مجلة جامعة طرطوس للبحوث والدراسات العلمية _ سلسلة العلوم الهندسية المجلد (7) العدد(1) 2023
Tartous University Journal for Research and Scientific Studies - engineering Sciences Series Vol. (7) No. (1) 2023

# تحليل زمن تدريب الشبكات العصبية الالتفافية على معالجات عالية الأداء

د. حسن البستاني*

م. نور خضر**

(تاريخ الإيداع 2022/10/30 . قُبِل للنشر في 2023/1/10 )

□ ملخّص□

نتيجة للتطور الكبير في حجم البيانات لم يعدّ أداء الشبكات العصبونية التقليدية مقنعاً، وخاصة بالنسبة لزمن تدريبها، لذلك فقد اتجه الباحثون لحلّ تطبيقات التصنيف classification applications باستخدام الشبكات العصبية الالتفافية CNN. نظرًا لزيادة معدل البيانات بشكل كبير وعمق هذه الشبكات وتعقيدها، يزداد الزمن اللازم لتدريب هذه الشبكات وتصبح كفاءتها أقل عند استخدام وحدة المعالجة المركزية CPU غير المتوازية كمعالج لتدريب الشبكات العصبونية الالتفافية، لحل هذه المشاكل لا بد من استخدام معالجات متوازية عالية الأداء، من بين المعالجات عالية الأداء الحديثة لاقت وحدة معالجة الرسوم GPU استخداماً وتطبيقاً لحل المشاكل المعقدة التي تتطلب زمن حوسبة كبير، وبين هذه التطبيقات تدريب الشبكات العصبونية الالتفافية.

طورت Nvidia منصة خاصة لاستخدام وحدة معالجة الرسوم GPU وهي منصة CUDA، حيث تعد CUDA منصة حوسبة متوازية ونموذج برمجي متعدد النياسيب multi-threads. سنقوم في هذا البحث باختيار قاعدة بيانات كبيرة وتدريبها على عدة وحدات معالجات الرسومات بقدرات حاسوبية مختلفة من تصنيع شركة Nvidia بالإضافة لتدريبها على وحدة المعالجة المركزية CPU. أكدت النتائج التجريبية كفاءة عالية لوحدة GPU كمعالج عالي الأداء في تقليل الزمن المستغرق لتدريب الشبكات العصبونية الالتفافية مع دقة عالية في التصنيف.

**الكلمات المفتاحية:** الشبكات العصبية الالتفافية، التدريب التسلسلي، التدريب المتوازي، وحدة GPU، منصة CUDA.

*مدرس في كلية هندسة تكنولوجيا المعلومات والاتصالات – جامعة طرطوس – طرطوس – سورية.

** طالبة ماجستير في كلية هندسة تكنولوجيا المعلومات والاتصالات – جامعة طرطوس – طرطوس – سورية.

مجلة جامعة طرطوس للبحوث والدراسات العلمية _ سلسلة العلوم الهندسية المجلد (7) العدد(1) 2023
Tartous University Journal for Research and Scientific Studies - engineering Sciences Series Vol. (7) No. (1) 2023

# Analysis of the training time of convolutional neural networks on high-performance processors

**Dr. Hasan Albustani***
**Eng. Noor khder****

**(Received 30/10/2022 . Accepted 10/1/2023)**

**□ ABSTRACT**

As a result of the large development in the big data, the performance of traditional neural networks is no longer convincing, especially in terms of their training time, so researchers have decided to solve the classification applications using CNNs. Due to increasing in data, depth and complexity of these networks, the time required to train it increases, and their efficiency becomes less when use non-parallel processors to train convolutional neural networks. To solve these problems, high performance parallel processors must be used, among the modern high-performance processors, The GPU has been the most widely used and applied to solve complex problems that require large computing time, and among these applications is the training of convolutional neural networks.

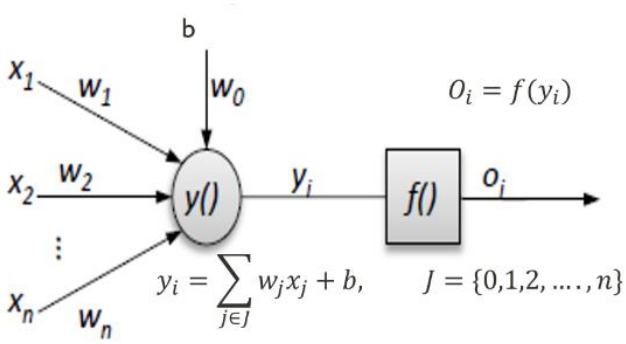
Nvidia developed a special platform for using the GPU, it is CUDA platform. CUDA is a parallel computing platform and software model multi-threaded. In this research, we will choose a large database and train this dataset on several graphics processor units with different computing capabilities from Nvidia, in addition to training it on the CPU. Experimental results confirmed the high efficiency of the GPU as a high-performance processor in reducing the time taken to train convolutional neural networks with high classification accuracy.

**Keywords:** Convolution neural network, Sequential training, Parallel training, Compute Unified Device Architecture (CUDA).

*Assistant Professor - Faculty of Information and Communication Technology Engineering - Tartous University - Tartous - Syria.
**Master Student - Faculty of Information and Communication Technology Engineering - Tartous University - Tartous - Syria.

## 1. المقدمة:

تستوحي الشبكات العصبية الاصطناعية اسمها وبنيتها من الدماغ البشري حيث تحاكي الطريقة التي تعمل بها الخلايا العصبية البيولوجية مع بعضها بعض. في عام 1943، اقترح العالمان McCulloch وPitts [1] مفهوم الشبكة العصبية الاصطناعية والنموذج الرياضي المعبر عنه كما في الشكل(1) . تتكون من عدة طبقات من العقد المتصلة ببعضها بعض، العقدة هي الخلية العصبية التي تستقبل الدخل وتطبق عليه وزن معين وانحياز، نسمي هذه العقدة بالعصبون، وعلى خرج العصبون تطبق دالة رياضية تدعى دالة التنشيط، تختلف بحسب التطبيق العملي المستخدم.




**الشكل (1): نموذج الشبكة العصبية الاصطناعية.**

تتكون الشبكة العصبونية من عدة طبقات بدءاً من طبقة الدخل وطبقة مخفية واحدة أو أكثر، وطبقة خرج، تتصل الطبقات مع بعضها بعض لحل مشكلات معقدة مثل التعرف على الصور والصوت والفيديو، وتحليل الصور وتصنيفها، والتعرف على الكلام، إلخ. وبسبب ظهور اهتمام هائل بالتعلم العميق في السنوات الأخيرة[2] نشأت أنواع مختلفة من الشبكات العصبونية، والتي تُستخدم في تطبيقات مختلفة وأنواع بيانات مختلفة، على سبيل المثال، تُستخدم الشبكات العصبونية العودية RNN بشكل شائع لمعالجة اللغة الطبيعية والتعرف على الكلام، بينما تُستخدم الشبكات العصبونية الالتفافية (CNN) غالبًا لمهام التصنيف والرؤية الحاسوبية.

في عام 1959، قام العالمان David Hubel وTorsten Wiesel بنشر تجارب حول خلايا القشرة البصرية الحيوانية وكيف تتعرف على الضوء في حقل استقبالي صغير [3]، حيث وجد أن الخلايا العصبية داخل دماغ قطة تنتظم على شكل طبقات. تتعلم هذه الطبقات كيفية التعرف على الأنماط المرئية عن طريق استخراج الميزات المحلية أولاً، ثم دمج الميزات المستخرجة للحصول على تمثيل أعلى مستوى. في وقت لاحق، أصبح هذا المفهوم أحد المبادئ الأساسية للتعلم العميق.

في عام 1980، ألهمت هذه التجارب Kunihiko Fukushima واقترح شبكة Neocog−nitron[4]، وهي شبكة عصبية ذاتية التنظيم تحتوي على طبقات متعددة، قادرة على التعرف على الأنماط المرئية بشكل هرمي من خلال التعلم، وأصبحت هذه الهندسة أول نموذج نظري لشبكة CNN.

في عام 1990، طور LeCun et al، الإطار الحديث لـ CNN المسمى LeNet−5 للتعرف على الأرقام المكتوبة بخط اليد MNIST [5]. تم تدريب LeNet−5 باستخدام خوارزمية الانتشار العكسي للخطأ والذي ساعد على التعرف على الأنماط المرئية من الصور الأولية مباشرةً دون استخدام أي آلية هندسية منفصلة للميزات[6].

