الجمهورية العربية السورية
المعهد العالي للعلوم التطبيقية والتكنولوجيا
قسم المعلوميات
العام الدراسي 2018/2019

مشروع السنة الرابعة

التعرف على لوحات السيارات بالأرقام العربية

تقديم

فارس غزاوي

إشراف

الدكتور: ياسر رحال

الأستاذ: شادي بلول

كلمة شكر

أتوجه بجزيل الشكر إلى أسرة المعهد العالي كمؤسسة تعليمية راقية. وأخص بالشكر كادر قسم المعلوميات.
وأخص بالذكر المشرفين على هذا المشروع، الدكتور ياسر رحال. والمهندس شادي بلول على ملاحظاتهم و توجيهاتهم في جميع خطوات هذا المشروع.
وأحب أن أوجه شكر أيضاً إلى المهندس علي عموري و المهندس بشار ونوس على مساعدتهم و تخصيص وقتهم لمساعدتي خلال فترة العمل على المشروع.

فارس غزاوي

الخلاصة

تعد عملية التعرف على المحارف من الصور من أبرز التطبيقات التي تشغل عصرنا الحالي، حيث أنه مع تزايد أعداد السيارات في الوقت الحالي، أصبح من الصعب تتبع كل سيارة في مجالات التحكم بالازدحام المروري واستعمالات القوى الأمنية. لذا تولدت الحاجة لنظام تعرف تلقائي على لوحات السيارات (ALPR) كونه قادر على توفير المعلومات اللازمة لحل مشاكل تتبع السيارات. يقدم البحث الآتي طريقة لإنتاج نظام تعرف تلقائي على الأرقام العربية في اللوحات السورية، يتعامل مع صور أو مقاطع فيديو لسيارات مختلفة ، تمر صورة الدخل بثلاث مراحل داخل النظام و هي تحديد موقع السيارة من صورة الدخل وذلك باستخدام YOLOv2 و DarknetAPI، تحديد موقع لوحة السيارة من الهيكل السيارة و ذلك باستخدام WPOD-NET ، التعرف على الأرقام العربية في لوحة و ذلك بتدريب Tesseract على التعرف على الأرقام العربية باستخدام معطيات تدريب مستخلصة من كاميرات المراقبة لفرع شركة سيرتيل في صحنايا و بالاستعانة بمكتبة open-ALPR، ثم استعماله داخل النظام المقترح للتعرف على الأرقام العربية في صور لوحات السيارات السورية.

Abstract

Character recognition is one of the most important applications in our modern era, with the number of vehicles increasing day by day; it is becoming difficult to keep track of each vehicle for the purpose of law enforcement and traffic management. Thus, we need Automatic license plate Recognition system (ALPRs) which is capable of providing appropriate information to solve vehicle-tracking issues. The following paper presents a Methodology to develop an ALPR system which is able to identify the Arabic numbers in Syrian license plates, this system deals with images and video tracks as input, each input image is passed through three phases inside the system, the first phase is Vehicle detection using yolov2 and DarknetAPI, second phase is license plate detection using WPOD-NET ,and the third phase is training tesseract to recognize Arabic numbers using our own Dataset which is extracted from the monitoring cameras in Syriatel branch in sahnaya , Syria ,then use the trained tesseract to identify the Arabic numbers in Syrian license plate image.

الفهرس

قائمة الأشكال

قائمة الجداول

الاختصارات

ALPR: Automatic License Plate Recognition.

OCR: Optical Character Recognition.

AI: Artificial Intelligence.

DL: Deep Learning.

SVM: Support Vector Machine.

CNN: Convolution Neural Network.

R-CNN: Regions with Convolution Neural Network.

LPB: Local Binary Pattern.

WPOD-NET: Wrapped Planer Object Detection Network.

Open CV: Open source Computer Vision library.

المصطلحات

• الذكاء الصنعي	Artificial intelligence
• تعلم الآلة	Machine Learning
• التعلم المشرف عليه	Supervised learning
• التعلم الغير مشرف عليه	Unsupervised learning
• التعلم بالتعزيز	Reinforcement learning
• التصنيف	Classification
• العنقدة	Clustering
• السمات	Features
• التعلم العميق	Deep Learning
• استخلاص الأغراض	Object Detection
• التعرف على الأغراض	Object Recognition
• العنونة	labeling
• مصنف	classifier
• الانكفاء	Regression
• توابع التفعيل	Activation Functions
• تلفيف	Convolution
• تجميع	Pooling
• شبكة عصبونية	Neural Network
• المربع المحيط	Bounding box
• المعالجة المسبقة	Preprocessing
• مجموعة المعطيات	Dataset

- معالجة الصور image Processing
- التجزئة Segmentation
- قيمة العتبة Threshold value
- منحنيات بيانية Histogram
- العناصر المتصلة Connected Component
- تقييس Scaling
- التحويل لقيم ثنائية Binarization

الفصل الأول

التعريف بالمشروع

يمهد هذا الفصل للمشروع، حيث يبين فكرة المشروع وأهميته والأهداف المرجوة منه. يذكر المتطلبات الوظيفية وغير الوظيفية للمشروع.

1-1- مقدمة

العديد من التطبيقات التي تعالج المشاكل المتعلقة بالمواصلات، كالعثور على السيارات المسروقة، منح صلاحيات لسيارات معينة لدخول المراب ومراقبة الطرقات، تتطلب في مرحلة من المراحل ضرورة التعرف على السيارة، والتي يمكن تنفيذها بشكل أتوماتيكي باستخدام نظم التعرف التلقائي على لوحات السيارات.

إن التقدم والتطور المتسارع في مجالات تعلم الآلة (ML) والتعلم العميق (DL) ساهمت في تحسين قدرة الحاسوب على استخلاص الأغراض (Object Detection) والتعرف عليها (Object Recognition) من صور المختلفة، كما أنها عززت قدرته على التعرف على المحارف بصريا (OCR optical character recognition)، والذي في النهاية يؤدي إلى تطور نظم ALPR وزيادة دقتها.

1-2- أهمية المشروع وتطبيقاته

تتنوع التطبيقات التي تستخدم نظم التعرف التلقائي للسيارات، يعد أبرزها في المجالات القانونية كملاحقة السيارات المسروقة، أو المخالفة لقوانين السير على الطرقات، بحيث يفيد النظام المقترح في عملية تتبع السيارات ومعرفة الأرقام العربية الموجودة على اللوحة بشكل أتوماتيكي.

كما يمكن أن يفيد المشروع أصحاب الشركات والمنشئات المختلفة، الذين يحتاجون إلى الاحتفاظ بأرقام السيارات الداخلة والخارجة من منشئات الشركة المعينة، بحيث يمكن ربط هذا التطبيق مع بنية معطيات الخاصة بالموظفين واستعمال النظام لمعرفة السيارات الداخلة للمنظمة وإلى من تتبع ومواعيد دخولها وخروجها بما يشكل نظام مراقبة كامل لهذه الشركة.

وفي حال الرغبة في استخدام tesseract (محرك ال OCR المستعمل في المشروع الحالي) للتعرف على الأرقام العربية في أي تطبيق آخر (التعرف على الأرقام العربية في صور أخرى غير صور لوحات السيارات)، يمكن للمطور الاستفادة من ملف arsy.traineddata والذي يشكل ملف تدريب لمحرك tesseract من أجل التعرف على الأرقام العربية، تم تطوير هذا الملف أثناء العمل على هذا المشروع باستخدام مجموعة التدريب الخاصة بنا.

1-3- المتطلبات

نسرد فيما يلي المتطلبات الوظيفية وغير الوظيفية للمشروع.

1-3-1- المتطلبات الوظيفية:

يجب تصميم نظام ALPR يتميز بالمزايا الآتية:

- أن يكون قادراً على استخلاص هيكل السيارة من واجهة الصورة والتخلص من الخلفية وإيضاح حدود السيارة.
- أن يكون قادراً على تحديد موقع لوحة السيارة السورية من هيكل السيارة (لوحات السيارات السورية بشكليها المستطيل والمربع) وإيضاح حدودها واستخلاصها.
- أن يكون قادراً على تحديد موقع الأرقام العربية من شكل اللوحة الكلي.
- أن يكون قادراً على استخلاص الأرقام العربية من اللوحة والتعرف عليها بدقة مقبولة.
- أن يكون قادراً على انتاج خرج يتضح فيه موقع كل من السيارة واللوحة والتمثيل النصي لأرقام لوحة السيارة بعد تعرف النظام عليها.
- أن يكون قادراً على التعرف على الصور الجديدة للوحات سيارات لم يدرب عليها.
- أن يكون قادراُ على التعامل مع مقاطع الفيديو كدخل.

1-3-2- المتطلبات الغير وظيفية:

- السرعة: يجب أن يتعرف النظام على الأرقام العربية بسرعة مقبولة.
- الوضوح وسهولة التعامل: يجب أن يوفر النظام سهولة في الاستخدام وآلية للمساعدة في حال عدم القدرة على الاستعمال.
- الفعالية والموثوقية: يجب أن يحقق النظام نسبة صحة مقبولة (تتجاوز 75%).

الفصل الثاني

الدراسة النظرية

يبين هذا الفصل أهم المفاهيم المتعلقة بهذا المشروع. يبدأ بتقديم مفاهيم تعلم الآلة وأنواعها، كما يقدم فكرة عن أبرز الخوارزميات المستعملة من أجل كل نوع من أنواع تعلم الآلة. ويوضح في النهاية المنهجية العامة لبناء نظام تعلم آلة.

2-1- تعلم الآلة Machine Learning

تعلم الآلة [18] هو أحد فروع الذكاء الصنعي AI وطريقة من طرق تحليل المعطيات، يقصد بتعلم الآلة مجموعة الأدوات والمفاهيم والمنهجيات المستخدمة لبرمجة الحواسيب بطريقة تسمح لهذه الحواسيب بالتعلم من المعطيات والاستفادة من المعلومات المستخلصة من مرحلة التعلم للتعرف على معطيات غير مدرب عليها بأقل قدر ممكن من التدخل البشري.

ازدادت الحاجة إلى تعلم الآلة بسبب زيادة حجوم وأنواع المعطيات المتوفرة، توفر تقنيات رخيصة نسبياً وبقدرات حسابية عالية، وأيضا تقنيات تخزين بمساحات كبيرة وأسعار منخفضة نسبياً، هذه التقنيات تعني أنه أصبح من الممكن إنتاج نماذج قادرة على تحليل معطيات كبيرة ومعقدة، وتوليد نتائج أكثر دقة بسرعة كبيرة مع تجنب الأخطاء بأكبر قدر ممكن.

معظم الشركات التي تتعامل مع كميات كبيرة من المعطيات لاحظت الحاجة لتوظيف تعلم الآلة في عملها مما يساعدها في العمل بطريقة فعالة أكثر والحصول على أفضلية على المنافسين. حيث لوحظ توظيف تعلم الآلة في مجالات مختلف من الحياة سواء كانت في الخدمات المالية، الاعمال الحكومية، الرعاية الصحية، مجالات الطاقة والنقل وغيرها الكثير.

2-2- العلاقة مع الذكاء الصنعي AI

يمتلك الذكاء الصنعي العديد من التعاريف منها أن الذكاء الصنعي هو دراسة تهدف إلى تدريب الحاسب على أداء مهام يستطيع الإنسان البشري أنجازها بطرق أفضل من الحاسب في حال برمجتها بطرق تقليدية، أي بمعنى آخر هي محاولة إضافة كافة القدرات التي يملكها الإنسان والتي تتصف بالذكاء إلى الحاسب.

من الأخطاء المنتشرة هو أن الذكاء الصنعي هو نظام برمجي، في الحقيقة ال AI يتم تضمينه داخل النظام لمساعدته على أداء مهام تتصف بالذكية وذلك باستعمال خوارزميات وطرائق متعددة.

على الرغم من كون الذكاء الصنعي هو العلم الأكثر ريادة في محاكاة قدرات البشر، إلا أن تعلم الآلة هو فرع من أفرع الذكاء الصنعي الذي يهدف إلى تدريب الآلات على التعلم ذاتيا دون الحاجة ليتم برمجتها على كافة أنواع المعطيات، حيث يهدف إلى تدريبها وتطويرها اعتمادا على الخبرة المكتسبة من المعطيات المدرب عليها.

2-3- مفاهيم أساسية في تعلم الآلة

عادة في تعلم الآلة يتم استعمال رموز المصفوفات والاشعة للدلالة على المعطيات، حيث يتم ترميز المعطيات باستخدام مصفوفات يمثل كل عمود فيها سمة من سمات الدخل ، وكل سطر يمثل معطى من معطيات الدخل مع قيم السمات الخاصة به (data point or observation) وعادة يوجد عمود نهائي في المصفوفة يسمى الهدف، العنوان أو الاستجابة (target، label، response)، وهو يمثل القيمة أو الصف الذي نحاول توقعه لأجل معطيات الدخل المختلفة.

← Features →					Label
Position	Experience	Degree	Country	City	Salary ($)
Developer	0	1	USA	New York	103100
Developer	1	1	UK	New York	104900
Developer	2	2	USA	New York	106800
Developer	3	1	USA	New York	108700
Developer	4	1	USA	New York	110400
Developer	5	1	UK	New York	112300
Developer	6	1	USA	New York	114200
Developer	7	1	Egypt	New York	116100
Developer	8	2	USA	New York	116100
Developer	9	1	USA	New York	119700
Developer	10	1	USA	New York	121600

الشكل 1: مثال على جدول السمات لمعطيات دخل مختلفة.

وقد تحوي بعض خوارزميات تعلم الآلة على بارامترات داخلية إضافية تسمى بالمتحولات الهجينة (hyper parameters).

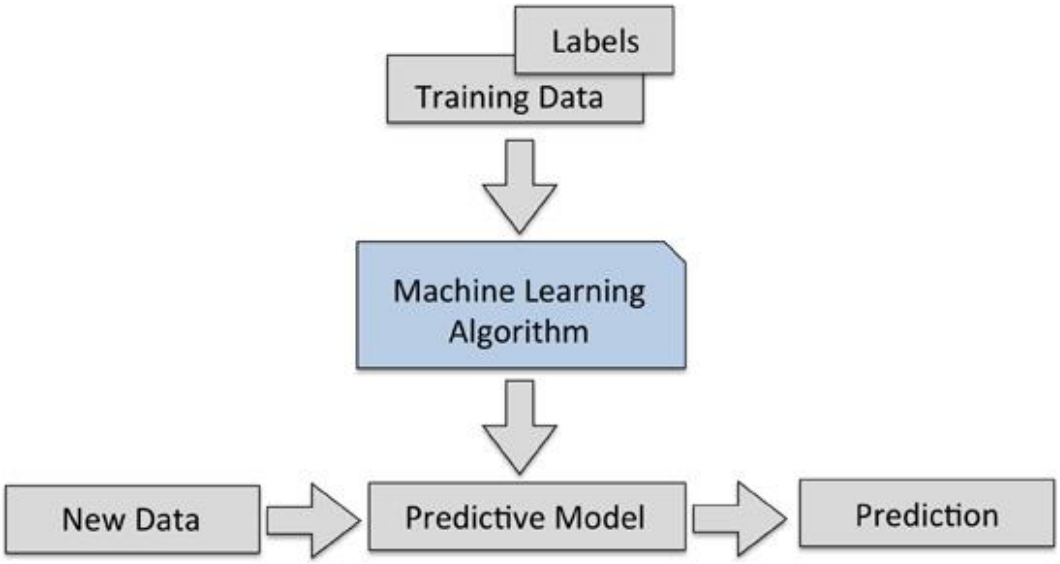
التعميم (أو Generalization) : هو قدرة النموذج الرياضي المولد على القيام بتوقعات على معطيات جديدة لم يتدرب عليها سابقاً.

2-4- طرائق تعلم الآلة

تقسم أنواع طرائق تضمين مفاهيم تعلم الآلة إلى عدة أقسام وهي:

2-4-1- التعلم المشرف عليه:

الخوارزميات التي تعتمد طريقة التعلم المشرف عليه تقوم بتدريب النظام على أمثلة معنونة (Labeled examples) ، بمعنى آخر، الاعتماد على معطيات دخل بحيث يكون من المعروف خرجها المطلوب، على سبيل المثال ليكن لدينا معطيات معنونة و التي نريد فرزها إلى صفين ، اما هذه المعطيات تعطي خرجا false أو true ، تُعطى خوارزمية التعلم مجموعة من معطيات الدخل مع الخرج الصحيح لها و تقوم الخوارزمية بالتعلم عن طريق مقارنة خرجها الفعلي مع الخرج المرغوب به، و عند الانتهاء من عملية المقارنة، تقوم الخوارزمية بتعديل النموذج الرياضي الخاص بها (بالتالي تعديل المتحولات الداخلية) بحيث تتأقلم مع معطيات الدخل أكثر. بمجرد انتهاء مرحلة التدريب على كمية من معطيات الدخل، عندها تقوم الخوارزمية باستعمال النموذج الرياضي النهائي لتوقع الخرج المعنون على دخل غير معنون (دخل غير معروف الخرج أي معطيات دخل لا نعلم إذا كانت false أو true في المثال السابق).



الشكل 2 : البنية العامة لنظم التعلم المشرف عليه

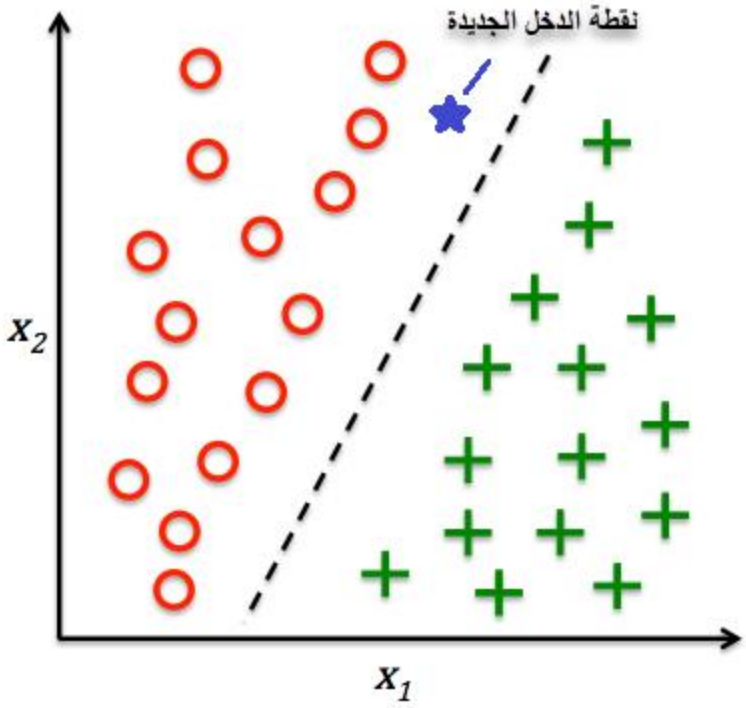
يُستعمل التعلم المشرف عليه في نوعين أساسيين من التطبيقات وهما:

2-4-1-1- التصنيف:

هو نوع من أنواع التعلم المشرف عليه حيث يكون الهدف هو توقع الصفوف المعنونة (Class Labels) أو التصنيفات التي ستخضع لها معطيات الدخل الجديدة اعتمادا على الملاحظات (observation) المبينة خلال عملية التعلم على معطيات التدريب.

من أبرز الأمثلة هو عملية تصنيف رسائل البريد الإلكتروني العشوائية (email spam detection)، وهو نوع من أنواع التصنيف الثنائي (أما أن يكون الرسالة الإلكترونية عشوائية وممثلة بالصف 0 أو ليست عشوائية ممثلة بالصف 1)، هنالك تصنيف متعدد الصفوف أيضاً مثل عملية تصنيف أرقام لوحات السيارات المستعمل في هذا المشروع (حيث أن الصفوف تتراوح بين الأرقام 0 إلى 9)

كمثال على عملية التصنيف الثنائي للحيوانات في الطبيعة إلى صفين (فقارية و غير فقارية) مع السمات (x1,x2)، بعد عملية التدريب يتمكن النموذج الرياضي من إيجاد علاقة بين السمات من أجل كل دخل معطى وصف الخرج المتوقع له، ورسم خط فاصل بين الصفين بحيث يتمكن من أجل معطيات دخل جديدة من تقدير الصف التي تنتمي له نقطة الدخل الجديدة (حيوان جديد) وفقا لقيم ال (x1,x2) (الشكل 3).



الشكل 3: عملية التصنيف لنقطة دخل جديدة اعتماداً على قيم السمات **x1** و **x2**

على افتراض أن الدوائر تمثل صف الحيوانات الفقارية و إشارات الجمع تشكل الحيوانات الغير فقارية (الشكل 3)، هنا يمكن ملاحظة أن نقطة الدخل تقع ضمن فضاء الفقاريات وعليه يقوم النظام بتصنيفها داخل هذا الصف.

