التعلم العميق Deep Learning

- **الدرس الأول : التعلم العميق و الشبكات العصبية**
  - الأسبوع الأول : مقدمة للتعلم العميق
  - الأسبوع الثاني : أساسيات الشبكات العصبية
  - الأسبوع الثالث : الشبكات العصبية المجوفة
  - الاسبوع الرابع : الشبكات العصبية العميقة
- **الدرس الثاني : تطوير الشبكات العميقة : المعاملات العليا**
  - الأسبوع الأول : السمات العملية للتعلم العميق
  - الأسبوع الثاني : الحصول علي القيم المثالية
  - الأسبوع الثالث : ضبط قيم الشبكات العميقة
- **الدرس الثالث : هيكلية مشاريع الـ ML**
  - الأسبوع الأول : استراتيجيات الـ ML - 1
  - الأسبوع الثاني : استراتيجيات الـ ML - 2

- **الدرس الرابع : الشبكات العصبية الملتفة CNN**
  - الأسبوع الأول : أساسيات الشبكات العصبية الملتفة
  - الأسبوع الثاني : حالات عملية من الشبكات العصبية الملتفة
  - الأسبوع الثالث : التعرف علي الأشياء
  - الاسبوع الرابع : التعرف علي الوجه
- **الدرس الخامس : الشبكات العصبية المتكررة RNN**
  - الأسبوع الأول : مفهوم الشبكات العصبية المتكررة
  - الأسبوع الثاني : المعالجة اللغوية الطبيعية NLP
  - الأسبوع الثالث : نماذج التتابع

درس 4: الشبكات العصبية الملتفة CNN

الأسبوع الرابع : التعرف علي الوجه

==========

الهدف من هذا الأسبوع :

- التعرف علي عدد من التطبيقات الخاصة بـ CNN , مثل الاستخدامات الفنية , و التعرف علي الوجه

-

في هذا الاسبوع سنتعرف علي تطبيقين هامين من تطبيقات الـ CNN , وهي التعرف علي الوجه , وثانيا , عدد من الاستخدامات في مجال الفن .

وتكنيك التعرف علي الوجه بلغ شأنا كبيرا , لدرجة تجعله حتي قادر علي التفريق بين التواجد الحقيقي للشخص أم أن احد يستخدم صورته المطبوعة

فهنا يتم فتح البوابة عبر وقوف الشخص امام الكاميرا

بينما هنا تكتشف الكاميرا ان هذه صورة مطبوعة فلا تفتح

وكأن الخوارزم لديه القدرة علي اكتشاف مدي "حيوية" الشخص امامه , وغالبا ما يتم هذا عبر استخدام تكنيك logistic regression عبر اعطائه كمية كبيرة من الصور الحقيقية و الصور المطبوعة ليقوم بالتفريق بينهما .

سنتعرف الان علي الفارق بين مصطلحين هامين في هذا المجال وهما :

التحقق من الوجه face verification
التعرف علي الوجه face recognition

الأمر الأول FV أسهل نوعا , وهو الخاص بالتأكد هل محمد الذي يقف امام الكاميرا , هو نفسه محمد المسجل في قاعدة البيانات , ام غيره .

و هذا النظام يطلق عليه أحيانا 1:1 أي نتحقق من أن هذا الشخص , هو شخص محدد .

الأمر التاني أصعب FR وهو اننا نقوم بادخال صورة شخص , ونريد من الخوارزم البحث في قاعدة بيانات تحتوي علي عدد معين من الأفراد لتحديد من فيهم هذا الشخص .

و هذا الأمر هو المستخدم بشكل مكثف في أغلب التطبيقات , مثل الـ tag لصور الفيسبوك , او بوابو دخول لشركة معينة و هكذا .

و مشكلة هذا النظام أن كفائته تقل رغما عنا , لأنه يقوم بتطبيق الخوارزم الأول FV مع كل الصور الموجودة لديه في قاعدة البيانات .

فلو كان لدينا 500 شخص في الشركة , وكانت كفاءة FV هي 99% (وهو رقم كبير) , فستكون كفاءة الـ FR هي 0.99 أس 500, أي 0.6% وهو رقم صغير جدا .

-

ننتقل لمسمي هام وهو : التعلم بصورة واحدة one shot learning .

وهو يتكلم عن امكانية الخوارزم في تحديد من صوحب الصورة عبر اعطاءه صورة واحدة , وهو الأمر الغير سهل , إذ أن ليس من السهل في الـ DL حساب نتيجة دقيقة من صورة واحدة .

فلو كان لدي في قاعدة البيانات الف موظف , فكل موظف فيهم لديه صورة واحدة , وحينما يأتي شخص ما لدي البوابة , فوجهه له صورة واحدة , وعلي الخوارزم مقارنة هذه الصورة الوحيدة , بالاف الصور الفرادي و اتخاذ قرار من فيهم هذا الشخص ام انه ليس منهم

و الفكرة التقليدية أن الشخص القادم للبوابة , سنأخذ صورته لإدخالها علي CNN وان يكون المخرج softmax مكون من 1001 مخرج , الف لكل موظف , و الاخيرة هي الاختيار انه ليس من العاملين .

