

# 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
- الأسبوع 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
- الأسبوع 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 الأول : أساسيات الشبكات العصبية الملتفة CNN



الرؤية بالكمبيوتر computer vision هي من المجالات شديدة الأهمية في الـ DL و التي تتزايد استخدامها يوما بعد يوم , سواء في الموبايل او اللاب توب او الاستخدامات الامنية او المطارات او سيارات ذاتية القيادة او كاميرات المراقبة و هكذا

و هناك عدد من الاسباب التي تدعم تعلم الـ CV بقوة , مثل :

- أولا ان وجود CV يجعل من الممكن صناعة منتجات , لم تكن لتتواجد اصلا بدون هذا الأمر , وهو ما ينبئ بنمو و تسارع صناعتها
- ان الافكار المتولدة من CV تفيد بشكل كبير في باقي مجالات الـ DL نظرها لتنوعها و ابداعها .

وبشكل عام , هناك عدد من الاستخدامات للـ CV :

- تصنيف الصور image classification :

وهي ان يقوم الخوارزم بتحديد ما اذا كانت الصورة تحتوي علي شئ محدد ام لا , فيها قطعة او لا

○ التعرف علي الأشياء : Object detection

وهي التي يقوم فيها الخوارزم , بتحديد اولا اذا ما كان هناك الشئ المحدد او لا , وبعدها يقوم بتحديد مكانه و مسافته من الكاميرا , وعمل مربع حوله , وقد يكون هناك اكثر من شئ في الصورة .

مثلا تقوم الكاميرا بتحديد وجه الشخص , او كاميرات السيارات بالتعرف علي السيارات و المشاة و هكذا

- تحويل الحالة : Neural style transfer

والتي يقوم فيها الخوازم بعمل دمج و تحويل في صورة معينة , بحيث تكون بنفس ستايل صورة اخري , فاذا اعطيناه صورة شخص عادي , وصورة اخري لدافينشي , فيقوم بعمل دمج فيهما بطريقة ما

و من أهم عيوب الـ CV هي كمية البيانات الكبيرة . .  
فلو تعاملنا مع صورة صغيرة (50 بيكسل طول و عرض ) فسيكون عدد البيانات فيها  $50 \times 50 \times 3$  اي 7500 عنصر feature , وهو رقم متوسط لكن الصورة صغيرة جدا

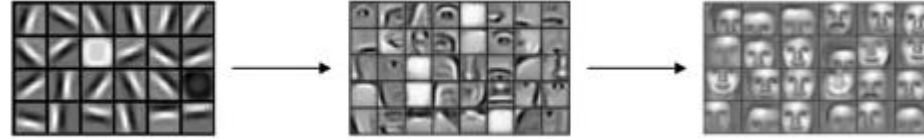
إذا ما تعاملنا مع صورة واضحة (1000 بيكسل طول و عرض ) , فهذا يعني 3 مليون feature , وهو رقم كبير جدا , خاصة ان الشبكة سيكون لها 3 مليون input , وبالتالي الطبقة الخفية الاولى وليكن ستكون الف unit اذن مصفوفة الاوزان للطبقة الخفية الاولى ستكون  $3,000,000 \times 1,000$  اي 3 مليار عنصر , وهو رقم مهول , ويحتاج لعدد كبير جدا من الصور و البيانات لتجنب الـ OF و أيضا يحتاج لامكانيات عالية للكمبيوتر .

وقد قامت الـ CNN بحل جزء من هذه المشاكل

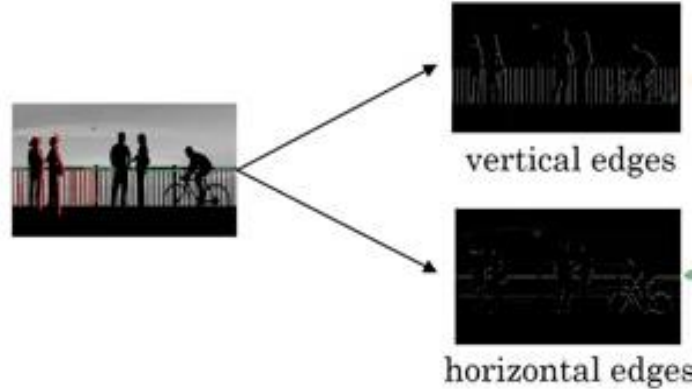
\* \* \* \* \*

واحد مراحل رؤية الكمبيوتر CV ما يسمى : التعرف علي الحواف edge detection

وهي تأتي من نفس فكرة ان رؤية الكمبيوتر , انه يقوم بتقسيم الصورة لاجزاء صغيرة , ثم يقوم بتجميعها لقطع اكبر , ثم التجميع للوجه كاملا .



فلو كان لدينا صورة ما , فيمكن القيام بالتعرف علي خطوط افقية او راسية , حسب المتاح اكثر :



و علينا ان نتعرف اولاً علي الصيغة الرياضية للشبكات الملتفة . .

اذا كان لدينا صورة صغيرة جدا , هي 6 بيكسيل في 6 بيكسل , وهي صورة ابيض و اسود , فتكون المصفوفة الخاصة بها بها 36 رقم فقط , حيث 6 في 6 , وكل خانة بها رقم واحد و ليس 3 ارقام لانها درجات الرمادي , وليكن هذه

3	0	1	2	7	4
1	5	8	9	3	1
2	7	2	5	1	3
0	1	3	1	7	8
4	2	1	6	2	8
2	4	5	2	3	9

فاول خطوة ان نبني مصفوفة صغيرة 3 في 3 , بحيث ان تقوم بعمل الالتفاف لها , وستكون كالتالي :

1	0	-1
1	0	-1
1	0	-1

لابد ان تكون بنفس الارقام . وهي تسمى الـ filter او الـ kernel

عمل الالتفاف يرمز له بالرمز \* , فتكون هكذا :

3	0	1	2	7	4
1	5	8	9	3	1
2	7	2	5	1	3
0	1	3	1	7	8
4	2	1	6	2	8
2	4	5	2	3	9

\*

1	0	-1
1	0	-1
1	0	-1

وقتها الناتج سيكون مصفوفة 4 في 4 , ويتم حسابه عبر بعد طرح المصفوفة الاصلية (6) من المصفوفة الملتفة (3) , ثم إضافة 1 , تكون 4 في 4  
و يكون الالتفاف بهذه الطريقة . .

