

الاسبوع الحادي عشر

Photo OCR

- في القسم الأخير ، هنعمل مثل عملی علي اغلب اللي درسناه في الـ ML عشان نعرف كذا حاجة زي
 - ازاي نعمل توصيف كامل للمشكلة
 - ازاي نبني البرنامج بالترتيب
 - ازاي نكتب كود كامل ، ونربط المصادر مع بعض
- و لاحظ ان في الحياة الطبيعية ، غالباً بيكون فيه فريق عمل كامل ، بيعملو تناول متكامل لحل المشكلة
- البرنامج هيكون عن الـ OCR اللي هو Optical character recognition ، وده الخاص بقراءة الـ text و الارقام المكتوبة يدوياً بالايد ، او المتصورة بكاميرا
- و استخدام الـ OCR بقى شديد الأهمية حالياً لأسباب كثيرة
 - مع انتشار الموبايل و التابلت في كل مكان
 - اعتمادنا الكامل على التكنولوجي في كل حاجة
 - استخدام تطبيقات للأكماء ، توصف لهم كل شيء ، وتقرأ لهم اي شيء مكتوب
 - في الـ self driving car محتاجين جداً كل شيء خاص بالـ OCR
- فمثلاً لو عندي الصورة ديه :



- الخوارزم هيبدأ الاول يحدد فين الكلام اساساً و يعلم عليه
- بعدها يبدأ يقارنه بالمكتبات اللي عنده عشان يحدد هي ايه



- يعني الخطوات بتكون كالتالي :
■ او لا يتم تحديد هو فين الكلام اساسا , ووضع مربع حوله



- الخطوة اللي بعدها هو تقسيم الكلمة نفسها لحروف متقطعة , كل حرق لوحده

ANTIQUE MALL

- ثالثا نمسك كل حرف , ونعرف معناه ايه بالتحديد , وبالتالي اقدر افهم الجملة بالكامل

A → A N → N T → T

- اخيرا اطبق احد الدوال اللي بتعمل spelling correction عشان لو فيه لغبطة بين مثلا او 1 او 0 و collaborate , فلو لقيت كلمة زي col1ab0rate اعرف انها

- الكلام ده اسمه machine learning pipeline

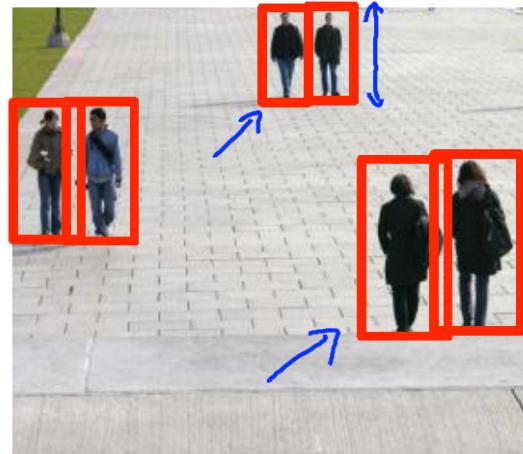


- و الـ pipeline معناها هنا , الـ flow chart اللي باستخدمها عشان اقسم المهام عندي , لخطوات متتالية
عشان اقدر انفذها

- وكل خطوة من ديه ممكن يكون فيها حاجات كتيرة من جوة , سواء خوارزم مباشر او NN او برنامج من لغة مختلفة , او وحدات معقدة
- كمان وارد جدا كل خطوة من ديه تحتاج لفريق عمل خمس افراد مثلا لانجازها

● التوافذ المتحركة Sliding Windwos

- وهو التكنيك الاساسي المصاحب لأغلب عمليات الـ OCR
- اول معضلة بتقابل الخوارزم اللي بيبدأ ببحث عن الكلمات هي اختلاف الـ Aspect Ratio من جملة لجملة
- يعني مثلا هنا ، لو فيه كاميرا بتدور علي ناس ماشية