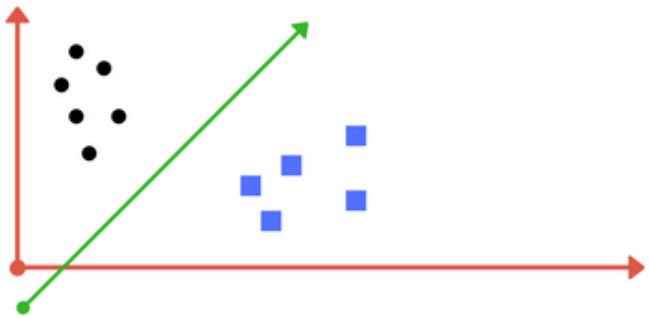
يمكن التعامل مع العديد من الخوارزميات في عملية التصنيف أبرزها:

Naïve Bayes Classifier: و هو مصنف يقوم بحساب احتمال كل معاملات الممكنة (حالات الدخل من أجل قيم السمات) و من ثم يختار الخرج بالاحتمال الأقوى وفق قانون Bayes المعطى بالعلاقة

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)}$$

SVM (Support Vector Machine):

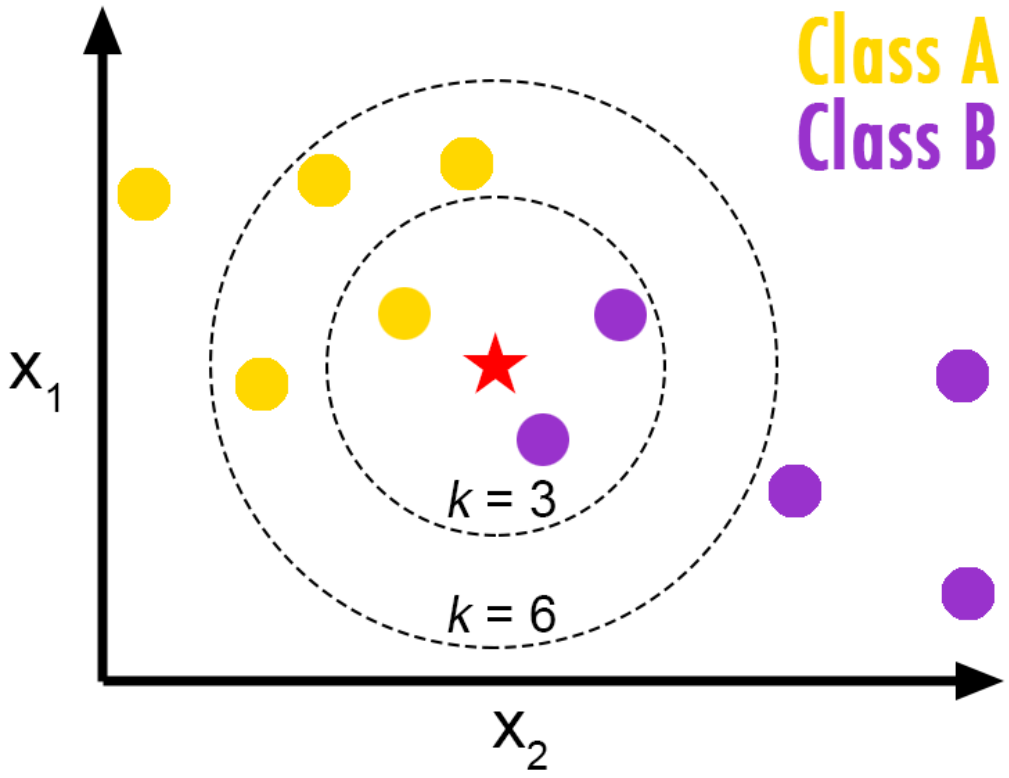
وهو مصنف تمييزي (discriminative classifier) يعتمد على إنشاء شكل هندسي متعدد الأبعاد (hyperplane) لفصل بين معطيات التدريب المعنونة (labeled data). فمثلاً من أجل فضاء ثنائي البعد (عينة تدريب ثنائية السمات) يعمل ال SVM على إنشاء خط مستقيم يقسم السطح إلى قسمين بحيث يتركز كل صف في قسم مختلف.



الشكل 4: فصل **SVM** للصفوف باستعمال خط مستقيم

K Nearest Neighbor:

في هذا النمط من المصنفات، يتم تصنيف معطى الدخل الجديد إلى الصف الذي يحوي أقرب k جار لهذا المعطى (حيث تمثل كلمة جار معطيات الدخل المدرب عليها النظام والتي تحمل قيم للسمات أقرب ما يمكن من المعطى الجديد) حيث k هو عدد طبيعي صغير غالبا، ففي حالة k=1 مثلا يوضع الدخل الجديد في الصف الذي يحوي أقرب جار له.



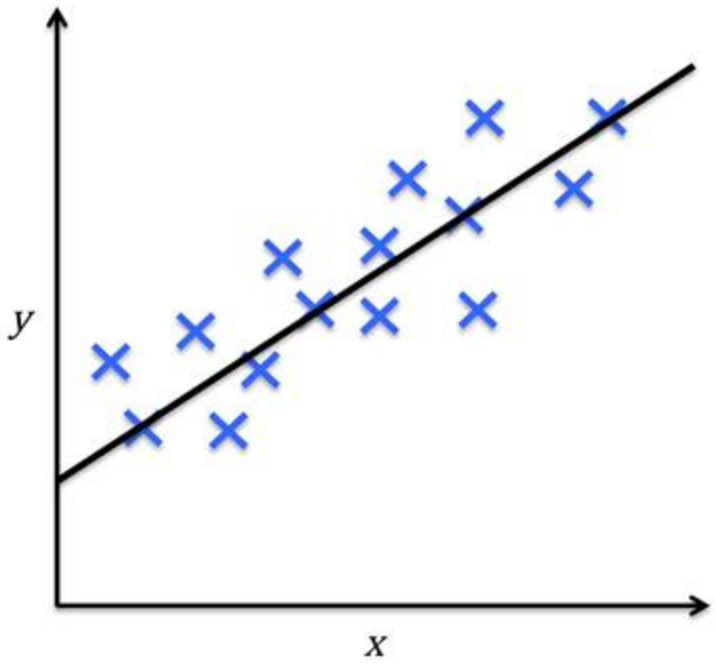
الشكل 5 : استعمال خوارزمية **k-Nearst Neighbor**

فمثلاً (الشكل 5)، العينة الجديدة الممثلة برمز النجمة قد تكون من الصف B في حال أخذ k=3 أو من الصف A في حال أخذ k=6.

2-1-4-2- الانكفاء:

يستعمل أيضاً لتصنيف المعطيات الغير معنونة (unlabeled data)، في هذه النوع من التعلم يعطى النظام مجموعة من متحولات التوقع (predictor variables) ومتحولات مستمرة للخرج (continuous response variables) ويحاول النظام إيجاد علاقة تمكنه من توقع الخرج المستمر.

كمثال يعطى X و Y كمتحولات توقع ويحاول النظام إيجاد منحني الذي يصغر المسافة بين متحولات المستمرة للخرج باستعمال اسلوب رياضي مثل (مثل متوسط المسافات average squared distance) بين نقاط الدخل والمنحني البياني ومن ثم باستعمال هذا الخط تتم عملية توقع لمعطيات دخل جديدة.

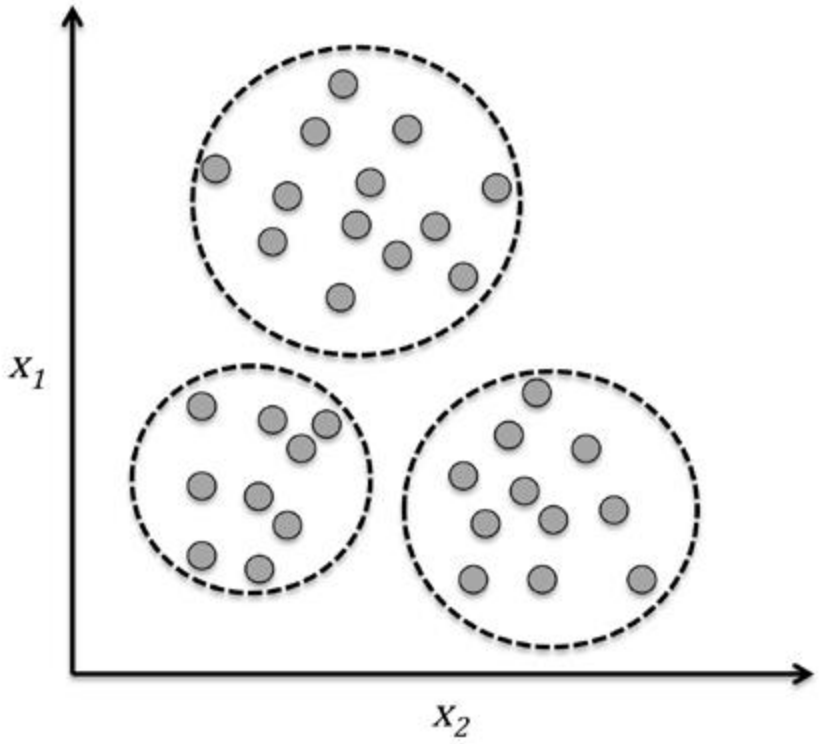


الشكل 6 : عملة الانكفاء من أجل **X** و **Y** كمتحولات توقع

2-4-2- التعلم الغير مشرف عليه:

يستعمل مع الأمثلة الغير معنونة، أي أن النظام لا يعرف الخرج المتوقع لأجل دخل معين ومنه يصبح هدف الخوارزمية هو استكشاف المعطيات وإيجاد بنية بداخل هذه المعطيات، بحيث تقسم هذه البنية المعطيات إلى معطيات معنونة اما عن طريق السمات المشتركة بين هذه المعطيات أو إيجاد السمات الأساسية التي تفصل بين المعطيات.

تستعمل التعلم الغير مشرف عليه في عدة أنواع من التطبيقات نذكر منها العنقدة، هو نوع من أنواع تحليل المعطيات الاستكشافي (exploratory data analysis) تستعمل بهدف تصنيف المعلومات (معطيات الدخل) إلى مجموعات جزئية دون معرفة مسبقة ببنية هذه المجموعات.



الشكل 7: عملية العنقدة

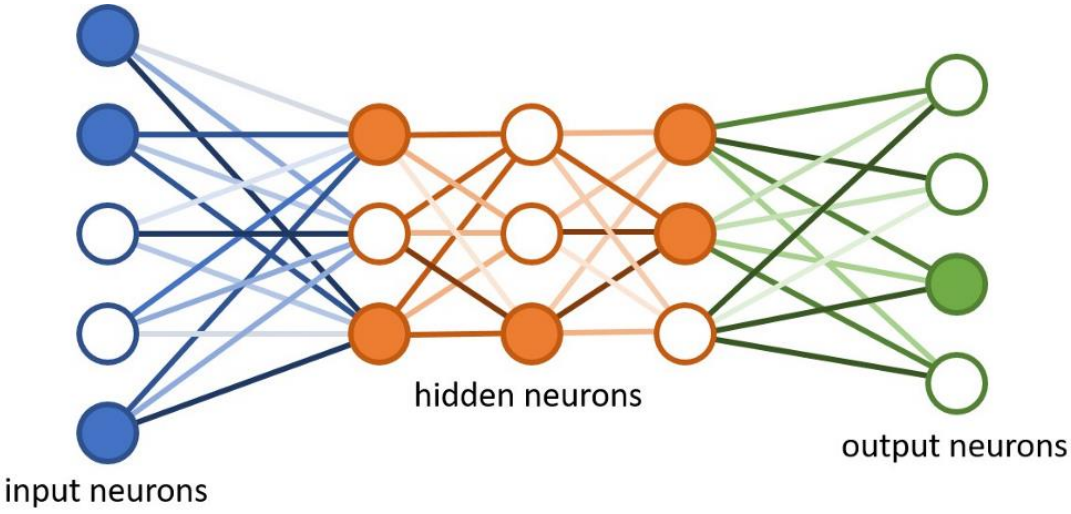
كل مجموعة (Cluster) تحوي عدد من الأغراض (objects) المختلفة عن الأغراض في المجموعات الأخرى.

يمكن تنفيذ العنقدة باستخدام عدد من الخوارزميات أشهرها ال K-means Clustering وهي واحد من أبسط وأشهر خوارزميات العنقدة، تعتمد هذه الخوارزمية على إيجاد عدد ثابت k من المراكز (Centroid) (وهي مواقع تشكل مراكز المجموعات المتشابهة للدخل (cluster)) واسناد كل من معطيات الدخل إلى أقرب مركز.

عند بداية خوارزمية k-means يتم اختيار المراكز بطريقة عشوائية، يتم اسناد معطيات دخل إلى أقرب مركز وثم يعاد حساب المراكز عن طريق إيجاد متوسط جميع النقاط ضمن المجموعة الواحدة، فتنشأ مجموعة من النقاط الجديدة والتي تشكل المراكز الجديدة للمجموعات، وتتكرر الخوارزمية عدة مرات حتى الوصول إلى مراكز ثابتة (لا تتغير مهما أعدنا حساب الخوارزمية) أو الوصول إلى عدد محدد مسبقاً من التكرارات.

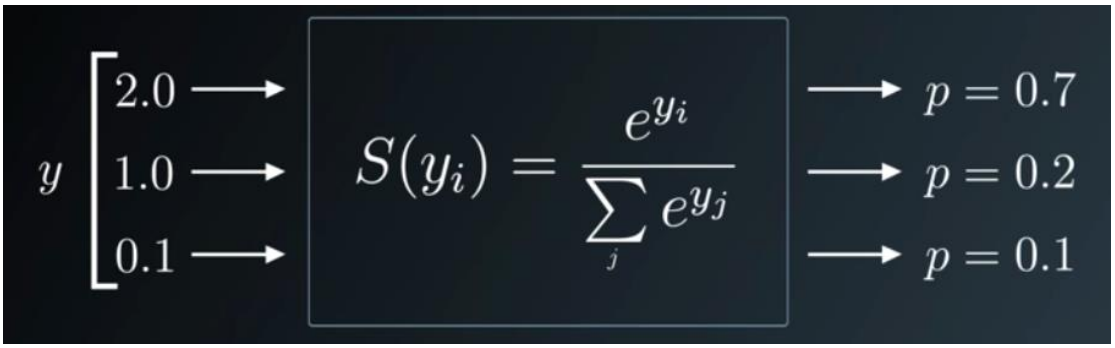
2-4-3- التعلم العميق:

هو فرع من أفرع تعلم الآلة، من أشهر تطبيقاته هو استعمال بنية الشبكات العصبونية ، وهي بينة تتكون من عدة طبقات (layers) من عصبونات (neurons) تتصل هذه الطبقات ببعضها بمتحولات تسمى الأوزان (weights).(الشكل 8)
الطبقة الأولى تأخذ مجموعة من المعطيات تعالجها وتستخلص منها مجموعة من المعلومات، ثم تمرر خرجها إلى الطبقة التالية وهكذا حتى الوصول للطبقة النهائية التي تقوم بالتوقع.



الشكل 8: البينة العامة للشبكات العصبونية

فيما بعد يتم مقارنة التوقع بالخرج المطلوب، ومن ثم باستعمال طريقة تسمى backpropagation يقوم النظام بتعديل قيم الأوزان بين الطبقات حتى الوصول للخرج الدقيق.
في الشبكات العصبونية، لا يتم تمرير خرج جميع العصبونات إلى الطبقة التالية، بل يتم استعمال توابع التفعيل للشبكات العصبونية وهي التوابع التي تحدد في حال كان العصبون تم تفعيله أو لا (سيستعمل خرجه في الطبقة التالية أم لا).
يوجد العديد من توابع التفعيل نذكر منها:
ReLu: وهو تابع تفعيل يعطى بالمعادلة A(x) = max (zero، x) يتميز بقلة التكلفة كونه يستعمل رياضيات بسيطة.
Step function: يعطى بالمعادلة A(x) = one if x > Threshold value else A(x) = zero و هو مفيد في تصميم المصنفات الثنائية (التصنيف علىً صفين لمعطيات الخرج).
Softmax: يعطى بالمعادلة الموضحة في الشكل و يهدف إلى حساب احتمالات الصفوف للخرج النهائي.



يوجد عدة أنواع للشبكات العصبونية نذكر منها:

2-4-3-1- الشبكة العصبونية التلفيفية (CNN):

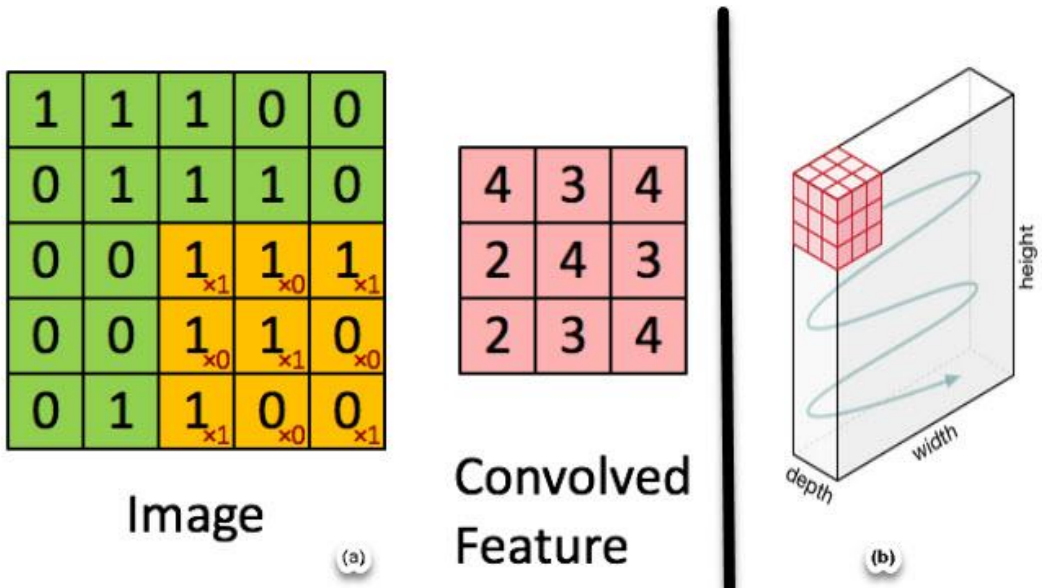
وهي خوارزمية تعلم عميق تأخذ صورة الدخل ومن ثم تعمل على إيجاد الأوزان (weights) من أجل عدة أغراض ضمن الصورة، وذلك لتمييز هذه الأغراض عن الخلفية وعن بعضها البعض.

في هذه الخوارزمية العصبون الواحد (neurons) يتعامل فقط مع منطقة معينة من الدخل تسمى (receptive fields) ومجموع هذه المناطق يشكل الدخل كاملاً.

تتمكن ال CNN من اكتشاف الارتباطات المكانية بين البيكسلات عن طريق تطبيق مجموعة من المرشحات (filters)، كما أن ال CNN تشكل وسيلة أفضل للتعامل مع المعطيات كونها تخفف من عدد المتحولات داخل الشبكة ولقدرتها على إعادة تعيين قيم الأوزان.

من أجل صورة دخل معينة تستطيع CNN أن تحول الصور إلى شكل أسهل معالجته وبدون خسارة أي من السمات، باستخدام عدة طبقات وهي:

1. طبقة التلفيفية (Convolution layer): وتسمى kernel/filter أحياناً هي عبارة عن اختيار مصفوفة k (وفقاً لطبيعة السمة المراد استخلاصها) ، تمر المصفوفة k على بيكسلات الصورة وتجري عملية ضرب بين قيم هذا البيكسل ومقابلاته في مصفوفة الk ,ثم تجمع نتائج الضرب السابقة لتنتج عنصراً في مصفوفة الخرج (الشكل 9) .



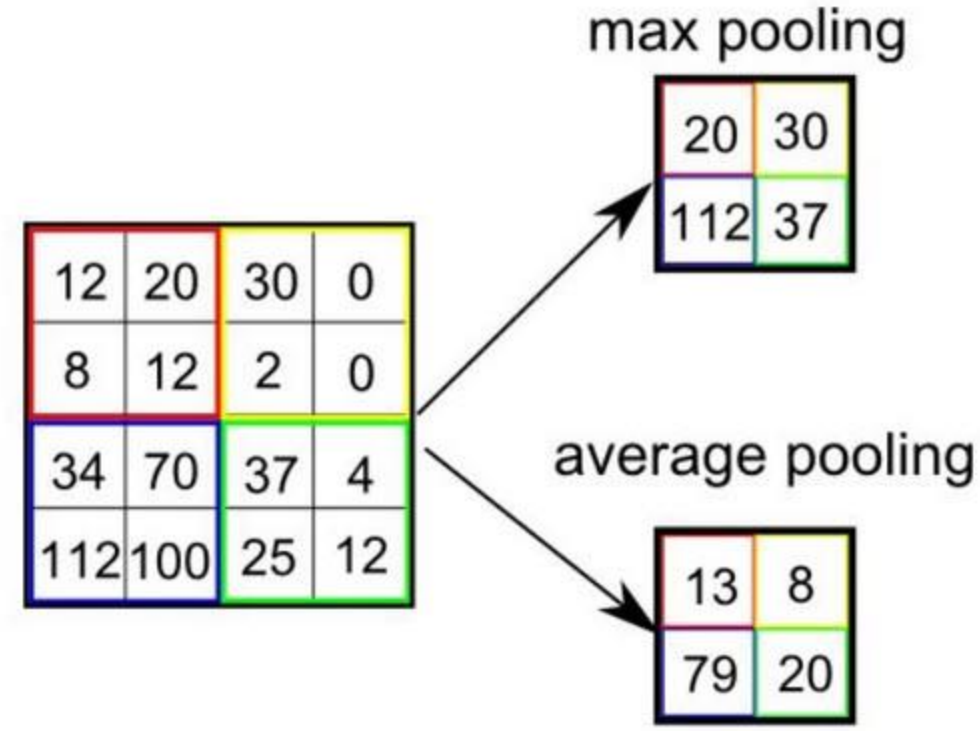
الشكل 9 **(a)** توضح مثال لضرب صورة بمصفوفة **k** (على اليسار) و السمة المستخرجة (على اليمين).**(b)** يوضح مرور المصفوفة **k** على الصورة الدخل ذات الطول و العرض و العمق (القيم اللونية).

الهدف من هذه الطبقة هو استخلاص السمات الأكثر أهمية (الحواف مثلاً) من صورة الدخل.

يجب ألا تقتصر عملية التلفيف على طبقة واحدة، حيث يمكن تقسيمها إلى طبقة لاستخلاص السمات الأساسية أمثال اللون والحواف وإضافة المزيد من الطبقات تباعا لاستخلاص سمات جديدة.

2. طبقة التجميع (Pooling Layer): مشابهة في عملها بالطبقة السابقة، مسؤولة عن تخفيف حجم خرج الطبقة السابقة (convolved feature) من أجل تخفيف الوقت المستهلك في عملية معالجة المعطيات عن طريق تخفيف الأبعاد (dimensionality reduction)، كما أنها مفيدة في استخلاص الخصائص التي لا تتأثر بالدوران وتغير الموقع.

يوجد نوعين من التجميع، Max يرد القيمة الأكبر من قسم الصورة المقابل للمصفوفة k، أما average فترد الأصغر. كما أن ال max يتخلص من الضجيج في الصورة عن طريق التخلص من القيم المشوهة والتخفيف من أبعاد الصورة المدخلة (الشكل 10) .



