCR e as coordenadas das células após a aplicação de restrições gerais às saídas dos modelos YOLOv8, para a melhoria da extração dos documentos.

No futuro, poderia ser interessante a implementação de um OCR, ou a exploração e desenvolvimento de modelos especializados para o reconhecimento de texto em estruturas tabelares.

Bibliografia

[1] Closer Consulting. Data science & ai solutions. Website, 2024. https://closer.pt , Consultado em setembro de 2024.

[2] Papers With Code. Max pooling. Papers With Code, 2024. https://paperswithcode.com/method/max-pooling , Consultado em junho de 2024.

[3] Papers With Code. Ms coco dataset. Papers With Code, 2024. https://paperswithcode.com/dataset/coco , Consultado em janeiro de 2024.

[4] Papers With Code. Yolov3 explained. Papers With Code, 2024. https://paperswithcode.com/method/yolov3 , Consultado em janeiro de 2024.

[5] Wikipedia contributors. Artificial neural network. Wikipedia, The Free Encyclopedia, 2024. https://en.wikipedia.org/wiki/Artificial$_neural_network, Consultadoemjaneirode$2024.

[6] Wikipedia contributors. Perceptron. Wikipédia, a Enciclopédia Livre, 2024. https://pt.wikipedia.org/wiki/Perceptron , Consultado em janeiro de 2024.

[7] Depositphotos. Flying seagull in the sky. Website, 2024. https://depositphotos.com/br/photo/flying-seagull-sky-273093204.html , Consultado em setembro de 2024.

[8] Eixiaoming and Weixiaolin02g. Yolov6: A single-stage object detection framework for industrial deployment. *arXiv*, arXiv:2209.02976, 2022. https://arxiv.org/abs/2209.02976 . Consultado em setembro de 2024.

[9] Jia Deng et al. Imagenet: A large-scale hierarchical image database. Papers With Code, 2024. https://paperswithcode.com/dataset/imagenet , Consultado em fevereiro de 2024.

[10] Joseph Redmon et al. Yolov1: You only look once: Unified, real-time object detection. Papers With Code, 2024. https://paperswithcode.com/method/yolov1 . Consultado em setembro de 2024.

[11] Evalyze. Plataforma de gestão de operações com inteligência artificial. Website, 2024. https://www.evalyze.com/pt/home-pt/ , Consultado em setembro de 2024.

[12] Diana Fonseca. Metodologia para detecção, reconhecimento e extração de tabelas. GitHub, 2022. https://github.com/Diana-Fonseca/metodologia$_detr_ec_ext_tabelas/tree/main, Consultadoemsetembrode$2024.

[13] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. arXiv preprint arXiv:1506.02640, 2015. https://arxiv.org/pdf/1506.02640 , Consultado em setembro de 2024.

[14] Y. Huang, Q. Yan, Y. Li, Y. Chen, X. Wang, L. Gao, and Z. Tang. A yolo-based table detection method. 2019 International Conference on Document Analysis and Recognition (ICDAR), 2019.

[15] IEEE. Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detection. *IEEE Xplore*, 2024. https://ieeexplore.ieee.org/document/10204762 . Consultado em setembro de 2024.

[16] Nickson Joel. Yolov8 - the ultimate guide (beginners to advanced). YouTube, 2024. https://www.youtube.com/watch?v=HQXhDO7COj8 , Consultado em junho de 2024.

[17] Keylabs. Under the hood: Yolov8 architecture explained. Keylabs Blog, 2024. https://keylabs.ai/blog/under-the-hood-yolov8-architecture-explained/ , Consultado em julho de 2024.

[18] H. Law and J. Deng. Cornernet: Detecting objects as paired keypoints. European Conference on Computer Vision (ECCV 2018), 2018. https://paperswithcode.com/method/cornernet , Consultado em janeiro de 2024.

[19] Neural Learn. Extract tables from pdf and convert to excel sheet with paddle ocr text detection and recognition. YouTube, 2022. https://www.youtube.com/watch?v=HZh31OGiQRQ , Consultado em março de 2024.

[20] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. arXiv preprint arXiv:1612.03144, 2017. https://arxiv.org/pdf/1612.03144 , Consultado em julho de 2024.

[21] X. Lu, B. Li, Y. Yue, Q. Li, and J. Yan. Grid r-cnn. IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2019), 2019. https://paperswithcode.com/paper/grid-r-cnn , Consultado em janeiro de 2024.

[22] C. Ma, W. Lin, L. Sun, and Q. Huo. Robust table detection and structure recognition from heterogeneous document images. arXiv preprint arXiv:2203.09056v2, 2022.