- هتلaci ان النسبة بين الطول و العرض هي هي لكل الاشخاص , سواء قريب او بعيد , البعيد هو اقصر , لكن ارفع , فالنسبة هي هي
- بينما في الكلمات , فيه كلمات طويلة و قصيرة حسب عدد الحروف و الكلمات



- وده اللي بيخلطي المهمة اصعب على الخوارزم للبحث عن الكلمات

• فلو بدانا نجرب في فحص المشاة

- هنوق اننا هنحدد ابعد هي 82 بيكسل طول في 36 بيكسل عرض ، ونهجيب الاف الصور الايجابية (فيها شخص وبالتالي $y=1$) والاف الصور السلبية (مفيهاش شخص وبالتالي $y=0$) وادرب عليها البرنامج هنا مثلا صور ايجابية ■



■ بينما هنا صور سلبية ■



- البرنامج هيمسك الصور الايجابية ، ويحول كل واحدة فيهم لصف فيها مصفوفة ، كل مصفوفة فيها 82 صف في 36 عمود ، وفي كل معلومة فيهم هيكون فيها رقم من 0 ل 10 بحيث يوضح رقم الـ grey قد ايه ، علي اساس ان صفر ابيض تماما ، و 10 اسود تماما (رقم الاسود ممكن يزيد او يقل عن 10 حسب الكفاءة المطلوبة)
- ويقوم الخوارزم بفحص المستطيل الذي سيصل له ، لتحديد اذا كان ايجابي ام سلبي

• وخطوات التنفيذ تكون كالتالي

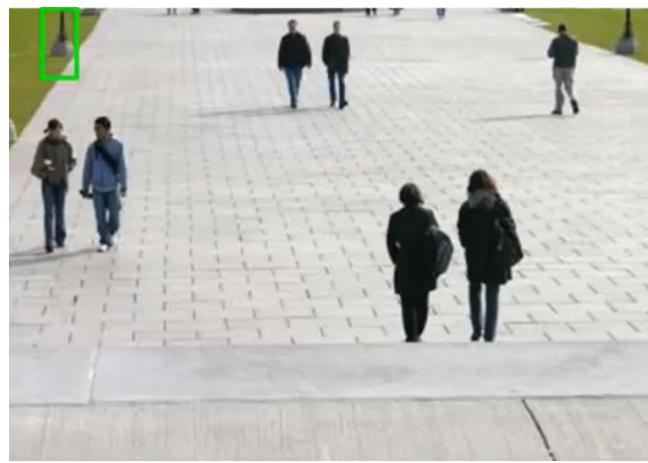
○ يتم البدء في فحص الصورة التالية



○ يقوم الخوارزم بتحديد اول مستطيل فيها هنا ، وفحص اذا كان به انسان ام لا ، ثم ينتقل يمينا مسافة صغيرة



○ يقوم بفحصها ايضا ، ثم ينتقل يمينا قليلا



○ اي انه يقوم بفحص الصور مستطيل مستطيل ، وينتقل يمينا قليلا قليلا ، ويري هل هي ايجابية ام سلبية

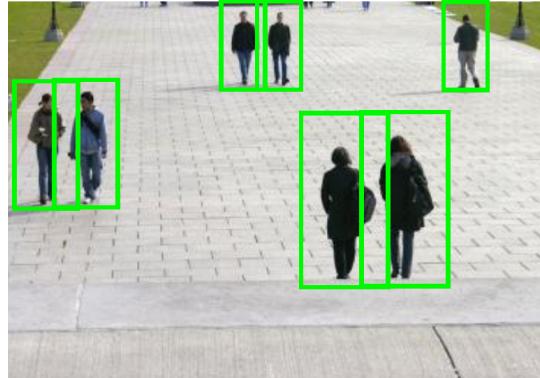
○ وبالتالي المستطيل الاخضر بيعمل scan للصورة بالشكل ده