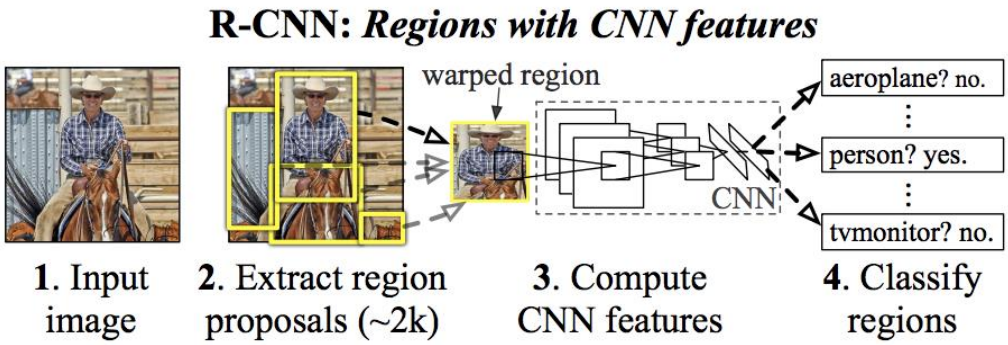
الشكل 10 يوضح تطبيق ال **Max Pooling** و ال **Average pooling** على صورة دخل معينة

3. Neural Network: بعد أجراء العمليات السابقة يتم تحويل الصورة إلى بعد واحد و إدخالها إلى شبكة عصبونية حيث تخضع للتدريب و يمتلك عندها النظام القدرة على التعرف على خصائص الصورة.

R-CNN –2–3–4–2

تعتمد هذه الخوارزمية على اختيار 2000 مقطع من الصورة (مجموعة بيكسلات) تسمى region proposal، ومن ثم تمريرها إلى CNN مما يخفف من عملية معالجة كل البيكسلات في حال الصورة كبيرة الحجم.

يتم اختيار هذه المقاطع باستعمال خوارزمية selective search، حيث يتم إنتاج مقاطع عشوائية في البداية، ثم باستعمال خوارزمية شرهة (greedy algorithm) وبشكل عودي يتم دمج المقاطع معاً لتشكيل مقطع أكبر، ليصبحوا فيما بعد region proposal.



الشكل 11 : مخطط يوضح آلية عمل **R-CNN**

بعد استخلاص ال region proposal يتم تمريرهم على CNN لاستخلاص السمات، ثم تمرير السمات المخرجة من CNN إلى SVM من أجل عملية التصنيف.

تعاني R-CNN من مشاكل أبرزها زمن التنفيذ الطويل، واستخدامها لخوارزمية شرهة مما قد يؤدي إلى اقتراح مقاطع غير مفيدة (لا تعبر عن أغراض في الصورة الحقيقة).

Fast R-CNN –3–3–4–2

مشابهة في عملها ل CNN، ولكن عوضاً عن إدخال ال region proposal إلى CNN، يتم إدخال صورة الدخل إلى CNN من أجل إنشاء Convolution feature map التي تساعد على إيجاد region proposal وإحاطتهم بمستطيلات باستعمال طبقة تسمى ROI pooling layer (Region of interest)، ثم يتم توحيد حجوم هذه المقاطع وتمريرها إلى الشبكة العصبونية.

تتفوق Fast R-CNN على R-CNN لعدم الحاجة لتمرير 2000 مقطع مقترح إلى CNN في كل مرة.

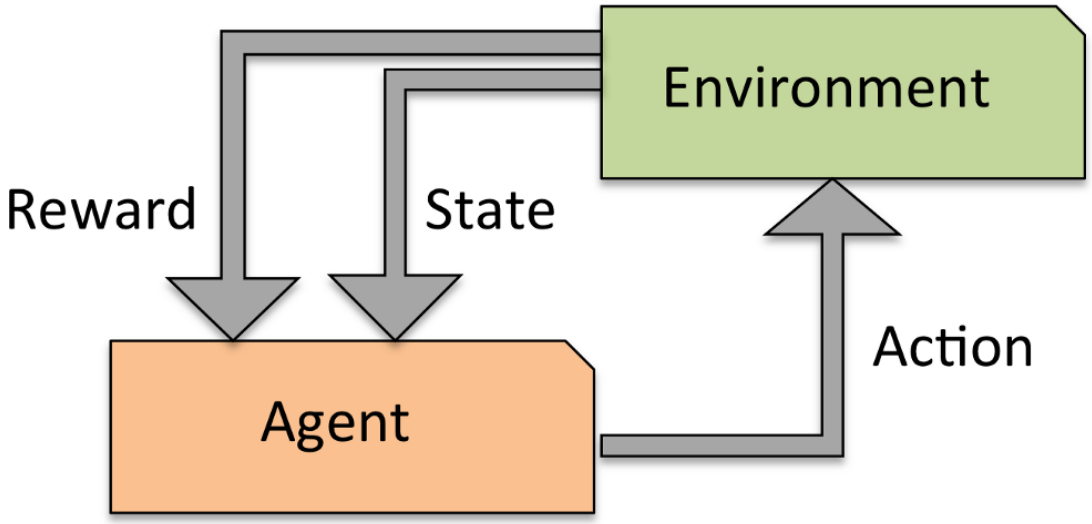
التعلم بالتعزيز: –4–4–2

يستعمل غالباً في الألعاب وتصميم الروبوتات، هذا النوع من التعلم يملك 3 مكونات أساسية وهم

- The agent : وهو المتعلم أو صانع القرار
- The environment : وهو كل ما يحتك معه صانع القرار
- Actions: وهو الأفعال التي يقوم بها صانع القرار.

يكمن الهدف في هذه الطريقة ببناء نموذج يهدف إلى تحسين أداء المتعلم، وتتم عملية التحسين عن طريق منح مكافأة (Reward) في كل مرة يقوم المتعلم بفعل من مجموعة الأفعال التي يحددها المطور، حيث تمثل المكافأة طريقة لقياس مدى صحة هذا الفعل

في سبيل تنفيذ مهمة معينة. عندها يقوم المتعلم باستخدام التغذية الراجعة (feedback) لتحسين تصرفاته المستقبلية من أجل الحصول على أكبر مكافأة ممكنة (الشكل 12).



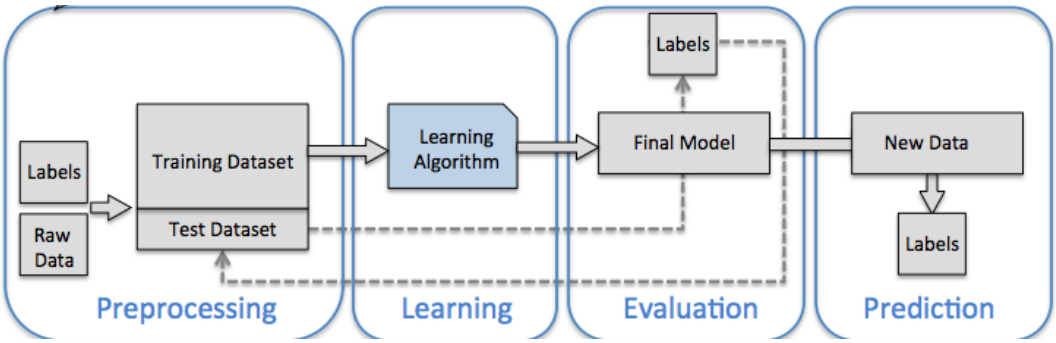
الشكل 12: مفاهيم التعلم بالتعزيز **reinforcment learning** و العلاقة بينهم

كمثال عنها محركات لعب الشطرنج حيث يقوم المتعلم باختيار تحركات لقطع الشطرنج بما يتلاءم مع توزيع الرقعة (والتي هي حالة من البيئة أو ما يسمى ب environment state) وتعطى المكافأة في حال الربح أو خسارة اللعبة.

.

2-5- المنهجية العامة في بناء نظام تعلم الآلة.

هنالك عدة خطوات أساسية خلال عملية بناء نظام تعلم الآلة (الشكل 13)



الشكل 13: الخطوات الأساسية لبناء نظام تعلم الآلة

المعالجة المسبقة (Preprocessing):

هي واحدة من الخطوات الأساسية في أي تطبيق تعلم آلة، تنبع أهمية هذه الخطوة كون معطيات الدخل تأتي غالبا في صيغ (Formats) غير ملائمة للنظام لكي يعالجها، حيث أنه هنالك العديد من الخوارزميات التي تتطلب من السمات أن تمتلك نفس المقياس أو للمعطيات أن تكون بصيغ معينة، بالتالي في هذه المرحلة تتم عملية معالجة المعطيات وتهيئتها بما يلائم النظام، وعند انتهاء المرحلة تقسم مجموعة المعطيات (Dataset) إلى قسمين وهم قسم التدريب (Training Dataset) وقسم الاختبار (testing dataset).

التعلم (Learning): وفيها يتم اختيار النموذج الرياضي أو الخوارزمية التي سيتم بواسطها تدريب الآلة على معطيات التدريب وتطبيقها على هذه المعطيات للخروج بالنموذج النهائي (Final Model).

التقييم (Evaluation): يتم فيها تقييم النموذج النهائي ومدى صحة النتائج الخرج وحساب الأخطاء على معطيات الاختبار (Testing dataset)، لتحديد مدى كفاءة النظام في التعامل مع معطيات جديدة.

التوقع (Prediction): وهو استعمال النظام ككل على معطيات خارجية للقيام بتنبؤات مستقبلية.

الفصل الثالث

نظم ALPR والدراسة المرجعية

يتضمن هذا الفصل تعريف بنظم التعرف التلقائي على لوحات السيارات، مراحل بناء هذه النظم والمشاكل التي تواجها، كما أنه يقدم دراسة مرجعية تتناول الأبحاث التي تم طرحها في مجال تطوير هذه النظم.

3-1- نظم التعرف التلقائي على لوحات السيارات ALPR

نظام التعرف التلقائي على لوحات السيارات (ALPR) [15]: هو نظام يعمل على استخلاص معلومات لوحات السيارات من صورة أو عدة صور.

المعلومات المستخلصة يمكن استعمالها في عدة تطبيقات مثل نظم مراقبة الطرق السريعة، نظم مراقبة مرآب السيارات ولأغراض أمنية.

يعمل نظام ALPR على التعرف على الأرقام الموجودة في لوحات السيارات من الصور المأخوذة من الكاميرات مستعملاً عدة تقنيات متل التعرف على الأغراض ، معالجة الصور والتعرف على الأنماط (pattern recognition).

3-1-1- المشاكل التي يواجها نظام ALPR:

إن تعدد أنواع لوحات السيارات واختلاف بيئة التصوير المحيطة تشكل أبرز العقبات في عملية التقاط والتعرف على لوحات السيارات من الصور. نذكر منها:

1. لوحات السيارات:

- الموقع: اختلاف مواقع لوحة السيارة باختلاف السيارة المعالجة وزاوية تصوير الصورة المدخلة.
- العدد: قد تحوي الصورة المدخلة للنظام على سيارة واحدة، عدة سيارات أو قد لا تحوي أي سيارة.
- الحجم: اختلاف حجم لوحة السيارة داخل الصورة بسبب بعد السيارة عن جهاز التصوير.

- الخطوط: قد يختلف الخط المستعمل في لوحة السيارة باختلاف الدول أو المدن أو حتى بين اللوحة والأخرى.
- أنواع مختلفة: قد تتعدد أنواع لوحات السيارات المستعملة داخل البلد الواحد (الشكل 14).

الشكل 14 : نوعي لوحات السيارات السورية المستعملة للسيارات المدنية

يمكن ملاحظة اختلاف الخط المستعمل في كتابة الأرقام ضمن النوع الواحد من لوحات السيارات (الشكل 14 وخصوصاً من أجل رقمين 2 و3).

وقد يختلف نوع لوحة السيارة تبعاً للجهة التي تتبع لها المركبة (الشكل 15).

الشكل 15 أنواع لوحات السيارات السورية المختلفة تبعاً لجهة عمل المركبة

- معامل الزمن: والذي يظهر على شكل تآكلات في معدن لوحة السيارة.

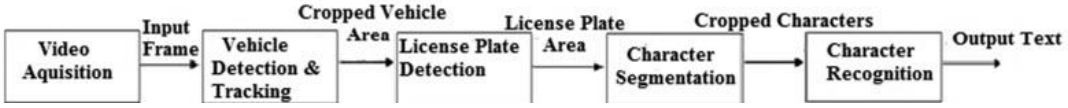
2. بيئة التصوير:

- ظروف الإضاءة: الصور المدخلة إلى النظام قد تختلف شدة وضوحها تبعا لإضاءتها (الصورة ملتقطة صباحاً أم مساءً، تأثير ظلال الأغراض المحيطة بالسيارة).
- الخلفية: قد تحتوي خلفية الصورة على أغراض بنمط (pattern) يشبه النمط المستعمل في التعرف على لوحة السيارة.

3-1-2- مراحل عمل نظام ALPR:

يمكن تلخيص مراحل عمل نظام ALPR بخمس مراحل وهم:

1. **المرحلة الأولى:** الحصول على صورة الدخل من المصادر المختلفة وهنا يجب الأخذ بعين الاعتبار العديد من المعاملات مثل نوع الكاميرا ودقتها، اتجاه الكاميرا والإضاءة المحيطة بالمشهد.
2. **المرحلة الثانية:** استخلاص جميع السيارات من صور الدخل المحضرة في المرحلة الأولى وإخراج عدة صور لكل سيارة مقتطعة.
3. **المرحلة الثالثة:** استخلاص لوحات السيارات من صور السيارات المولدة من المرحلة وتوليد عدد من الصور لكل لوحة سيارة مقتطعة.
4. **المرحلة الرابعة: تجزئة** لوحة السيارة من أجل الحصول على المحارف.
5. **المرحلة الخامسة:** التعرف على المحارف المستخلصة من المرحلة السابقة باستخدام المصنفات.



الشكل 16: مراحل عمل نظام **ALPR** في حال الدخل مقطع فيديو

يمكن تنفيذ المراحل السابقة باستخدام عدة خوارزميات من أجل كل مرحلة من المراحل، وإن قوة نظام التعرف على لوحات السيارات يعتمد على دقة ومتانة الخوارزميات المستعملة في تطويره. تبرز الفقرة التالية مجموعة من نظم ال ALPR والطرائق المستعملة في كل نظام.

3-2- الدراسة المرجعية

بعد الاطلاع على المفاهيم الأساسية المتعلقة بنظم التعرف التلقائي على لوحات السيارات، كان لا بد من قراءة عدد من الأبحاث التي تتقاطع مع المشروع الحالي سواء في مجال تصميم نظام ال ALPR بشكل عام، أو في التعرف على المحارف العربية بشكل خاص، وذلك للاستفادة من الدراسات المطبقة على هذا المجال من التطوير، واختيار خوارزمية مناسبة لإنجاز القسم العملي.

تقدم هذه الفقرة أهم الأبحاث والخوارزميات المستعملة من أجل كل مرحلة من مراحل تطوير نظم الALPR (انظر فقرة 3-1-2-).

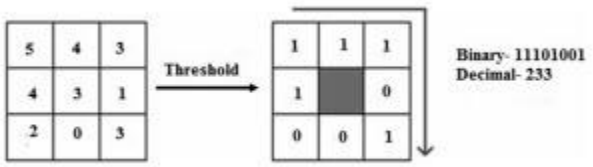
3-2-1- مرحلة استخلاص السيارة:

تهدف في هذه المرحلة على فصل هيكل السيارة عن خلفية المشهد للحصول على صورة واضحة للسيارة.

في [1] تم تنفيذ عمليات معالجة قبل دخول الصورة للنظام (preprocessing) عن طريق تحويل الصورة إلى تدرجات اللون الرمادي أولا (grayscale)، من ثم تحويل الصورة إلى النظام الثنائي عن طريق تحديد قيمة للعتبة (وهي قيمة التي تساعد على قسم البيكسلات على قسمين الواجهة (foreground image) والخلفية المحيطة (background image)). يذكر أن تواجد الأغراض في الخلفية وظلال هذه الأغراض قد يؤثر على عملية اختيار قيمة العتبة.
و من أجل استخراج السيارة، اقترح [1] عملية دراسة كثافة البيكسل أثناء معالجة الفيديو للتمكن من تحديد حواف السيارة داخل الواجهة، و ذلك بدراسة تابع الكثافة الاحتمالية لكل بيكسل بين اللقطات داخل الفيديو (frames),حيث أن كثافة البيكسل في الواجهة للأجسام المتحركة تتغير مع الزمن ، و منه يمكن تحديد موقع السيارة بحساب مجموع كثافات البيكسلات البيضاء في الصورة الثنائية من أجل كل أسطر الصورة و أعمدتها، تليها عملية رسم المنحنين الأفقي و العمودي للمجموع (horizontal and vertical histograms) ، و منه بالتالي الحصول على النقاط الإحداثية التي تحدد المستطيل المحيط بالسيارة من تقاطع هذين المنحنيين (إذا كانت قيمة الفرق بين سطرين متتاليين أقل من قيمة العتبة بالتالي يمثل السطر نقطة احداثية موقع البداية بالنسبة للجسم المتحرك وفقاً للاتجاه الأفقي أما إذا كانت أكبر من العتبة فالسطر يمثل نقطة احداثية موقع النهاية بالنسبة للجسم المتحرك و بنفس الطريقة للأعمدة).

3-2-2- مرحلة استخلاص لوحة السيارة:

يقترح [1] استخدام خوارزمية LPB(local binary pattern) من أجل تحديد أطراف لوحة السيارة من صورة السيارة المعالجة في الخطوة السابقة، حيث تعتمد الخوارزمية على تقسيم صورة الرمادية (grayscale) إلى مجموعات (block) بأبعاد 3x3، من ثم مقارنة تعيين قيمة واحد أو صفر إلى البيكسلات المتواجدة على أطراف هذه المجموعات ,بحيث يأخذ البيكسل قيمة صفر إذا كان أصغر تماما بالقيمة من قيمة البيكسل الواقع في منتصف الـ block، و قيمة الواحد في حال كان بقيمة أكبر أو تساوي قيمة البيكسل الواقع في المنتصف، و من ثم تحويل قيم البيكسلات الواقعة على المحيط إلى رقم عشري (الشكل17).



الشكل 17: مثال على تطبيق **LBP**

يشكل الفرق بين القيمة 1 و0 في الشكل السابق احتمال وجود فرق أو حافة (edge) في الصورة الحقيقة، ومن أجل التأكد في حال كانت هذه الحواف هي حواف لوحة السيارات وبمجرد الحصول على جميع القيم العشرية لجميع المجموعات (blocks)، قام [1] برسم المنحنيات البيانية (histograms) من أجل كل المجموعات السابقة (عدد مرات ظهور العدد 233 في الصورة مثلاً)، ومقارنتها مع عينة تدريب مكونة من 50 knowledge histogram بحيث يتمكن في النهاية من استنتاج موقع اللوحة ضمن الصورة. يذكر أن من مزايا خوارزمية LPB أنها لا تتأثر بالإضاءة المسلطة على المشهد ككل لأن الفرق بين البيكسلات لا يختلف كون الإضاءة ستسلط على جميع البيكسلات المجاورة.

أما في [4][3][2] [14] فقد تم استخدام معامل يدعى (sobel) من أجل الحصول على الحواف، حيث اعتمد على فرق اللون بين لوحة السيارة وجسم السيارة لاستخلاص اللوحة.

يعتمد معامل sobel على مصفوفتين تسميان بkernel 3x3 Gx Matrix and kernel 3x3 Gy Matrix (الشكل 18).

-1	0	+1
-2	0	+2
-1	0	+1

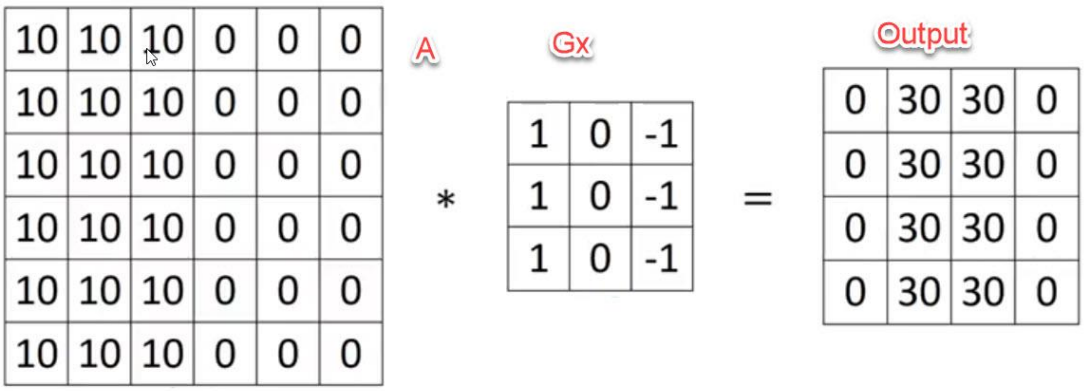
Gx

+1	+2	+1
0	0	0
-1	-2	-1

Gy

الشكل 18: المصفوفتان **Gx** و **Gy**

تستعمل المصفوفتين السابقتين في عملية إيجاد الحواف الأفقية في حالة Gy والعمودية في حالة Gx، فمثلاً في حالة صورة دخل (الشكل 19)، ينتج لدينا الخرج بإجراء عملية تلفيف بين المصفوفتين A وGy فينتج لدينا المصفوفة output الذي يمكن بوضوح تحديد الحافة عليها.



الشكل 19: مثال على عملية **Convolution**

وبالتالي باستعمال الخوارزمية السابقة، تمكن [14] [4][3][2] من إيجاد المواقع المتوقعة لمستطيل لوحة السيارة باستعمال الحواف العمودية فقط كون الحواف الأفقية قد تسبب تشابه مع ممتص الصدمات الخاص بالسيارة، ومنه بالحصول على الحواف العمودية والتخلص من الحواف في الخلفية يمكن الحصول على الموقع المتوقع للوحة السيارة.

في [5] تم اقتراح استعمال تحويل هوف (Hough transform) وهي طريقة من طرائق استخلاص السمات في معالجة الصور. يعمل التحويل على استخلاص الأشكال البسيطة (أشكال يمكن تمثيلها بمتحولات قليلة كالخطوط والدوائر) من الصور بأشكالها المختلفة (خط عمودي، أفقي، مائل بدرجة معينة)، مما يجعل عملية استخلاص اللوحات لا تتأثر بميلان زاوية التصوير، ولكن المشكلة الأساسية تكمن في استهلاك هذه الخوارزمية للوقت والذاكرة.