لم تقدم CNN أداءً جيدًا في المشكلات المعقدة المتنوعة بسبب نقص بيانات التدريب الكبيرة ونقص الابتكار في الخوارزمية وقوة الحوسبة غير الكافية، ولكن في عصر البيانات الضخمة، يوجد مجموعات بيانات تدريب كبيرة، وخوارزميات أكثر ابتكارًا، ووحدات معالجة رسومية ضخمة أدت لظهور شبكة CNN تسمى AlexNet في عام 2012، صممها Kievsky et al وأظهرت أداءً ممتازًا في تطبيقات التعرف البصري على نطاق واسع المعروفة باسم ImageNet Large Scale Visual Recognition Competition (ILSVRC) [7]. أدى نجاح AlexNet إلى تمهيد الطريق لابتكار العديد من نماذج CNN [8] بالإضافة إلى تطبيق تلك النماذج في مجالات مختلفة من الرؤية الحاسوبية ومعالجة اللغة الطبيعية.

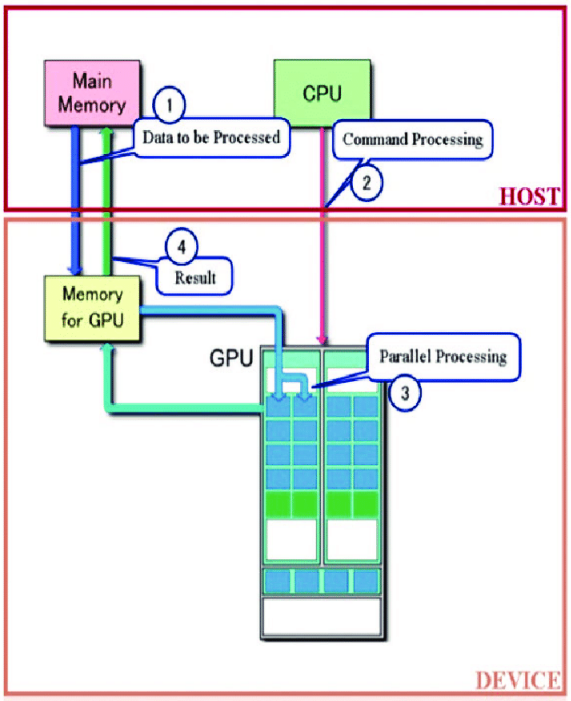
قدمت CNN نهجًا أكثر قابلية للتوسع لتصنيف الصور ومهام التعرف على الكائنات objects، والاستفادة من مبادئ الجبر الخطي، وتحديداً حوسبة المصفوفة، لتحديد الأنماط داخل الصورة بدلاً من استخدام طرق استخراج الميزات اليدوية والمستهلكة للوقت لتحديد الكائنات في الصور، ولكنها مكلفة من الناحية الحسابية، وتتطلب وحدات معالجة رسومية لتدريب النماذج بكفاءة أكبر[9] . وبسبب زيادة حجم البيانات المراد معالجتها يمكن أن يستمر التدريب على وحدة المعالجة المركزية CPU لعدة ساعات أو أيام. من أجل تسريع هذه العملية، يتم تنفيذ التدريب في بيئة متوازية باستخدام وحدة معالجة الرسومات GPU إلى جانب وحدة المعالجة المركزية، وكانت Nvidia قد طورت منصة CUDA وهي منصة حوسبة متوازية ونموذج برمجي.

تتكون وحدة معالجة الرسومات من مجموعة من المعالجات المتعددة المتدفقة streaming multiprocessors (SM)، كل منها قادر على دعم الآلاف من نياسيب threads الأجهزة المتزامنة، حتى 2048 نواة على وحدات معالجة الرسومات الحديثة. يتم تنفيذ جميع عمليات إدارة الخيوط، بما في ذلك الإنشاء والجدولة والمزامنة بالكامل في الأجهزة بواسطة SM بدون أي تكلفة إضافية. للحصول على الأداء الأمثل، قد نحتاج إلى استخدام كل من وحدة المعالجة المركزية ووحدة معالجة الرسومات للتطبيق، حيث تنفذ الأجزاء المتسلسلة على وحدة المعالجة المركزية، بينما تنفذ أجزاء البيانات المكثفة المتوازية على وحدة معالجة الرسومات.

لا تعد وحدة معالجة الرسومات حاليًا منصة قائمة بحد ذاتها ولكنها معالج مشترك لوحدة المعالجة المركزية. لذلك، يجب أن تعمل وحدات معالجة الرسومات جنبًا إلى جنب مع مضيف قائم على وحدة المعالجة المركزية من خلال ناقل PCI–Express. تسمى وحدة المعالجة المركزية المضيف Host وتسمى وحدة معالجة الرسومات بالجهاز Device.

يوضح الشكل (2) آلية عمل CUDA وفقاً للخطوات الآتية:

1. تخصيص الذاكرة على المضيف والجهاز بشكل منفصل. تكون ذاكرة الجهاز قابلة للقراءة والكتابة بواسطة المضيف من خلال وظائف نسخ الذاكرة. يتم نسخ البيانات من المضيف إلى الجهاز باستخدام واجهة برمجة تطبيقات CUDA.
2. يتم استدعاء وظيفة النواة عن طريق الكود التسلسلي من وحدة المعالجة المركزية.
3. يتم تنفيذ وظيفة Kernel بشكل متوازي على كل نواة.
4. نسخ النتائج من الجهاز إلى المضيف باستخدام واجهة برمجة تطبيقات CUDA.




**الشكل :(2) آلية عمل منصة CUDA.**

في هذا البحث سنستخدم شبكة Vgg16 بنسخة مصغرة لتصنيف فئتين من الصور هما القطط والكلاب، تتم المعالجة المسبقة للصور وتهيئة الشبكة بمعدل التعلم، ومن ثمّ تنفيذ التدريب المقترح باستخدام إطار عمل TensorFlow.

سنقوم بتدريب الشبكة بشكل تسلسلي على وحدة المعالجة المركزية فقط، ثم نقارن النتيجة مع التدريب بشكل متوازي على المعالجات NVIDIA GeForce 930M، NVIDIA GeForce GTX 1050، Tesla T4، من حيث كمية الزمن المطلوب لعملية التدريب.

لا بد من الإشارة إلى المرجع [10]، الذي قام بدراسة أجراها لاكتشاف إشارات المرور والتعرف عليها في الزمن الحقيقي وتصنيفها باستخدام شبكة CNN مكونة من ثلاث طبقات التفاف وطبقة متصلة بالكامل، تم تنفيذها على معالج Intel Core i7 CPU والمعالجين Nvidia GeForce GTX 650، Nvidia GeForce GTX 650M للاستفادة من منصة CUDA وحصلوا على النتائج المبينة كما في الجدول (1). ونتيجة الاختلاف في قاعدة البيانات المستخدمة، والاختلاف في نوع المعالجات المستخدمة، فقد تعذر اجراء مقارنة في زمن تدريب الشبكات العصبونية الالتفافية. والميزة التي أضافها البحث والتي لم تؤخذ بعين الاعتبار في البحث المذكور من حيث بنية المعالجات، فقد استخدمنا معالجات جديدة من انتاج شركة Invidia هي معالج Tesla T4، واستخدمنا إطار العمل TensorFlow.

**جدول (1) يبين المقارنة بين تنفيذ تدريب شبكة CNN على أجهزة مختلفة وفقاً للمرجع [19].**

| المعالج | زمن التدريب بالدقائق |
|---|---|
| (CPU) Intel Core i7 | 16 min |
| Nvidia GeForce GTX 650 | 12 min |
| Nvidia GeForce GT 650M | 7 min |

## 2. أهمية البحث وأهدافه:

نتيجة للتطور الكبير في حجم البيانات، أصبح زمن تدريب الشبكات العصبونية الالتفافية على وحدة المعالجة المركزية كبير جداً، لذلك سنقوم في هذا البحث باستخدام وحدة معالجة الرسوم GPU كمعالج عالي الأداء لتدريب شبكات CNN بهدف تقليل زمن تدريب هذه الشبكات.