- لاحظ ان كل خطوة لليمين بالنسبة للمستطيل بيكون اسمها **step** و القيمة المثلثي هي 1 بيكسل ، عشان منفوتش اي حاجة تروح مننا ، لكن طبعا ده هيكون ابطئ و بالتالي اغلي كتير
- بينما ممكن تعملها مثلا قيمة 5 بيكسل ، وتعمل فحص بصريا علي الصورة ، عشان تشفو لو فيه حاجة وقعت
- بعد ما يخلص الصف الاول بالكامل ، ينزل تحت خطوة واحدة ، برضه اما يكون ببيكسل واحد ، او اكتر شوية ، ويمشي يمسح الصف بالكامل ، وينزل كمان و هكذا لغاية لما يخلص الصورة ، بحيث تكون كدة



- ده السبب اللي بنسمى التكنيك ده **sliding windows** النوافذ المتحركة ، عشان عندي مستطيل اخضر بيتحرك يعمل مسح كامل للصورة
- لاحظ ان عندي كذا حاجة لابد من ضبطها او لا
 - قيمة الـ **step** الافقى و الراسى ، كل ما نقل ، كل ما الكفاءة تزيد اكتر، بس هيكون ابطئ ، وبالتالي اغلى
 - حجم المربع نفسه ، معروف نسبة بس مازا عن حكمه ، يعني لو كانت نسبة العرض للطول 1:5 ، فممكن يكون 20:100 ، او 25:125 ، او 30:150 ، كل الابعاد ديها 1 الي 5 ، لكن لابد من تحديد الحجم المناسب اللي هيقدر يشوف الصورة ، واللي غالبا هيكون مماثل لحجم عينة التدريب ، بحيث يكون زى ده



◦ كل ده كان عن فحص المشاه ، فمادا عن الحروف

- مشكلة الحروف الكبري ، ان مفيش ابعاد ثابتة اقدر احدد لها مستطيل يمسح الصورة عشان يشوفها و يختبرها
- عشان كدة مش هنتعامل مع الكلمات ولا الجمل ، ولكن مع الحروف ، لأن غالباً الحرف بيكون ليه بعد ثابت اشبه بالمربع
- هنعمل برضه training data للخوارزم ، بحيث نقوله ان ديه بيانات ايجابية ($y=1$)



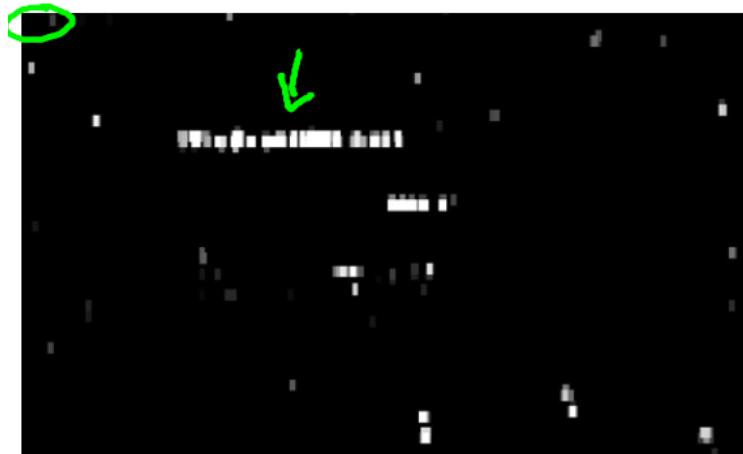
◦ و ديه بيانات سلبية ($y=0$)



- تكون الخطوات كالتالي :
- او لا امسك الصورة ، و اعمل مسح ليها لفحص الحروف في المربعات



- يبدا الخوارزم بتحويل المربعات الى الوان ، فالبيضاء هي التي بها حروف ($y=1$) ، والسوداء التي تخلو من الحروف ($y=0$) ، ولو كان هناك مساحة لم يحددها الخوارزم بدقة ، يقوم بعملها رمادي