اولا ان نمسك بالمصفوفة الصغيرة (3\*3) ونجعلها فوق الجزء العلوي الايسر من المصفوفة الاصلية , هنا :

3	0	1	2	7	4
1	5	8	9	3	1
2	7	2	5	1	3

0	1	3	1	7	8
4	2	1	6	2	8
2	4	5	2	3	9

و ان يتم ضرب كل رقم في الرقم الذي وقع فوقه , فيكون :

$$1*3 + 1*2 + 1*2 + 0*0 + 0*5 + 0*7 + -1*1 + -1*8 + -1*2 == -5$$

فنتكون أول قيمة اعلي يسار مصفوفة الناتج (4\*4) تساوي -5

-5			

ثم نقوم بإعادة العملية , ولكن ان تتحرك المصفوفة الملتفة (3\*3) خطوة لليمين , اي هنا :

3	0	1	2	7	4
---	---	---	---	---	---

1	5	8	9	3	1
2	7	2	5	1	3
0	1	3	1	7	8
4	2	1	6	2	8
2	4	5	2	3	9

ونعيد الحساب , والذي سيكون بـ 4-

ونكرر الأمر يمينا و يمينا هكذا

3	0	1	2	7	4
1	5	8	9	3	1
2	7	2	5	1	3
0	1	3	1	7	8
4	2	1	6	2	8
2	4	5	2	3	9

3	0	1	2	7	4
1	5	8	9	3	1



2	7	2	5	1	3
0	1	3	1	7	8
4	2	1	6	2	8
2	4	5	2	3	9

بعدها نقوم بالاتجاه يسارا واكن ان ننزل خطوة لتحت , هنا :

3	0	1	2	7	4
1	5	8	9	3	1
2	7	2	5	1	3
0	1	3	1	7	8
4	2	1	6	2	8
2	4	5	2	3	9

و هكذا نستمر في حساب المربع الناتج (4\*4) , حتي يتحول لهذا :

-5	-4	0	8
-10	-2	2	3
0	-2	-4	-7

-3	-2	-3	-16
----	----	----	-----

وهذه العملية علي الرغم من تعقيدها الرياضي , الا انها تتم بسهولة بعدد من الدوال , مثل

python	:	conv-forward
tensorflow	:	tf.nn.conv2d
keras	:	Conv2D

و لمعرفة كيف يقوم هذا الأمر بالتعرف علي الحواف , دعنا ننظر للمثال التالي :

اذا فرضنا وجود صورة حادة , 6 في 6 , مثل هذه



حينما نقوم بتحويلها لمصفوفة ستكون هكذا :

10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0

وحينما نقوم بعمل الالتفاف لها ستكون هكذا :

10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0

\*

1	0	-1
1	0	-1
1	0	-1


والتي ستكون نتيجتها هكذا :



## واستكمالا لشرح مفهوم الـ edge detection

علمنا ان هذه الارقام , لو تم عمل التفات لها ستكون هكذا :

$$\begin{array}{|c|c|c|c|c|c|c|} \hline 10 & 10 & 10 & 0 & 0 & 0 & \\ \hline 10 & 10 & 10 & 0 & 0 & 0 & \\ \hline 10 & 10 & 10 & 0 & 0 & 0 & \\ \hline 10 & 10 & 10 & 0 & 0 & 0 & \\ \hline 10 & 10 & 10 & 0 & 0 & 0 & \\ \hline 10 & 10 & 10 & 0 & 0 & 0 & \\ \hline \end{array} * \begin{array}{|c|c|c|} \hline 1 & 0 & -1 \\ \hline 1 & 0 & -1 \\ \hline 1 & 0 & -1 \\ \hline \end{array} = \begin{array}{|c|c|c|c|} \hline 0 & 30 & 30 & 0 \\ \hline 0 & 30 & 30 & 0 \\ \hline 0 & 30 & 30 & 0 \\ \hline 0 & 30 & 30 & 0 \\ \hline \end{array}$$



ولكن ماذا عن لو تم عكس الارقام هكذا :

$$\begin{array}{|c|c|c|c|c|c|c|} \hline 0 & 0 & 0 & 10 & 10 & 10 & \\ \hline 0 & 0 & 0 & 10 & 10 & 10 & \\ \hline 0 & 0 & 0 & 10 & 10 & 10 & \\ \hline 0 & 0 & 0 & 10 & 10 & 10 & \\ \hline 0 & 0 & 0 & 10 & 10 & 10 & \\ \hline 0 & 0 & 0 & 10 & 10 & 10 & \\ \hline \end{array} * \begin{array}{|c|c|c|} \hline 1 & 0 & -1 \\ \hline 1 & 0 & -1 \\ \hline 1 & 0 & -1 \\ \hline \end{array}$$

وقتها نتيجة الأرقام ستكون هكذا :

0	-30	-30	0
0	-30	-30	0
0	-30	-30	0
0	-30	-30	0

وهو ما يعني ان تكون الصورة هكذا :



وهو ما يشير إلي أن هناك حافة بين الألوان , لكنها معكوسة إذ أن الابيض علي اليسار و الاسود علي اليمين

كل هذا الأمر عن الحواف الرأسية , فماذا عن الحواف الأفقية ؟ ؟

يتم كشفها عبر استخدام مصفوفة مشابهة , لكن توزيع الارقام يكون افقيا و ليس رأسيا .

ففي حين أن المصفوفة اليسري للكشف الرأسي , فالمصفوفة اليمني للكشف الأفقي .

1	0	-1
1	0	-1
1	0	-1

Vertical

1	1	1
0	0	0
-1	-1	-1

Horizontal

ولرؤية هذا , نري المثال التالي :

10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
0	0	0	10	10	10
0	0	0	10	10	10
0	0	0	10	10	10


\*

1	1	1
0	0	0
-1	-1	-1

إذا قمنا بفحص هذه الصورة , والتي فيها توزيع للابيض و الاسود بهذا الشكل :

10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
0	0	0	10	10	10
0	0	0	10	10	10
0	0	0	10	10	10

\*



ستكون النتيجة هكذا

0	0	0	0
30	10	-10	-30
30	10	-10	-30
0	0	0	0

تلاحظ ان رقم 30 (صف 2 عمود 1) هو ما يشير الي المنطقة التي تنتقل فيها الالوان من الابيض للاسود , وهي من الصفوف 2 الي 4 , ومن الاعمدة 1 الي 3 في المصفوفة الاصلية



10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
0	0	0	10	10	10
0	0	0	10	10	10
0	0	0	10	10	10

اما رقم 30- والذي هو في الصف الثالث العمود الرابع , فيشير للانتقال من الاسود الي الابيض في الصفوف من 2 - 5 , وفي الاعمدة من 3 – 6 وهكذا في باقي الارقام .