وسنقوم بمقارنة زمن تدريب شبكة CNN على وحدة معالجة مركزية CPU، مع عدة أنواع من المعالجات الرسومية GPU التي تستخدم CUDA كإطار تعلم عميق للاستفادة من سرعة معالجات GPU لتقليل الزمن اللازم للتدريب وزيادة دقة الشبكة وتقليل الخسارة.

## 3. مشكلة البحث:

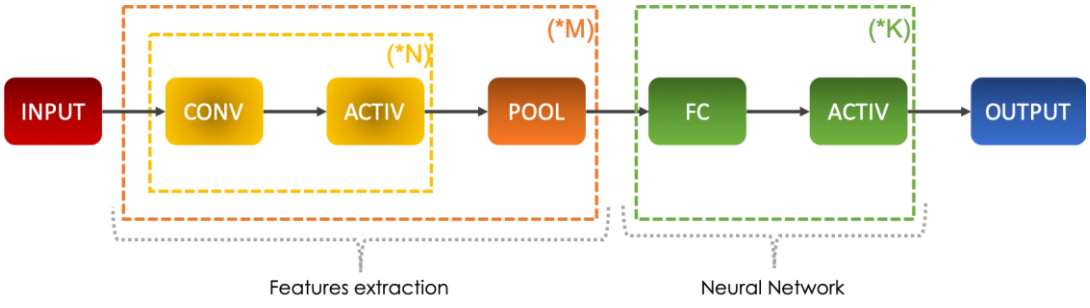
تأثرت كفاءة شبكات CNN بزيادة تعقيد المهام ومجموعات البيانات الضخمة التي تحتاجها للتدريب، مما يجعل تدريب هذه الشبكات على أجهزة كمبيوتر تقليدية غير عملية، فهي مكثفة للغاية من الناحية الحسابية ويمكن أن تستغرق عدة ساعات وحتى أيام لتشغيلها على وحدات المعالجة المركزية CPU. لحل هذه المشكلة سنقوم بتدريب شبكة CNN بشكل متوازي من خلال منصة الحوسبة المتوازية CUDA والتي تسخر قوة GPU لموازنة عمليات الالتفاف وغيرها من العمليات الحسابية التي تحتاج قوة معالجة كبيرة فنحصل على أداء أفضل بكثير مقارنة بالتدريب على وحدة المعالجة المركزية CPU.

## 4. طرائق البحث ومواده:

### 4-1-التعريف بشبكات CNN:

شبكات CNN هي نوع من نماذج التعلم العميق لمعالجة البيانات التي تحتوي على نموذج شبكي، مثل الصور، مستوحاة من الإدراك البصري للكائنات الحية، ومصممة لتعلم التسلسل الهرمي المكاني للسمات تلقائيًا وبشكل تكيفي، من المستوى المنخفض إلى الأنماط عالية المستوى.

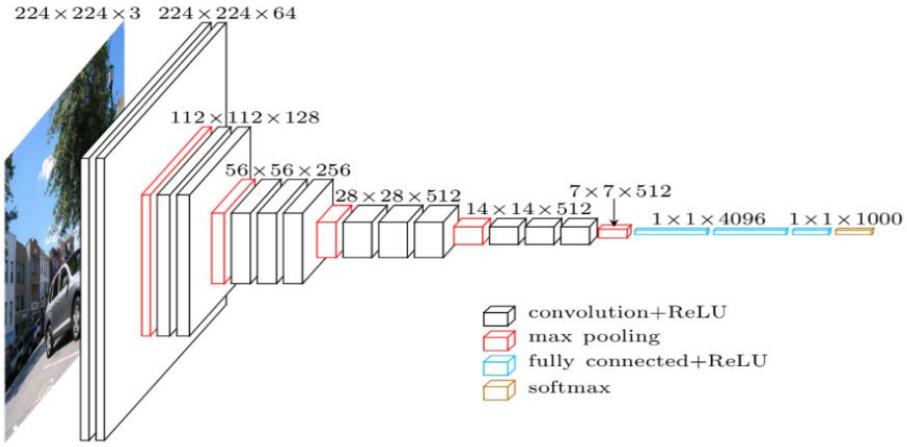
تتكون شبكة CNN عادةً من ثلاثة أنواع من الطبقات: طبقة الالتفاف Convolution layer، وطبقة التجميع pooling، وطبقة الاتصال الكامل full connection. تقوم طبقتا الالتفاف والتجميع، باستخلاص الميزات feature extraction، بينما تقوم الطبقة الاتصال الكامل، بتعيين الميزات المستخلصة في الخرج النهائي، مثل التصنيف [11]. تتكون CNN من كتلة واحدة أو أكثر من طبقات الالتفاف والتجميع، تليها طبقة أو أكثر من الطبقات الاتصال الكامل (FC) ثم طبقة الخرج كما هو موضح في الشكل (3).




**الشكل (3): بنية الشبكة العصبية الالتفافية.**

على مدى السنوات الماضية، تم تقديم العديد من بنى CNN المختلفة [12]. تعد بنية النموذج عاملاً حاسمًا في تحسين أداء التطبيقات المختلفة. تم إجراء تعديلات مختلفة في بنية CNN من عام 1989 حتى اليوم. تتضمن هذه التعديلات إعادة الصياغة الهيكلية، والتنظيم، وتحسينات استخلاص الميزات، الخ.

اقترح Simonyan و Zisserman [13] بنية سهلة وفعالة لشبكة CNN. كان هذا التصميم المبتكر يسمى Visual Geometry Group (VGG) كما هو موضح في الشكل (4). يتميز هذه التصميم بعدد طبقات أكثر ومرشحات ذات أحجام أصغر وبنفس فعالية المرشحات ذات الأحجام الأكبر التي كانت تستخدم في التصاميم السابقة. يكون الدخل في CNN عبارة عن صورة متعددة القنوات (بالنسبة لصورة RGB، يكون لدينا 3 قنوات والصور الرمادية لدينا قناة واحدة). سنعتمد هذه البنية في هذا البحث كنموذج للشبكات العصبونية الالتفافية، وسنقوم بتدريب هذه الشبكات على معالجات GPU.



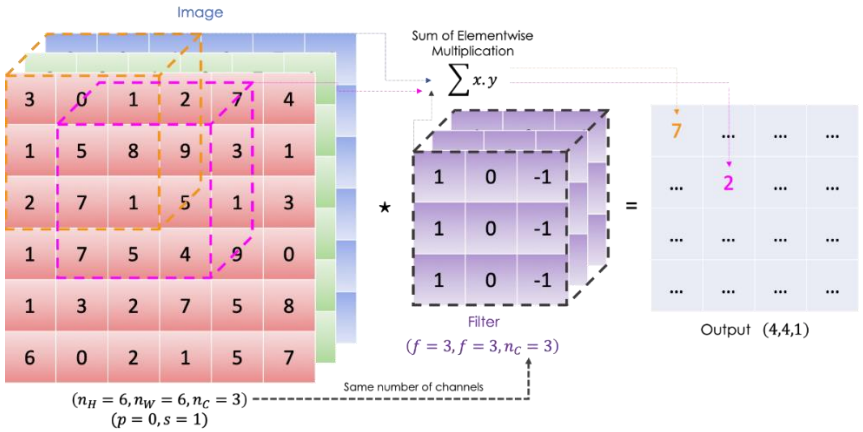

**الشكل (4): هيكل شبكة VGG16 المعتمدة في هذا البحث.**

### 1. الطبقة الالتفافية Convolution layer:

الطبقة الالتفافية هي لبنة البناء الأساسية لشبكة CNN تحتوي على مجموعة من الأنوية (المرشحات) والتي يتم ربطها مع صورة الدخل (مصفوفة ذات بعد n) لإنشاء مصفوفة خريطة الميزات كخرج. تُعرف النواة بأنها شبكة من القيم المنفصلة، حيث كل قيمة هي وزن لهذه النواة. يتم ضبط جميع أوزان النواة بأرقام عشوائية عند بدء عملية التدريب لنموذج CNN، (من الممكن ضبطها بطرق أخرى)، تغذي الطبقات بعضها مما يجعل الميزات المستخرجة هرمية وأكثر تعقيد حيث مع كل فترة تدريب، يتم ضبط الأوزان وتتعلم النواة لاستخراج ميزات ذات معنى [14].