- ثم يقوم الخوارزم بجمع المربعات المجاورة معا ، لتكون كلمة او جملة ، ويقوم باستبعاد التكون غير المنطقي ، زي مستطيل طوله اكبر بكثير من عرضه
- كمان لو الخوارزم شاف ان فيه خمس مربعات جنب بعض بيضاء ، ثم مساحة مجاورة سوداء ، ثم 3 مربعات بيضاء ، فياتي الشك ان المساحة السوداء هي حرف مكمل لهم لكن لم يتم قراءته



- اخيرا يقوم الخوارزم بتطبيق آلية spelling correction عشان يصح اي خطأ حدث

- كل ده كان الخطوة الاولى اللي اسمها text detection اللي هي مختصة بـس اني اشوف فين الحروف



- تعالي نروح للخطوة الثانية character segmentation اللي اقسم صورة الكلام عندي ، لحروف ورا بعض متقسمين بشكل معين ، يعني احول ده :

ANTIQUE MALL

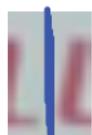
◦ لده



- و ده هيتم عن طريق البحث عن المسافات بين الحروف و بعضها ، عشان اعمل خطوط بينية
- فالاول هاعمل نماذج ايجابية ($y=1$)



- عشان الخوارزم يعرف يعمل خطوط بينية زي كدة



- واعمله نماذج سلبية ($y=0$) سواء هي في قلب الحرف (مفيش مسافة بينهم) ، او مفيش حرف اصلا



- بعدها اخلي الخوارزم يعمل خطوط بينهم بحيث تكون كدة



- و ده عن طريق نفس الـ sliding window اللي هتتحرك في بعد واحد افقيا بس
- بعدها ندخل علي المرحلة التاللة اللي هي character recognition اللي فيها الخوارزم يمسك حرف حرف
- و يقدر يعرف ده حرف ايه زي كدة



- اخيرا نجي مرحلة الـ spelling correction

● تصنيع البيانات artificial data synthesis

- من أفضل الطرق للوصول لخوارزم بكافأة عالية ، اننا نعمله تدريب على كم كبير من البيانات
- طيب هنجيب منين الكميه الكبيرة من البيانات ، هنسخدم ما يسمى تصنيع البيانات
- الطريقة ديه مش هيتفع تستخد في كل انواع الـ ml لكن في بعضها فقط ، لكن لو تم تطبيقها بتجيباك كميات كبيرة من بيانات التدريب
- و بيكون فيه تناولين ليها
 - إما معنديش اصلا بيانات و عايز اجيبيها من الصفر
 - او عندي بالفعل ، بس بيانات قليلة و عايز ازودها
- عملية الـ OCR و الـ character recognition تفع معاها فكرة تركيب البيانات

● النموذج الأول ، ايجاد بيانات من الصفر

- معناها ان معنديش بيانات اصلا ، و عايز اجيب بيانات من لا شئ ، فمثلا عايز بيانات كتيرة زي ديه عشان اعمل بيها تدريب لخوارزم عندي



- فممكن اني احمل كمية كبيرة من الخطوط Fonts و اعمل كود ب بحيث يكتب حروف عشوائية بخطوط عشوائية و يحطها على خلفيات عديدة

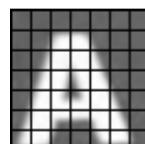
Abcdefg
Abcdefg
Abcdefg
 Abcdefg
Abcdefg

- بحيث تكون كدة

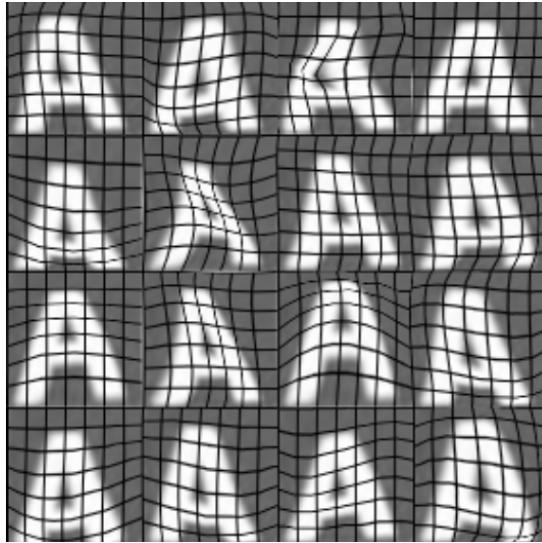


- وكل ما اعمل صورة , اكون مسجل معاها الحرف الخاص بيها , بحيث يكون عندي قيمة y في الـ classification و اقدر اعمل لها تدريب بشكل سليم