لاحظ ان هذا الفلتر , ليس الوحيد المستخدم :

1	0	-1
1	0	-1
1	0	-1

هناك فلتر آخر يسمى sobel filter هكذا :

1	0	-1
2	0	-2

1	0	-1
---	---	----

ويختلف أنه تم استبدال الصف الثاني ب 2 مكان 1 , وذلك لعمل ثقل أكبر في المنتصف

كما يمكن تغيير الارقام بحيث يكون هناك ثقل أكبر في المنتصف , وابتعاد عن ارقام الصفر هكذا :

3	0	-3
10	0	-10
3	0	-3

وهو ما يسمى sharr filter

و الشيء الحديث , انه احيانا يتم عمل مصفوفة الفلتر من ارقام غير معروفة اشبه بالاوزان , وان ندع الخوارزم نفسه هو ما يحددها , اي ان تكون هكذا :

W1	W2	W3
W4	W5	W6
W7	W8	W9

\* \* \* \* \*

نتناول الآن عملية هامة , تستخدم كثير في فحص الصور, وهو ما يسمى : الحشو Padding

ولكي نفهمها , علينا اولا أن نتناول عيين يحدثان في عملية الالتفاف كما شرحنا سابقا . .

قلنا ان الصورة لو كان لها بعد معين  $6*6$  مثلا , وكانت مصفوفة الالتفاف  $3*3$  وقتها سيكون الناتج  $4*4$  , ذلك لان الناتج يكون

$$n-f+1$$

واذا تكررت العملية سيتم تصغير الصورة اكثر و اكثر , حتي تختفي , وقتها لن نمكن من تنفيذ العملية . .

هذا هو العيب الأول

الثاني هو ان عملية الالتفاف , تجعل الارقام التي علي الاطراف تستخدم قليلا .

فالارقام الحمراء تستخدم مرة واحدة فقط , والخضراء مرتين , والصفراء ثلاث مرات , بينما الارقام الداخلية يتم استخدامها اكثر , لمرور المصفوفة الملتفة عليها مرات عديدة

3	0	1	2	7	4
1	5	8	9	3	1
2	7	2	5	1	3
0	1	3	1	7	8
4	2	1	6	2	8
2	4	5	2	3	9

ولحل المشكلتين معا , نقوم بعمل الحشو Padding

ويقصد به ان يتم عمل إحاطة للمصفوفة 6\*6 بسطر كامل من الارقام , بحيث هذا يزيد الصفوف و الاعمدة كلا منهما بمقدار 2 , بحيث تكون هكذا

	3	0	1	2	7	4	
	1	5	8	9	3	1	
	2	7	2	5	1	3	
	0	1	3	1	7	8	
	4	2	1	6	2	8	
	2	4	5	2	3	9	

وغالبا ما يتم وضع ارقام 0 في الخانات الفارغة , وهذا سيحل المشكلتين معا .

فمن ناحية , صارت المصفوفة  $8*8$  , فحينما يتم حساب المصفوفة الناتجة ستكون  $6*6$  , وهذا معناه ان الحجم لم يتغير . لان المصفوفة الجديدة  $6*6$  , سيتم عمل حشو لها قبل التفافها مرة اخري .

و من ناحية فإن الارقام التي كانت علي الاطراف صارت داخلية و هو ما معناه انها ستقرأ مرات عديدة .

وهنا في هذه الحالة نقول ان مقدار الحشو يساوي 1 (سطر واحد تم اضافته حول المصفوفة)

$$\text{padding} = p = 1$$

ومن الممكن عمل حشو اكبر , بحيث تكون  $p = 2$  وهو ما معناه ان الصفوف و الأعمدة ستزداد بمقدار 4 كلا منهما

ومن هنا نصل أن هناك نوعين هامين للتفاف . .

النوع الأول : الالتفاف المسموح Valid convolutions :

وهو ألا نقوم باي عملية للحشو , وبالتالي , اذا كانت ابعاد المصفوفة الاصلية  $n*n$  و الفلتر  $f*f$  يكون الناتج :

$$n*n * f*f ===== n-f+1 * n-f+1$$

النوع الثاني , وهي : الالتفاف المشابه Same convolutions :

و هي التي تظل الصورة بعد الالتفاف , مشابهة في الابعاد للصورة قبل الالتفاف , وبالتالي لابد من استخدام الحشو .

اي أن ابعاد المصفوفة ستكون  $n+2p*n+2p$  و الفلتر سيكون  $f*f$  , فتكون :

$$n+2p * n+2p * f*f ===== n+2p-f+1 * n+2p-f+1$$

ولأن ابعاد المصفوفة النهائية سيكون مطابق لاصلية قبل عمل الحشو :

$$\begin{aligned} n+2p-f+1 &= n \\ 2p-f+1 &= 0 \\ p &= \frac{1}{2}(f-1) \end{aligned}$$

لذا حينما كان الفلتر يساوي 3 , كان المطلوب من الحشو ان يكون 1

واذا كان الفلتر يساوي 5 , سيكون الحشو 2

و غالبا ما يكون الفلتر رقم فردي , نادرا ما يكون زوجي .

\* \* \* \* \*

فكرة اخري تستخدم و هي : الالتفاف بالخطوة Strided convolution

و المقصود بها , ان الفلتر , بدلا من أن يقوم بعمل مسح كامل للمصفوفة الاصلية و يتحرك افقيا و رأسيا خطوة خطوة , فمن الممكن ان يتحرك خطوتين بدلا من خطوة واحدة .

فإذا كان لدينا مصفوفة  $7 \times 7$  هكذا :

2	5	9	9	6	5	1
5	4	2	5	8	9	6
4	5	8	9	6	3	2
5	8	9	6	3	2	5
0	4	2	3	5	2	4
5	8	9	6	6	2	5
2	5	8	9	6	6	2

و الفلتر هو  $3 \times 3$  .

فبدلا من ان يقوم الفلتر بالتحرك بشكل بطيء عليها , تكون اول خطوة له :



2	5	9	9	6	5	1
5	4	2	5	8	9	6
4	5	8	9	6	3	2
5	8	9	6	3	2	5
0	4	2	3	5	2	4
5	8	9	6	6	2	5
2	5	8	9	6	6	2

2	5	9	9	6	5	1
5	4	2	5	8	9	6
4	5	8	9	6	3	2
5	8	9	6	3	2	5
0	4	2	3	5	2	4
5	8	9	6	6	2	5
2	5	8	9	6	6	2

2	5	9	9	6	5	1
5	4	2	5	8	9	6
4	5	8	9	6	3	2
5	8	9	6	3	2	5
0	4	2	3	5	2	4
5	8	9	6	6	2	5
2	5	8	9	6	6	2

وحتي رأسيا يكون خطوتين

2	5	9	9	6	5	1
5	4	2	5	8	9	6
4	5	8	9	6	3	2
5	8	9	6	3	2	5
0	4	2	3	5	2	4
5	8	9	6	6	2	5
2	5	8	9	6	6	2

و هكذا , وهذا سينتج مصفوفة الناتج  $3 \times 3$  فقط , ولن تكون  $5 \times 5$  كما المتوقع .