تكون طبقة الالتفاف الأولى مسؤولة عن التقاط الميزات ذات المستوى المنخفض مثل الحواف واللون ومع الطبقات المضافة، تتكيف بنية الشبكة مع الميزات عالية المستوى، مما يوفر شبكة تتمتع بفهم كامل للصور في مجموعة البيانات.

تتحرك النواة على الصورة بدءاً من أعلى اليسار إلى اليمين وتتحرك نحو الأسفل قليلاً بعد تغطية عرض الصورة وتتكرر نفس العملية حتى ينتهي مسح الصورة بأكملها وفي كل مرة نأخذ حاصل الضرب النقطي بين النواة وصورة الإدخال بضرب القيم المقابلة لبعضها بعض، ثم جمع هذه القيم لإنشاء قيمة مقياس واحد في خرج خريطة الميزات كما في الشكل(5) . تستمر هذه العملية حتى تتوقف النواة عن الانزلاق على الصورة. كل قيمة في خريطة الميزات هي المجال الاستقبالي لمنطقة معينة في الصورة الأصلية[15] .




**الشكل (5): تحرك المرشح على صورة الدخل لاستخراج خريطة الميزات.**

الطبقات اللاحقة في الشبكة العصبية قادرة على البناء على خريطة الميزات التي اكتشفتها الطبقات السابقة وتحديد الأشكال الأكثر تعقيدًا. يمكن تعريف الالتفاف وفق العلاقة الرياضية الآتية:

$$\boldsymbol{conv(I,K)= \sum_{i=1}^{n_H} \sum_{j=1}^{n_W} \sum_{k=1}^{n_C} K_{i,j,k} I_{x+i-1,y+j-1,k}} \quad (1)$$

حيث:

$n_H$: عدد أسطر مصفوفة الدخل $I$: مصفوفة الدخل $n_H \dots 3,2,1 = i$

$n_W$: عدد أعمدة مصفوفة الدخل $K$: مصفوفة المرشح $n_W \dots 3,2,1 = j$

$n_C$: عمق مصفوفة الدخل $n_C \dots 3,2,1 = k$

تمتلك طبقة الالتفاف معاملين هما معامل الحشو (padding) ومعامل الخطوة (stride)، تمثل الخطوة عدد وحدات البيكسل التي تنزاح عبر مصفوفة الدخل ، وكلما كانت الخطوة كبيرة سيتم تقليص حجم المخرجات والعكس صحيح، نشير لها بالرمز s. يستخدم الحشو لحل مشكلة البكسلات الموجودة في زاوية الصورة كونها أقل استخدامًا من البكسلات في منتصف الصورة فيتم التخلص من المعلومات في الحواف بسرعة كبيرة، لذلك نضيف حشو حول الصورة لأخذ وحدات البيكسل الموجودة على الحواف في الاعتبار، نشير لها بالرمز P. النواة هي المميزة الوحيدة التي يتم تعلمها تلقائيًا أثناء عملية التدريب في طبقة الالتفاف؛ من ناحية أخرى، فإن حجم النواة وعدد الأنوية والحشو والخطوة هي معلمات فائقة يجب ضبطها قبل بدء عملية التدريب. تتميز طبقات الالتفاف بالخاصيتين الآتيتين:

(1) اتصال ضئيل: في الشبكة العصبونية ضمن طبقة الاتصال الكامل، تتصل كل خلية عصبية من طبقة واحدة بكل خلية عصبية من الطبقة التالية ولكن في CNN يوجد عدد صغير من الأوزان بين طبقتين، لذلك، فإن عدد الأوزان التي تحتاجها صغير، بالتالي حجم الذاكرة لتخزين تلك الأوزان صغير أيضًا، لذا فهي مناسبة للذاكرة. إضافة إلى أنها أرخص من الناحية الحسابية من مضاعفة المصفوفة.

(2) تقاسم الوزن: لا توجد أوزان مخصصة بين كل خليتين عصبيتين من الطبقات المتجاورة بل تعمل جميع الأوزان مع كل بكسل من مصفوفة الدخل. يمكن تعلم مجموعة واحدة من الأوزان لجميع المدخلات وهذا يقلل بشكل كبير من وقت التدريب بالإضافة إلى التكاليف الأخرى [14].

**2. وظيفة التنشيط:**

بعد تطبيق المرشح على الصورة الأصلية، يتم تمرير خرج الالتفاف من خلال وظيفة رياضية تسمى وظيفة التنشيط. وظيفة التنشيط المستخدمة غالباً في CNN هي RELU (Rectified Linear Unit) وتعطى بالعلاقة الآتية [14]:

$$f(x) = max(0, x) \quad (2)$$

**3. التجميع pooling:**

توفر طبقة التجميع عملية اختزال نموذجية حيث تقلل الحجم المكاني للميزة الملتقة وبالتالي تقليل القدرة الحسابية المطلوبة لمعالجة البيانات من خلال تقليل الأبعاد (عدد أعمدة مصفوفة الدخل $n_W$،عدد أسطر مصفوفة الدخل $n_H$ ) وتحافظ على عدد المرشحات المستخدمة في الالتفاف $n_C$، بينما يتم تقليص خرائط الميزات، فإنها تحافظ دائمًا على السمات الأكثر شيوعًا في كل خطوات التجميع، حيث تستخرج الميزات المهيمنة التي تكون ثابتة الدوران والموضع فتحافظ على عملية تدريب فعال للنموذج وتقلل زمن التدريب. يتم تنفيذ عملية التجميع عن طريق تحديد حجم المناطق المجمعة وخطوة العملية، على غرار عملية الالتفاف [14]. اعتمادًا على التعقيدات الموجودة في الصور، يمكن زيادة عدد هذه الطبقات لالتقاط تفاصيل ذات مستويات منخفضة أكثر، ولكن على حساب المزيد من الطاقة الحسابية. توجد أنواع مختلفة من تقنيات التجميع مثل التجميع الأعظمي (Max Pooling)، والتجميع المتوسط (average pooling)، والتجميع الأصغر(Min Pooling) ، وغيرها. ولكن أكثرها شيوعاً هو التجميع الأعظمي المبين في الشكل (6).

يمكن حساب أبعاد خريطة الميزات كالآتي:

$$\dim(pooling(image)) = \left(\left\lfloor \frac{n_H+2p-f}{s} + 1 \right\rfloor , \left\lfloor \frac{n_W+2p-f}{s} + 1 \right\rfloor , n_C \right) \quad : s>0 \quad (1-3)$$

$$= n_H + 2p - f, n_W + 2p - f, n_C \quad : s=0 \quad (2-3)$$

حيث

$n_H$: عدد أسطر مصفوفة الدخل

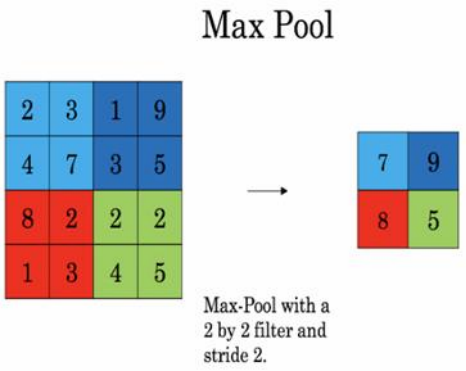
$n_W$: عدد أعمدة مصفوفة الدخل

$n_C$: عدد المرشحات المستخدمة في الالتفاف

$p$: حجم منطقة التجميع

$f$: أبعاد مصفوفة المرشح

s: خطوة عملية التجميع




**الشكل (6): عملية التجميع Max pooling.**

### 4. الطبقة الاتصال الكامل Fully connected layer:

يتم تسطيح خرائط الميزات لطبقة الالتفاف أو التجميع النهائية، أي تحويلها إلى مصفوفة أحادية البعد (متجه)، ومتصلة بطبقة واحدة أو أكثر من الطبقات المتصلة بالكامل، حيث يتم توصيل كل دخل بكل خرج بوزن قابل للتعلم[16] . تحتوي الطبقة الاتصال الكامل النهائية على نفس عدد عقد الإخراج مثل عدد الفئات. كل طبقة متصلة بالكامل تتبعها عادةً وظيفة التنشيط RELU، الطبقة المتصلة بالكامل الأخيرة تتبعها وظيفة تنشيط تبعاً لنوع المشكلة، نستخدم تابع Sigmoid للتصنيف الثنائي، ويستخدم تابع SoftMax للتصنيف متعدد الطبقات [12].