و هذه الفكرة غير مناسبة لان دقتها ستكون قليلة جدا لندرة الصور المستخدمة , كما ان في حالة إضافة موظفين جدد في الشركة , سنضطر لتغيير الـ softmax لجعلها برقم اكبر , وهو شئ غير مناسب

و التناول الصحيح هو تفعيل دالة التشابه similarity function , وهي التي تتناول صورتين في كل مرة , و تقوم بحساب كمية الفروق بينهما و يكون لها رمز تاو t

و في حالة كانت الفروق كبير , فالشخصين مختلفين , في حالة كانت الفروق قليلة فهما نفس الشخص .

d(img1 , img2) < t	same person
d(img1 , img2) > t	different person

وبالتالي سنستخدم هذه الدالة , بحيث تتناول الصورة الاولي (الشخص علي البوابة) , وتكون الصورة الثانية الف مرة لالف صورة في قاعدة البيانات , وستعطي كلها ارقام كبيرة , الا شخص محدد سيكون t برقم قليل هو هذا الشخص

و اذا كانت كل ارقام t كبيرة للالف شخص , فهذا معناه ان الشخص علي البوابة ليس من العاملين

و مع إضافة شخص جيد في قاعدة بيانات العاملين , سيكون الأمر سهل جدا في التعامل

-

ماذا عن دالة التشابه التي ستقوم بالمقارنة بين الصورتين ؟

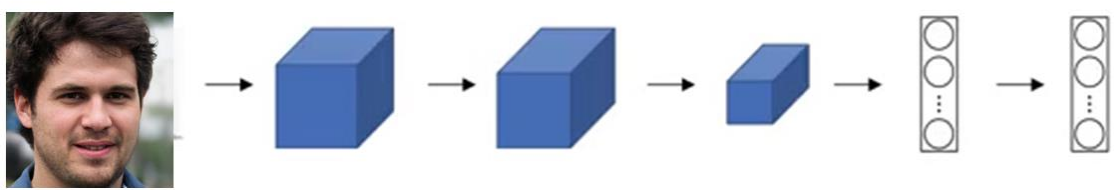
أحد الدوال المستخدمة في هذا الأمر , ما تسمي شبكة سياميس Siamese network

و يبدأ عملها عبر تناول كلا من الصورتين , كل واحدة علي حدة , و إدخالها في شبكة ملتفة عميقة لنخرج بها بالمصفوفة النهائية (التي يتم إدخالها فيما بعد في الـ softmax لكننا سنلغي الـ softmax هنا)

هذه المصفوفة الأخيرة و ليكن 1*128

فهذه الصورة الأولي

وهذه الصورة الثانية :



ثم نتناول المصفوفتين النهائيتين , التي كلا منهما 1*128 , ونقوم بعمل مقارنة بينهم , وأفضل مقارنة هي مقارنة النورم . .

$$\|f(x^{(i)}) - f(x^{(j)})\|^2$$

واذا كان نفس الشهص , فستكون المصفوفتين قريبتين من بعضهما , وبالتالي قيمة الفارق قليلة و بالتالي التاو , والعكس اذا كان الشخص مختلف .

مع التأكيد علي ان هذه الشبكة سيكون لها نفس المعاملات parameters حينما تفحص نفس الصورتين , والا اختل القياس .

و احيانا تسمي هذه الطريقة deep face

-

و من الطرق الجيدة المستخدمة في تفعيل شبكة الـ CNN لمقارنة الوجوه , هو استخدام ما يسمي دالة الخطأ الثلاثي Triplet loss function .

وهي تسمي ثلاثية لأنها تتعامل مع ثلاث صور بالتوازي :

أولا الصورة الـ anchor و هي الصورة الأصل المخزنة في قاعدة البيانات والتي نقارن من خلالها , وسنرمز لها بالرمز A

ثانيا صورة لنفس الشخص (صورة مختلفة لكن نفس الشخص) , و تسمي Positive و يرمز لها P

ثالثا صورة لشخص مختلف , وتسمي Negative و يرمز لها N

و ستكون الدالة بكفاءة عالية , حينما تكون قيمة الـ norm بين الصورة الأصل A و الحقيقي P اقل من الـ norm بين الأصل A و الشخص المختلف N

لذا :

$$d(A,P) < d(A,N)$$

أي أن :

$$\| f(A) - f(P) \|^2 < \| f(A) - f(N) \|^2$$

وبالتالي يمكن نقل الطرف الأيمن الجهة اليسري :

$$|| f(A) - f(P) ||^2 \; -- \; || f(A) - f(N) ||^2 \; < \; 0$$

أي أن الفارق بين الصورة الحقيقية و الشخص السليم , يجب أن يكون أقل من الفارق بين الصورة الحقيقة و الشخص المختلف