في [6] تم تطبيق خوارزمية العناصر المتصلة التي تعتمد على مسح الصورة وعنونة كل بيكسل ضمن مجموعات اعتماداً على اتصال بيكسلات المجموعة الواحدة ببعضها البعض، ومن ثم حفظ العناصر المتصلة التي تمتلك احداثيات هندسية تشبه لوحة السيارات (الشكل المستطيل والأبعاد). تشكل هذه الطريقة واحدة من أشهر الطرق في استخلاص لوحات السيارات.

في [7] تم اقتراح طريقة للحصول إلى لوحة السيارة عن طريق تعريف أحرف السيارات وتصنيفها باستعمال شبكة عصبونية. من أجل صورة الدخل، يتم البحث عن جميع مواضع الأحرف المختلفة ومعالجة هذه المواضع لرؤية في حال كانت تمثل أرقام داخل لوحة السيارة (بمعنى آخر العثور على اللوحة من خلال المحارف).

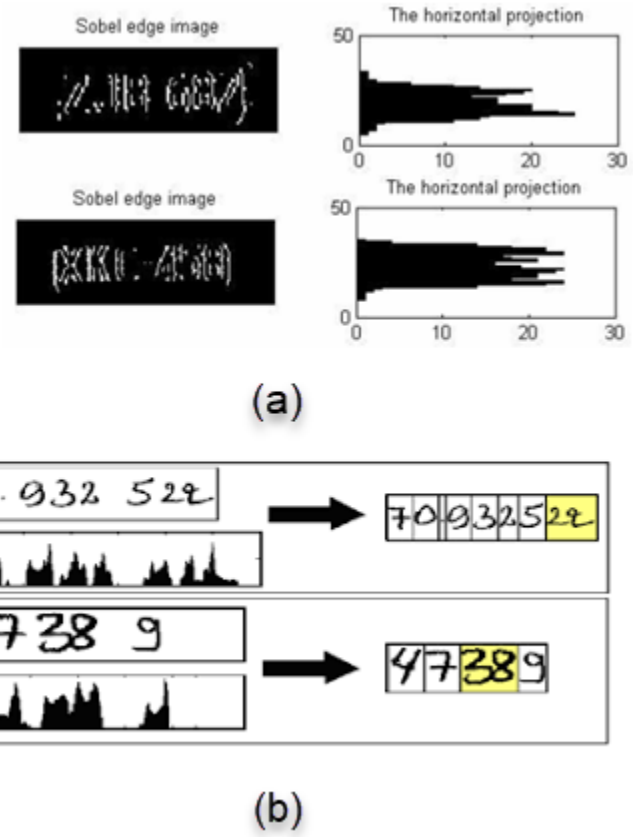
استعمل [13] عدة عمليات preprocessing على صور لواجهات أمامية للسيارات للحصول على لوحات السيارات العراقية، حيث قام بعمليات قص وإعادة تحجيم لحجم معين ثم التمرير على مرشح top-hat للتخلص من الضجيج، يليها عملية التخلص من العناصر المتصلة التي لا تمتلك أكثر من 70 بيكسل ومن ثم اقتصاص أكبر عنصر متصل من صورة السيارة الأصلية.

3-2-3- مرحلة استخلاص المحارف من لوحة السيارة:

يقترح [1] احاطة اللوحة بمربعات محيطة (وهي أكبر مساحة يشغلها جسم ثنائي أبعاد ضمن المحاور الثنائية المتوضع بها)، تبدأ العملية بتحويل صورة اللوحة الرمادية إلى صورة ثنائية باستعمال adaptive threshold، ثم تقسيم المربع المحيطي الذي يحيط اللوحة إلى مجموعة من المحارف الصغيرة بالاعتماد على المعرفة المسبقة بالمحارف والمسافات الثابتة بينهم في لوحات السيارات الهندية.

وفق [4] عن طريق عنونة البيكسلات المتصلة (label Connected Pixels) لصورة ثنائية للوحة سيارة، يمكن الاستخلاص المحارف حيث أن البيكسلات المتصلة التي تمتلك حجماً وأبعاد مقيسه شبيه بالمحارف المدرب عليها النظام يمكن اعتبارها محارف لوحة سيارة. نكمن المشكلة في هذا الأسلوب في معالجة المحارف المتصلة أو تلك التي تعاني تشوهات بسبب التصوير أو بسبب تلف اللوحة، و هذه المشاكل تمت معالجتها بشكل جزئي من قبل [14] (الذي استعمل خوارزمية مشابه لخوارزمية [4] عن طريق إزالة كل العناصر (Component) التي لا تحمل أبعاد مشابه لأبعاد المحارف و بما ذلك النقاط، لم يؤثر هذا القرار على العملية استخلاص المحارف كون المحارف العربية التي تمتلك نقطة و مستعملة في اللوحات السعودية هم ال (ب ,ن ,ق)و يمكن تمييزهم بسهولة دون النقاط، و هذا لم يؤثر على التعرف على الرقم 0 كون الخوارزمية السابقة لم تتخلص من العناصر التي تقع في مكان ينصف ارتفاع لوحة السيارة .

بعض الطرق تقترح الاعتماد على فرق اللون بين خلفية لوحة السيارة والمحرف نفسه كما في [9]، حيث تم القيام بحساب مجموع البيكسلات البيضاء ضمن جميع أعمدة الصورة الثنائية للوحة السيارة وهذا ما يعرف ب (vertical projection) من أجل معرفة نقطة بداية المحرف ونقطة نهايته، ومن ثم القيام بحساب مجموع البيكسلات البيضاء من أجل جميع الأسطر (horizontal projection) من أجل فصل كل محرف على حدا. يذكر أنه من محاسن استخدام هذه الطريقة أنها لا تختلف باختلاف مواقع المحارف في اللوحة ولكنها تعتمد بشكل واضح على نوعية الصور لأن الضجيج على الصور يؤثر على خرج الprojection .



الشكل 20: (**a**) **horizontal projection** (**b**) **vertical projection**

بعض الخوارزميات المقترحة تعتمد على المعرفة المسبقة بمواضع المحارف في صورة لوحة السيارة كما في [10]، حيث يتم تعديل حجم صورة لوحة السيارة إلى حجم معياري محدد مسبقاً، بحيث تصبح مواقع جميع المحارف معروفة في هذا الحجم وبالتالي يمكن استخلاصها. يمكن ملاحظة بساطة هذه الخوارزمية، لكن في حالة أي خطأ في مرحلة استخلاص اللوحة (أخذ قليل من جسم السيارة ضمن صورة اللوحة، انزياح اللوحة في الصورة) سيسبب في أخذ الخلفية البيضاء على أنها محارف وبالتالي فشل الخوارزمية.
قام [13] بعدة عمليات قبل استخلاص المحارف العربية من صور لوحات السيارات، منها تحويل الصورة إلى صورة ثنائية باستعمال خوارزمية Otsu threshold التي تعمل على إيجاد القيمة الأفضل للعتبة من أجل الفصل الصحيح بين الخلفية و الواجهة ، من ثم إجراء Complement للصورة الثنائية الناتجة (قلب الأبيض إلى أسود و العكس) ,يليها اجراء عملية dilation من أجل توسيع أو تضخيم الأحرف الموجودة في الواجهة، ثم التخلص من جميع الأغراض بأقل من 50 بيكسل من أجل جعل اللوحة أكثر وضوحاً، وفي النهاية استخلاص أجزاء اللوحة المختلفة(الأرقام العربية ، المحافظة)و استخلاص المحارف باستخدام خوارزمية labeling connected component.

3-2-4- مرحلة التعرف على المحارف:

وهي المرحلة الأخيرة من مراحل نظام ALPR التي تهدف إلى تحويل صور المحارف المستخلصة من المرحلة السابقة على معطيات.
استعمل [1] قاعدة معطيات مكونة من 300 محرف (محارف إنكليزية كون اللوحات الهندية تحمل أرقام ومحارف إنكليزية)، من أجل كل صورة محرف جديد يتم رسم vertical and horizontal histogram واستعمالهم كشعاع سمات (feature vector) لمقارنتهم مع قاعدة معطيات التدريب باستخدام خوارزمية k-nearest neighbor.

أما عند [2] فقد تم استخدام template matching ، حيث يتم توحيد حجوم صور المحارف المستخرجة من مرحلة السابقة و تحويلها إلى مجموعة صور ثنائية، ثم قياس مدى شبه هذه المحارف المستخرجة بالقالب template باستخدام hamming distance[1] ، حيث تعتبر مجموعة البيكسلات مشابه لأخرى فقط إذا كانت الفرق بين قيمها لا تتجاوز قيمة عتبة معينة.

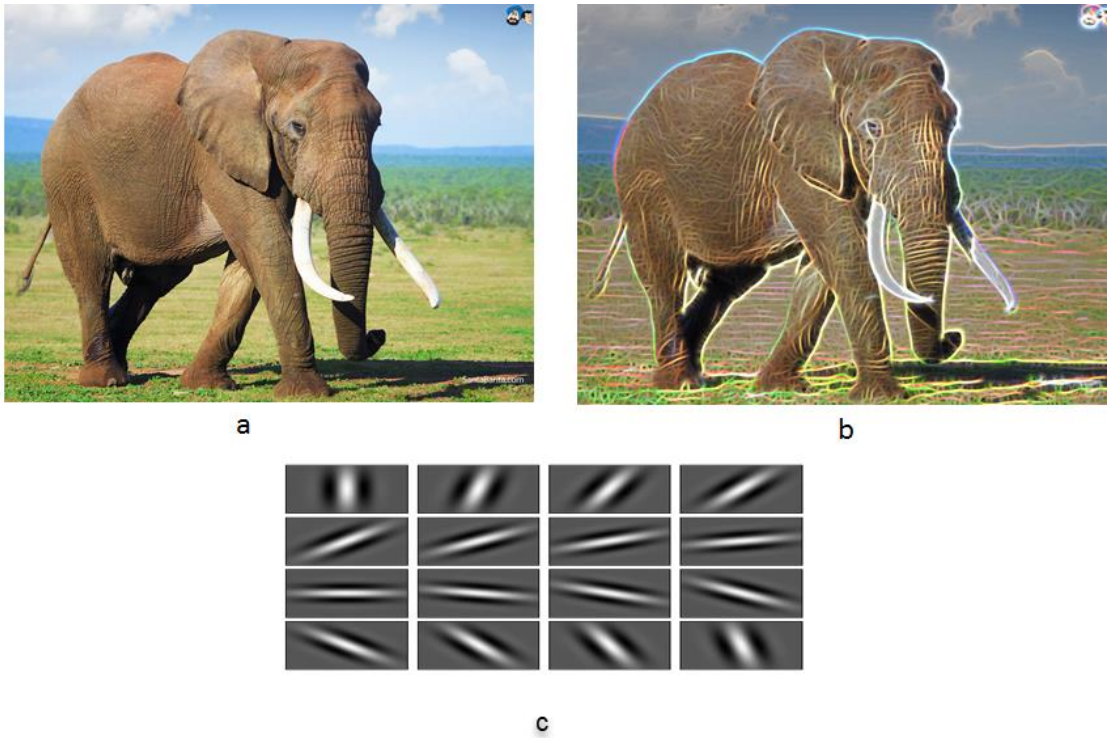
في [14] [13] تم أيضاً استعمال template matching من أجل مقارنة الحروف العربية المستخلصة من المرحلة السابقة مع templates (عبارة عن مجموعة من صور المحارف العربية الثنائية) من أجل التعرف على ال محارفOCR.

يذكر أن ال Template matchingمفيدة في التعرف على المحارف وحيدة الخط الخالية من الكسور أو التشويه والغير مدورة، حيث أن الاختلاف بين المحرف والقالب بالزاوية أو بسبب الضجيج او الخط قد يؤدي إلى نتائج خاطئة.

بما أن ليس كل البيكسلات في المحرف بنفس الأهمية يمكن الاعتماد على تقنيات استخراج السمات (feature extraction technique) حيث أنها لا تحتاج إلى جميع البيكسلات و منه أقل استهلاكا للذاكرة و الوقت من ال template matching ، كما تستطيع التغلب على بعض مشاكل ال template matching كالتدوير (rotation) ، فمثلاً في [11] تم استخلاص قيمة شعاع السمات (feature vector) عن طريق تقسيم الصورة الثنائية للمحرف إلى مجموعة من الأقسام مكونة من تسع بيكسلات (3x3) ، تم عد قيم البيكسلات السود في كل قسم و من ثم تمريرها إلى مصنف SVM.

في [12] تم استعمال مرشح Gaborحيث يعتمد هذا المرشح على تمرير حزم من الترددات واهمال الترددات الأقل بشكل مشابه لمرشح التمرير السفلي (low pass filter) ولكن بالاعتماد أيضا على التوجه (orientation) من أجل كشف المحارف، حيث أن حواف المحرف التي لديها توجه بنفس زاوية مرشح Gabor تمتلك حينها الاستجابة الأقوى(الشكل 21) .

[1] يوجد العدد من طرائق قياس مدى التشابه نذكر منها Mahalanobis distance [14]، Bayes decision technique و Hausdorff distance.



الشكل 21: التعرف على حواف جلد الفيل باستعمال **Gabor Filter**

3-2-5- النتائج:

بتطبيق على 300 صورة ملتقطة خلال فترتين مختلفتين لمراعاة ظروف الإضاءة المختلفة (180 صورة في الصباح و120 صورة في المساء)، تمكن [1] من إنشاء نظام ALPR بدقة 96.14% من أجل عمليات استخلاص السيارات ولوحاتهم (detection)، ودقة 89.35% من أجل عمليات التعرف على المحارف (recognition).

من أجل [13] وبتطبيق على عينة من 40 صورة للواجهة الأمامية لسيارات عراقية، تمكن النظام من استخراج اللوحة بدقة 87.5% ومن التعرف على المعارف بدقة 85.7%، حيث لوحظ تراجع الدقة بين المرحلتين بسبب أن عمليات ال preprocessing المنفذة على الصور أدت لحذف بعض المعلومات المهمة وأيضا عدم وضوح بعض صور الدخل.

أما في حالة استخلاص اللوحات السعودية من صور السيارات في [2]، فقد حقق النظام نتائج بدقة 96.2% ل1165 صورة بظروف مختلفة من الإضاءة وبزمن تنفيذ من أجل صورة بأبعاد 384x288 من مرتبة 40ms.

أما من أجل الفيديو المطبق في [6]، فقد أظهر النظام دقة بقيمة 96.62% من أجل فيديو تصل مدته إلى أربع ساعات.

ومن أجل خوارزمية Segmentation المقترحة في [9]، وبالتطبيق على ما يقارب 30000 صورة، وصلت دقة هذه الطريقة إلى 99.2% وبزمن تنفيذ قدرة 10-20ms.

ومن أجل [14]، تم اختبار النظام على 470 صورة لسيارة سعودية حيث تم استخلاص اللوحة بدقة 98.3% والتعرف على المحارف بدقة 98.63%.

3-2-6- الملخص[15] :

تلخص الجداول الثلاث الآتية (جدول 4,3,2)إيجابيات وسلبيات الطرائق المختلفة (لاستخلاص لوحات السيارة، استخلاص المحارف من لوحة السيارة، للتعرف على المحارف في لوحات السيارات) على التتالي.

الطريقة	شرح بسيط	إيجابيات	سلبيات
Boundary features	الاعتماد عل الشكل المستطيل للوحة السيارة	الابسط و الأسرع	صعب التطبيق على الصور المعقدة كونها حساسة جدا للحواف.
Global image feature	البحث عن عنصر متصل يحمل أبعاداً تشابه أبعاد لوحة السيارة	مستقل عن موقع اللوحة ضمن الصورة	قد يولد صورا غير صالحة Broken
Texture features	اختلاف الألوان بين المحارف و خلفية اللوحة أو بين اللوحة و السيارة (اكتشاف حواف)	القدرة على استخلاص اللوحة حتى في حالة تشوه محيط اللوحة	حسابه معقد في حالة عدة حواف
Character features	تواجد النص داخل لوحة السيارة	لا يواجه مشاكل مع أي شكل من أشكال التدوير	يحتاج إلى وقت تنفيذ كبير و قد يولد أخطاء في حال تواجد النص في أكثر من موقع

جدول 1: ملخص عن طرائق استخراج لوحة السيارة

الطريقة	الإيجابيات	السلبيات
Pixel connectivity	بسيطة و لا تتأثر بالتدوير	عدم القدرة على استخلاص المحارف المتصلة أو المتضررة
Projection profiles	مستقلة عن موضع المحرف في اللوحة	الضجيج يؤثر على الخرج بشكل كبير. الحاجة لمعرفة مسبقة بعدد المحارف في اللوحة
Prior knowledge of characters	بسيطة	محدودة تبعاً للمعرفة المسبقة و منه أي تغيير في المواضع سيسبب أخطاء

جدول 2: ملخص عن طرائق استخلاص المحارف من لوحة السيارة

الطريقة	الإيجابيات	السلبيات
Template matching	بسيطة	عمليات معالجة للبيكسلات غير مهمة كما أن هذه الطريقة تتأثر بسهولة بمعاملات الضجيج و الدوران
Gabor filter	أسرع من سابقتها كون عدد السمات أقل من عدد البيكسلات	يجب البحث عن السمات بقيم واضحة لتجنب التعرف السيء

جدول 3: للتعرف على المحارف في لوحات السيارات

الفصل الرابع

الدراسة العملية والتضمين البرمجي

يذكر هذا الفصل النموذج المقترح للتعرف على لوحات السيارات السورية، يليها استعراض بسيط للأدوات والمكتبات المستعملة، ثم توضيح كيفية تضمين النموذج المقترح برمجياً.

4-1- النموذج المقترح

بعد التعرف على نظم ALPR وأقسامها وطرق تضمينها المختلفة في دراسات وتجارب مرجعية، سيقدم هذا المشروع في قسمه العملي طريقة مقترحة لتصميم نظام للتعرف على الأرقام العربية في لوحات السيارات السورية عبر تقسيم العمل على ثلاث مراحل مهمة وأساسية:

1- التعرف على هيكل السيارة: وذلك باستعمال yolov2 وDarknetAPI.

2- التعرف على لوحة السيارة: وذلك باستعمال WPOD-NET المقترحة من قبل المرجع [16].

3- التعرف على الأرقام العربية: وذلك باستعمال tesseract-ocr للكشف عن الأرقام وتدريبه باستعمال مكتبة open-ALPR.

حيث تم توليد عدة ملفات مكتوبة بلغة python، تمثل المراحل المختلفة من النظام المقترح باستعمال برنامج spyder (بيئة لتطوير وبناء ملفات برمجية بلغة python).

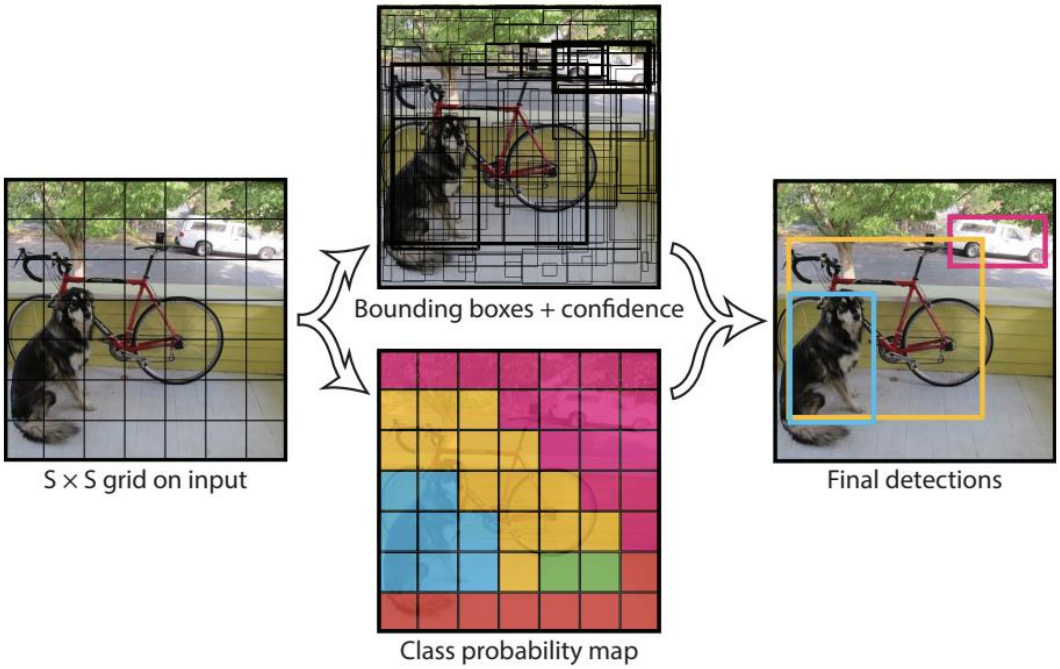
4-2- الأدوات والمكتبات المستعملة

4-2-1- Yolov2 And DarknetAPI :

يعرف yolov2 على أنه نظام تعرف على أغراض واقعية من صور الدخل وتصنيفها في مجموعة صفوف Classes بالاعتماد على Darknet Neural Network API المكتوبة بلغة C.

يتميز yolov2 بسرعة في الأداء ودقة في استخلاص الأغراض ما يميزه عن باقي خوارزميات اكتشاف الأغراض، فمثلاً ال R-CNN و Fast R-CNN يعتمدان على البحث ضمن أجزاء من الصورة التي تمتلك احتمالية وجود غرض عالية، أما YOLO (you only look once) فيتم الاعتماد على CNN واحدة لتوقع المربعات المحيطة واحتمالات الصفوف المختلفة لها.

يعمل yoloمن أجل أي صورة دخل على تقسيم الصورة إلى SxS شبكة (grid) (الشكل 22),من أجل كل شبكة يتم تحديد m مربع محيط وإدخالهم في الشبكة العصبونية، تخرج الشبكة العصبونية احتمال الصف (class probability)، وفي حال حصل مربع محيط على احتمال صف أكبر من عتبة معينة فسيتم استعمال هذا المربع للحصول على موقع الغرض من الصورة.