### 5. تدريب الشبكة العصبية الالتفافية CNN:

تدريب الشبكة هو عملية العثور على الأنوية في طبقات الالتفاف والأوزان في الطبقات المتصلة بالكامل، مما يقلل الاختلافات بين تنبؤات الخرج وتسميات الحقيقة الأساسية على مجموعة بيانات التدريب. تعد خوارزمية الانتشار العكسي backpropagation هي الطريقة المستخدمة بشكل شائع لتدريب الشبكات العصبونية، حيث تلعب وظيفة الخطأ وخوارزمية التحسين أدوارًا أساسية [16]. يتم حساب أداء النموذج تحت نواة وأوزان معينة من خلال دالة الخطأ (الفرق بين الخرج الهدف والخرج الفعلي للشبكة) من خلال الانتشار الأمامي على مجموعة بيانات التدريب، ويتم تحديث الميزات القابلة للتعلم، أي النوى والأوزان، وفقًا لقيمة الخطأ من خلال خوارزمية الأمثلة optimization.

- دالة الخطأ: تقيس التوافق بين تنبؤات الخرج من خلال الانتشار الأمامي وتسميات الحقيقة الأساسية. يستخدم في مشاكل التصنيف المتعدد دالة خسارة هي cross entropy، بينما يتم تطبيق متوسط الخطأ التربيعي mean squared error عادةً على مشاكل الانحدار إلى القيم المستمرة. نوع وظيفة الخطأ هو أحد المعلمات الفائقة ويجب تحديده وفقًا للمهام المحددة.
- الأمثلة optimization: تستخدم خوارزمية الهبوط المنحدر Gradient descent بشكل شائع كخوارزمية أمثلة، حيث تقوم بشكل متكرر بتحديث الميزات القابلة للتعلم، أي النواة والأوزان، للشبكة لتقليل الخطأ. يوفر لنا التدرج لوظيفة الخطأ اتجاه الذي تتمتع فيه الوظيفة بأعلى معدل زيادة، ويتم تحديث كل ميزة قابلة للتعلم في الاتجاه السلبي للتدرج مع حجم خطوة كيفي يتم تحديده بناءً على ميزة فائقة تسمى معدل التعلم أو نسبة التعلم. التدرج، رياضيًا، مشتق جزئي من الخسارة فيما يتعلق بكل معلمة قابلة للتعلم.

### 4-2-الحوسبة باستخدام GPU:

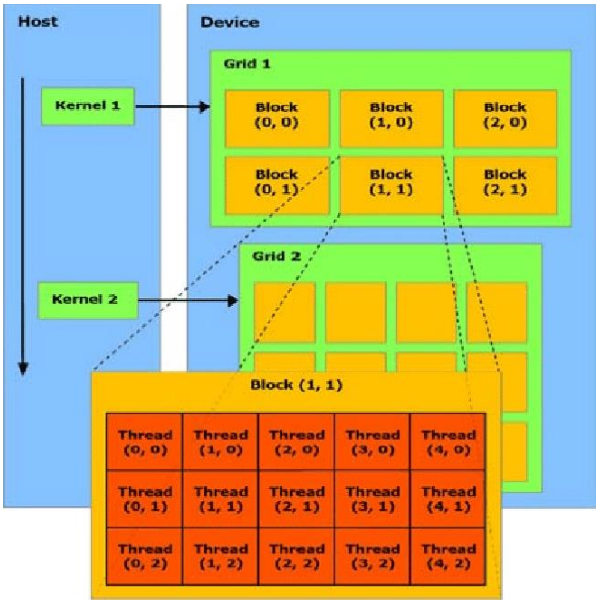
وحدة معالجة الرسوم GPU عبارة عن معالج متوازي مخصص لتسريع العمليات الحسابية الرسومية وإجراء عمليات الفاصلة العائمة اللازمة لعرض الرسوم ثلاثية الأبعاد، والتي تفوق قدرة وحدة المعالجة المركزية CPU لتنفيذها.

تم تصميم وحدات معالجة الرسومات لتسريع عرض الرسومات ثلاثية الأبعاد في الزمن الحقيقي، مثل تطبيقات الألعاب. أصبحت الألعاب مع مرور الوقت أكثر كثافة من الناحية الحسابية مع رسومات فائقة الواقعية وعوالم واسعة ومعقدة داخل اللعبة. لذا تزايدت الطلبات على معالجات رسومية أكثر سرعة خصوصاً بعد ظهور تقنيات العرض المتطورة مثل شاشات 4K وظهور ألعاب الواقع الافتراضي. ازدادت مرونة وحدة معالجة الرسومات GPU وأصبحت قابلة للبرمجة، مما سمح لمبرمجي الرسومات بإنشاء تأثيرات بصرية أكثر إثارة للاهتمام ومشاهد واقعية باستخدام تقنيات الإضاءة والتظليل المتقدمة. بدأ المطورون في الاستفادة من قوة وحدات معالجة الرسومات لتسريع أعباء العمل الإضافية بشكل كبير في الحوسبة عالية الأداء (HPC) والتعلم العميق.

تتطلب شبكات CNN الأكبر حجماً وعمقاً من حيث عدد الطبقات والميزات مزيداً من قوة المعالجة الحاسوبية، مما يجعل تدريبها باستخدام جهاز كمبيوتر تقليدي غير عملياً [17]. يمكن تمثيل الحسابات في CNN على أنها عمليات مصفوفية يمكن موازنتها بكفاءة. وبالتالي، فإن قوة المعالجة المتوازية الهائلة لوحدات معالجة الرسومات GPU تجعل تدريب شبكات CNN أكثر كفاءة، وتدعم أطر التعلم العميق الشائعة تسريع GPU افتراضيًا [18].

منصة التعلم العميق الأكثر شعبية لمثل هذه الأطر هي CUDA (Compute Unified Device Architecture) التي تم تطويرها من قبل شركة NVIDIA ومعظم أطر التعلم العميق الشائعة تستخدمها كواجهة خلفية لوحدة GPU. تعدّ CUDA منصة حوسبة متوازية ونموذج برمجي يمكن استخدامها لأداء مهام الحوسبة العامة مثل ضرب المصفوفات وإجراء عمليات الجبر الخطي، بدلاً من مجرد إجراء الحسابات الرسومية من خلال تسخير قوة GPU لموازنة العمليات الحسابية وتسريع التطبيقات التي تحتاج إلى قدرة معالجة كبيرة، مما يسمح بأداء أفضل بكثير مقارنة بوحدة المعالجة المركزية CPU. يتم تشغيل الأجزاء المتسلسلة من حمل العمل في سلسلة واحدة على وحدة المعالجة المركزية للجهاز، وتعمل الأجزاء كثيفة الحوسبة بالتوازي على آلاف من نوى وحدة معالجة الرسومات.

تختلف نواة GPU تمامًا عن نواة CPU. تم تصميم نواة CPU بمنطق تحكم معقد للغاية لتنفيذ البرامج التسلسلية، بينما تم تصميم نواة GPU بمنطق تحكم أبسط يركز على تشغيل مهام البيانات المتوازية، حيث تحتوي GPU على آلاف النوى الصغيرة والأكثر كفاءة من CPU متعددة الأنوية multicores. توفر CPU متعددة الأنوية ذاكرة تخزين مؤقت كبيرةK وتنفذ تعليمات بطول x86 كاملة على كل نواة، في حين أن نوى وحدة معالجة الرسومات الأصغر مخصصة للنقطة العائمة للإنتاجية. تتكون بنية CUDA كما في الشكل (7) من ثلاثة أجزاء أساسية تساعد مطوري النظام على الاستفادة من إمكانات الحوسبة الفعالة لوحدة GPU وهي شبكات grids وكتل blocks ونياسب threads في هيكل هرمي. نظرًا لوجود عدد من النياسب في كتلة وعدد الكتل في شبكة وعدد من الشبكات في GPU واحدة، لذلك يتم تحقيق العملية الموازية باستخدام بنية هرمية كبيرة جدًا[19] .