و لمزيد من الدقة , نقول ان كون الفارق بينهم اقل من الصفر هو ليس دقيق , فجيب ان يكون الفارق بعيد , فنستبدل قيمة ألفا α مكان الصفر

$$|| f(A) - f(P) ||^2 \; -- \; || f(A) - f(N) ||^2 \; < \; - \alpha$$

أو

$$|| f(A) - f(P) ||^2 \; -- \; || f(A) - f(N) ||^2 \; + \; \alpha < 0$$

و يطلق علي ألفا : الهامش margin و هي مشابهة للقيمة المناظرة لها في الـ SVM

ولمعرفة تأثير ألفا , فلو تعاملنا مع المتباينة بدونها , وكان الفارق بين الصورة الحقيقية و الشخص السليم هو 0.5 , وكان الفارق بين الصورة الحقيقية و الشخص المختلف هي 0.51 , فستكون المتبانية صحيحة , وهذا شئ غير مناسب , لأن الفارق بسيط جدا , وسيؤدي حتما الي إعطاء نتائج غير سليمة في الحياة الواقعية ,

والتي ستكون إما False Positive يعني الدالة توقعت ان هو بالفعل نفس الشخص بينما ليس هو , أو مشكلة False Negative أي أن الدالة توقعت ان هذا ليس الشخص , بينما كان هو نفسه .

لكن مع استخدام الفا وليكن بقيمة 0.3 فهذا معناه ان الفارق بين الصورة الحقيقية و الشخص المختلف لو كان 0.5 فجيب أن يكون الفارق بين الصورة الحقيقية و الشخص السليم لا يزيد عن 0.2 , وهذا يدفع لزيادة دقة الدالة .

و غالبا ما يتم استخدامها في بايثون بهذه الطريقة :

$$L(A,P,N) = \max(\| f(A) - f(P) \|^2 - \| f(A) - f(N) \|^2 + \alpha , 0)$$

و المقصود لها , ان قيمة الخطأ loss و المرمز لها بالرمز L سيكون الأعلي بين رقمين , الرقم الأول هو كل المعادلة السابق ذكرها (بالأحمر) والرقم الثاني هو الصفر .

فإذا كانت المعادلة الحمراء تتوافق مع الشرط بالفعل , وهي أقل من الصفر , فدالة الماكس ستختار الصفر مما يعني ان الخطأ صفر بالفعل .

وإذا كانت المعادلة الحمراء موجبة (غير مناسبة) , فستختارها الماكس , وهذا يعني أن هناك خطأ بقيمة موجبة , يجب أن يقل .

و تكون معادلة الخطأ الكلية J هي :

$$\mathbf{J = \Sigma\ L(A,P,N)}$$

حيث يكون السمشن لكل العناصر الموجودة .

و غالبا ما يتم تدريب الدالة كالتالي , اذا كان لدي الف شخص مثلا في الشركة , فاقوم بتصوير كل شخص 10 صور مختلفة , (باشكال و تنويعات مختلفة لكل واحد) , فيكون لدي 10 الاف صورة .

ثم أقوم بإعطاء هذه الصور كلها للخوارزم , الذي يقوم بمعالجتها , مع إعطاءه قيم y حيث يتعرف علي الصور المطابقة و الغير مطابقة .

بعدها يكون جاهز للتعرف علي اي شخص من الألف , وفي لحظة التشغيل , لن يحتاج لصور عديدة للشخص بل تكفيه صورة واحدة one shot , وهي التي يراها حينما يدخل الشخص من البوابة .

ولاحظ أنه , ثد نقوم بإعطاء الخوارزم اثناء التدريب صور غير متلائمة , مما يؤدي لقلة كفائته .
و هذا يحدث اذا ما كانت صور الـ A قريبة من صور الـ P و بعيدة تماما عن صور الـ N , وقتها سيكون من السهل علي الخوارزم ان يحقق الشرط المذكور , ولن يتعلم بشكل جيد .

بينما لو قمنا بإعطاءه صور الـ A , P بعيدة قليلا عن بعضهم (اي ان الشخص نفسه له صور عديدة غير مطابقة او متشابهة) , وكذلك إعطاءه الشخص المختلف يشبه الشخص الحقيقي الي حد ما , فهذا سيجعل الخوارزم يتعلم ان يفرق بين الاشخاص المتشابهة , وكذلك ان يميز الشخص نفسه حتي لو كان بهيئة مختلفة .