- النموذج الثاني , الخاص بتكبير البيانات
 - يعني انا عندي بالفعل مجموعة من البيانات , وعايز اخليها اكتر بكثير
 - فمثلا لو عندي صورة حرف كدة



- فممكنا اعمل deformation للصورة , بحيث يكون فيها بعض التتوّع زي كدة



- و ده هيفرق جدا ايجابا ، لما يكون في بيانات التدريب الاشكال ديه ، عشان يخلی الخوارزم قادر انه يغطي اكتر قدر من الحروف الملتوية
- و في نفس موضوع تكبير البيانات ، ممكن تعمل نفس الفكرة في الصوت
 - يعني امسك ملف صوت نقي ، لشخص بيقول (zero , one , two three , four , five)
 - بعدها اضيف عدد من المؤثرات اللي هتبظ الصوت ، زي مثلا دوشة ، او اصوات تانية ، او صوت هوا ، او صوت اتوبیسات
 - لما اضيف التأثيرات اللي بتاثير سلبا في الصوت ، و اعطي الخوارزم ، قيمة γ اللي هي النتيجة كذا كذا ، كدة انا باعطي بيانات متعددة و متعددة للخوارزم ، فلما هو يسمع صوت فيه دوشة او موسيقي او ايا كان هيفقدر يميشه
- وكأن الفكرة العامة لتكبير البيانات كالتالي :
 - اني باعمل جميع انواع التشويه المتوقع حدوثها في البيانات الحقيقية عندي (اللي عايزة اقيسها) بقوم بتطبيقاتها على بيانات التدريب ، عشان اخلی الخوارزم يفهمها ، ويتدرب عليها ، وبالتالي يقدر يعالج زيها
 - ولاحظ حاجة ، ان مش اي نوع تشويه و خلاص اعمله ، لكن اعمل التشويه اللي متوقع حدوثه في البيانات اللي هقيسها ، فلو انا هاقيس اصوات بتتكلم في مصنع ، فمتوقع يكون خلفية الصوت فيه صوت هوا ، او صوت ماكينات ، فمش هاعمل تشويه ان فيه وراهم صوت موسيقي ، لأن فعليا مش هاستخدمها
 - ولو انا عايزة اعمل قراءة حروف متصرورة في حوائط فيها شمس ، فاعمل تشويه لبيانات التدريب ان يكون فيه لسعة اضاءة زيادة ، ومعملتش ان هيكون فيه اضاءة اقل
 - و فكرة اني معملش حاجة ملهاش استخدام ، مش اني اوفر وقت في البرمجة ، لكن لما الخوارزم باديله بيانات محددة و مركزه ، فهبيطلع قيم الثيتات متطابقة بشكل كبير مع المطلوب ، وهيكون الـ variance اقل فاتجنب اي UF متوقع ، بينما لما اديله حاجات ملهاش استخدام عندي ، فلازم كفاءة الاستنتاج ه تكون اقل ، وفي وقت اكتر

- و خد بالك ان قبل ما تعمل اي تكبير للبيانات ، اتأكد او لا ان اصلا البيانات عندك قيمة الانحراف bias فيها قليل ، والا التشويف هيضرها اكتر ، فيا اما تضييف عدد اكتر من الـ features في البيانات ، او تضييف طبقة خفية جديدة في الـ NN ، لغاية لما تتأكد ان قيمة الـ bias قليلة قبل ما تكبر
-