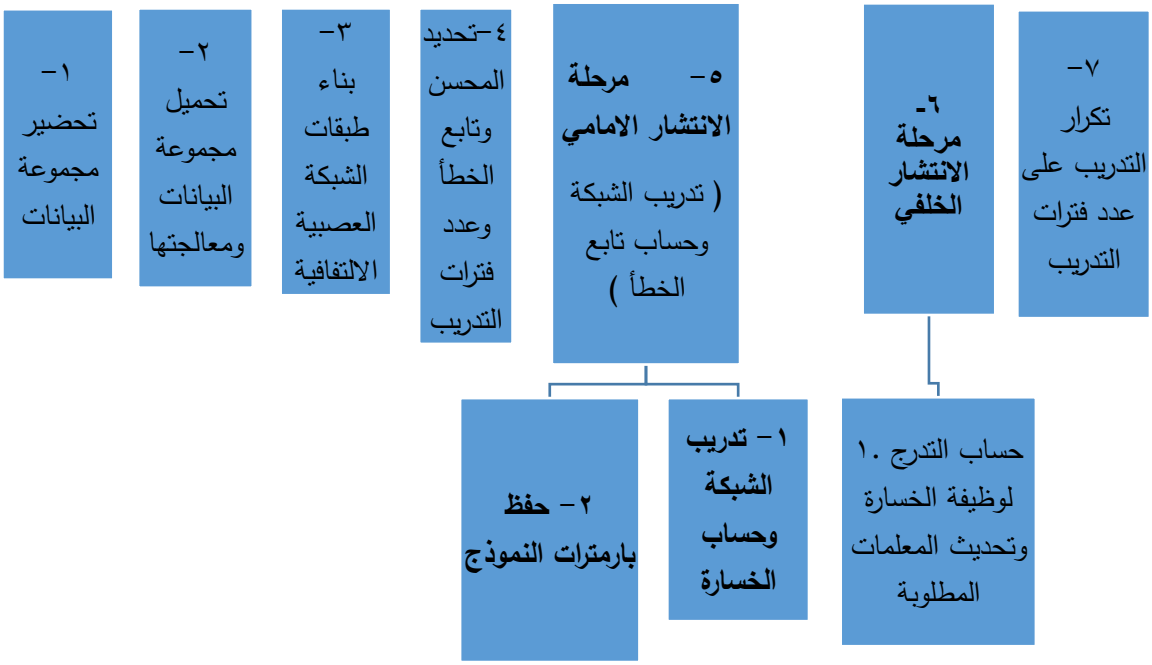



**الشكل (7) بنية CUDA**

في حالة وجود مشكلات مثل معالجة الصور والفيديو، يكون حجم البيانات المراد معالجتها كبيراً جداً. في حال استخدام وحدة المعالجة المركزية التسلسلية لحل هذه المشكلات، يمكن أن تستمر أجزاء التدريب والاختبار لعدة أيام. من أجل تسريع هذه العملية، يتم تنفيذ حساب موازٍ على التطبيق الأساسي [20]، حيث تسمح بنية الشبكات العصبونية الالتفافية بالبرمجة المتوازية التي تقلل من التعقيد الزمني لعمليات الالتفاف والتجميع، كمثال بدلاً من تنفيذ كل عملية ضرب نقطي بشكل تسلسلي يمكن أن تحدث جميع العمليات في نفس الوقت ولا يعتمد أي منهما على نتائج أي حساب آخر حيث تحدث الحسابات بالتوازي على وحدة معالجة الرسومات.

## 5- تدريب الشبكة العصبونية على معالجات عالية الأداء

الخطوات العملية التي سنقوم بها في هذا البحث لتدريب الشبكات العصبونية الالتفافية CNN على معالجات عالية الأداء (معالجات GPU)، تتلخص كما هو مبين في الشكل (8).



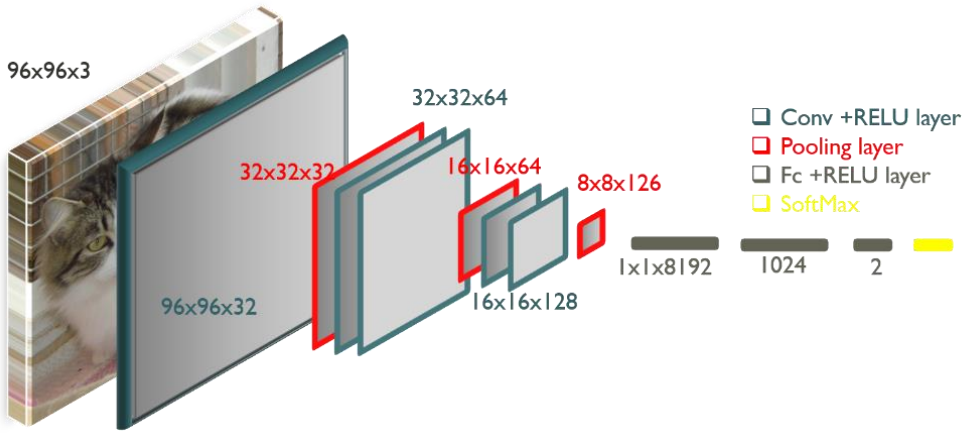

**الشكل (8) آلية عمل الخوارزمية المقترحة**

**تحضير مجموعة البيانات:** تحميل مجموعة بيانات للتدريب وهي صور للقطط والكلاب من موقع Kaggle [21] واخترنا منها 1000 صورة لكل صنف. يوضح الشكل (9) عينة من صور القطط والكلاب المستخدمة للتدريب والتحقق.

**الشكل (9) عينة من صور القطط والكلاب المستخدمة للتدريب والتحقق**

**معالجة مجموعة البيانات:** نحتاج إلى معالجة مسبقة للصور فيتم ضبط حجمها لتلائم النموذج، يتم تحويل التسميات إلى متجهات ترميز، وتقسيم البيانات إلى مجموعات تدريب وتحقق بنسبة 80% للتدريب و20% للتحقق، وزيادة للبيانات من البيانات الموجودة أثناء التدريب.

**بناء الشبكة العصبونية الالتفافية:** تظهر بنية الشبكة العصبية الالتفافية المعتمدة في التطبيق وهي نسخة مصغرة من VGG في الشكل (10). تتكون هذه الشبكة من الطبقات الآتية:




**الشكل (10) بنية الشبكة العصبية الالتفافية المصغرة من شبكة VGG16 المستخدمة في التدريب**

تمت تهيئة النموذج بالمحسن المراد استخدامه وهو Adam، وبمعدل تعلم $1e^3$ وهو معدل التعلم الابتدائي للمحسن Adam، وعدد فترات التدريب Epochs يساوي 100 فترة تدريب، وتابع الخطأ وهو Binary cross entropy.

يتم تنفيذ الشبكة المقترحة باستخدام إطار العمل TensorFlow الذي يستخدم وحدة معالجة الرسومات من NVIDIA في حال توافرها بشكل افتراضي، ويمكن أن نجعل التدريب على وحدة المعالجة المركزية فقط.

في التجربة قمنا بمقارنة الدقة والخطأ والزمن عند تدريب الشبكة العصبونية المعالجات الآتية:

- وحدة معالجة مركزية واحدة Intel Core i5
- معالج NVIDIA GeForce 930M
- معالج Tesla T4 من Google Colab
- معالج NVIDIA GeForce GTX 1050

نوضح في الجدول (2) خصائص هذه المعالجات.