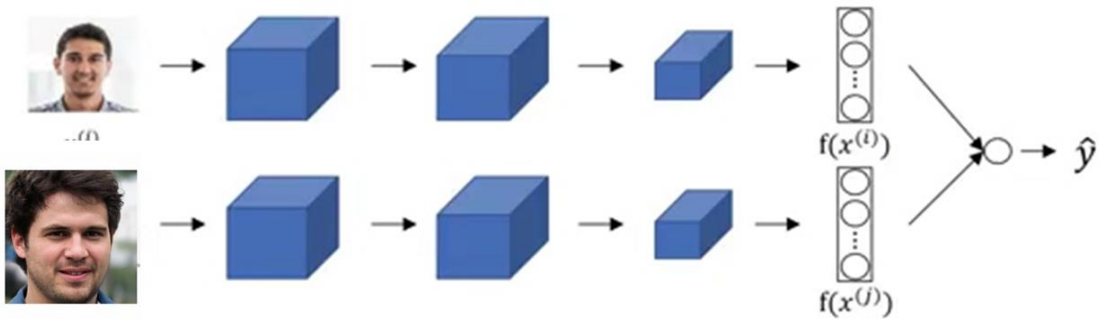
أيضا التدريب لا يتم فقط علي صور الموظفين , بل علي عشرات الملايين من الصور المتنوعة للناس حول العالم , حتي يكون مهئ لكل انواع الصور , ولأن بالطبع صور الموظفين غير كافية للتدريب , و حينما يكون الخوارزم جاهز يتم تغذيته بصور الموظفين و يقوم هو بالتمييز بينهم فورا و تحديد من علي البوابة .

وعدد من الشركات بعدما قامت بتدريب الخوارزم علي عشرات الملايين من الصور , قامت بنشر الباراميترز مجانا , فيمكنك استخدامها بالفعل

-

لكن دالة الخطأ الثلاثي Triplet loss function ليست الوحيدة المستخدمة للتعرف علي الوجوه و التفريق بينها , هناك أيضا دالة التصنيف اثنائي Binary classification

و تقوم فكرتها علي تناول الصورتين المطلوب المقارنة بينهما و عمل CNN لكل واحدة , وفي النهاية امساك المصفوفتين النهائيتين , و عمل تصنيف ثنائي لهما , لتكون قيمة y إما صفر لو كانو مختلفين , او 1 لو كانو متشابهين



وتكون المعادلة الخاصة بالـ logistic regression هي :

$$\hat{y} = \sigma \left(\Sigma \quad Wi \quad | f(xi) - f(xk) | \quad + b \right)$$

وهي ما تعني أننا نقوم بإيجاد سيجمويد لمجموع قيم w مضروبة في الفارق بين قيمة xk , xi (وهي قيم مصفوفة النهاية الـ 128) , بعد عمل قيمة مطلقة لها , ثم جمعها علي b

وأحيانا يتم استبدال الدالة | f(xi) – f(xk) | بدالة اخري تسمي كاي chi و هي :

$$\frac{(f(xi) - f(xk))^2}{f(xi) + f(xk)}$$

مع التأكيد علي أن الشبكة الملتفة CNN التي ستقوم بمعالجة الصورة الأولي , يجب ان تكون بنفس معاملات الشبكة التي ستقوم بمعالجة الثانية

و من الأساليب الذكية المستخدمة , هو أن تقوم بعمل تطبيق للـ CNN لكل صور العاملين عندك مرة واحدة , وتقوم بتخزين المصفوفة النهائية الـ 128 رقم , بحيث في كل مرة , تقوم البوابة فقط بحساب صورة الشخص الذي يقف عندها فقط , ومقارنتها بالارقام السابق حسابها و تخزينها لديك , وهذا يسرع العملية

وبالتالي حينما نقوم بتدريب الخوارزم في هذا النوع من الدوال , نعطيه ازواج من الصور لنفس الشخص (صور مختلفة) , وتكون قيمة y تساوي 1 , ونعطيه ازواج اخري من صور لاشخاص مختلفين , وتكون قيمة y هي 0 , حتي يتدرب الخوارزم عليها و يقوم بحساب w , b

-

من التطبيقات الأخري للـ CNN هو إضافة طابع معين علي صور او علي رسوم , اي دمج ذكي بين صورة , وطابع .

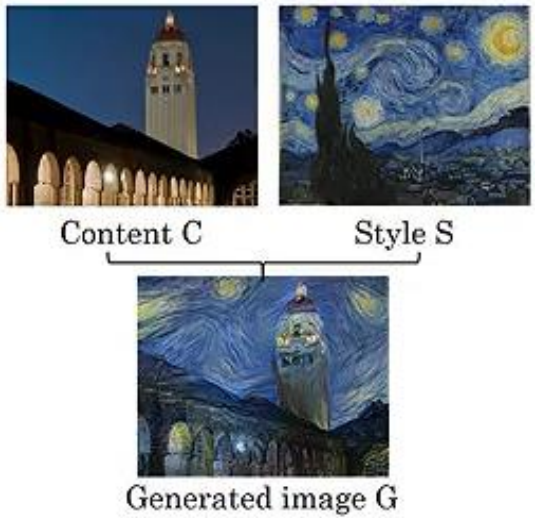
فلو كان لدينا صورة مثل هذه لجامعة ستانفورد :

