VERY DEEP CONVOLUTIONAL NETWORKS FORLARGE-SCALEIMAGERECOGNITION

備考

著者

Karen Simonyan, Andrew Zisserman

掲載

"Very Deep Convolutional Networks For Large-ScaleImageRecognition," arXiv:1409.1556[cs.CV]，2014.

Abstract

本研究では、大規模画像認識における畳み込みネットワークの深さが精度に与える影響を調べる。私たちの主な貢献は、非常に小さな (3×3) 畳み込みフィルタを備えたアーキテクチャを用いてネットワークのた深さを増した場合の徹底的な評価であり、深さを 16 〜 19 の重み層に押し上げることで、従来の構成に比べて大幅な改善が達成できることを示しています。これらの調査結果は、ImageNet Challenge 2014 に参加した際に得られたもので、我々のチームはローカリゼーションおよび分類のトラックでそれぞれ 1 位と 2 位を獲得しました。また、私たちの表現が他のデータセットでもうまく一般化され、最先端の結果が得られることも示しています。コンピュータビジョンにおける深層視覚表現の使用に関する研究を促進するために、最高の性能を持つ 2 つの ConvNet モデルを公開しました。

1. Introduction

近年、コンボリューションネットワーク (ConvNets) は、ImageNet のような大規模な公開画像リポジトリ (Deng ら、2009 年) や、GPU や大規模な分散クラスタ (Dean ら、2012 年) のような高性能なコンピューティングシステムによって可能になった大規模な画像・映像認識 (Krizhevsky ら、2012 年, Zeiler & Fergus、2013 年, Sermanet ら、2014 年, Simonyan & Zisserman、2014 年) において大きな成功を収めています。特に、深層視覚認識アーキテクチャの進歩において重要な役割を果たしてきたのは、ImageNet の大規模視覚認識チャレンジ (ILSVRC) (Russakovsky ら、2014) です。このコンペティションでは，高次元の浅い特徴符号化 (Perronnin ら、2010) (ILSVRC-2011 の優勝者) から Deep ConvNet (Krizhevsky ら。、2012) (ILSVRC-2012 の優勝者) まで、数世代にわたる大規模画像分類システムのテストベッドとして機能してきた。

ConvNets がコンピュータービジョンの分野でより一般的なものになるにつれて、より良い精度を達成するために Krizhevsky ら (2012) のオリジナルアーキテクチャを改良する試みが数多く行われてきました。例えば、ILSVRC-2013 (Zeiler＆Fergus、2013, Sermanet ら、2014) への最も優れた提出では、より小さい受容ウィンドウサイズ (＝フィルタサイズ) と最初の畳み込み層より小さいストライドを利用しました。もう一つの改善点は、画像全体と複数のスケールにわたってネットワークを密にトレーニングおよびテストしました (Sermanet ら、2014, Howard、2014) 。本論文では、ConvNet アーキテクチャ設計のもう一つの重要な側面である深さについて説明します。この目的のために、我々はアーキテクチャの他のパラメータを固定し、よ

り多くの畳み込み層を追加することによってネットワークの深さを着実に増やします。これは、すべての層で非常に小さい (3×3) たたみ込みフィルターを使用することで実現可能です。

その結果、我々はより精度の高い ConvNet アーキテクチャを開発しました。これは、ILSVRC 分類および ローカリゼーションタスクで最先端の精度を実現しただけでなく、他の画像認識データセットにも適用可能であり、比較的単純なパイプラインの一部として使用した場合でも優れた性能を達成しました (例えば、微調整なしで線形 SVM によって分類された深部特徴など) 。私たちは、さらなる研究を促進するために、2 つの最も優れた性能を持つモデル 1 を公開しました。

2. ConvNet Configurations

公平な環境で ConvNet の深さの増加による改善を測定するために、Ciresan ら (2011) 、Krizhevsky ら (2012) に触発された同じ原理を用いて ConvNet 層の構成を設計しました。本節では、まず私たちの ConvNet の一般的なレイアウト (2.1 節) について説明し、次に評価で使用される特定の構成の詳細を (2.2 節) で説明します。続いて、2.3 節では、我々の設計の選択について議論し、先行技術との比較を行う。

2.1. Architecture

トレーニング中、ConvNet への入力は、固定サイズの 224×224 RGB 画像です。私たちが実行する唯一の前処理は、トレーニングセットで計算された平均 RGB 値を各ピクセルから減算することです。画像は、3×3 (左右、上下、中央の概念を捉えるための最小サイズ) という非常に小さな畳み込みフィルターを使用した畳み込み (conv.) レイヤーを通過します。演算の 1 つには、1×1 の畳み込みフィルターも使用します。これは、入力チャネルの線形変換 (非線形性が後に続く) と見なすことができます。畳み込みストライドは 1 ピクセルに固定されており、畳み込み層の入力の空間パディングは、畳み込み演算後も空間分解能が維持されるようになっています。つまり、パディングは 3×3 畳み込み演算に対して 1 ピクセルです。空間プーリングは、5 つの max プーリングレイヤーによって実行されます。これらは、いくつかの畳み込み層の後に続きます (すべての畳み込み層の後に max-pooling 層があるわけではない) 。max プーリングは、2×2 ピクセルのウィンドウ上で、ストライド 2 で実行されます。