الشكل 22: آلية عمل **yolov2**

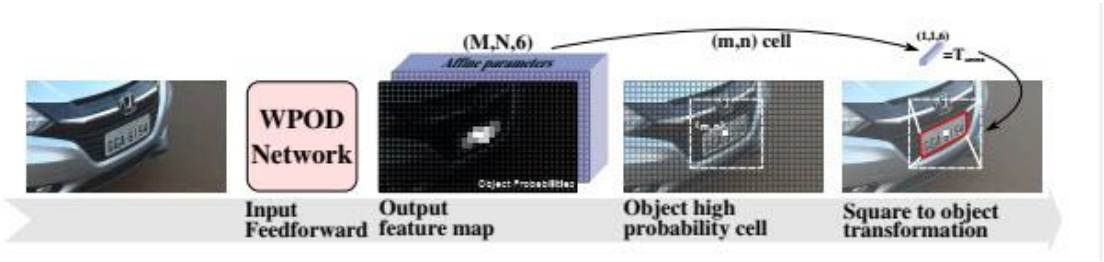
يعد yolo من أسرع خوارزميات العثور على الأغراض الموجودة في الوقت الحالي، حيث تصل سرعة معالجته إلى 45 frames/s، ولكن يواجه yolo مشكلة في التعرف على الأجسام الصغيرة في الصورة بسبب القيود المكانية على الخوارزمية.

: WPOD-NET -2-2-4

من أجل الاستفادة من شكل لوحة السيارات (المستطيل غالباً) تم استعمال WPOD-NET، وهي شبكة عصبونية يمكن تدريبها على اكتشاف لوحات السيارات بمختلف المواضع والحالات التي تشغلها في الصورة (الدوران والحجوم الصغيرة).

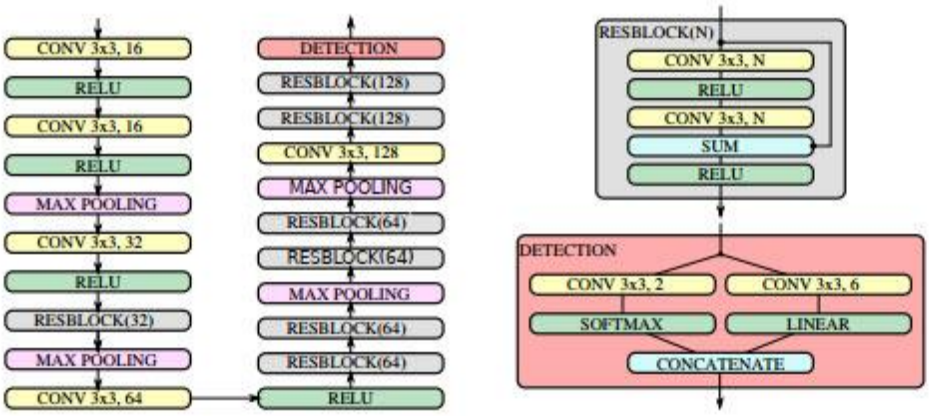
تم تطوير WPOND [16] باستعمال مزيج من YOLO و STN (spatial transformer Networks)، حيث يستطيع yolo اكتشاف والتعرف على الأغراض المختلفة من الصورة مرة واحدة وبسرعة كبيرة، ولكن دون الاهتمام للتحولات المكانية للغرض، حيث يتم احاطة الغرض بمستطيل فقط من أجل كل جسم متعرف عليه، في المقابل، STN تستطيع اكتشاف المواقع الغير مستطيلة في الصورة، ولكنها لا تستطيع تنفيذ عدة تحويلات مكانية على الغرض في الوقت نفسه، إنما تنفذ تحويل واجد على الدخل كله.

تمر عملية الحصول على لوحة السيارة بعدة مراحل (الشكل 23)، تبدأ العملية بإدخال المدخلات الى الشبكة CNN وينتج الخرج على شكل 8-channel feature map وهي مصفوفة تحوي احتمالات وجود غرض ومتحولات المكانية النسبية من أجل صورة الدخل، ومن أجل مربع خيالي محيط بالبيكسل صاحب الإحداثيات (m,n)، إذا كان احتمال وجود غرض في البيكسل (m,n) أكبر من عتبة معينة، عندها يتم توليد مصفوفة تحويل مكاني نسبية (affine transformation matrix) التي تعمل على تحويل المربع الخيالي المحيط ب (m,n) إلى منطقة لوحة سيارة (LP region).



الشكل 23: مراحل عمل **WPOD-NET**

تحتوي شبكة WPOD-Net على 21 Convolution layer ، 14 طبقة منهم تم استعمالها داخل (residual blocks) (الشكل 24)، بمرشح k بأبعاد 3x3. يوجد 4 Max Pooling Layers بأبعاد 2x2 بهدف تخفيف أبعاد الدخل، تم استعمال العديد من ReLu Layers، وفي النهاية يوجد طبقة التعرف وهي عبارة عن طبقتي تلفيف تعملان على التوازي: الأولى من أجل إيجاد الاحتمالات وتستعمل للتفعيل تابع Softmax والثانية من أجل مراجعة المتحولات النسبية (regressing the affine parameters) وبدون تابع تفعيل.



الشكل 24: البينة الداخلية لشبكة **WPOD-NET**

: Open-CV -3-2-4

وهي مكتبة مفتوحة المصدر تحتوي على أكثر من 2500 خوارزمية تستعمل في مجال Computer Vision وتعلم الآلة، يمكن استخدام هذه الخوارزميات في التعرف على الأوجه، تصنيف تحركات الإنسان ضمن ملف فيديو، تتبع الأجسام المتحركة وغيرها. هذه المكتبة مكتوبة لغة C++ وتمتلك واجهات للتعامل مع لغات مثل c++,python,java,Matlab.

تكمن الاستفادة العظمى من هذه المكتبة في عملية تطوير نظم ALPR في مرحلة preprocessing حيث توفر هذه المكتبة سلسة من التوابع للتخلص من الضجيج في الصور (cv2.fastNlMeansDenoisingColored)، أو لتحويل الصورة إلى صورة ثنائية (cv2.threshold) وغيرها الكثير من عمليات التي يمكن استخدامها لتهيئة صور الدخل.

:Tesseract OCR -4-2-4

Tesseract هو محرك OCR مفتوح المصدر تم تطويره في مخابر شركة HP في Bristol بين عامي 1984 و 1994 ، و قد أمكن الرؤية أن tesseract سيشكل إضافة مهمة لخط انتاج الطابعات الخاص ب HP كون جميع محركات OCR كانت تفشل فشل ذريع في أي صورة ضعيفة الدقة على عكس Tesseract .

بعد بحث مشترك بين مخابر HP في Bristol وقسم الماسحات الخاص ب HP أيضا في Colorado أصبح Tesseract يتصدر محركات الOCR في الدقة ولكنه لم يصبح منتجا بعد، حيث تم ارساله في نهاية 1994 إلى UNLV (University of Nevada، Las Vegas) من أجل الاختبار ال سنوي لدقة محركات ال OCR، و تفوق حينها على جميع محركات ال OCR، و في نهاية عام 2005 أصدرت HP نسخة Tesseract مفتوحة المصدر، ولا يزال يطور في google حتى الوقت الحالي.

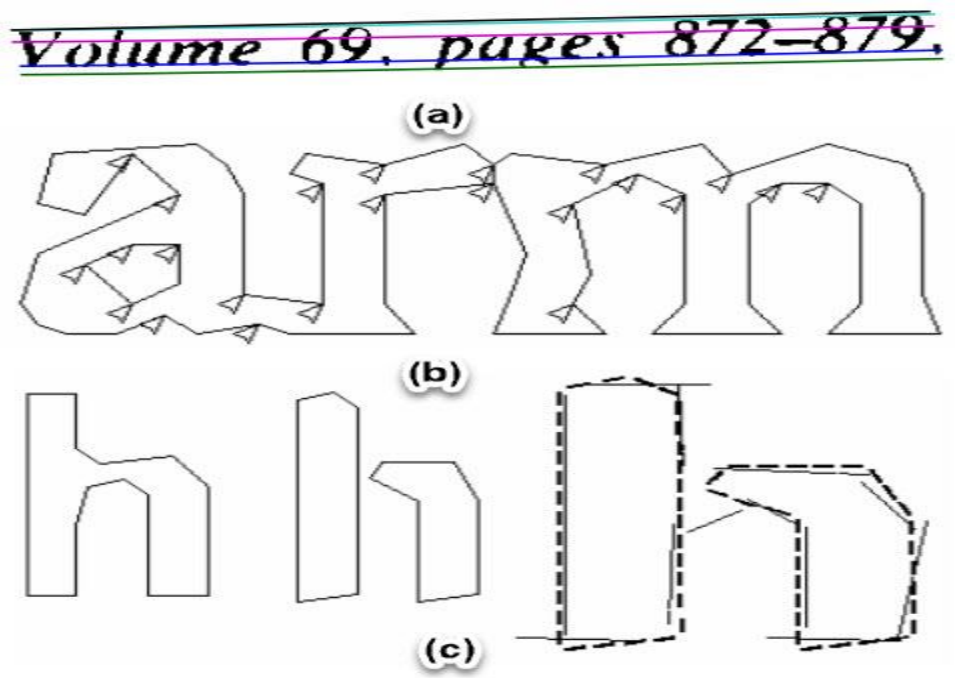
يعمل نظام tesseract على معالجة الصور الثنائية القادمة له من الدخل، التعرف على المحارف الموجودة في الصورة، ومن ثم اخراج تمثيل نصي لمحتويات هذه الصورة النصية.

تمر عملية التعرف على النص باستعمال tesseract بعدة مراحل [17] وهي:

1. يقوم الTesseract بالبحث عن المكونات المتصلة في الصورة حيث يتم تخزين حواف المكون (outlines).
2. يتم تجميع الحواف السابقة في blobs (منطقة في الصورة كل البيكسلات فيها تمتلك خصائص متشابهة مثل الإضاءة واللون).
3. يتم تنظيم blobs ضمن خطوط نصية (text lines).
4. تقسم الخطوط النصية إلى كلمات (words).
5. مرحلة التعرف على الكلمات والمكونة من عبورين على الكلمات، العبور الأول يعمل على التعرف على كل الكلمة على حدي باستعمال مصنف ثابت static classifier، ومن ثم الكلمات التي تم التعرف عليها بشكل صحيح تمرر إلى adaptive classifier حيث يتم تعزيز المعطيات المدربة training data، وفي النهاية عبور ثان على الكلمات لمحاولة التعرف على الكلمات التي فشل في التعرف عليها في العبور الأول.
6. إخراج النص الرقمي الذي يمثل النص الموجود في الصورة.

يستعمل خلال هذه العملية العديد من الخوارزميات:

- خوارزميات العثور على الأسطر النصية وخاصة في الصفحات المائلة (skewed pages).
- خوارزميات للبحث والتعرف على الكلمات المتناسبة وغير المتناسبة (proportional words وهي الكلمات التي تمتلك كل محارفها العرض نفسه) وذلك من أجل تجميع المحارف في كلمات مناسبة.
- خوارزميات لتقسيم المحارف المتصلة وربط المحارف المقطوعة (المتضررة نتيجة التصوير، الإضاءة ....).



الشكل 25: (**a**) **line-finding Algo** (**b**) **Chopping Joined Characters** (**c**) **Associating Broken Characters**

يدعم tesseract العديد من اللغات عن طريق مجموعة من الملفات ذات اللاحقة traineddata، وهي ملفات تحمل جميع المعلومات عن محارف اللغة التي يحتاجها ال Tesseract (أشكال المحارف، نوع الخط، حجوم المربعات المحيطة لكل محرف .....)، كما أنه يوفر الأدوات اللازمة للمطور لتدريب ال tesseract على لغات ومحارف معينة وإنشاء ملف traineddata الخاص به.

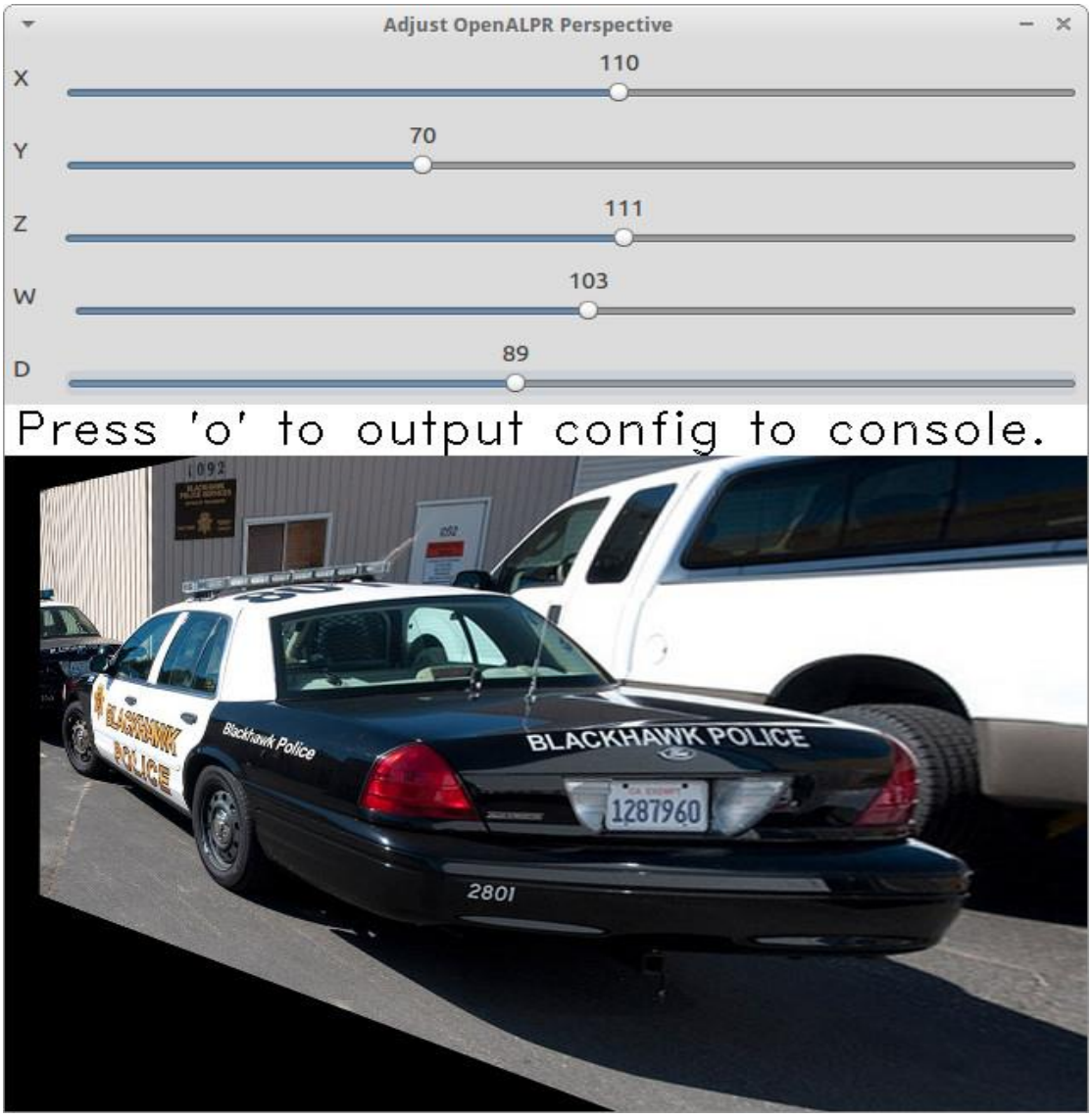
4-2-5- Open-ALPR:

هي مكتبة مفتوحة المصدر مصممة لأغراض إنشاء نظم التعرف التلقائي للوحات السيارات، مكتوبة بلغة c++ ومدعومة من قبل لغات python,Java,C#.

توفر هذه المكتبة العديد من الأدوات التي تساعد في تصميم نظام ALPR، حيث تستطيع هذه المكتبة التعامل مع الصور أو مقاطع الفيديو كدخل لها، معالجة هذا الدخل وإخراج تمثيل نصي يعبر عن المحارف الموجودة في لوحة سيارة الدخل. نذكر من هذه الأدوات:

- openalpr-utils-calibrate: و هي أداة تستعمل لتحسين الصور القادمة من كاميرات ثابتة بزاوية رؤية مائلة (صور السيارات مائلة و اللوحة لا تظهر بشكل مستطيل)، حيث تأخذ هذه الأداة صور دخل واحدة لسيارة ملتقطة من الكاميرا المائلة، و تتيح للمستخدم عبر واجهة GUI التعديل على قيم زوايا ميلان الصورة، و عند انتهاء المستخدم

تتيح هذه الأداة إظهار إعدادات تمثل انتقالات دورانية، يمكن تطبيق هذه الإعدادات على جميع صور اللوحات القادمة من كاميرا .



الشكل 26: **openalpr-utils-calibrate**

- Openalpr-utils-prepcharsfortraining: و هي أداة تتيح للمستخدم توليد ملفا box. و tif. اللازمين لتدريب محرك tesseract على عينة تدريب للغة معينة (إنشاء ملف traineddata).

Box file: هو ملف نصي يوصف فيها صور المحارف المستعملة في التدريب (كل صورة يقابلها سطر يمثل المحرف الذي تعبر عنه)، مع احداثيات المربع المحيط ، بالنسبة لمبدأ احداثيات مركزه هو الزاوية اليسرى العليا من الصورة، و رقم نهائي يوضح رقم

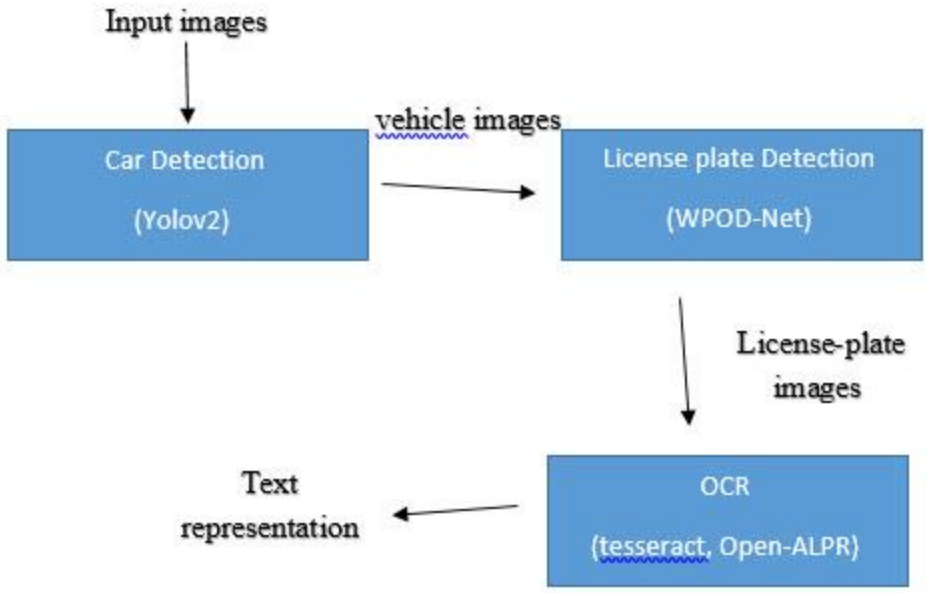
الصفحة و ذلك في حال توليد ملف tif. متعدد الصفحات (من أجل الفصل بين الخطوط حيث لا يجب أن تتدمج محارف الخطوط المختلفة في ملف tif. واحد لأنه يسبب أخطاء في خرج tesseract).

Tif file: هو ملف صورة يحوي جميع صور جميع المحارف المستخدمة للتدريب.

الشكل 27: **tif file** (**b**) **box file** (**a**)

4-3- التضمين البرمجي للمراحل المقترحة

بعد الاطلاع على الشكل العام لنموذج ALPR المقترح في المشروع، والتعريف بأبرز الأدوات والمكتبات المستخدمة، أصبح بالإمكان تحديد الشكل العام للمراحل المشروع مع الأدوات المستعملة في كل مرحلة من المراحل المقترحة (الشكل 28). ستتناول الفقرة الآتية تحويل كل مرحلة من المراحل المقترحة (في الفقرة 4-1-) إلى نص برمجي مكتوب بلغة python.



الشكل 28: الشكل العام لمراحل المشروع، الخرج المتوقع و الأدوات المستعملة في كل مرحلة

4-3-1- التعرف على هيكل السيارة:

تم استعمال yolov2 كونه يشكل أحد أسرع النظم في مجال التعرف على الأغراض ، كما أنه يلائم مهمة الكشف عن السيارة كونه دقيق جداً في استخلاص الأغراض الواضحة في المشهد وتصنيفها إلى صفوف يمكن للمطور من التعامل معها دون الحاجة لتدريبها على هذه الصفوف.

في البداية تمت الاستعانة ببرنامج anaconda (وهي بيئة تطوير data science) لإنشاء بيئة افتراضية بلغة python على نظام Linux (python virtual machine تفيد في فصل المشروع البرمجي الذي يتم تطويره داخل البيئة الافتراضية عن النظام ككل)، كما أنه يوفر قدرة على التعامل مع الحزم البرمجية المختلفة مثل Tenserflow (وهي مكتبة مفتوحة المصدر من أجل بناء نظم تعلم الآلة) وkeras (وهو مكتبة مصممة لتطوير الشبكات العصبونية باستعمال python) التي يحتاجها yolo v2 في عمله.

بعد إنشاء البيئة الافتراضية لpython وتنصيب الحزم البرمجية الآتية (Tenserflow 1.14، Keras 1.5.0، opencv)، قمنا بتنصيب Darknet (وهي إطار عمل لتطوير الشبكات العصبونية يعتمد عليها yolov2) وجميع الحزم التي يعتمد عليها وإجراء عملية تصريف (compile) لها.

لم يتم تدريب الشبكة العصبونية التي يوفرها Darknet على اكتشاف السيارات بل اعتمدنا في هذا البحث على ملفات الأوزان (yolo-voc.weights) وملف التشكيل (yolo-voc.cfg) للشبكة العصبونية الخاصة بyolo (ملفات يمكن تحميلها من

موقع yolo) ، كون نظام ال yolo يستطيع تصنيف الأغراض في المشهد إلى صفوف مختلفة (مذكورة داخل ملف voc.names) ومن ضمن هذه الصفوف هي 'car' و 'bus'، بالتالي يمكن استعمال yolo دون الحاجة لتدريبه على عملية استخلاص هيكل السيارة من صور الدخل.