ونريد إعادة رسمها , بحيث تكون مشابهة لستايل رسمة لفان جوخ هذه :

فتكون الصورة النهائية هكذا :

و يطلق علي الصورة الأصلية c لأنها content و علي الطابع s لأنه style بينما علي الصورة الناتجة g لانها generated

فتكون :



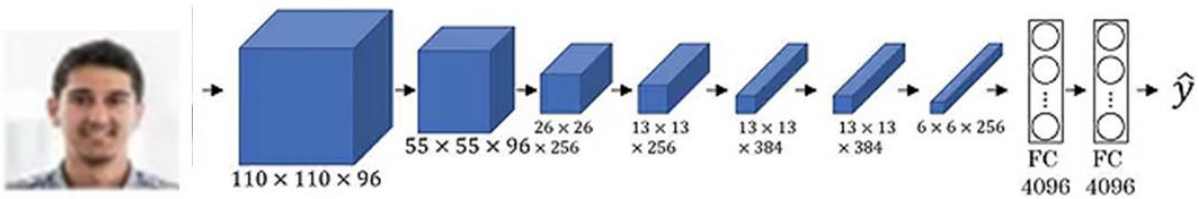
مثال آخر , لصورة كوبري في سان فرانسيسكو , مع طابع للوحة لبيكاسو :

وتسمي هذه التقنية تحويل الطابع الشبكي Neural style transfer

-

و لنتعرف علي كيفية دمج الصورة مع الطابع , علينا ان نتعرف أولا علي الوظيفة التي تقوم بها الطبقات العميقة في الـ CNN لكي نتعلم فن التعامل معها .

إذا تناولنا صورة مثل هذه , وأردنا إدخالها في شبكة CNN عميقة :



فإذا قمنا بتحليل البيانات في الطبقة الأولي فسنجد أنها مصفوفات ارقام , ويمكن ان تتمثل بهذه الصور.

فهذه الوحدة مثلا تتناول الخطوط المائلة \\\\

وهذه الخطوط المائلة العكسية ////

وهذه الخطوط الرأسية : ||||

وهذه الأفقية :

وهكذا في باقي وحدات الطبقة الأولي (لا تنس أن هذه وحدات neurons لشبكة عصبية) :

وكأن الطبقة الأولي تتعامل مع أجزاء صغيرة من الصورة , بينما الطبقات الأكثر عمقا تتناول أجزاء أكبر و أوسع في الصورة

فالطبقة الثانية مثلا ستكون هكذا :

وهو ما نري فيها اشكال اكثر تعقيدا , سواء خطوط مائلة او مستقيمة او دائرية وهي التي تظهر صور أكثر وضوح ولو بمقدار بسيط .

ثم الطبقة الثالثة والتي بدأت تظهر بشكل واضح أجزاء كاملة من وجوه و ايدي و قطع من السيارات :

يليها الطبقة الرابعة :

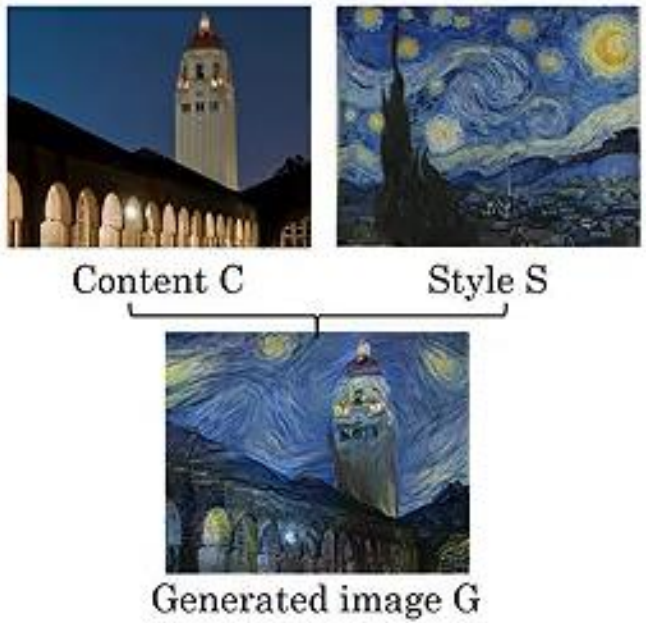
ثم الخامسة :

واللتان تشتملان علي صور أكثر وضوحا و نقاء

-

بعد ما اقتربنا من فهم فكرة دمج الصور مع الطابع , علينا تحديد دالة الخطأ cost function المطلوب من الخوارزم تقليلها , للوصول للدمج السليم .

إذا كان لدينا صورة content مع طابع style و تصنع منتج generated بهذا الشكل :



فيمكن تحديد دالة الخطأ علي أنها مدي اقتراب الصورة الناتجة من كلا من الصورة الاصلية و الطابع .