و يمكن عمل هذه الفكرة مع الحشو في نفس الوقت , ويكون القانون العام لهم , في حالة المصفوفة هي  $n \times n$  الفلتر  $f \times f$  و الحشو  $p$  و الخطوة  $s$

$$[(n+2p-f)/s] + 1 * [(n+2p-f)/s] + 1$$

$$(7+0-3/2) + 1 = 3$$

ففي هذه الحالة ,  $s=2$  و  $p=0$  و  $n=7$  و  $f=3$  تكون القيمة

حسنا , ماذا في حالة كان هذا الرقم ليس صحيح , وكان كسر عشري . .

هذا معناه ان الفلتر لم يتمكن من الوصول للنهاية بشكل سليم مع المشي بهذه الخطوة .

فإذا كانت المصفوفة  $8*8$  مثلا :

2	5	9	9	6	5	1	5
5	4	2	5	8	9	6	6
4	5	8	9	6	3	2	9
5	8	9	6	3	2	5	3
0	4	2	3	5	2	4	2
5	8	9	6	6	2	5	5
2	5	8	9	6	6	2	3
8	5	1	2	3	6	5	8

فإن الخطوة الأولى ستكون :

2	5	9	9	6	5	1	5
5	4	2	5	8	9	6	6
4	5	8	9	6	3	2	9
5	8	9	6	3	2	5	3
0	4	2	3	5	2	4	2
5	8	9	6	6	2	5	5
2	5	8	9	6	6	2	3
8	5	1	2	3	6	5	8

الثانية :

2	5	9	9	6	5	1	5
5	4	2	5	8	9	6	6
4	5	8	9	6	3	2	9
5	8	9	6	3	2	5	3
0	4	2	3	5	2	4	2
5	8	9	6	6	2	5	5
2	5	8	9	6	6	2	3
8	5	1	2	3	6	5	8

الثالثة :

2	5	9	9	6	5	1	5
5	4	2	5	8	9	6	6
4	5	8	9	6	3	2	9
5	8	9	6	3	2	5	3
0	4	2	3	5	2	4	2
5	8	9	6	6	2	5	5
2	5	8	9	6	6	2	3
8	5	1	2	3	6	5	8

الخطوة الرابعة لا يمكن ان تتم , لانها يتخرج خارج المصفوفة , وبالتالي سيكون عدد من عناصرها صفر و هذا غير صحيح , فيتم الغاء هذه الخطوة

2	5	9	9	6	5	1	5
5	4	2	5	8	9	6	6
4	5	8	9	6	3	2	9
5	8	9	6	3	2	5	3
0	4	2	3	5	2	4	2
5	8	9	6	6	2	5	5
2	5	8	9	6	6	2	3
8	5	1	2	3	6	5	8

وهذا معناه انه في حالة كان الخطوات غير كافية , يتم عمل تقريب للأسفل دائما floor round للرقم , وهنا ستظل مصفوفة الناتج  $3 \times 3$  , مع ان حاصل العملية الرياضية هو : 3.5

و تكون الصيغة النهائية كالتالي :

$$n \times n \text{ image} \quad f \times f \text{ filter}$$
padding  $p$       stride  $s$ 

$$\left\lfloor \frac{n+2p-f}{s} + 1 \right\rfloor \quad \times \quad \left\lfloor \frac{n+2p-f}{s} + 1 \right\rfloor$$

ولا تنس أن هذه العلامة تشير إلى التقريب لأسفل floor round

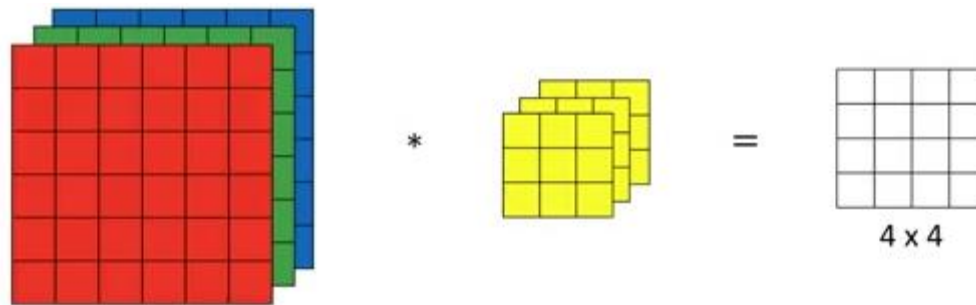
\* \* \* \* \*

كل ما سبق كان عمل الالتفاف من بعدين , بحيث تكون المصفوفة لها بعدين و كذلك الفلتر , فماذا عن الابعاد الثلاثة , والتي تتحقق حينما تكون الصورة بألوان ثلاثة , و بالتالي هناك بعد ثالث هو الـ channel , وهي الالوان الثلاثة RGB

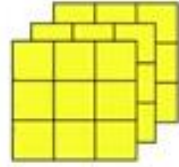
وقتها تكون الصورة لها ثلاث ابعاد ,  $6*6*3$  الأولي height الثانية width والثالثة channels

كذلك لابد أن تكون مصفوفة الفلتر لها نفس العمق (عدد الـ channels) حتي تتمكن من الالتفاف حولها , بهذا الشكل .

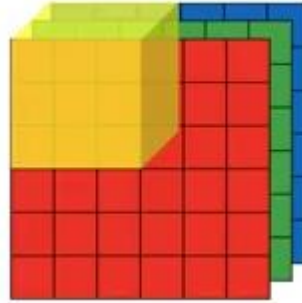
والناتج لن يكون بأبعاد ثلاثة , لكن مصفوفة ببعدين فقط و ستكون  $4*4$



لاحظ أن الفلتر هنا , ممكن ان يشار له بشكل المكعب :



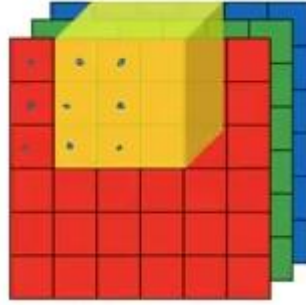
و لعمل التفاف لها , سيقوم الفلتر ثلاثي الابعاد , بعمل مسح مستمر , كما كان يقوم به في ثنائي الابعاد , بأن يذهب في اول مكان له فوق علي اليسار , بهذا الشكل :



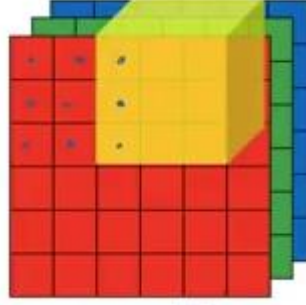
وقتها سيكون مكمل الفلتر عبارة عن 27 رقم , يتم ضرب كل رقم فيها , في الرقم المناظر له , وجمع الارقام معا , كما كنا نفعل في ثنائي الابعاد , ويكون هذا هو اول رقم فوق علي اليسار من مصفوفة الناتج (4\*4)

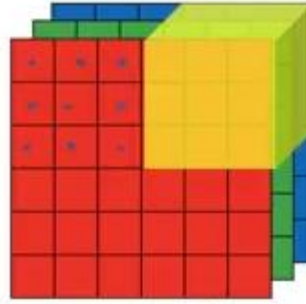
ثم يتحرك الفلتر خطوة اخري لليسار .