تكمن الخطوة التالية في التعامل مع الشبكة والتوابع التي توفرها Darknet وذلك ما يوفره ملف vehicle-detection.py (انظر الملحق 1)، حيث يأخذ مجلد دخل ومجلد خرج كدخل له، يعمل على معالجة كافة الصور في مجلد الدخل باستعمال yolo وإخراج صورة لمحيط السيارة مقتطعة عن الخلفية في مجلد الخرج.

أولا يتم إنشاء شبكة ال yolo باستعمال تابع load-net (توفره مكتبة Darknet)، الذي يأخذ ملف الأوزان وملف التشكيل كدخل له ويخرج غرض يمثل neural network model، وتحديد قيمة العتبة التي سيتم استعمالها لاعتبار الغرض المكتشف هو من الصف سيارة أو لا.

يوفر تابع detect المضمن ضمن مكتبة Darknet داخل ملف darknet.py وسيلة لاستعمال yolo من أجل الكشف عن الأغراض، حيث يأخذ كدخل له ال neural network model والصور المدخلة وقيمة العتبة ويخرج الأغراض المكتشفة في الصورة (إحداثيات المربع المحيط الخاص بالغرض واحتمال انتماءه للصفوف المختلفة المقدمة من yolo) ومن ثم يتم معالجة الأغراض التي تنتمي للصف 'car' أو 'bus' فقط.

إضافة لإخراج هذا الرماز لصورة مقتطعة للسيارة من المشهد إلا أنه يخرج أيضا ملفات نصية توضح مواضع هذه السيارات المكتشفة داخل الصور الخاص بهم، من أجل استعمالها في المرحلة القادمة (الشكل 29).

الشكل 29:خرج مرحلة اكتشاف هيكل السيارة من أجل صورة دخل معينة

4-3-2- التعرف على لوحة السيارة:

كون yolov2 يفشل في عملية الحصول على الأغراض صغيرة الحجم في المشهد، ونظراً للحاجة لاستخلاص لوحات السيارات من الصور (قد تكون صغيرة الحجم بسبب بعد السيارة عن الكاميرا الملتقطة)، تم استعمال نظام WPOD-net المقترح في [16] كونه قادر على استخلاص لوحات السيارات من المشهد مهما كان حجمها من الصورة الكلية ومهما كانت قيم زوايا ميلانها على المحاور المختلفة، عوضاً عن إحاطة اللوحة المائلة بمربع محيط كما يفعل نظام yolo.

يمثل ملف license-plate-detection.py الرماز البرمجي المسؤول عن تنجيز مرحلة استخلاص لوحات السيارات (انظر ملحق 1)، حيث يأخذ كدخل له مجلد الدخل يحوي جميع صور السيارات المخرجة من المرحلة السابقة ومسار نموذج الشبكة المستعمل في [16] (wpod-net_update1.h5)، يخرج هذا الملف صور تمثل لوحة السيارة بعد التخلص من زوايا الدوران والقيام بعمليات التقييس.

أولاً يتم تحديد قيمة للعتبة من أجل تحديد فيما إذا كان الغرض المكتشف من قبل النظام هو لوحة سيارة أم لا، ثم تهيئة الشبكة باستعمال تابع load-model الذي توفره مكتبة keras (يأخذ كدخل له ملف نموذج الشبكة المستعمل)، من ثم من أجل كل صورة في مجلد الدخل يتم استخراج لوحة السيارة من الصورة باستعمال تابع detect_lp الذي يأخذ كدخل له الشبكة و الصورة و العتبة و يخرج مصفوفتين تعبران عن أبعاد اللوحة ضمن الصورة و الصورة المقايسة المقتطعة للوحة من الصورة(يتم إجراء عملية تقييس من أجل جعل صور جميع اللوحات المستخرجة بالحجم نفسه لاستعمالها كدخل مرحلة التعرف على الأرقام).

يتم استخلاص منطقة الأرقام العربية من لوحة السيارة وتخزينها كصور، كما ينتج هذا الرماز ملف نصي يحوي أبعاد اللوحة ضمن الصورة وذلك باستعمال تابع writeShapes الذي يعمل على طباعة احداثيات المربعات المحيطة التي عثرت عليها الشبكة على ملف نصي(الشكل 30).

الشكل 30: خرج مرحلة استخلاص لوحة السيارة من أجل صورة دخل معينة

4-3-3- التعرف على الأرقام العربية:

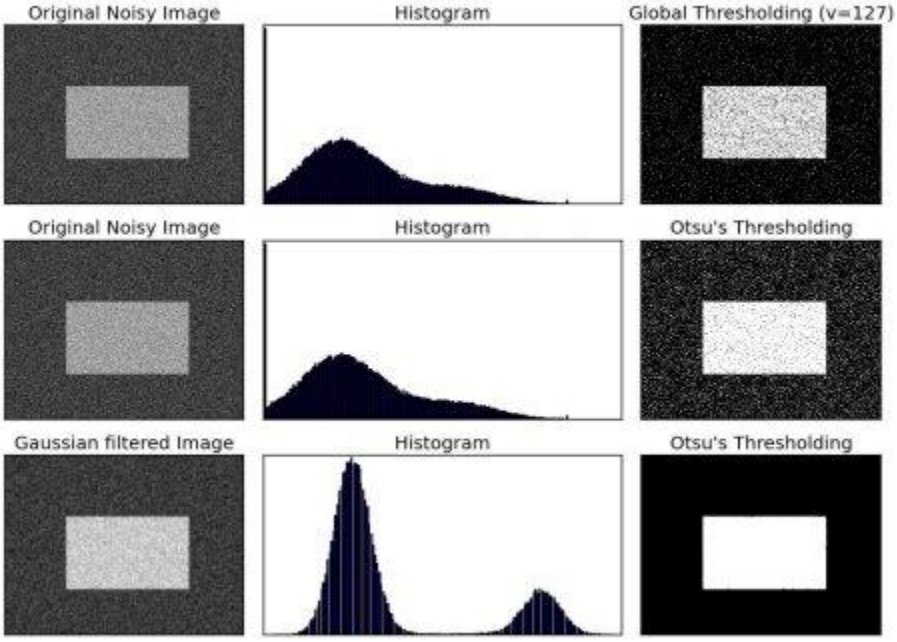
من أجل إتمام هذه المرحلة تم الاعتماد على tesseract-ocr للتعرف على الأرقام العربية، وقد تم تقسيم العمل في هذه المرحلة إلى قسمين:

4-3-3-1- تدريب ال Tesseract:

كما ذكرنا سابقاً أن tesseract يدعم العديد من اللغات عن طريق ملفات تدريب بلاحقة (traineddata .)، و على الرغم من أن الTesseract يوفر ملف (ar.traineddata) من أجل دعم اللغة العربية، إلا أنه بعد التجريب على عدد من الأرقام العربية ، اتضح أن الاعتماد على دعم tesseract للغة العربية غير كافي لما يولده من نتائج كارثية في مرحلة الخرج، مما استدعى العمل على تدريب tesseract على الأرقام العربية.

بالاعتماد على 178 صورة ملتقطة للسيارات الداخلة إلى مركز شركة سيرتيل في صحنايا (العينة الكاملة هي 242 صورة بالإضافة لملفات فيديو مختلفة، ولكن تم قسم العينة إلى قسم تدريب 178 صورة وقسم اختبار 63 صورة)، تم توليد مجموعة معطيات التدريب (training data set) حيث تم استخلاص لوحات السيارات من الصور السابقة وتحويلها إلى صور ثنائية باستعمال مكتبة opencv وخوارزمية Otsu's Binarization.

تعمل خوارزمية Otsu's Binarization على رسم المنحنيات الرمادية (grayscale histogram وهو منحني يوضح عدد مرات تكرار قيم تدرجات اللون الرمادي في الصورة) من أجل إيجاد قيمة العتبة الملائمة لتحويل الصورة إلى قيم 1,0 الثنائية من أجل جميع صور النسقين (bimodal image وهي الصور التي تمتلك قمتين في منحنياتها البيانية) (الشكل 31).



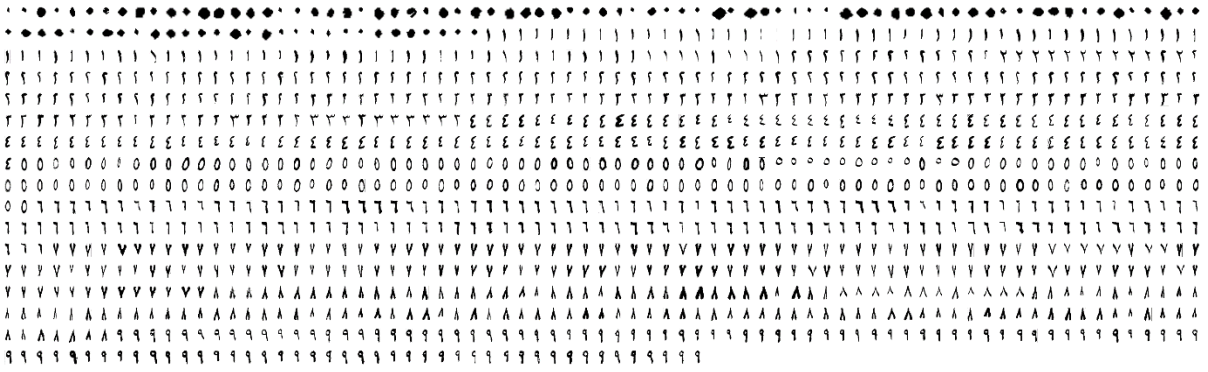
الشكل 31: يوضح تطبيق **otsu threshold** على مجموعة من صور الدخل و المنحنيات الخاصة بكل صورة

في البداية تم اقتراح استخدام open-ALPR من أجل عملية استخلاص الأرقام من صورة الجزء العربي من اللوحة (باستعمال أداة classifychars التي توفرها المكتبة و التي تخرج صور للمحارف المستخلصة بأحجام موحدة) ، و لكن تنوع الخطوط و بالتالي تنوع المسافات بين الأرقام في اللوحات السورية بالإضافة للتشوهات التي تم العثور عليها في بعض صور عينة التدريب (الشكل 32) أدى لظهور نتائج سيئة و غير دقيقة في عملية استخراج الأرقام من اللوحة، تم تفادي هذه المشكلة عن طريق

اقتصاص محارف من صورة اللوحة الثنائية يدوياً و حفظهم في صور خرج موحدة الحجم للأرقام ليتم استعمالهم في عملية التدريب.

الشكل 32: أمثلة عن الصور متضررة اللوحة في عينة التدريب

بعد الانتهاء من عملية استخلاص المحارف تم استعمال أداة openalpr-utils-prepcharsfortraining من أجل توليد ملفي box و tif (الشكل 33)الضروريين لتدريب tesseract على المحارف العربية.



الشكل 33:ملف **tif** المولد من **dataset** الخاصة بنا

تم تنفيذ عدة تعليمات متتالية على ملفين الbox والtif وهم بالترتيب:

- تشغيل Tesseractفي نمط التدريب:

(ملاحظة lang و fontname هما اسمان اختياريان يحددهما المستخدم في كل أجزاء هذه الفقرة)

وذلك باستعمال تعليمة الآتية في مفسر الأوامر الخاص بنظام Linux (terminal).

tesseract [lang].[fontname].exp[num].tif [lang].[fontname].exp[num] box.train

خرج هذه المرحلة هو ملف fontfile.tr والذي يحوي سمات كل المحارف المدخلة في ملف tif.

ملاحظة: يجب أن يتطابق طريقة تسمية ملف box مع ملف tif وإلا لن يتعرف عليهم tesseract.

- توليد ملف unicharset:

وهو ملف يحوي معلومات عن كل رمز نحاول أن ندرب tesseract عليه داخل ملف box يتم توليده باستعمال التعليمة

```
unicharset_extractor lang.fontname.exp0.box lang.fontname.exp1.box
```

وذلك لمساعدة tesseract على معرفة مجموعة المحارف التي يتم تدريبه عليها.

- توليد ملف font_properties:

الغرض من هذا الملف هو توفير معلومات عن الخطوط التي يتم تدريب tesseract عليها، كل سطر من هذا الملف يتم تقسيمه بالشكل الآتي

```
fontname italic bold fixed serif fraktu
```

حيث يعبر fontname عن اسم الخط المستعمل على شكل سلسة محارف string، أما باقي الخيارات فهي عبارة قيمة صفر أو واحد تعبر عن كون الخط يحمل الصفة المذكورة أم لا.

ملاحظة: كل حقل fontname في ملف tr يجب أن يقابله سطر في ملف font_properties.

- العنقدة (clustering):

يعد ما تم استخلاص السمات في المراحل السابقة، لا بد من القيام بعملية عنقدة لهذه الخصائص من أجل انتاج نماذج التعرف (recognition prototypes)، تتم عملية العنقدة على ثلاث مراحل

- ✓ Shapeclustring : لإنتاج ملف يجدول الأشكال المختلفة للمحرف الواحد (shapetable file) باستعمال التعليمة الآتية

shapeclustering -F font_properties -U unicharset lang.fontname.exp0.tr lang.fontname.exp1.tr …

- ✓ Mftraining : من أجل توليد ملفين الأول (inttemp) و يحوي نماذج التعرف، والثاني (pffmtable) و يمثل عد السمات المتوقعة للمحرف الواحد و ذلك باستعمال التعليمة الآتية

mftraining -F font_properties -U unicharset -O lang.unicharset lang.fontname.exp0.tr lang.fontname.exp1.tr…

- ✓ Cntraining: يهدف لتوليد ملف (normproto) الذي يوضح نموذج الحساسية لكل محرف يتم تدريبه (sensitivity prototypes).

- التجميع: تجميع جميع الملفات المولدة بالخطوات السابقة لتوليد ملف traineddata النهائي وذلك بتنفيذ التعليمة:

combine_tessdata lang.

ملف traineddata المولد في النهائية يمكن استعماله في tesseract وذلك بتفعيل الخيار l– عند تشغيل tesseract على صورة معينة.

يوضح ملحق (2) التعليمات المنفذة على terminal على ملف box و tif من أجل توليد ملف arsy.traineddata النهائي.

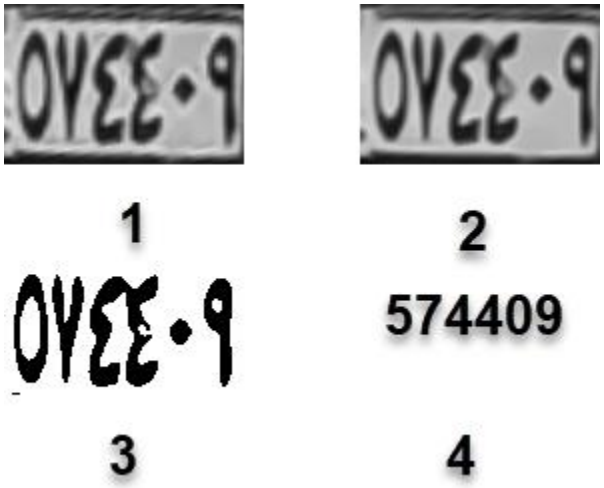
تم تجربة ملف ال traineddata المنتج من خرج مرحلة التجميع على صور دخل تجريبية لم يدرب عليها، لوحظ أخطاء في عملية التعرف على الأرقام (التعرف على الرقم 1 على أنه 6 و الرقم 2 على أنه 3 و ذلك بسبب التشابه بينهم في الخط) فتمت تصحيح العينة عن طريق التخلص من بعض صور الأرقام التي تعاني من تشوه و قد تسبب تضارب في عمل Tesseract، و أيضاً تم تعزيز مجموعة معطيات التدريب عن طريق إضافة صور الأرقام العربية المستعملة في الخطوط المختلفة العربية التي يدعمها برنامج Microsoft word وذلك من أجل تدريب tesseract على خطوط مختلفة و زيادة دقة الخرج، و على مجموعة من لوحات السيارات السورية المستخلصة عن الإنترنت.

بعد الانتهاء من عملية استخلاص المحارف، ومن 220 صورة للوحات سيارات في عينة التدريب نتج لدينا 104 صورة للرقم (0) ,93 صورة للرقم (1) ، 120 صورة للرقم (2) ، 84 صورة للرقم (3) ,122 صورة للرقم (4) ، 151 صورة للرقم (5) ,150 صورة للرقم (6) ,161 صورة للرقم (7) ,143 صورة للرقم (8) و111 صورة للرقم (9)، حيث تم باستعمال خوارزمية التدريب السابقة انتاج ملف arsy.traineddata وذلك لتدريب tesseract على الأرقام العربية.

4-3-3-2- التعامل مع tesseract:

يشكل الرماز OCR.py النص البرمجي المسؤول عن تنفيذ المرحلة الثالثة من نظامنا المقترح (انظر الملحق 1)، يأخذ كدخل له مسار مجلد الدخل، ومن أجل كل صورة يعمل هذا الرماز على تهيئتها كدخل لمحرك ال tesseract من خلال مجموعة من العمليات المعالجة المسبقة (preprocessing operation).

من أجل كل صورة في مجلد الدخل ، يتم التخلص من الضجيج (بيكسلات إضافية تتولد بسبب ضعف دقة الصورة أو بسبب خيالات الأجسام في الخلفية على لوحة السيارة) من خلال تابع cv2.fastNlMeansDenoisingColored الذي توفره مكتبة open-cv، و ذلك لتجنب التعرف على بيكسلات الضجيج على أنها الرقم (0) من أرقام المحارف العربية، ثم يتم تحويل الصورة مخففة الضجيج إلى صورة ثنائية باستعمال تابع cv2.threshold و تفعيل الخيار cv2.THRESH_OTSU من أجل اختيار خوارزمية Otsu-threshold للحصول على أفضل قيمة للعتبة و منه صور ثنائية أوضح، و من ثم تطبيق tesseract على الصور المعالجة و طباعة خرج ال tesseract إلى ملف نصي.



الشكل 34:(1) الصورة المقتطعة للأرقام العربية ، (2) الصورة بعد إجراء إزالة الضجيج ، (3) الصورة الثنائية ، (4) الخرج المصي من **tesseract**

يمكن التعامل مع tesseractعن طريق تنفيذ الأمر الآتي على مفسر أوامر Linux.

tesseract FILE OUTPUTBASE [OPTIONS]… [CONFIGFILE]…

حيث:

FILE الصورة التي يراد معالجتها.

OUTPUTBASE: الملف الذي سيحوي خرج ال tesseract أو يمكن استعمال stdout لطباعة ناتج معالجة ال tesseract على مفسر الأوامر.

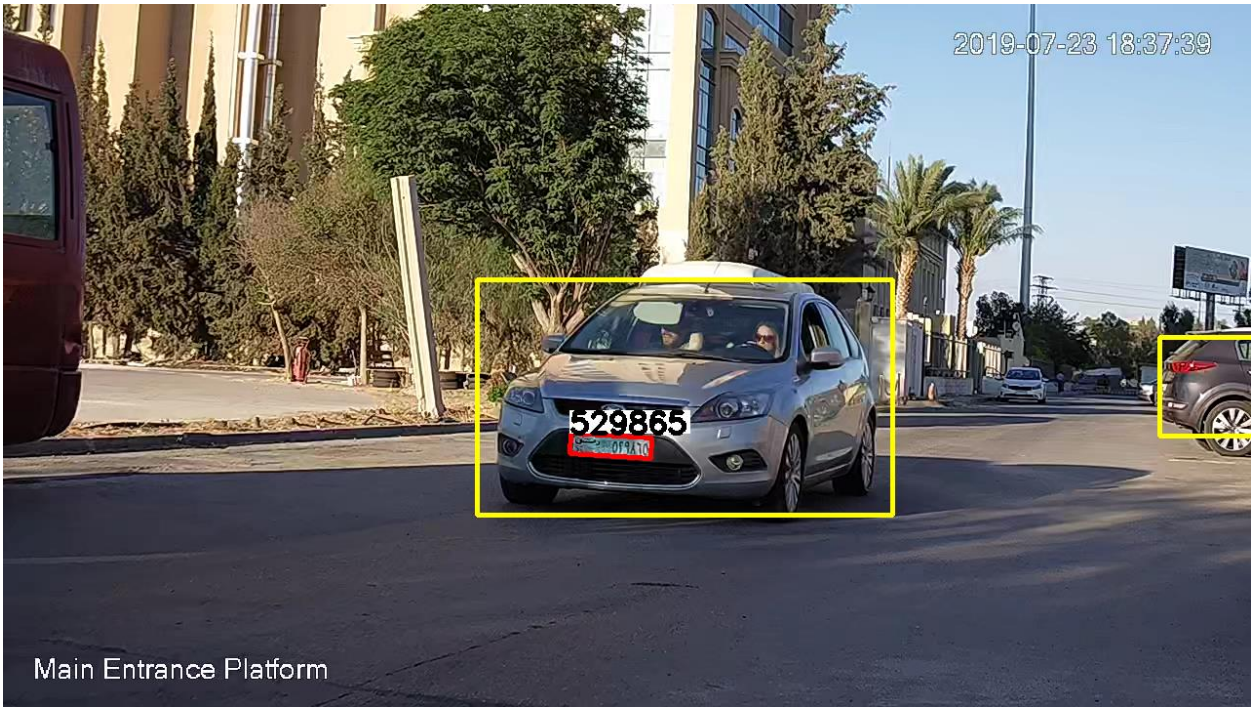
OPTIONS: تفعيل عدة خيارات أثناء معالجة الصورة نذكر منها:

Script l-: من أجل تحديد اللغة المستعملة في عملية التعرف (اسم ملف traineddata).

psm N--: من أجل مساعدة ال tesseract على معرفة طبيعة توضع النص في الصورة، فمثلاً في حالة N= 7 يفترض tesseract أن النص في الصورة يقع على سطر واحد فقط، N=10 النص في الصور هو محرف واحد فقط.