فإذا كانت الصورة الناتجة قريبة من كلا من الأصل و الطابع معا , فقد تم تنفيذ المهمة بنجاح .

لذا :

$$J(G) = J_{content}(C , G) + J_{style}(S , G)$$

فالدالة الكلية المراد تقليلها , هي مقدار ابتعاد الصورة الناتجة عن كلا من الأصل و الطابع .

إلا أن جمع الخطأين معا دون عمل نسبة محددة بينهم لن يكون دقيق , فيتم إضافة عاملين مضروين فيهما (الفا , بيتا) , حتي يقوم الخوارزم بضبط الكمية المطلوبة من كلا منهما بالقدر المناسب .

$$J(G) = \alpha J_{content}(C, G) + \beta J_{style}(S, G)$$

ماذا عن خطوات التنفيذ ؟

- أول خطوة تكون إضافة الصورة الاصل C و الطابع S للخوارزم , وليكن هكذا :

- ثانيا يتم تحديد معادلة الخطأ cost function , والتي ستكون بالقيمة السالف شرحها
- ثم يقوم الخوارزم بعمل صورة ارقام عشوائية تماما , بنفس المقاس المطلوب , والذي يكون غالبا نفس مقاس الصورة الأصل , فلنقل أنه سيكون 100*100*3 , هكذا

- بعدها يتم استخدام الاشتقاق التدريجي GD لعمل تعديل في قيمة G (مصفوفة الصورة الناتجة) عبر تقليل دالة الخطأ , هكذا :

$$G = G - d/dG\ J(G)$$

- وهي التي تقوم باشتقاق دالة الخطأ , وطرحها من قيمة G لتعديلها و هكذا .
- فنجد أن الصورة ستتحول بالتدريج هكذا :

ثم

ثم

حتي نصل للصورة المثلي .

_

إذن علينا أن نتناول بشئ من التفصيل معادلة الخطأ cost function , لدمج الصور , والتي تشتمل علي جزئين , الأول الخاص بمعادلة خطأ صورة المحتوي content cost function , والتي سنتكلم عنها الآن , والثاني معادلة خطأ الطابع style cost function والتي سنتكلم عنها في الفيديو القادم .

و لفهم مبدأ معادلة الخطأ , علينا أن نتعرف علي تأثير عمق الطبقة في تغير صورة المحتوي (الصورة الأصلية) .

فإذا كنا في الطبقات الأولي , فهذا معناه ان الصورة في هذه الطبقات لازالت قريبة بدرجة ما من الصورة الأصلية , بينما كلما يزداد العمق , كلما ابتعدت الصورة الناتجة عن الأصلية .

و هذا معناه انه ينبغي اختيار طبقة محددة , لا تكون سطحية للغاية , فتكون الصورة الناتجة قريبة من الاصلية تماما , ولا عميقة للغاية فتكون بعيدة جدا , ولكن رقم في الوسط .

و فكرة : معادلة خطأ المحتوي content cost function تقوم علي الفارق بين قيمة الـ activation الخاصة بالصورة الناتجة , و قيمة الـ activation الخاصة بالصورة الأصلية

وبالتالي فإن قيمة الـ activation الخاصة بالصورة الأصلية يمكن أن نطلق عليها :

$$a^{[l](C)}$$

بينما قيمة الـ activation الخاصة بالصورة الناتجة يمكن أن نطلق عليها :

$$a^{[l](G)}$$

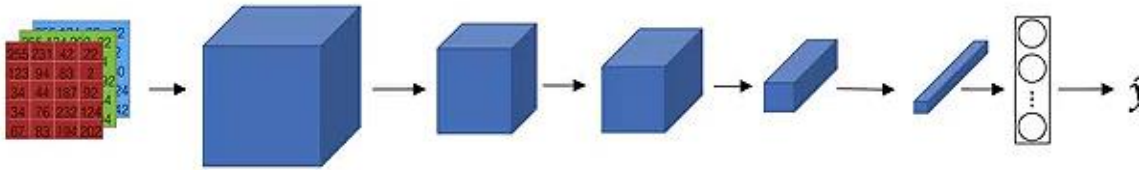
ويكون الهدف تقليل الفارق بينهما , بأسلوب النورم , بهذه المعادلة :

$$J_{content}(C,G) = \frac{1}{2}\| a^{[l](C)} - a^{[l](G)} \|^2$$

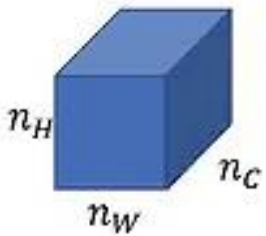
-

لفهم مبدأ معادلة خطأ الطابع style cost function , علينا أن ننظر في هذا المثال

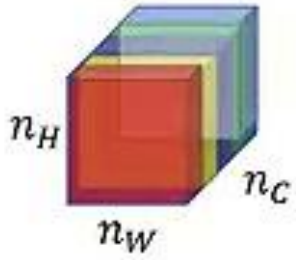
فإذا كان لدينا صورة الطابع تدخل في شبكة CNN هكذا .