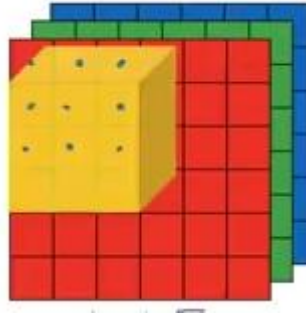


و يتم ضرب الارقام بنفس الطريقة , وملء مصفوفة الناتج , وهكذا

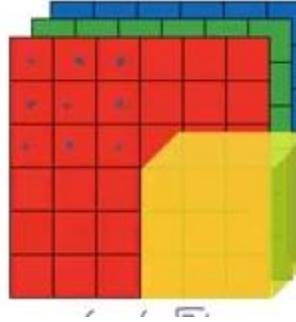




ثم ننزل لسطر جديد



بحيث يكون اخر رقم فيها :



اما بالنسبة لقيم مصفوفات الفلتر , فهي تكون مثل القيم العادية , فكل channel فيها تكون :

1	0	-1
1	0	-1
1	0	-1

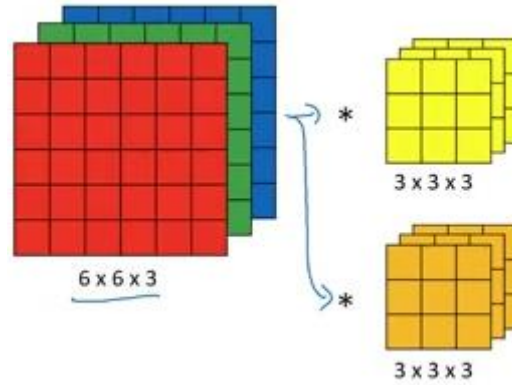
واحيانا نحتاج ان نقوم بعمل فلتر للون محدد فقط , وليكن الأحمر , وقتها نقوم بجعل الطبقة الاولى للفلتر بنفس الارقام بينما طبقتي GB تكون كلها اصفار , وهكذا

وهذه الطريقة هي لعمل فحص الحواف الرأسية vertical edge detection , وفي حالة اردنا ان نقوم بعمل فحص افقي او بزاوية معينة في نفس الوقت , فيمكن عمل اكثر من فلتر هكذا :

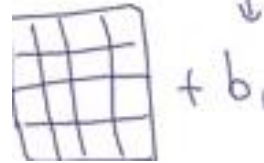


نبدأ الآن بعمل تطبيق حقيقي لطبقة واحدة من شبكة ملتفة CNN

يبدأ الأمر بالتعامل مع مصفوفة الأصل :  $6 \times 6 \times 3$  وعمل عدد 2 فلتر كلا منها  $3 \times 3 \times 3$  بحيث ينتج عدد اثنين مصفوفة ناتج كلها منها  $4 \times 4$  واحدة للفحص الافقي و الثانية للرأسي .



إلا أن مصفوفتي الناتج سنقوم فيها بعملية رياضية مبسطة , أولاً انه سيتم جمع رقم  $b$  وهو معامل الخطأ bias عليها هكذا :



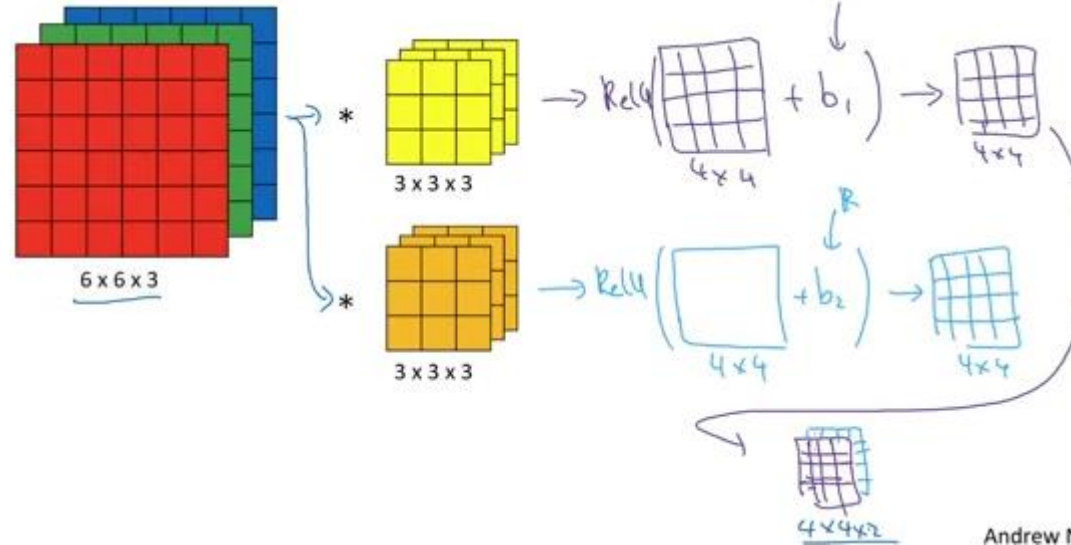
وحيثما يتم جمع رقم علي مصفوفة , فنحن نعلم من مبدأ الـ **broadcasting** في بايثون انه سيقوم بعمل "فرد" للرقم بحيث يكون مصفوفة بنفس البعد , وكاننا سنقوم بجمع نفس رقم الـ **bias** علي كل عنصر من عناصر المصفوفة .