4-4- توليد الخرج

يعمل النظام على توليد خرج يوضح موقع السيارة من الصورة عن طريق إحاطتها بمربع في صورة الخرج، ومن ثم تحديد موقع لوحة السيارة من خلال إحاطة حواف لوحة السيارة بمربع، وفي النهاية إظهار خرج مرحلة ال OCR فوق المحارف العربية في الصورة (الشكل 35).



الشكل 35: مثال خرج النهائي لأحد الصور المدخلة

من أجل تضمين ذلك تم توليد ملف gen-outputs.py (انظر الملحق 1) الذي يأخذ مسار مجلدي الدخل والخرج كدخل له ويخرج في مجلد الخرج مجموعة لصور السيارات المعالجة، حيث يتم رسم مربع أصفر اللون محيط بهيكل السيارة في الصورة ومستفيداً من الملف النصي الذي تولده المرحلة الأولى (الذي يوضح موضع السيارة في الصورة) وتابع draw_label (تابع يأخذ كدخل له الصورة، الأبعاد التي يراد إحاطتها بمستطيل، لون المستطيل وعرضه).

يتم الاستفادة من الملفات النصية المولدة في المرحلة الثانية (الذي يحوي أبعاد اللوحة ضمن الصورة) من أجل رسم مربع أحمر اللون حول محيط لوحة السيارة باستعمال تابع draw_losangle الذي يأخذ كدخل الصورة والنقاط المراد رسم القطع المستقيمة فيما بينها واللون.

كما يتم كتابة خرج محرك tesseract (يظهر على شكل ملف نصي من أجل كل صورة دخل يوضح خرج ال tesseract من أجل الأرقام العربية في اللوحة) على صورة الخرج وذلك برسم مربع أبيض يعلو القسم العربي من اللوحة وكتابة خرج ال Tesseract على هذا المربع باستعمال تابع write2img.

4-5- دليل استخدام التطبيق

بما أن النظام متاح للاستخدام على الصور أو مقاطع الفيديو، ووجب توليد آلية تمكن للمستخدم من استعمال نظام ال ALPR المقترح على نوعي الدخل وبطريقة واضحة.

يوفر النظام للمستخدم واجهتين للتعامل:

1- التعامل مع الصور: وذلك بتشغيل ملف run.sh على مفسر الأوامر الخاص ب Linux.

2- التعامل مع مقاطع الفيديو: وذلك بتشغيل ملف video-processing.py.

4-5-1- run.sh:

و هو bash script (انظر الملحق 1) يهدف إلى سلسلة عمل نظام ال ALPR المقترح عن طريق التأكد أولا من وجود المكتبات المطلوبة (Darknet...) ثم التحقق من وجود مجلد الدخل و من وجود مجلد الخرج (في حال عدم وجود مجلد خرج يتم إنشاء مجلد جديد) ثم تنفيذ مراحل الثلاثة لنظام ال ALPR المقترح عن طريق تنفيذ ملفات python البرمجية الموافقة لكل مرحلة (vehicle-detection.py,license-plate-detection.py,OCR.py) ، ثم توليد الخرج و ذلك بتشغيل ملف gen-outputs.py، و بمجرد الانتهاء من توليد الخرج يقوم ملف run.sh بحذف جميع الملفات المؤقتة التي تولدت خلال المراحل الثلاث (صور هيكل السيارة المقتطع، صور اللوحات المقتطعة من سياراتها، ملفات النصية التي تحوي أبعاد لوحة السيارة ضمن الصورة، ملفات النصية التي توضح خرج ال tesseract.....) والمحافظة على صور المولدة من ملف gen-outputs.py كونها تشكل خرج النهائي للنظام المقترح.

يوفر ملف run.sh مجموعة من المحددات (arguments) وآلية مساعدة للتعرف على طريقة استعماله ونوعية الخيارات (options) المتاحة للمستخدم للتعامل مع ملف run.sh (الشكل 36).

```
(alpr) faresgh@faresgh-Inspiron-3543:~/Downloads/Alpr-System$ bash run.sh
Input dir not set.

 Usage:

   bash run.sh -i input/dir -o output/dir -c csv_file.csv [-h] [-l path/to/model
]:

   -i   Input dir path (containing JPG or PNG images)
   -o   Output dir path
   -c   Output CSV file path
   -l   Path to Keras LP detector model (default = data/lp-detector/wpod-net_upd
ate1.h5)
   -h   Print this help information

(alpr) faresgh@faresgh-Inspiron-3543:~/Downloads/Alpr-System$
```

الشكل 36 : آلية استعمال ملف **run.sh** مع الخيارات المتاحة للمستخدم

4-5-2- Video-processing.py :

ملف بلغة ال python (انظر ملحق 1) مهمته تجزئة مقاطع الفيديو الموجودة في مجلد الدخل إلى عدد من الإطارات (frames)، يتم حفظ هذه الإطارات في مجلد مؤقت ثم معالجة كل إطار على حدا باستعمال run.sh وفي النهاية، تم تجميع هذه الإطارات المعالجة إلى الخرج على شكل مقطع فيديو بلاحقة avi في مجلد الخرج.

يعمل video-processing.py على التقاط الإطارات باستعمال تابع VideoCapture الذي توفره مكتبة open-cv (يأخذ كدخل له مسار مقطع الفيديو أو الرقم 0 من أجل الحصول على الفيديو باستعمال الكاميرا الموصولة)، والذي يعمل على إنشاء غرض من الصف VideoCapture، واستعمال هذا الغرض من خلال المرور بحلقة تكرارية وتنفيذ عملية القراءة من هذا الغرض [VideoCapture Object name].read().

بعد الانتهاء من عملية معالجة الإطارات (تنفيذ ملف run.sh على إطارات مقطع الفيديو) يتم تجميع الإطارات المعالجة باستعمال تابع VideoWriter الذي توفره مكتبة open-cv والذي يأخذ كدخل له مسار ملف الخرج المرغوب، المرمز المستعمل في عملية إنشاء الفيديو (codec)، عدد الإطارات في الثانية بالنسبة لمقطع الخرج (FPS: Frame Per Second) وحجم الإطار.

يقوم تابع VideoWriter بإنشاء غرض من الصف VideoWriter، يستعمل هذا الغرض في عملية كتابة الإطارات المعالجة إلى ملف الخرج المرغوب باستعمال تابع [VideoWriter Object name].write().

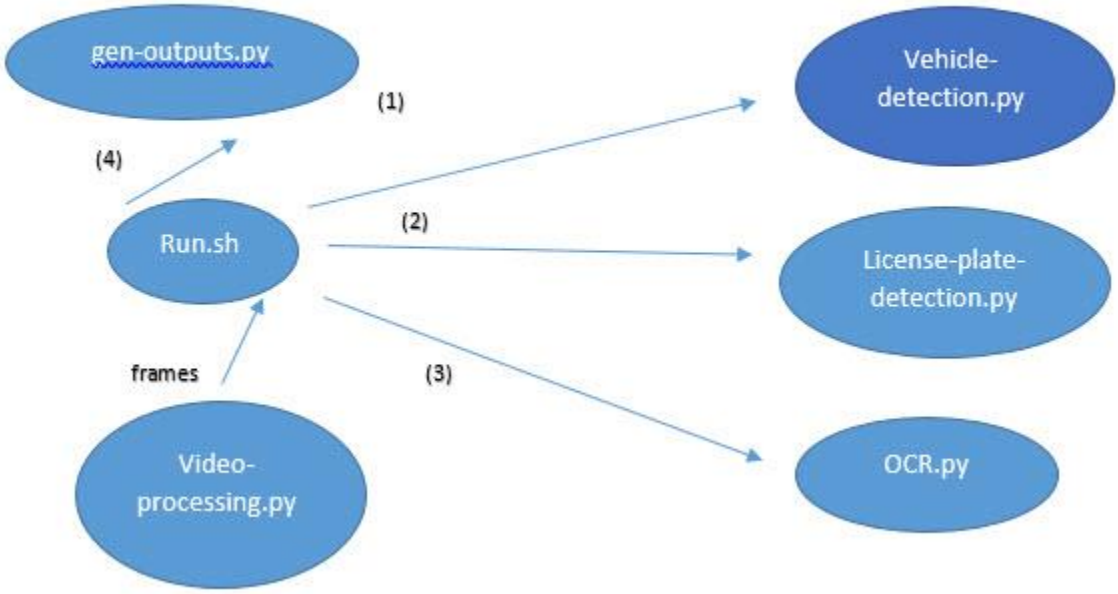
توفر مكتبة argparse في لغة python آلية للتعامل مع المحددات تسهل على المستخدم التعرف على المحددات المستخدمة في النص البرمجي واستعمالات هذه المحددات، يأخذ ملف video-processing محددين وهما indir يمثل مسار مجلد الدخل وoutdir الممثل لمسار مجلد الخرج (يوجد عملية تأكد من وجود مجلد الدخل والخرج وإنشاء الأخير في حال عدم وجوده) (الشكل 37)

```
(alpr) faresgh@faresgh-Inspiron-3543:~/Downloads/Alpr-System$ python video-proce
sseing.py -h
usage: video-processeing.py [-h] indir outdir

Processing Videos as input for alpr System

positional arguments:
  indir       Input dir for videos
  outdir      Output dir for Processed Videos

optional arguments:
  -h, --help  show this help message and exit
(alpr) faresgh@faresgh-Inspiron-3543:~/Downloads/Alpr-System$
```

الشكل 37: آلية استعمال ملف **video-processing.py** و الخيارات المتاحة للمستخدم

4-6- النتائج:

بعد الاطلاع ع الملفات البرمجية بلغة python (انظر الفقرة السابقة و ملحق 1) والتي تمثل التضمين البرمجي لكل مرحلة من المراحل المقترحة في المشروع، يمكن تلخيص آلية استدعاءات الملفات البرمجية (الشكل 38)



الشكل 38: المخطط الإستدعاءات البرمجية لنظام **ALPR** المقترح

تم اختبار النظام المقترح على عينة من 63 صورة، وتم تقسيم مناقشة نتائج كل مرحلة على حدي.

4-6-1- مرحلة استخلاص هيكل السيارة باستعمال yolo:

تم الاعتماد على عدً الصور التي يتمكن yolo من اكتشاف السيارة الأساسية (foreground car) بنجاح، دون اهتمام للسيارات الثانوية (background cars)، بالرغم أن yolo تمكن من الكشف عن السيارات الثانوية في معظم الصور، ولكننا لا نهتم لهذا الكشف كون السيارة الثانوية هي السيارة المتواجدة في الخلفية والتي لا تظهر لوحتها، بالتالي هي غير مهمة بالنسبة لتقييم نظام ALPR المقترح.

من أجل 63 صورة لسيارات ضمن المشهد (عينة التدريب)، تمكن yolo من اكتشاف 60 هيكل لسيارات في واجهة الصورة، حيث أخفق yolo في اكتشاف السيارات البعيدة في صورتين، وأيضاً اكتشاف خاطئ لحدود السيارة في أحد الصور مما أثر على خرج المراحل التي تليه (الشكل 39).

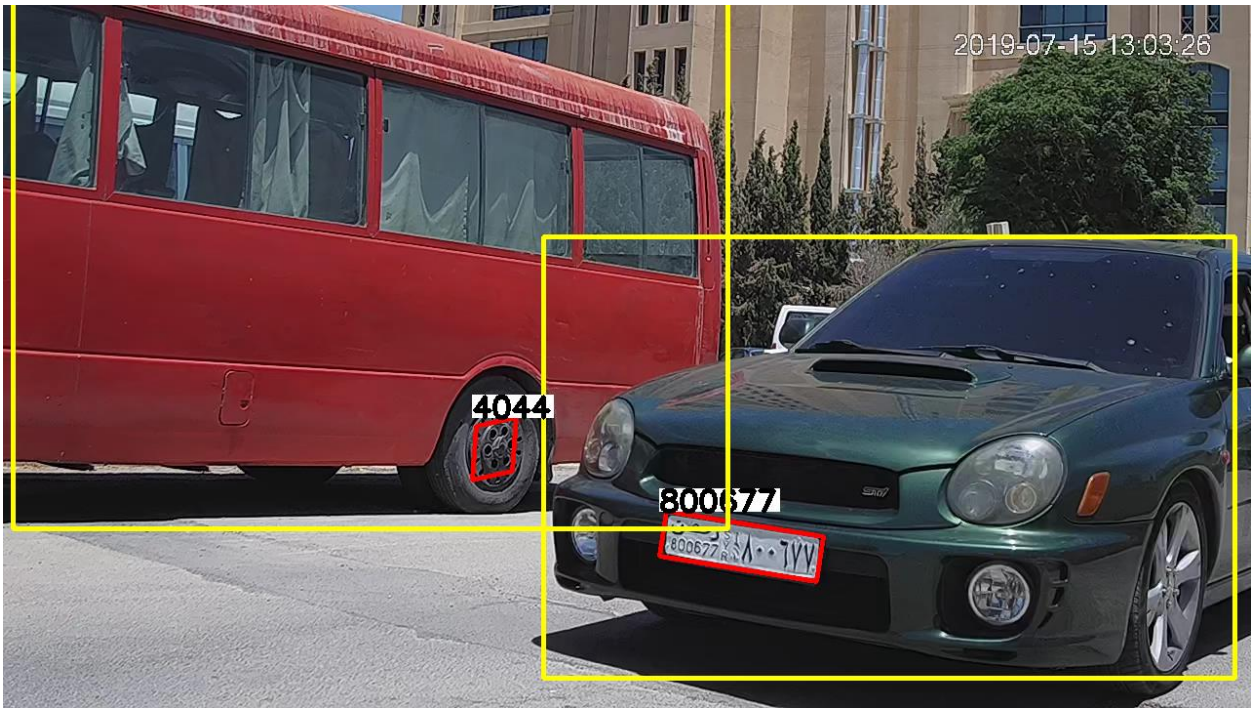
الشكل 39: اكتشاف خاطئ لهيكل السيارة قد يؤثر على خرج المراحل التي تليه

ومنه يستطيع yolo اكتشاف واستخلاص هيكل السيارة من صورة الدخل بدقة 95.24%.

4-6-2- مرحلة استخلاص لوحة السيارة من هيكلها باستعمال WPOD-Net

تم تقييم هذه المرحلة بنفس الطريقة السابقة، وهي عدً الصور التي يتمكن النظام من تحديد موقع لوحة السيارة فيها. من أجل 60 صورة دخل لهياكل السيارات المستخلصة بنجاح من yolo تمكن WPOD-NET من التعرف على 55 لوحة سيارة، بدقة 91.67%.

بالنسبة للصور ال 5 المستبعدة، وعلى الرغم من التعرف الصحيح ل WPOD-NET على لوحات السيارات في هذه الصور، إلا أنه تعرف على أشكال في هذه الصور على أنها لوحات سيارات وهي ليست كذلك، مما قد يسبب أخطاء فادحة في حال وضع النظام المقترح قيد العمل (الشكل 40)



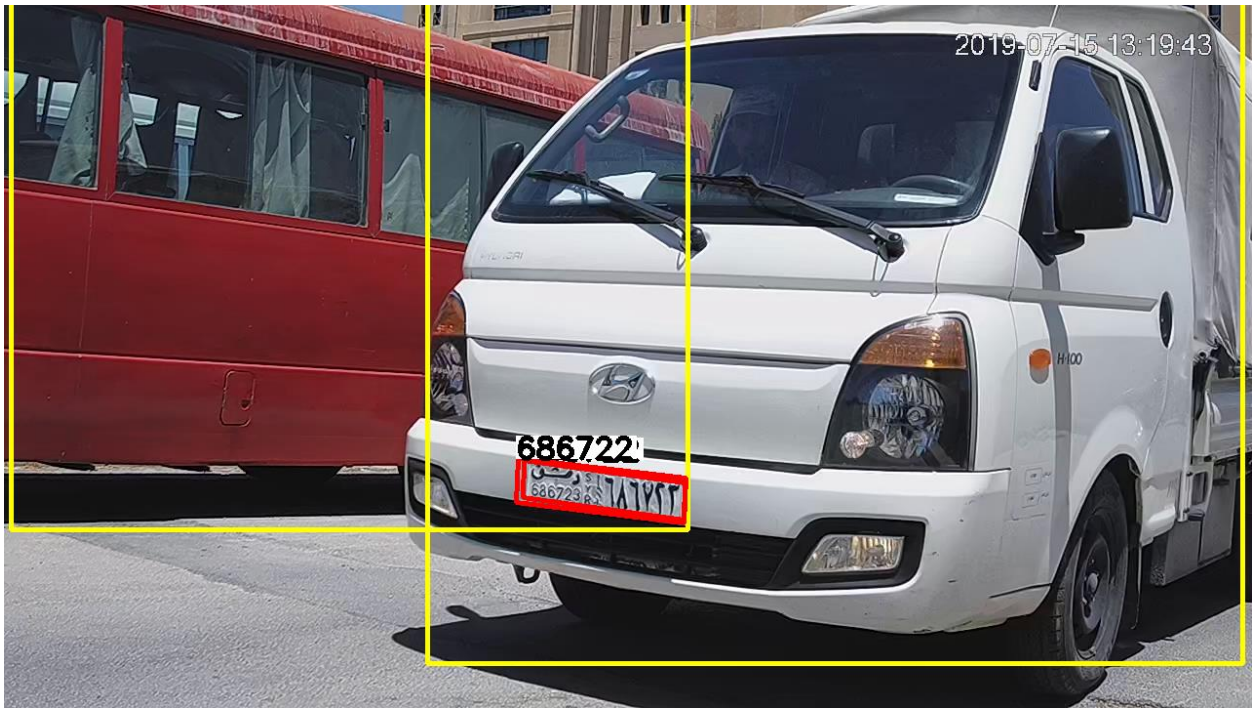
الشكل 40: التعرف الخاطئ ل **WPOD-NET** على إطار السيارة باعتباره لوحة للسيارة الثانوية

4-6-3- التعرف على الأرقام العربية:

تم تقييم مرحلة التعرف على الأرقام OCR وفق منهجيتين

- **التعرف الصحيح على كل محرف لوحده**: وذلك بمعرفة عدد الأرقام المتعرف عليها بشكل صحيح من 360 رقم مدخل (60 لوحة كل لوحة تحوي 6 أرقام)، حيث تمكن ال tesseract من التعرف الصحيح على 336 رقم بمعدل دقة 93.34%.

من أبرز الأخطاء في خرج ال Tesseract كانت في عملية التمييز بين الرقمين 2 و3 باللغة العربية (نظراً للتشابه الحاد بينهما)، حيث لوحظ نقص في الرقم 3 في عينة التدريب (84 صورة للرقم 3 فقط) مما يفسر التعرف الخاطئ وعدم قدرة ال Tesseract على التمييز بين الرقمين 2 و3 (الشكل 41).



الشكل 41: التعرف الخاطئ على الرقم 3 على أنه الرقم 2

- **التعرف الصحيح على لوحة كاملة**: بسبب النقص في عينة الرقم 3 في التدريب، فإن النظام غير قادر (في وضعه الحالي) على التعرف على جميع لوحات السيارات، حيث تمكن النظام من التعرف 41 لوحة بشكل صحيح بدقة 68.2%.

من أجل 20 صورة التي لم يتعرف على لوحاتها، 18 صورة منهم كانت المشكلة في عدم التمييز بين الرقمين 2 و3 الموضحة سابقاً، بالتالي يمكن بزيادة عينة الرقم ال 3 (تدريب ال tesseract على عينة أكبر من العينة المقترحة) زيادة قدرة النظام على التعرف على لوحة السيارة بدقة أكبر.

الفصل الخامس

خاتمة المشروع

يختتم هذا الفصل المشروع. فيبين الفائدة المكتسبة من المشروع، والمشكلات والصعوبات التي واجهتنا خلاله، ويبين بعض الآفاق المستقبلية.

5-1- الخاتمة والفائدة المكتسبة:

في هذا المشروع، تم بناء نظام ALPR. بالإضافة إلى توليد ملف تدريب لمحرك tesseract (arsy.traineddata) أفضل من ملف التدريب (ar.traineddata) المقدم من tesseract في مجال التعرف على الأرقام العربية.

تم التعرف على عدد واسع من المفاهيم والطرائق خلال تنفيذ هذا المشروع. فتم التعرف على مفاهيم تعلم الآلة المختلفة لأول مرة، كما تم التعرف على مفاهيم ALPR والخوارزميات المختلفة لتنجيز هذا النوع من النظم، والتعرف على عدد من الأدوات المفيدة أثناء التنفيذ العملي. كما تم تطوير النظام باستعمال لغة python مما أدى إلى إغناء المعرفة بهذه اللغة.

5-2- المشكلات والصعوبات:

تتمثل الصعوبات الأساسية التي واجهتنا في المشروع ب:

- الحاجة لدراسة مرجعية مطولة، وذلك كونه الاحتكاك الأول مع مفاهيم تعلم الآلة وخوارزمياتها.
- الدقة الغير عالية نسبياً للصور المستعملة في إنشاء Dataset الخاصة بنا، مما أدى إلى استهلاك وقت كبير لإنشاء ملف التدريب.
- التجارب والنماذج السابقة: مثل اقتراح ال open-ALPR ليقوم بعملية استخلاص الأرقام من السيارة، والذي انتهى بالفشل والاستعاضة عنه بالاستخلاص اليدوي للأرقام لتدريب tesseract عليها.
- صعوبة الموضوع المدروس: حيث أن نظام ال ALPR هو نظام حساس جداً يتأثر خرجه بشكل واضح بمعايير المحيطية (زاوية التصوير والظلال المسقطة من الأشياء)، ويحتاج إلى خوارزميات قوية من أجل معالجة الصور بظروف إضاءة وتصوير مختلفة.

5-3- التوسع والآفاق المستقبلية:

توجد العديد من الآفاق المستقبلية التي يمكن طرحها بناءً على هذا المشروع. فيمكن تطوير هذا المشروع بتمديده للتعرف على الأرقام الإنجليزية في لوحة السيارات السورية باستخدام محرك OCR آخر، ثم المقارنة بين خرجين محركي ال OCR من أجل رفع قدرة النظام على استخلاص لوحات السيارات.

يمكن أيضاً تطوير النظام ليصبح قادراً على استخلاص لوحات السيارات السورية بكافة أنواعها (الحكومية – ذوي الاحتياجات الخاصة) بحيث يتمكن استعماله في الحياة العملية.