**جدول (2) خصائص المعالجات المستخدمة في تدريب التطبيق المقترح**

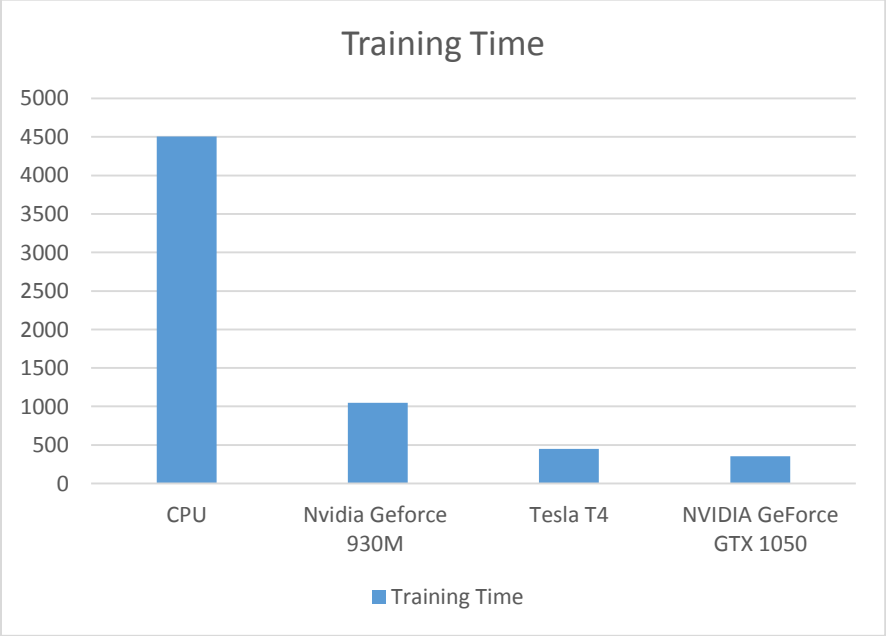
| المعالج | Intel Core (i5-6200U) | NVIDIA GeForce 930M | Tesla T4 | NVIDIA GeForce GTX 1050 |
|---|---|---|---|---|
| RAM | 16 GB | 16 GB | 16 GB | 16 GB |
| Cores | 2 | 384 | 2560 | 640 |
| نظام التشغ | Windows10 | Windows10 | Windows10 | Windows10 |
| CUDA | لا يوجد | 10.1 | 10.1 | 10.1 |

بالتدريب على CPU كان الزمن المستغرق كبيراً جداً بالنسبة للشبكة المصغرة من Vgg16. عند التدريب بشكل متوازي على معالج NVIDIA GeForce 930M، NVIDIA GeForce GTX 1050، Tesla T4 حصلنا على النتائج الموضحة في الجدول(3) .

**جدول (3) نتائج التدريب للشبكة المقترحة على معالجات مختلفة**

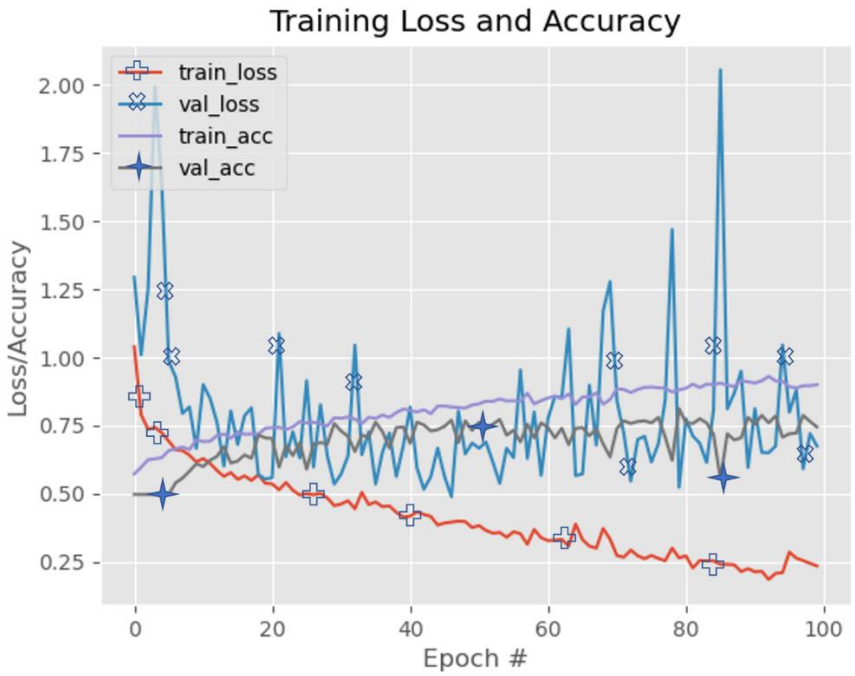
| المعالج | النتائج | زمن التدريب بالدقائق |
|---|---|---|
| Intel Core (i5-6200U) | loss: 0.2351 / accuracy: 0.9000 | 75.11 |
| NVIDIA GeForce 930M | loss: 0.1941 / accuracy: 0.9229 | 17.44 |
| Tesla T4 | loss: 0.1757 / accuracy: 0.9233 | 7.46 |
| NVIDIA GeForce GTX 1050 | loss: 0.1678 /accuracy: 0.9323 | 5.86 |

يبين الشكل (11) مخطط بياني لتوضيح الفرق بين الزمن اللازم لتنفيذ شبكة CNN على معالجات مختلفة.



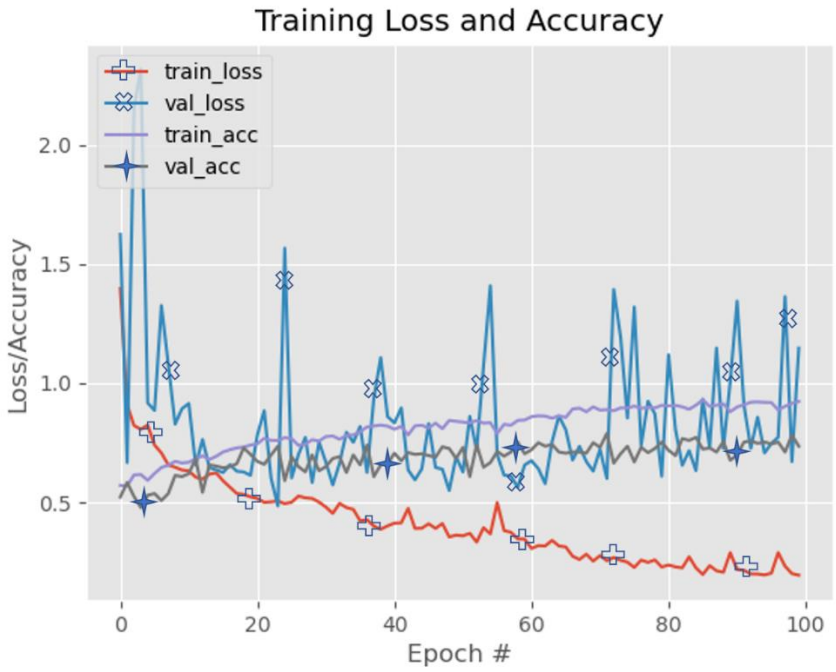

**الشكل(11) مخطط بياني يوضح الفرق بين الزمن اللازم لتنفيذ شبكة CNN على معالجات مختلفة**

توضح الأشكال التالية (12)،(13) ، و(14) منحنيات الدقة والخطأ عند التدريب على أجهزة مختلفة ( Intel Core (i5-6200U ، NVIDIA GeForce 930M، NVIDIA GeForce GTX 1050، Tesla T4) لنجد أن قيم الدقة والخطأ في الشبكة متقاربة من بعضها بعض في نهاية التدريب.



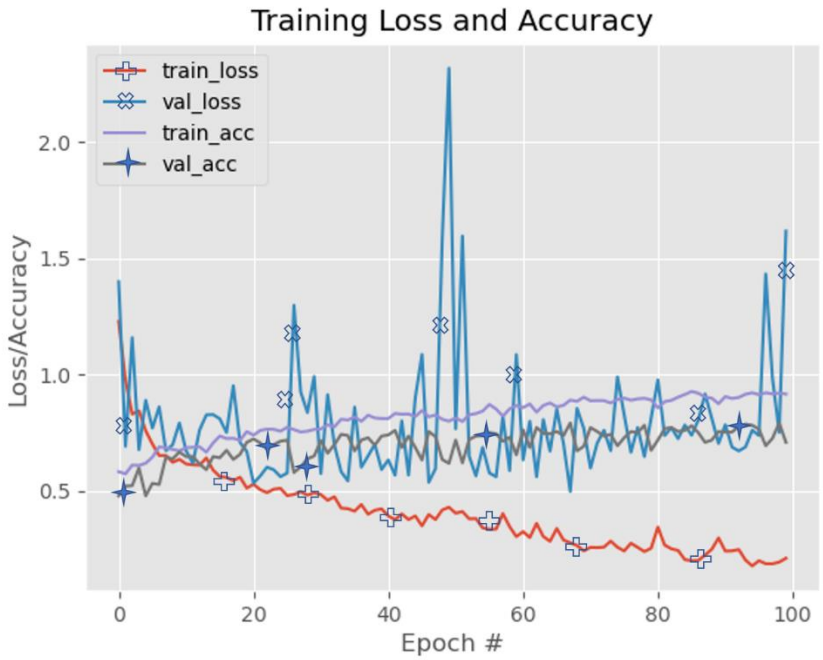

**الشكل (12) التدريب على معالج CPU**

نلاحظ من الشكل (12) أنه تم تدريب النموذج على معالج CPU (i5-6200U) Intel Core على مدى 100 فترة تدريب وحققت خسارة منخفضة 0.23 وبلغت الدقة عند التدريب 0.90. تعتبر نتيجة مرضية ولكن زمن التدريب كبير.