[23] Microsoft. Document intelligence - azure ai services. Microsoft Learn, 2024. https://learn.microsoft.com/en-us/azure/ai-services/document-intelligence/?view=doc-intel-4.0.0 , Consultado em setembro de 2024.

[24] Microsoft. Ncast4_v3 size series - azure virtual machines. Microsoft Learn, 2024. https://learn.microsoft.com/en-us/azure/virtual-machines/nct4-v3-series , Consultado em setembro de 2024.

[25] Microsoft. Ocr - optical character recognition - azure ai services. Microsoft Learn, 2024. https://learn.microsoft.com/en-us/azure/ai-services/computer-vision/overview-ocr , Consultado em setembro de 2024.

[26] MMYOLO. Yolov8 description: Inference process. MMYOLO Documentation, 2024. https://mmyolo.readthedocs.io/en/latest/recommended$_{t}opics/algorithm_{d}escriptions/yolov8_{d}escription.htm$ $process, Consultadoemsetembrode2024.$

[27] M. O. Perez-Arriaga, T. Estrada, and S. Abad-Mota. Tao: System for table detection and extraction from pdf documents. Proceedings of the Twenty-Ninth International Florida Artificial Intelligence Research Society Conference, 2016.

[28] Roboflow. A thorough breakdown of yolov4. Roboflow Blog, 2024. https://blog.roboflow.com/a-thorough-breakdown-of-yolov4/ , Consultado em setembro de 2024.

[29] Roboflow. What's new in yolov8? Roboflow Blog, 2024. https://blog.roboflow.com/whats-new-in-yolov8/ , Consultado em julho de 2024.

[30] J. A. Rodrigues. Métodos matemáticos para o processamento de imagens. Mestrado em Matemática Aplicada para a Indústria, 2023. Dezembro de 2023.

[31] Adrian Rosebrock. A better, faster, and stronger object detector: Yolov2. PyImageSearch, 2024. https://pyimagesearch.com/2022/04/18/a-better-faster-and-stronger-object-detector-yolov2/ . Consultado em setembro de 2024.

[32] Rafael Sakurai. Implementação a estrutura de uma rede neural convolucional utilizando o mapreduce do spark, 2024. https://www.sakurai.dev.br/introducao-big-data/ , Consultado em maio de 2024.

[33] A. Simlina. Yolov8 architecture explained. Medium, 2024. https://abintimilsina.medium.com/yolov8-architecture-explained-a5e90a560ce5 , Consultado em setembro de 2024.

[34] PyTorch Team. torch.nn.silu. PyTorch Documentation, 2024. https://pytorch.org/docs/stable/generated/torch.nn.SiLU.html , Consultado em junho de 2024.

[35] TechTarget. Ocr (optical character recognition). TechTarget, 2024. https://www.techtarget.com/searchcontentmanagement/definition/OCR-optical-character-recognition; Consultado em setembro de 2024.

[36] TensorFlow. tf.image.non_max_suppression. TensorFlow Documentation. https://www.tensorflow.org/api$_{d}ocs/python/tf/image/non_{m}ax_{s}uppression, Consultadoemmarode2024.$

[37] Ultralytics. Coco dataset. Ultralytics Documentation, 2024. https://docs.ultralytics.com/datasets/detect/coco/ , Consultado em setembro de 2024.

[38] Ultralytics. How can ultralytics yolo be used for real-time object tracking. Ultralytics Documentation, 2024. https://docs.ultralytics.com/how-can-ultralytics-yolo-be-used-for-real-time-object-tracking , Consultado em setembro de 2024.

[39] Ultralytics. Yolo performance metrics. Ultralytics Documentation, 2024. https://docs.ultralytics.com/guides/yolo-performance-metrics/ , Consultado em setembro de 2024.

[40] Ultralytics. Yolov5: Pretrained models on pytorch hub. PyTorch Hub, 2024. https://pytorch.org/hub/ultralytics$_yolov5/.Consultadoemsetembrode2024.$

[41] Viso.ai. Yolov3: Real-time object detection algorithm. Viso.ai, 2024. https://viso.ai/deep-learning/yolov3-overview/ . Consultado em setembro de 2024.

[42] A. Yadav. From yolo to yolov8: Tracing the evolution of object detection algorithms. Medium, 2024. https://medium.com/nerd-for-tech/from-yolo-to-yolov8-tracing-the-evolution-of-object-detection-algorithms-eaed9a982ebd , Consultado em setembro de 2024.

[43] YOLOv8. Yolov8 architecture explained. YOLOv8.org, 2024. https://yolov8.org/yolov8-architecture-explained/ , Consultado em setembro de 2024.