يمكن زيادة دقة النظام بزيادة العينات (وخاصة الرقم 3)، كما يمكن توسيع ملف (arsy.traineddata) ليشمل المحارف العربية للتمكن من التعرف على المحافظة التي تتبع لها المركبة من جهة، ومن جهة أخرى توفير ملف تدريب أقوى من ملف التدريب المقترح من tesseract لكافة محارف اللغة العربية.

المصادر والمراجع

[1] Bachchan A.K., Gorai A., Gupta P. (2017) Automatic License Plate Recognition Using Local Binary Pattern and Histogram Matching. In: Huang DS., Jo KH., Figueroa-García J. (eds) Intelligent Computing Theories and Application. ICIC 2017. Lecture Notes in Computer Science, vol 10362. Springer, Cham

[2]M. Sarfraz, M. J. Ahmed, and S. A. Ghazi, "Saudi Arabian license plate recognition system," in Proc. Int. Conf. Geom. Model. Graph.,2003, pp. 36–41.

[3]D. Zheng, Y. Zhao, and J. Wang, "An efficient method of license plate location," Pattern Recognit. Lett., vol. 26, no. 15, pp. 2431–2438,2005.

[4]K. Kanayama, Y. Fujikawa, K. Fujimoto, and M. Horino, "Development of vehicle-license number recognition system using real-time image processing and its application to travel-time measurement," in Proc.IEEE Veh. Tech. Conf., May 1991, pp. 798–804

[5]V. Kamat and S. Ganesan, "An efficient implementation of the Hough transform for detecting vehicle license plates using DSPs," in Proc. Real-Time Tech. Applicat. Symp., 1995, pp. 58–59

[6]P. Wu, H.-H. Chen, R.-J. Wu, and D.-F. Shen, "License plate extraction in low resolution video," in Proc. Int. Conf. Pattern Recognit., vol. 1. 2006, pp. 824–827

[7]J. Matas and K. Zimmermann, "Unconstrained license plate and text localization and recognition," in Proc. IEEE Conf. Intell. Transp. Syst., Sep. 2005, pp. 572–577

[9]S. Zhang, M. Zhang, and X. Ye, "Car plate character extraction under complicated environment," in Proc. IEEE Int. Conf. Syst. Man Cybern.,vol. 5. Oct. 2004, pp. 4722–4726

[10]I. Paliy, V. Turchenko, V. Koval, A. Sachenko, and G. Markowsky," Approach to recognition of license plate numbers using neural networks," in Proc. IEEE Int. Joint Conf. Neur. Netw., vol. 4. Jul. 2004,pp. 2965–2970.

[11]F. Aghdasi and H. Ndungo, "Automatic license plate recognition system," in Proc. AFRICON Conf. Africa, vol. 1. 2004, pp. 45–50.

.[12]M.-K. Kim and Y.-B. Kwon, “Recognition of gray character using Gabor filters,” in Proc. Int. Conf. Inform. Fusion, vol. 1. 2002, pp.419–424.

[13]S. S. Omran and J. A. Jarallah, "Iraqi car license plate recognition using OCR," 2017 Annual Conference on New Trends in Information & Communications Technology Applications (NTICT), Baghdad, 2017, pp. 298-303.

[14]Alginahi, Yasser. (2011). Automatic Arabic License Plate Recognition. International Journal of Computer and Electrical Engineering. 10.7763/IJCEE.2011.V3.360.

[15]S. Du, M. Ibrahim, M. Shehata and W. Badawy, "Automatic License Plate Recognition (ALPR): A State-of-the-Art Review," in IEEE Transactions on Circuits and Systems for Video Technology, vol. 23, no. 2, pp. 311-325, Feb. 2013.

[16]S. M. Silva and C. R. Jung,"License Plate Detection and Recognition in Unconstrained Scenarios", pp. 580-596,sep. 2018.

[17] R. Smith, "An Overview of the Tesseract OCR Engine," Ninth International Conference on Document Analysis and Recognition (ICDAR 2007), Parana, 2007, pp. 629-633.

[18] Aurélien Géron. Hands-on machine learning with Scikit-Learn and TensorFlow: concepts, tools, and techniques to build intelligent systems. ” O’Reilly Media, Inc.”, 2017

الملحق 1

الرماز البرمجي

ملاحظات :

تتوفر آلية أدوات و المكتبات الخاصة ب yolov2 و Darknet عبر الموقع الرسمي لهم مع شرح آلية التنصيب. https://pjreddie.com/darknet/yolo/

يجب بداية انشاء python virtual environment باستعمال anaconda و تنصيب كل من tensorflow و ال keras من مواقعهم الرسمية(الحاجة لهم لعمل البرنامج). https://www.tensorflow.org / / https://keras.io/

يمكن تنصيب ال tesseract و كل المكتبات التي يعتمد عليها و ذلك عن طريق مراجعة الصفحة الإلكتروني ل tesseract على github. https://github.com/tesseract-ocr/tesseract/wiki .

يمكن تنصيب مكتبة open alpr من خلال مراجعة موقعهم على الإنترنت http://doc.openalpr.com/opensource.html#training-ocr .

ينصح بقراءة الفقرة (4-3-) للفهم المصلحات و التوابع المستخدمة ضمن الأكواد البرمجية.

لدعم تشغيل النظام على gpu (يؤدي إلى نتائج أسرع) يجب تنزيل Cuda, Cunnd , Nvidia Driver من موقع Nvidia الرسمي .

```
Vehicle-detection.py
"""
Created on TUS Aug  20 15:50:12 2019
@author: faresgh
"""
import sys
import cv2
import numpy as np
import traceback
import darknet.python.darknet as dn

from src.label                  import Label, lwrite
from os.path                    import splitext, basename, isdir
from os                         import makedirs
```

```
from src.utils                    import crop_region, image_files_from_folder
from darknet.python.darknet import detect
if __name__ == '__main:'___
        try:
                #specifying input and output directories
                input_dir  = sys.argv[1]
                output_dir = sys.argv[2]
                #Specifiying the threshold for an object to be considered as vehicle
                vehicle_threshold = .5
                #Specifiying the yolo model weights and configuration files
                vehicle_weights = 'data/vehicle-detector/yolo-voc.weights'
                vehicle_netcfg  = 'data/vehicle-detector/yolo-voc.cfg'
                vehicle_dataset = 'data/vehicle-detector/voc.data'
                vehicle_net  = dn.load_net(vehicle_netcfg, vehicle_weights, 0)
                vehicle_meta = dn.load_meta(vehicle_dataset)
                #getting images files from input directory and sort them
                imgs_paths = image_files_from_folder(input_dir)
                imgs_paths.sort()
                #verify the existense of output directory and create it (if not existed)
                if not isdir(output_dir):
                        makedirs(output_dir)
                print 'Searching for vehicles using YOLO'…
                for i,img_path in enumerate(imgs_paths):
                        print '\tScanning %s' % img_path
                        bname = basename(splitext(img_path)[0])
                        #using detect function to retrive the boundry boxes and its class
names
                        R,_ = detect(vehicle_net, vehicle_meta, img_path
,thresh=vehicle_threshold)
                        #verify if the box is in class car or bus
                        R = [r for r in R if r[0] in ['car','bus']]
                        print '\t\t%d cars found' % len(R)
                        if len(R):
                                Iorig = cv2.imread(img_path)
                                #getting the width and hight of an array reverserd order
                                WH = np.array(Iorig.shape[1::-1],dtype=float)
                                Lcars[] =
                                for i,r in enumerate(R):
                                        #getting the diminsion of the car reigon according to
the bounding box diminsion
                                        cx,cy,w,h = (np.array(r[2])/np.concatenate( (WH,WH)
)).tolist()
```

```
                                        tl = np.array([cx - w/2., cy - h/2.])
                                        br = np.array([cx + w/2., cy + h/2.])
                                        label = Label(0,tl,br)
                                        Icar = crop_region(Iorig,label)
                                        Lcars.append(label)
                                        cv2.imwrite('%s/%s_%dcar.png' %
(output_dir,bname,i),Icar)
                                #printing the class number and left-top and right-bottom
cordinates of a veahicle in a file
                                lwrite('%s/%s_cars.txt' % (output_dir,bname),Lcars)
        except:
                traceback.print_exc()
                sys.exit(1)
        sys.exit(0)
----------------------------------------------------------------------------------------------

License-plate-detection.py
import sys, os
import keras
import cv2
import traceback
from PIL import Image, ImageDraw
from src.keras_utils                    import load_model
from glob                                       import glob
from os.path                                    import splitext, basename
from src.utils                                  import im2single
from src.keras_utils                    import load_model, detect_lp
from src.label                                  import Shape, writeShapes
image_path_output = '/tmp/output'/
def adjust_pts(pts,lroi):
        return pts*lroi.wh().reshape((2,1)) + lroi.tl().reshape((2,1))
if __name__ == '__main:'___
        try:
                #verifiying input directory which is the output directory of the privious step
                input_dir  = sys.argv[1]
                output_dir = input_dir
                #verifying the threshold value for a shape to be considered as plate
                lp_threshold = .5
                #spicifying the wpod_net module path
                wpod_net_path = sys.argv[2]
                wpod_net = load_model(wpod_net_path)
                #getting previous step car images from the input
```

```
                imgs_paths = glob('%s/*car.png' % input_dir)
                print 'Searching for license plates using WPOD-NET'
                for i,img_path in enumerate(imgs_paths):
                        print '\t Processing %s' % img_path
                        bname = splitext(basename(img_path))[0]
                        Ivehicle = cv2.imread(img_path)
                        #specifying the diminsion and ratio
                        ratio = float(max(Ivehicle.shape[:2]))/min(Ivehicle.shape[:2])
                        side  = int(ratio*288.)
                        bound_dim = min(side + (side%(2**4)),608)
                        print "\t\tBound dim: %d, ratio: %f" % (bound_dim,ratio)
                         #retriving of the licencse plate an its dimintions
                        Llp,LlpImgs,_ =
detect_lp(wpod_net,im2single(Ivehicle),bound_dim,2**4,(240,80),lp_threshold)
                        if len(LlpImgs):
                                Ilp = LlpImgs[0]
                                Ilp = cv2.cvtColor(Ilp, cv2.COLOR_BGR2GRAY)
                                Ilp = cv2.cvtColor(Ilp, cv2.COLOR_GRAY2BGR)
                                 #defineing object from class shape in label.py which spicify
the four cordinates for the lp rectangler after resizing
                                s = Shape(Llp[0].pts)
                                #writing the lp image
                                cv2.imwrite('%s/%s_lp.png' % (output_dir,bname),Ilp*255.)
                                #extracting the arabic nymbers erea of a lp and save it
                                im = Image.open('%s/%s_lp.png' % (output_dir,bname))
                                im_width, im_height = im.size
                                im = im.crop((im_width/2, 0, im_width, im_height))
                                image_name_output = '%s_lp_Croped.png' % (bname)
                                image_path_output = output_dir"/" +
                                im.save(image_path_output + image_name_output)
                                writeShapes('%s/%s_lp.txt' % (output_dir,bname),[s])
        except:
                traceback.print_exc()
                sys.exit(1)
        sys.exit(0)
-------------------------------------------------------------------------------------------
OCR.py
/!#usr/bin/env python2
 -*- #coding: utf-8-*-
"""
Created on Mon Sep  2 17:16:18 2019
@author: faresgh
```

```
"""
import sys
import cv2
import numpy as np
import traceback
import os
from PIL import Image, ImageDraw
from os.path                    import splitext, basename
from glob                       import glob
if __name__ == '__main:'___
        try
                input_dir  = sys.argv[1]
                output_dir = input_dir
                 #getting the arabic numbers area from the privous step
                imgs_paths = sorted(glob('%s/*lp_Croped.png' % output_dir))
                print 'Performing OCR'…
                for i,img_path in enumerate(imgs_paths):
                        print '\tScanning %s' % img_path
                        bname = basename(splitext(img_path)[0])
                        img = cv2.imread(img_path)
                         #denoising of image saving it into dst image
                        dst = cv2.fastNlMeansDenoisingColored(img, None, 10, 10, 7, 15)
                        cv2.imwrite('%s/%s_dn.png'% (output_dir,bname),dst)
                         #converting the previous image to binary
                        im_gray = cv2.imread('%s/%s_dn.png'% (output_dir,bname),
cv2.IMREAD_GRAYSCALE)

        imbw2=cv2.threshold(im_gray,100,255,cv2.THRESH_BINARY+cv2.THRESH_OTSU)]
[1
                        cv2.imwrite('%s/%s_outs.png'% (output_dir,bname),imbw2)
                        #Modifying the input to tesseract
                        tmp = Image.open('%s/%s_outs.png'% (output_dir,bname))
                        t_width, t_height = tmp.size
                        t = tmp.crop((t_width/50, t_height/7, t_width*49/50, t_height*6.5/7))
                        image_name_output = '%s_final.png' % (bname)
                        t.save('%s/'%(output_dir) + image_name_output)
                        #excuting tesseract command where arsy is the .traineddata file that
we make
                        command = "tesseract %s/%s_final.png %s/%s_str -l arsy --psm 7"%
(output_dir,bname,output_dir,bname)
                        os.system(command)
        except:
                traceback.print_exc()
```

```
                sys.exit(1)
        sys.exit(0)
        ---------------------------------------------------------------------------
Run.sh
/!#bin/bash
check_file ()
}
        if [ ! -f "$1" ]
        then
                return 0
        else
                return 1
        fi
{
check_dir ()
}
        if [ ! -d "$1" ]
        then
                return 0
        else
                return 1
        fi
{
 #Check if Darknet is compiled
check_file "darknet/libdarknet.so"
retval?$=
if [ $retval -eq 0 ]
then
        echo "Darknet is not compiled! Go to 'darknet' directory and 'make"!'
        exit 1
fi
lp_model="data/lp-detector/wpod-net_update1.h5"
input_dir"=
output_dir"=
csv_file"=
 #Check # of arguments
usage} ()
        echo""
        echo " Usage":
        echo""
        echo "   bash $0 -i input/dir -o output/dir -c csv_file.csv [-h] [-l path/to/model]":
        echo""
```

```
        echo "   -i   Input dir path (containing JPG or PNG images)"
        echo "   -o   Output dir path"
        echo "   -c   Output CSV file path"
        echo "   -l   Path to Keras LP detector model (default = $lp_model)"
        echo "   -h   Print this help information"
        echo""
        exit 1
{
while getopts 'i:o:c:l:h' OPTION; do
        case $OPTION in
                i) input_dir=$OPTARG;;
                o) output_dir=$OPTARG;;
                c) csv_file=$OPTARG;;
                l) lp_model=$OPTARG;;
                h) usage;;
        esac
done
 #Cheking if input , output and csv files are specified in the command
if [ -z "$input_dir"  ]; then echo "Input dir not set."; usage; exit 1; fi
if [ -z "$output_dir" ]; then echo "Ouput dir not set."; usage; exit 1; fi
if [ -z "$csv_file"   ]; then echo "CSV file not set." ; usage; exit 1; fi
 #Check if input dir exists
check_dir $input_dir
retval?$=
if [ $retval -eq 0 ]
then
        echo "Input directory ($input_dir) does not exist"
        exit 1
fi
 #Check if output dir exists, if not, create it
check_dir $output_dir
retval?$=
if [ $retval -eq 0 ]
then
        mkdir -p $output_dir
fi
 #End if any error occur
set -e
 #Detect vehicles
python vehicle-detection.py $input_dir $output_dir
 #Detect license plates
python license-plate-detection.py $output_dir $lp_model
```

```
#OCR
#python license-plate-ocr.py $output_dir
python OCR.py $output_dir
#Draw output and generate list
python gen-outputs.py $input_dir $output_dir > $csv_file
#Clean temporary files and draw output
rm $output_dir/*_lp.png
rm $output_dir/*car.png
rm $output_dir/*_cars.txt
rm $output_dir/*_lp.txt
rm $output_dir/*_str.txt
rm $output_dir/*_dn.png
rm $output_dir/*_final.png
rm $output_dir/*_outs.png
rm $output_dir/*_Croped.png
```

Video-processing.py

```
/!#usr/bin/env python2
-*- #coding: utf-8-*-
"""
Created on Wed Sep  4 13:42:17 2019
@author: faresgh
"""
import cv2
import numpy as np
import os
import argparse
import sys
import traceback
from os.path                    import isdir
from os                         import makedirs
from src.utils                  import videos_files_from_folder
from PIL import Image, ImageDraw
#preparing argument parser to help with using the code
parser = argparse.ArgumentParser(description='Processing Videos as input for alpr
System')
parser.add_argument('indir', type=str, help='Input dir for videos')
parser.add_argument('outdir', type=str, help='Output dir for Processed Videos')
args = parser.parse_args()
if __name__ == '__main:'___
        try:
```

```
        input_dir = args.indir
        output_dir = args.outdir
#           getting mp4 videos ( the function videos_files_from_folder is self_made)
        video_paths = videos_files_from_folder(input_dir)
        video_paths.sort()

        if not isdir(output_dir):
            makedirs(output_dir)
        print  'Processing Videos                '
        for i,video_path in enumerate(video_paths):
            print 'Scanning Videos'
            cap = cv2.VideoCapture(video_path)
            count = 0
            if (cap.isOpened()== False) :
                print("Error opening video stream or file")
            frame_width = int(cap.get(3))
            frame_height = int(cap.get(4))
               #Define the codec and create VideoWriter object.The output is stored in
'outpy.avi' file.
            out_str = '%soutpy%s.avi'% (output_dir,i)
            out = cv2.VideoWriter(out_str,cv2.VideoWriter_fourcc('M','J','P','G'), 10,
(frame_width,frame_height))
               #Read until video is completed
            while(cap.isOpened()):
               #       Capture frame-by-frame
                ret, frame = cap.read()
                if ret == True:
                    cv2.imwrite("Video/Input/frame%d.jpg" % count, frame)
                    count += 1
                else :
                    break
              #running the run.sh on all the frames extracted from the video
            command = "bash run.sh -i ./Video/Input/ -o %s -c %sresults.csv"
%(output_dir,output_dir)
            os.system(command)
               #writing the processed frames into the output video
            c = 0
            while(c != count ):
                img = cv2.imread("%sframe%d_output.png"% (output_dir,c))
                out.write(img)
                c += 1
        cap.release()
```

```
            out.release()
            cv2.destroyAllWindows()
        except:
            traceback.print_exc()
            sys.exit(1)
sys.exit(0)
-----------------------------------------------------------------------------------
 Gen-outputs.py
#!/usr/bin/env python2
# -*- coding: utf-8 -*-
"""
Created on Wed Sep  4 09:05:33 2019
@author: faresgh
"""
import sys
import cv2
import numpy as np
import os
from glob                                   import glob
from os.path                                import splitext, basename, isfile
from src.utils                              import crop_region, image_files_from_folder
from src.drawing_utils                      import draw_label, draw_losangle, write2img
from src.label                              import lread, Label, readShapes
from pdb import set_trace as pause
YELLOW = (  0,255,255)
RED    = (  0,  0,255)
input_dir = sys.argv[1]
output_dir = sys.argv[2]

img_files = image_files_from_folder(input_dir)

for img_file in img_files:
        bname = splitext(basename(img_file))[0]
        I = cv2.imread(img_file)
        detected_cars_labels = '%s/%s_cars.txt' % (output_dir,bname)
        Lcar = lread(detected_cars_labels)
        sys.stdout.write('%s' % bname)
        if Lcar:
                for i,lcar in enumerate(Lcar):
                        # drawing a rectangle around the top-left and bottom-right of the
vechicle
                        draw_label(I,lcar,color=YELLOW,thickness=3)
                        lp_label                = '%s/%s_%dcar_lp.txt'              %
(output_dir,bname,i)
```

```
                        lp_label_str    = '%s/%s_%dcar_lp_Croped_str.txt'          %
(output_dir,bname,i)
                        if isfile(lp_label):
                                # drawing red rectangle around the lp after returning it to the
image cordinate
                                Llp_shapes = readShapes(lp_label)
                                pts = Llp_shapes[0].pts*lcar.wh().reshape(2,1) +
lcar.tl().reshape(2,1)
                                ptspx = pts*np.array(I.shape[1::-1],dtype=float).reshape(2,1)
                                draw_losangle(I,ptspx,RED,3)
                                # writing the output of tesseract into a white box by default
                                if isfile(lp_label_str):
                                        with open(lp_label_str,'r') as f:
                                                lp_str = f.read().strip()
                                        llp = Label(0,tl=pts.min(1),br=pts.max(1))
                                        write2img(I,llp,lp_str)
                                        sys.stdout.write(',%s' % lp_str)
        cv2.imwrite('%s/%s_output.png' % (output_dir,bname),I)
        sys.stdout.write('\n')
```

الملحق 2

التعليمات المنفذة لتدريب Tesseract

يوضح هذا الملحق , التعليمات المنفذة على خرج تعليمة openalpr-utils-preparecharfortraning (ملف combined.box و combined.tif) باستعمال مفسر الأوامر terminal في ال linux .

يمكن الاستعاضة عن التعليمات الآتية بقراءة توثيق تدريب tesseract على github https://github.com/tesseract-ocr/tesseract/wiki/Training-Tesseract-3.03%E2%80%933.05

ملاحظة : ينصح بقراءة الفقرة (4-3-3-1-) لمعرفة العمليات و معناها و تسلسلها الصحيح.

```
#generating combined.box and combined.tif
openalpr-utils-prepcharsfortraining /home/faresgh/Desktop/Processing/final\ tiles/ --
tile_width 55
tesseract combined.tif arsy.Arabic.exp0 nobatch box.train
unicharset_extractor combined.box
#font name <italic> <bold> <fixed> <serif> <fraktur<
echo "Arabic 0 0 0 0 0" > font_properties
shapeclustering -F font_properties -U unicharset arsy.Arabic.exp0.tr
mftraining -F font_properties -U unicharset -O arsy.unicharset arsy.Arabic.exp0.tr
cntraining arsy.Arabic.exp0.tr
#prefix "relevant" files with our language code
mv inttemp arsy.inttemp
mv normproto arsy.normproto
mv pffmtable arsy.pffmtable
mv shapetable arsy.shapetable
combine_tessdata arsy .
#copy the created arsy.traineddata to the tessdata folder
#so tesseract is able to find it
sudo cp arsy.traineddata /usr/local/share/tessdata/
```