و إذا تناولنا الوحدة الرابعة , وليكن هذه :

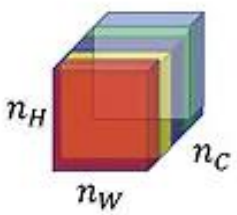


فلأنها ثلاثة أبعاد , فالعمق الخاص بها هو متعلقة بالقنوات channels (لذا يرمز لها nc) , أي أنها هكذا (وليكن 5 قنوات هنا) :

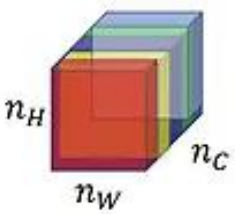


فمعادلة خطأ الطابع style cost function تقوم بمقارنة كل قناة من القنوات الموجودة بين الطابع , وبين الصورة الناتجة هكذا :

Style image



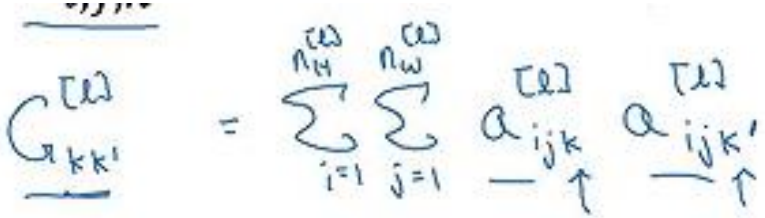
Generated Image



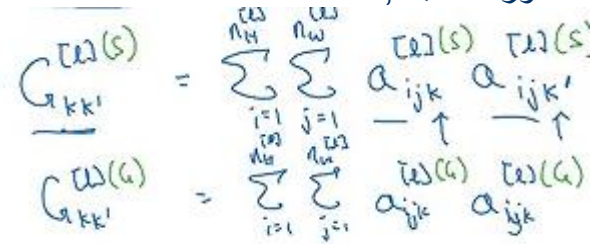
بعدها نقوم بحساب مصفوفة هامة , هي مصفوفة الـ activation والتي تكون معتمدة علي كل قيم الـ activation للمكعب , بدلالة i,j,k بهذا الشكل :

$$\text{Let } a^{[l]}_{i,j,k} = \text{activation at } (i,j,k).\ G^{[l]} \text{ is } n_c^{[l]} \times n_c^{[l]}$$

ومنها : يتم حساب مصفوفة المجموع الكلي , بما يسمي gran matrix



وهذه المصفوفة سيتم حسابها مرتين , مرة للطابع و مرة للصورة الناتجة , هكذا :



و تكون معادلة الخطأ , هي الفارق بينهم , بعد قسمته علي ضعف حاصل ضرب nw*nw*nc , وذلك لعمل تسوية للبيانات normalization

$$J_{style}^{[l]}(S,G) = \frac{1}{\left(2n_H^{[l]}n_W^{[l]}n_C^{[l]}\right)^2}\sum_k\sum_{k'}(G_{kk'}^{[l](S)} - G_{kk'}^{[l](G)})$$

وهذه الـ **J** تكون لطبقة واحدة , فيتكم جمع قيم **J** لكل الطبقات , مع ضربها في قيمة لمدا كمعامل لها

$$J_{style}(S,G) = \sum_l \lambda^{[l]} J_{style}^{[l]}(S,G)$$

فنكون قد حصلنا علي معادلة خطأ الطابع , والتي يتم جمعها مع معادلة خطأ الصورة الأصلية بالصيغة الأصلية هنا :

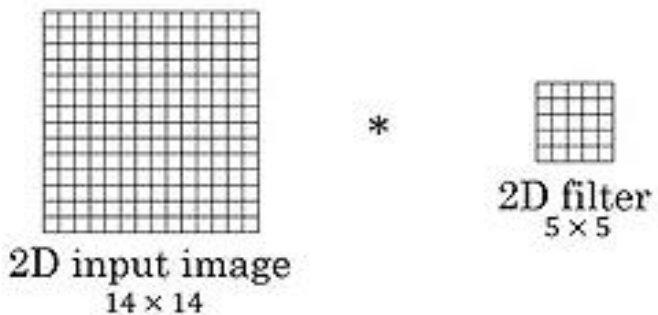
$$J(G) = \alpha\, J_{content}(C,G) + \beta\, J_{style}(S,G)$$

-

و في نهاية الكورس نتناول تطبيق مهم للـ CNN , والتي تختلف نوعا عن التعامل مع الصور , فالتعامل مع الصور و إن كان هو أغلب التطبيقات للـ CNN , إلا أن هناك تطبيقات أخري .