• Ceiling Analysis

- العامل الأهم الذي يجب وضعه دائمًا في الاعتبار هو الوقت ، والذي يقاس بالأيام والأسابيع والأشهر
- فممكن فريق من العاملين في الـ ml يضيع شهورًا عديدة في محاولة الوصول لنتيجة أفضل من النتيجة الحالية ، ويجدوا أنهم لم يتقدمو شيئاً ، وما فعلوه هو اضاعة الوقت ، وبالتالي زيادة التكاليف
- وفي هذا الأمر نقطتين يجب مراعاتها
 - أحياناً نكتشف أنه كان من الأفضل استخدام خوارزم أو آلية ، غير المستخدمة حالياً ، وبعد شهور من العمل ، كان الأفضل أن يتم استخدامها من البداية
 - وفي نفس المعنى ، ربما يتم حالياً استخدام الطريقة المناسبة ، لكن بعوامل مختلفة عن القيم المناسبة (قيمة الفا ، أبعاد الـ NN و هكذا) ، فجipp تعديلها
 - وفي أحيان معينة ، يكون الخوارزم وقيمه بالفعل مناسب ، ولكن قد وصلنا لافتقار قيم متاحة حالياً ، وای خطوات أخرى في عمل المزيد من الـ loops هي ضياع وقت ، وزيادة قليلة جداً في الكفاءة ، في حين الوقت الضائع كان أغلي منها ، فالافتراض أن تتوقف هنا ، ولا نكمل في المزيد من الـ loops
- و المقصود بيها هنا : اي محطة من محطات او مرحلة من المراحل ، المفروض اقضى فيها اكتر وقت ، وایه المرحلة او المحطة اللي مدیهاش وقت اكتر من احتياجها عشان هي بالفعل مش هتعطي كفاءة اكتر من كدة

• فلو عندي الـ pipeline الخاص بالـ OCR



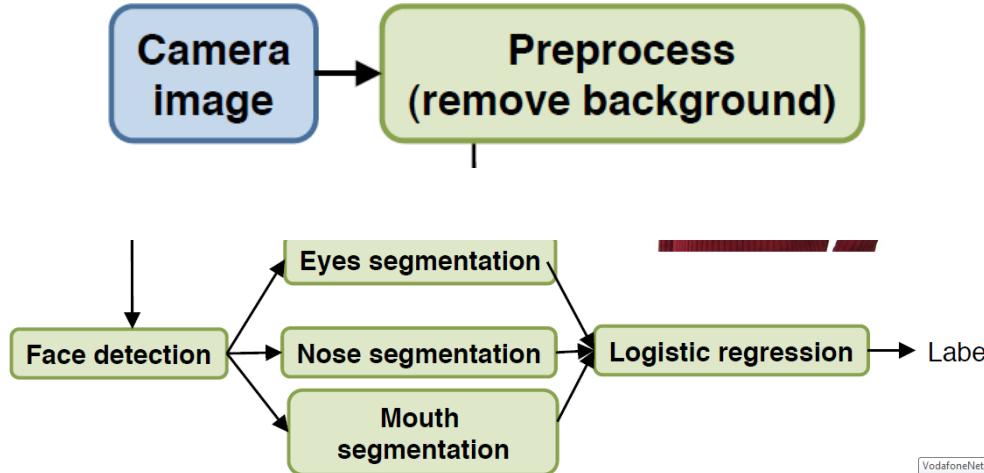
- أول خطوة عندي عشان اعرف انهي اهم محطة ، اني اجيip الكفاءة الكلية للنظام يعني اعرف هو قدرته قد ايه لقراءة الحروف والكلمات ، وهي ببساطة هتعتمد على حاصل قسمة ، قدر يحدد صح كام حرف ، علي مجموع الحروف اللي استلمها
- فلو هو استلم 5000 حرف ، وعرف يجيب منهم 4500 صح ، والباقي غلط ، ببقي كفائه 90 % نفترض ان الكفاءة الكلية 72 %
- امسك اول محطة او اول مهمة عندي (Text Detection) ، واعملها بيدي مش هاخلي الخوارزم يعملها ، وده يخليني متأكد انها ه تكون بكافأة 100 %