**الشكل (13) التدريب على معالج Nvidia GeForce 930M**

نلاحظ من الشكل (13) أنه تم تدريب النموذج على معالج Nvidia GeForce 930M على مدى 100 فترة تدريب وحققت خسارة منخفضة 0.19 وبلغت الدقة عند التدريب 0.92. تعتبر نتيجة جيدة من حيث الدقة والخسارة بالمقارنة مع معالج CPU لوحده إضافة إلى زمن التدريب أقل بكثير.




**الشكل (14) التدريب على معالج NVIDIA GeForce GTX 1050**

نلاحظ من الشكل (14) أنه تم تدريب النموذج على معالج NVIDIA GeForce GTX 1050 على مدى 100 فترة تدريب وحققت خسارة منخفضة 0.16 وبلغت الدقة عند التدريب 0.92. تعتبر نتيجة جيدة من حيث الدقة والخسارة بالمقارنة مع المعالجين السابقين إضافة إلى زمن التدريب أقل بكثير منهما.

## 6. النتائج والتوصيات:

تتناول هذه المقالة تطبيق شبكة Vgg16 بنسخة مصغرة لتصنيف فئتين من الصور هما القطط والكلاب، تتم المعالجة المسبقة للصور وتهيئة الشبكة بمعدل التعلم يساوي $1e^3$ ، وتابع الخطأ، وعدد فترات التدريب، والأمثلة، وتحويل التسميات إلى متجهات ترميز، وتقسيم البيانات إلى مجموعات تدريب وتحقق، وزيادة للبيانات من البيانات الموجودة أثناء التدريب.

تم تنفيذ التدريب المقترح باستخدام إطار عمل TensorFlow. بعدئذ يتم تدريب الشبكة بشكل تسلسلي على وحدة المعالجة المركزية فقط، ثم قارَنا النتيجة مع التدريب بشكل متوازي على المعالجات NVIDIA GeForce 930M، NVIDIA GeForce GTX 1050، Tesla T4. وكان أفضل معالج أعطى نتائج فعالة في تقليل الزمن هو NVIDIA GeForce GTX 1050 حيث استغرق 5.86 دقيقة مع دقة تصل إلى 93.2%. وعند اختبار تصنيف الشبكة لصور غير موجودة في مجموعة بينات التدريب حصلنا على النتائج التالية الموضحة بالشكل (15)، تظهر الدقة ضمن كل صورة.

بالرغم من ان عدد الأنوية في المعالج NVIDIA GeForce GTX 1050 (640 نواة) أقل من عدد الأنوية في المعالج Tesla T4 (2560 نواة)، يمكن تعليل هذه النتيجة للسببين الاتيتين:

(1) – إن توزيع الكتل blocks والنياسيب threads يؤثر بشكل أساسي على الأداء، ففي المعالج NVIDIA GeForce GTX 1050 يتم توزيع الكتل والنياسيب بشكل أمثلي لتوزيع المهام التي تنفذ بشكل متوازي.

(2) – إن معالج Tesla T4 متاح من قبل شركة Google عبر شبكة الانترنيت، وبالتالي هناك أزمان إضافية تتعلق بالاتصال لمخدمات هذه الشبكة.

ومن التوصيات التي يمكن العمل عليها مستقبلاً لإغناء البحث، يمكن استخدام بيئة الأنظمة الموزعة distributed systems لتدريب الشبكات العصبونية الالتفافية، كما يمكن استخدام نظام متعدد من وحدة معالجة الرسوم multi–GPU، ومقارنة زمن تدريب الشبكات العصبونية الالتفافية مع المعالجات المستخدمة في هذا البحث.


**الشكل (15) اختبار الشبكة المدربة على مجموعة صور من خارج صور التدريب وتظهر الدقة على كل صورة.**

## 7-المراجع:

1. McCulloch, WS; Pitts, W. (1943), *A logical calculus of the ideas immanent in nervous activity* . Bull Math Biol 1990; 52: 99-115

2. LeCun,Y; Bengio,Y; Hinton,G. (2015), *Deep learning*. Nature 521:436–444.

3. Hubel,H,D; Wiesel, N,T. (1968), *Receptive fields and functional architecture of monkey striate cortex*. Journal of Physiology London, 195:215–243.

4. Fukushima, K. (1980), *Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position*. Biological Cybernetics, 36(4):193–202.

5. Lecun, Y; Bottou, L; Bengio, Y; Haffner, P. (1980), *Gradient-based learning applied to document recognition*. Proceedings of the IEEE, 86(11):2278–2324.

6. Rumelhart, E, D; Hinton, E, G; Williams,J, R.( 1986*), Learning internal representations by error propagation*. In Rumelhart, E, D; Mcclelland, L, J. editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, Volume 1: Foundations, pages 318–362. MIT Press, Cambridge, MA.

7. Krizhevsky, A.; Sutskever, I. ; Hinton, G. E. (2012), *Image net classification with deep convolutional neural networks*. In Proceedings of the 25th International Conference on Neural Information Processing Systems. 1097–1105 (ACM, LakeTahoe, Nevada).

8. Sultana, F; Sufian, A; Dutta, P. (2018), *Advancements in image classification using convolutional neural network*. In 2018 Fourth International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN), pages 122–129.

9. Marques, J; Falcao, G; Alexandre, A, L. (2017), *Distributed learning of CNNs on heterogeneous CPU/GPU architectures*. arXiv:1712.02546v1, pages 33.

10. Shustanova, A; Yakimova, P. (2017), *CNN Design for Real-Time Traffic Sign Recognition*. Procedia Engineering 201, 718–725.

11. Russakovsky, O; Deng, J; Su, H, et al. (2015), *ImageNet Large Scale Visual Recognition Challenge*. International Journal of Computer Vision,115:211–252.

12. Alzubaidi, L; Zhang, J; Humaidi, J, A; Al-Dujaili, A; Duan, Y; Al-Shamma, O; Santamaría, J; Fadhel, A, M; Al-Amidie, M; Farhan, L. (2021), *Review of deep learning: concepts, CNN architectures, challenges, applications, future directions*. Journal of Big Data volume 8,53.

13. Simonyan, K; Zisserman, A. (2014), *Very Deep Convolutional Networks for Large-Scale Image Recognition*. arXiv:1409.1556, pages 14.

14. Ghosh, A; Sufian, A; Sultana, F; Chakrabarti, A; De, D. (2020), Fundamental Concepts of Convolutional Neural Network. Intelligent Systems Reference Library book series (ISRL, volume 172), pages 519-567.

15. Montesinos-López, o; Montesinos, A; Crossa, J. (2022), *Convolutional Neural Networks. Multivariate Statistical Machine Learning Methods for Genomic Prediction book*, pages 533-577.

16. Yamashita, R; Nishio, M ; Gian Do ,K,R ;Togashi,R.(2018), *Convolutional neural networks: an overview and application in radiology*. Insights into Imaging, 9:611–629.

17. Chetlur,S; Woolley,C; Vandermersch,P; Cohen,J; Tran,J; Catanzaro,B; Shelhamer,E. (2014), *cudnn: Efficient primitives for deep learning*. [Online]. Available: http://arxiv.org/abs/1410.0759.

18. Kim, H; Nam, H; Jung, W; Lee, J. (2017), Performance Analysis of CNN Frameworks for GPUs. 2017 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS), Seoul National University, Korea, pages 10.

19. Aliady,H; Utari,T,D. (2018), *GPU Based Image Classification using Convolutional Neural Network Chicken Dishes Classification.* Int. J. Advance Soft Compu. Appl, Vol 10, No 2, pages 13.

20. Cengil,E; Çinar,A; Zafer Güler,Z. (2017), *A GPU-based convolutional neural network approach for image classification*. 2017 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS), Malatya, Turkey, pages 5.

21. https://www.kaggle.com/c/dogs-vs-cats