بعدها نقوم بعمل دالة **ReLU** عليها (وهي التي تجعل القيمة الموجبة دالة خطية , والسالبة خطية سالبة تقترب من الصفر) , هكذا

$$\text{ReLU} \left( \begin{array}{|c|c|c|c|} \hline & & & \\ \hline & & & \\ \hline & & & \\ \hline & & & \\ \hline \end{array} + b_1 \right) \rightarrow \begin{array}{|c|c|c|c|} \hline & & & \\ \hline & & & \\ \hline & & & \\ \hline & & & \\ \hline \end{array}$$

$4 \times 4$        $4 \times 4$

و يتم تكرار هذا في المصفوفة الثانية (من الفلتر الثاني) , ومن ثم , ضم الناتجين معا , ليكونا طبقتين وراء بعضهما , فتكون الصورة النهائية هكذا :



Andrew Ni

و هنا لابد من ربط كل ما نقوم به الآن , مع الفكرة الاساسية للشبكات العصبية NN

لا نريد ان ننسى , ان المساق الطبيعي للشبكات العصبية , هو اننا نتناول المدخل  $x$  و الذي يمكن ان نسميه  $a_0$  ثم نقوم بضربه في  $w_1$  ثم جمعه علي  $b_1$  ليكون هو قيمة  $z_1$

ثم نقوم بإدخال قيمة الـ  $z_1$  في دالة التفعيل (والتي كانت سيجمويد او تانش) لنحصل علي قيمة  $a_1$  والتي نكرر بها نفس الفكرة في الطبقات التالية في الشبكة العصبية .

هنا الفكرة نفسها , الصورة الاصلية هي مكان الـ  $x$  و بالتالي المصفوفة ثلاثية الابعاد هنا  $6*6*3$  هي قيم الإكس .

و الفلتر (سواء واحد او اكثر ) هي مكان الـ  $w$  لأنه يتم ضربها في قيم الإكسات (لا تنس ان الفلترز يتم ضربها بالفعل في المصفوفة الاصلية)

اذن حاصل ضرب المصفوفة الاصلية في الفلترز كانه  $w_1a_0$

وبعدها نقوم بجمع  $b_1$  علي المصفوفة المضروبة , لتكون  $w_1a_0 + b_1$  والتي نعتبر  $z_1$

بعدها بدلا من دالة السيجمويد , سنقوم بادخالها في دالة ReLU فيكون الناتج منها هو  $a_1$

فعملية الشبكة الملتفة , هي بالكامل مشابه لعملية الاوزان في الـ NN

و اذا اردنا التكلم عن عدد المعاملات parameters وهي الازان w+b

فإذا كان لدينا مثلا 10 فلترز , كل فلتز هو  $3*3*3$  , وكأن كل فلتز يقيس الالوان الثلاثة , و لأنهم عشرة فهي سيقس الافقي و الراسي و بزوايا عديدة .

فكل فلتز فيهم سيكون له 27 رقم لـ w مضاف إليه قيمة واحدة للـ b يكون 28 قيمة , واذا كانو 10 فلترز , يكون المجموع 280 معامل .

و الميزة ان هذا الرقم قليل نوعا , لأن الصورة مثلا اذا كانت  $1000*1000$  , فيكون بها 3 مليون رقم , وهو الذي يعني احتياجنا لـ 3 مليون معامل في الحالة العادية , و هذا الـ 3 مليون لوحدة واحدة في الطبقة الخفية الاولى , فاذا كان لدينا مثلا 100 وحدة كحد ادني , يكون لدينا 300 مليون معامل وهو رقم مخيف .

بينما تكنيك الـ CNN يجعلنا نستخدم فقط 280 معامل يسهل الحصول علي قيم دقيقة لهم , في وقت بسيط .

وبالتالي علينا ان نقوم بعمل سرد كامل للرموز المستخدمة :

أولا هناك قيمة f الخاصة بحجم الفلتز

$$f^{[l]} = \text{filter size}$$

ثم قيمة p الخاصة بحجم الحشو padding

$$p^{[l]} = \text{padding}$$

ثم قيمة s الخاصة بعدد الخطوات



$$s^{[l]} = \text{stride}$$

أما مصفوفة الـ input فستكون :

$$\text{Input: } n_H^{[l-1]} \times n_W^{[l-1]} \times n_C^{[l-1]} \leftarrow$$

وهي التي تشير الي قيمة nh اي عدد الصفوف وهي قيمة الارتفاع , ثم nw عدد الاعمدة , قيمة العرض , و nc عدد الـ channels ولا تنس ان كل هذه القيم لـ L-1 وهي للطبقة السابقة

اما مصفوفة الـ output فستكون مشابهة , لكن هي لقيم L

$$\text{Output: } n_H^{[l]} \times n_W^{[l]} \times n_C^{[l]}$$

والعلاقة بينهما بالمعادلة :

$$n_H^{[l]} = \left\lfloor \frac{n_H^{[l-1]} + 2\rho^{[l]} - f^{[l]}}{s^{[l]}} + 1 \right\rfloor$$

حيث ان هذا الأمر ينطبق علي العرض او الطول .

ولا تنس أن nc المستخدمة في مصفوفة الـ output هي نفسها عدد الفلترز المستخدمة من العملية في الطبقة السابقة لها

ولا تنس أن أي فاطر ستكون ابعاده :

$$f(L) * f(L) * nc(L-1)$$

حيث ان عدد طبقات اي فلتر لا بد و ان يتساوي اساسا مع عدد طبقات الـ output من الطبقة السابقة

و بالتالي يكون الـ output بالابعاد :

$$a(L) = nh(L) * nw(L) * nc(L)$$

وإذا كان هذا الأمر لعدد من الصور في العينة  $m$

$$A(L) = m * nh(L) * nw(L) * nc(L)$$

ماذا عن الأوزان  $w$  , وهي نوعين  $w$  ,  $b$

نبدأ بالـ  $w$  وهي ستكون بنفس الابعاد الخاصة بالفلتر  $((f(L) * f(L) * nc(L-1))$  لكن مضروبة في عدد الفلترز المستخدمة و هي  $nc(L)$

$$w = f(L) * f(L) * nc(L-1) * nc(L)$$


أما قيمة  $b$  فهي فقط عدد الفلترز الحالي :

$$b = nc(L)$$

\* \* \* \* \*

فأخذ مثال كامل لنفسه الصورة بشكل افضل

نفرض أن هناك صورة لقطة , نريد بناء CNN لتقوم بعمل التصنيف , و ان هذه الصورة ابعادها  $39 \times 39 \times 3$


$$\begin{array}{r} \times \\ 39 \times 39 \times 3 \\ \hline n_h = n_w = 39 \\ n_c = 3 \end{array}$$

وقتها نقول ان كلاه من  $nh(0)$  و  $nw(0)$  تساوي 39 , ان  $nc(0)$  هي 3 لانها ثلاث اللوان

سنختار مصفوفة فلتر بحيث تكون 3 في 3 , وبها 10 فلتر , و الخطوة 1 , ولا خطوات

$$\begin{aligned} p^{(0)} &= 3 \\ s^{(0)} &= 1 \\ p^{(0)} &= 0 \\ 10 \text{ filters} \end{aligned}$$

وقتها يكون الناتج هكذا :



$$37 \times 37 \times 10$$

$$n_h^{(0)} = n_w^{(0)} = 37$$

$$n_c^{(0)} = 10$$

سيكون  $37 \times 37 \times 10$  , حيث قيمة 37 , هي  $39 - 3 - 1$  , والـ 10 هي عدد فلاتر الفلتر السابق .