- و بعدها هخلي الخوارزم يكمل باقي الخطوات عادي بنفسه , يعني انا بس اللي عملت اول خطوة , وهو اللي هيعمل الباقي
- ده غالبا هيؤدي لزيادة الكفاءة شوية , لأن كان فيه شوية عناصر وقعت مني بسبب قصور الخوارزم في تأدية اول خطوة , ونقول ان الكفاءة بقت 89 %
- بعدها اروح علي تاني خطوة , واعمل فيها نفس اللي عملناه من شوية , هخليها manual بنفسي بحيث تكون بكفاءة 100 % , من غير ما اخلي الخطوة الاولى للخوارزم , يعني ه تكون اول خطوتين يدويا وبكفاءة 100 %
- اقيس ساعتها الكفاءة , ونقول انها بقت 90 %
- وطبعا لو عملت الخطوة الثالثة والاخيرة يدويا , لازم يعملي كفاءة 100 % , ولو عمل اقل من كدة بيقى فيه مشكلة اساسا في البرنامج

- دلوقتي نحل النتائج
- لما تخيلنا ان المحطة الاولى , وصلت لكفاءة 100 % , ساعتها كفاءة النظام كله , نحط من 72 لـ 89 % , يعني هتزود كفاءة النظام بـ 17 %
- لما قلنا ان المحطة الثانية ه تكون 100 % , الكفاءة الكلية هتزيد بـ 1 %
- ولما قلنا ان المحطة الثالثة ه تكون 100 % , الكفاءة الكلية هتزيد 10 %

- ومعنى كدة
- إن اكتر مجهد ممكن ابذله هيكون في المحطة الأول , ثم الثالثة , واتجاهل اني اعمل اي مجهد في الثانية , لأن اي مجهد ووقت يبذل في الاولى او الثالثة معناها زيادة كبيرة في الكفاءة , بينما في الثانية تضيع وقت
- وبالتالي , الفرق بين قيمة الكفاءات , هي اللي هتحددلي انهي محطة ابذل فيها وقت اكتر , وايه اللي اتجاهله ○

- مثل تاني : فحص الوجه
- لو جيت اعمل برنامج لفحص الوجه , ه تكون الخطوات كالتالي :
 - اولا الكاميرا هتاخذ الصورة
 - ثانيا هنشيل منها الخلفية
 - بعدها اعمل فحص للوجه عشان يحدد
 - بعدها هنعمل تلات حاجات مهمة هي :
 - فحص العين
 - فحص الانف
 - فحص الفم
 - لما اجيبي كل حاجة عن الوجه , اعمل classification عشان احدد ده مين



◦ نفرض الكفاءة الكلية 85 %

- ها عمل نفس الموضوع , اني او لا اشيل الخلفية بالفوتوشوب مش هو اللي يعملها , ونشوف الكفاءة بقت كام , قول مثلا بقت 85.1 %
- بعدها اقول للخوارزم علي مكان العينين و اشوف الكفاءة
- اكرر الموضوع مع الانف
- اكرره مع الفم
- اخيرا اعمل انا الـ classification بنفسي , هلاقي ان الكفاءة بقت 100 %

◦ و عن طريق الطرح , اعرف انهي محطة مهم اني اشتغل فيها اكتر

Component	Accuracy
Overall system	85%
Preprocess (remove background)	85.1% → 5.9%
→ Face detection	91% → 4%
→ Eyes segmentation	95% → 1%
Nose segmentation	96% → 1%
Mouth segmentation	97% → 3%
→ Logistic regression	100%

Andrew Ng

◦ و حصل ان مجموعة من المهندسين ضيغو 18 شهر كاملة في عمل خوارزميات لفصل الخلفية , وتم بنجاح ,
بعدها اكتشفوا ان وقتهم ضاع علي زيادة تافهة في الكفاءة