و لا تنس أن الصور تعتبر بعدين 2D لكن هناك تطبيقات أخري من بعد واحد 1D وتطبيقات أخري ثلاثة ابعاد 3D

فلو كان لدي مصفوفة 2D مثل هذه (مثل التي درسناها) , وكانت 14*14 , فيمكن عمل شبكة ملتفة 5*5 , بحيث تنتج لنا 10*10 (وإذا كان هناك عدد من الفلترز في الشبكة الملتفة وليكن 16 فستكون 16*10*10) , ثم نقوم بعمل شبكة ملفتة أخري 5*5 , لتنتج لنا 6*6



بينما لو كانت البيانات بعد واحد 1D مثل رسم القلب والذي يكون بعدها واحدا لأنه دالة مع الوقت , فلو كان لدينا رسم قلب مثل هذا و كان مثلا 14 رقم , فيكون لدينا مصفوفة 14 فقط

1	20	15	3	18	12	4	17

فيمكن عمل شبكة ملفتة من بعد واحد بمقدار 5 مثل هذه :

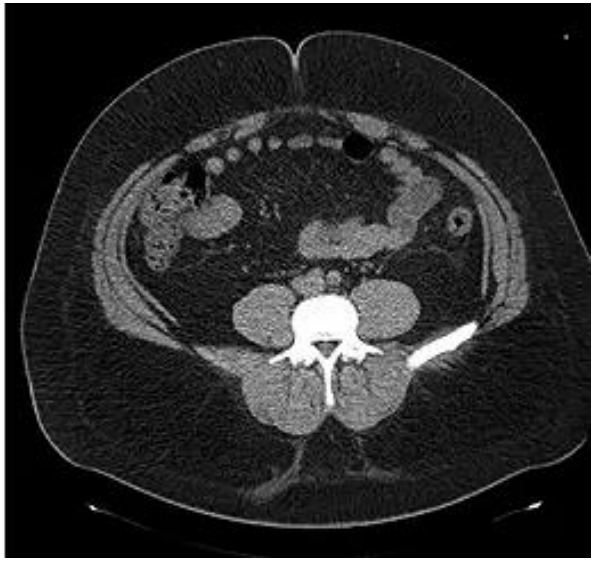
1	3	10	3	1

والتي ستقوم بالتعامل معها , وستنتج مصفوفة من بعد واحد بمقدار 10 , ويمكن كذلك عمل مصفوفة ملتفة من بعد واحد بمقدار 5 لتنتج لنا مصفوفة نهائية بمقدار 6 ارقام

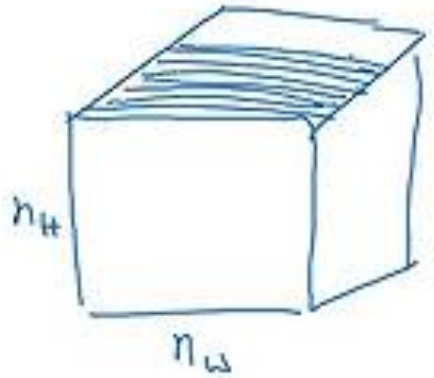
ففكرة التعامل مع البيانات ببعد واحد مشابهة الي حد كبير لبعدين , وغالبا ما يتم استخدام الـ RNN للبيانات ببعد واحد بدلا من الـ CNN

ماذا عن الأبعاد الثلاثة ؟

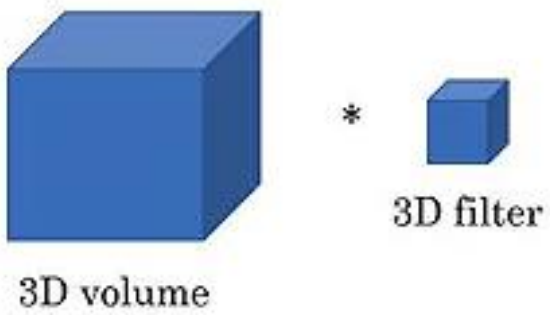
من تطبيقاتها الرسم المقطعي للجسم , حينما يكون هناك عدد من الصور المختلفة للجسم كلما قامت الأشعة بالدخول في العمق و هو ما يسمي CAT Scan



فلأن هناك العديد من الصور , فيتم رصها جميعا وراء بعضها مما يعطي مصفوفة ثلاثية الابعاد :



وقتها تكون البينات ثلاثية الأبعاد , فجيب أن تكون الشبكة الملتفة كذلك :



فلو كان لدينا المصفوفة المجسمة 14*14*14 , فتكون الشبكة الملتفة 5*5*5 , فيكون الناتج 10*10*10 , والتي ندخل لها شبكة ملتفة اخري 5*5*5 , فتنتج لنا 6*6*6

و إذا كان هناك مثلا 16 فلتر في الاخيرة فتكون : 16*6*6*6 , و هكذا

و من التطبيقات أيضا , الافلام و الفيديوهات , والتي تحتوي علي عدد من الصور وراء بعضها البعض

-

نهاية الاسبوع الرابع