لا تنس ان 37 هي  $nh(1)$  و  $nw(1)$  , بينما  $nc(1)$  تساوي 10 , ايضا ان هذه القيمة تسمى  $a_0$

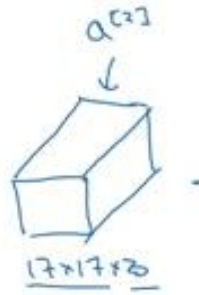
نقوم باختيار فلتر تالي ,  $5 \times 5$  , وخطوة 2 , وعدد الفلاتر 20

$$\begin{aligned}
 f^{01} &= 5 \\
 s^{01} &= 2 \\
 p^{01} &= 0 \\
 \text{20 filters}
 \end{aligned}$$

بتطبيق القانون :

$$\frac{n+2p-f}{s} + 1$$

تكون ابعاد الصورة المصفوفة الجديدة : 17\*17\*20



$$n_h^{(2)} = n_w^{(2)} = 17$$

$$n_c^{(2)} = 20$$

الـ 17 هي  $nh(2)$  و  $nw(2)$  , والـ 20 لانها عدد الفلاتر السابقة والتي ستكون هنا  $nc(2)$  , وان هذه المصفوفة هي  $a_2$

نستخدم فلتر جديد  $5*5*20$  , وبخطوة 2 , وعددهم 40 فلتر

$$f^{(3)} = 5$$

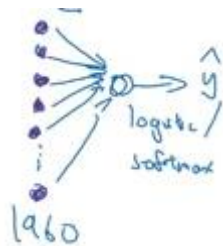
$$s^{(3)} = 2$$

$$40 \text{ filters}$$

يكون الناتج  $7 \times 7 \times 40$

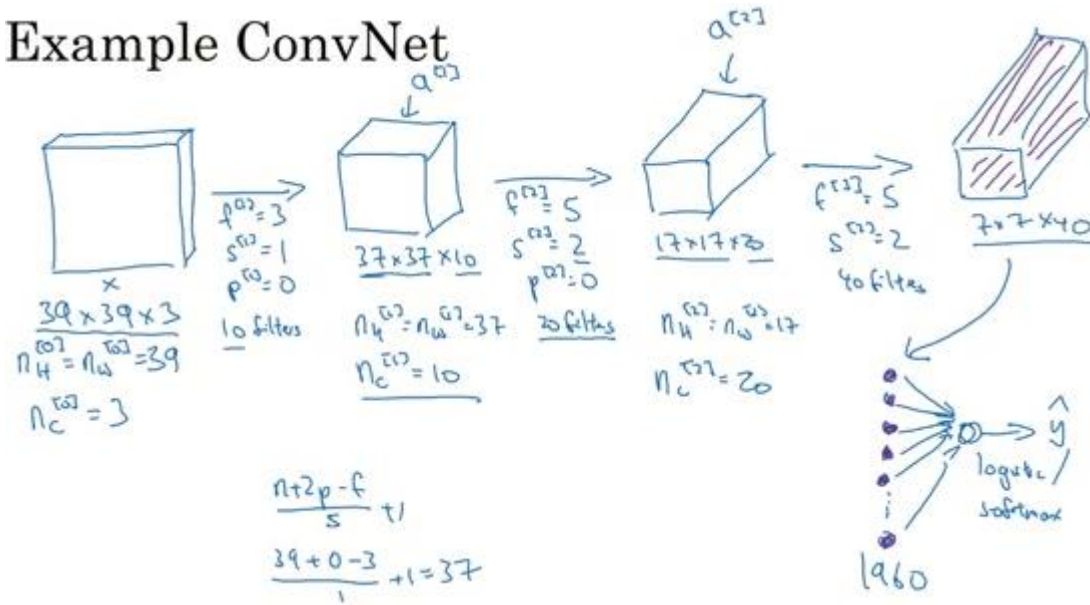


الان صارت المصفوفة معقولة , وعدد الارقام فيها ( $7 \times 7 \times 40$ ) سيكون 1960 رقم , وهي التي يمكن ان ندخلها في دالة logistic regression بسهولة , للحصول علي  $\hat{y}$



وتكون الصورة الكاملة . .

## Example ConvNet



لا تنس ان كمية كبيرة من الوقت المستخدم في بناء الشبكة , يكون في تحديد المعاملات العليا الخاصة بالـ CNN مثل :

- الحجم النهائي للشبكة
- الخطوة stride
- الحشو padding
- عدد الفلاتر و ابعادها



وستنعم لاحقاً كيفية تحديد هذه التفاصيل , لكن ستجد بشكل عام ان الشبكة كلما تعمقنا فيها اكثر , كلما قل الطول و العرض , و كلما زاد العمق

وكل هذا كان شرح ما يسمى الطبقات الملتفة convolutional layers

لكن هناك نوعين آخرين , احدها يسمى Pooling layer و الاخر Fully connect layer , وهما اقل استخداما و اكثر بساطة من الـ conv لكن احيانا يتم الاستعانة بهم , وهو ما سنشرحه في الفيديوها القادمة .

\* \* \* \* \*

نتكلم عن الطبقات من نوع Pooling layer .

و كما ذكرنا فهي اقل في الاستخدام , لكنها اسهل و ابسط . وتقوم بعمل اختصار للبيانات , وتسريع العملية .

و اذا اخترنا منها نوع Max pooling , تكون فكرته كالتالي :

اذا كان لدينا مصفوفة 4 في 4 هكذا :

1	3	2	1
2	9	1	1
1	3	2	3
5	6	1	2

فبالسبب الى max pooling يمكن اختصارها لمصفوفة اثنين في اثنين , عبر تقسيم المصفوفة لاربعة مربعات , و اختيار الرقم الاعلى في كل مربع كالتالي :

1	3	2	1
2	9	1	1
1	3	2	3
5	6	1	2

فتكون مصفوفة الناتج :

9	2
6	3

بحيث ان كل ربع من المصفوفة تم اختيار الرقم الاكبر منه ليتم وضعه في مصفوفة الناتج .

و هذا يشابه لو اننا قمنا باختيار فلتر  $2 \times 2$  , وبخطوة 2 , لكن بالطبع هنا لا يوجد ضرب في ارقام لكن فقط اختيار الاكبر .

و عملية الـ **max pooling** هي بالفعل تقوم بتسهيل الشبكة و تسريعها , وقد يكون السبب , ان الرقم الاعلي دائما ما يشير الي شئ بارز و ظاهر , فيحتمل ان يكون شئ مهم (عين , سيارة , مسدس . )

و هذا معناه انه عند تطبيق الـ **MP** لابد من تحديد رقمين :  $f, s$  اي حجم المربع الذي سيقوم بتغطية مساحة ما , لفحص الرقم الاكبر , والخطوات التي سيمشي بها المربع (قد يتم تحديد  $p$  للحشو كذلك لكن هذا نادر)

كذلك الأمر , اذا تناولنا مصفوفة  $5 \times 5$  , هكذا :

1	3	2	1	3
2	9	1	1	5
1	3	2	3	2
8	3	5	1	0
5	6	1	2	9

وإذا قلنا ان  $f=3$  ,  $s=1$  فتكون مصفوفة الناتج  $3*3$  , وذلك لان الناتج سيكون بنفس القانون :

$$\frac{n+2p-f}{s} + 1$$

وتكون اول خطوة :

1	3	2	1	3
2	9	1	1	5
1	3	2	3	2
8	3	5	1	0
5	6	1	2	9

الثانية :

1	3	2	1	3
2	9	1	1	5
1	3	2	3	2
8	3	5	1	0
5	6	1	2	9

الثالثة :

1	3	2	1	3
2	9	1	1	5
1	3	2	3	2
8	3	5	1	0
5	6	1	2	9

ثم السطر التالي :

1	3	2	1	3
2	9	1	1	5
1	3	2	3	2
8	3	5	1	0
5	6	1	2	9

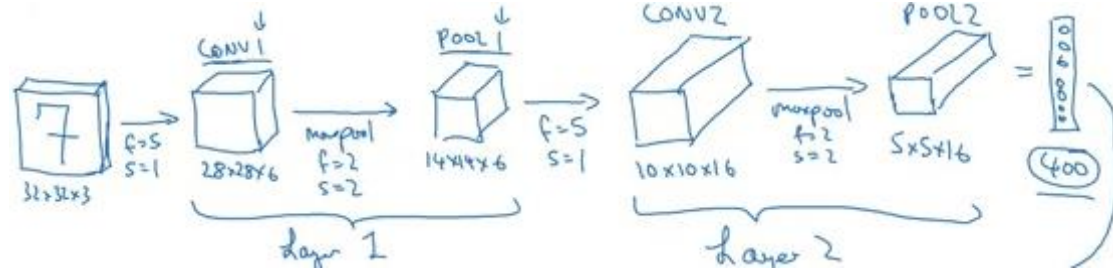
و تستمر بهذه الطريقة حتي تصل للنهاية , وتكون مصفوفة الناتج :

9	9	5
9	9	5
8	6	9



وإذا تناولنا مثالا عمليا , يكون كالتالي :

نفترض أن لدينا خوارزم لعمل تصنيف لرقم مكتوب بخط اليد , في صورة  $32 \times 32 \times 3$  لأنها صورة الوان , وليكن الرقم المكتوب هو 7  
إذا قمنا بتطبيق أول خطوة وهو عمل فلتر  $5 \times 5$  وبخطوة  $s=1$  وقيمة الـ channels تساوي 6 يتحول لمصفوفة :  $28 \times 28 \times 6$  , وتسمى conv 1



منها نقوم بعمل max pool عبر تحديد  $s=2$  ,  $f=2$  تتحول لمصفوفة :  $14 \times 14 \times 6$  , وتسمى pool 1

لاحظ ان كلا من conv 1 & pool 1 كلا منهما نسميها الطبقة الاولى layer 1 , وذلك لان مرحلة الـ pool ليس فيها عوامل يتم حسابها فكانها ليست طبقة منفصلة .

ثم نقوم بعمل فلتر جديد  $5 \times 5$  , بعدد channels يساوي 16 وقيمة  $s=1$  , يتحول لمصفوفة  $10 \times 10 \times 16$  وتسمى conv 2

ثم نطبق pool بفلتر  $s=2$  ,  $f=2$  تتحول لمصفوفة  $5 \times 5 \times 16$  وتسمى pool 2

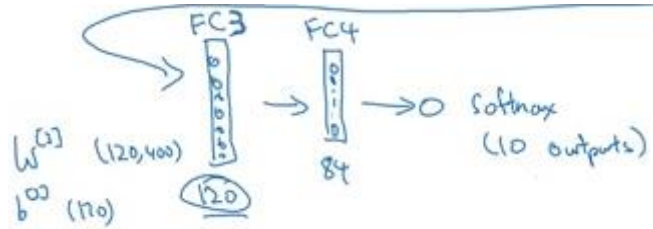
و ايضا كلا من conv 2 & pool 2 تسمى الطبقة الثانية layer 2

هنا نقوم بعمل افراد للقيم , والتي هي  $5 \times 5 \times 16$  تساوي 400 قيمة في مصفوفة واحدة , و تعتبر الـ inputs

و منها نقوم بإنشاء طبقة عادية بعدها تسمى FC3 و الـ FC اختصار Fully Connected و برقم 3 , لانه تم تسمية طبقة 1 و 2 سافا . وستكون 120 وحدة .

فتكون الاوزان  $w_3$  هي مصفوفة  $120 \times 400$  بينما  $b_3$  هي فيكتور 120

بعدها نقوم ببناء FC 4 من 84 وحدة , ويكون لها  $w_4$  بقيمة  $84 \times 120$  و  $b_4$  بقيمة 84 , ومنها نستخدم softmax بعدد 10 مخارج , لاننا نريد تصنيف الرقم بين 10 اختيارات , من 0 الي 9 .

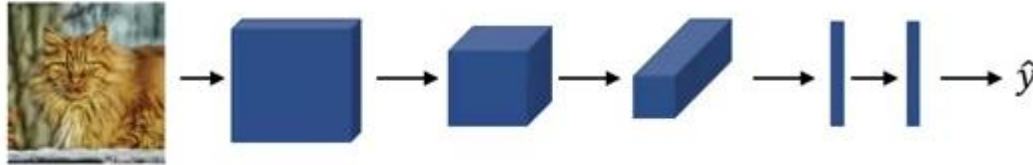




و بشكل عام , فالخريطة العامة للـ CNN تكون طبقة او اكثر conv يليها طبقة واحدة pool و طبقة او اكثر conv يليها طبقة واحدة pool واخيرا عدد من طبقات الـ FC يليها الـ softmax

وتكون هكذا :

Training set  $(x^{(1)}, y^{(1)}) \dots (x^{(m)}, y^{(m)})$ .



وإذا اخذنا نظرة سريعة علي الطبقات و المعاملات :