畳み込み層 (アーキテクチャによって深さが異なる) の演算の後に、3 つの全結合 (FC) 層が続きます。最初の 2 つのレイヤはそれぞれ 4096 チャネルを持ち、3 番目のレイヤは 1000 クラスの 画像分類を実行するため、1000 チャネル (各クラスに 1 つ) を含みます。最後の層はソフトマックス層です。全結合層の構成は、すべてのネットワークで同じです。

すべての隠れ層は、非線形性(ReLU(Krizhevsky ら、2012))の補正機能が備わっています。私たちのネットワーク (1 つを除く) にはローカル応答正規化 (LRN) が含まれていないことに注意してください (Krizhevsky ら、2012) 。図 4 に示すように、このような正規化は ILSVRC データセットの性能を向上させず、メモリ消費と計算時間の増加につながります。該当する場合、LRN 層のパラメーターは (Krizhevsky ら、2012) のパラメーターです。

2.2. Configurations

本論文で評価した ConvNet の構成を表 1 に列ごとに 1 つずつ示します。以下では、ネットの名称を(A-E)と呼ぶことにします。すべての構成は第 2.1 節で示した一般的な設計を踏襲しており、深さだけが違います. 具体的には，ネットワークＡの 11 個の重み層 (８つの畳み込み層と３つの FC 層) からネットワークＥの 19 個の重み層 (16 個の畳み込み層と 3 つの FC 層) まであります．畳み込み層の幅 (チャネル数) はかなり小さく、最初の層の 64 から始まり、最大プーリングレイヤーの後は 2 倍ずつ 512 に達するまで増加します。

表 1：ConvNet 構成 (列に表示)

レイヤーが追加されると、構成の深さが左 (A) から右 (E) に増加します (追加されたレイヤーは太字で示されています) 。畳み込み層パラメーターは、「(畳み込みフィルタのサイズ)-(チャネルの数)」として示されます。ReLU 活性化関数は簡潔にするために表示されていません。

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

表 2 では、各構成のパラメーターの数を報告しています。深度が大きいにも関わらず、ネットのウェイトの数は、変換が大きく、浅いネットのウェイトの数よりも多くありません。層の幅と受容野((Sermanet ら、2014)で 144M 個の重み)。

表 2：パラメータの数 (百万単位)

Network	A,A-LRN	B	C	D	E
Number of parameters	133	133	134	138	144

2.3. Discussion

私たちの ConvNet の構成は、ILSVRC-2012 (Krizhevsky ら、2012) や ILSVRC-2013 の大会の上位入賞作品で使用されたもの (Zeiler & Fergus、2013, Sermanet ら、2014) とは大きく異なっています。 最初の畳み込み層で比較的大きな畳み込みフィルタを使用するのではなく、2 つ目の畳み込み層で比較的大きな畳み込みフィルタサイズを持つ畳み込み演算を行う．層ではなく (例えば、(Krizhevsky ら. 2012)でフィルタサイズ 11×11 、ストライド数 4、または(Zeiler & Fergus, 2013, Sermanet ら，2014)フィルタサイズ 7×7 ストライド数 2)、ネット全体で非常に小さな$3×3$の畳み込みフィルタを使用し、すべてのピクセル (ストライド 1 で) で入力と畳み込み演算されます。 フィルタサイズ 3×3 の 2 つの畳み込み層 (間に空間的なプーリングがない) は、 5×5 の畳み込みフィルタサイズを持つ畳み込み層と同じ効果があることが簡単にわかります．同様に、フィルタサイズ 7×7 の代わりに、例えば、フィルタサイズ 3×3 の畳み込み層 3 つを使用することで何が得られるのでしょうか？第一に、単一の層の代わりに 3 つの ReLU を組み込むことで、決定関数をより識別性の高いものにします。第二に、パラメータの数を減らします。3 層の 3×3 畳み込み演算の入力と出力の両方に C チャンネルあると仮定すると、演算は 3 つの $(3^2C^2) = 27C^2$ の重みでパラメータ化されます．同時に、単一のフィルターサイズ 7×7 の畳み込み層は $7^2C^2 = 49C^2$ のパラメータを必要とし、畳み込み層を 3 層重ねる場合と比較してパラメータ数が 81% 多くなります。これは、 7×7 畳み込みフィルタに正則化を課し、 3×3 フィルタ (間に非線形性が挿入されている) を使って強制的に分解させます。

フィルタサイズ 1×1 畳み込み層 (構成 C、表 1) を組み込むことは、畳み込み層のフィルタサイズに影響を与えることなく、決定関数の非線形性を増加させる方法です。私たちの場合、 1×1 畳み込みは、本質的に同じ次元の空間への線形投影であるが (入力チャンネルと出力チャンネルの数は同じ)、ReLU によって追加の非線形性が導入されます。 1×1 畳み込み層は、最近、Lin ら (2014) の「Network in Network」アーキテクチャで利用されていることにも注目すべきです。

小型の畳み込みフィルターは、Ciresan ら (2011) によって既に使用されていますが、彼らのネットは私たちのものよりもかなり浅く、大規模な ILSVRC データセットでの評価を行っていません。Goodfellow ら (2014) は、深い ConvNet (11 重み層) を番地認識のタスクに適用し、深度の増加がパフォーマンスの向上につながることを示しました。 ILSVRC-2014 分類タスクのトップパフォーマンスである GoogleNet (Szegedy ら、2014) は、私たちの研究と独立して開発されましたが、非常に深い ConvNet (22 の重みレイヤー) と小さな畳み込みフィルタ (3×3 は別として、1×1 および 5×5 の畳み込みも使用します) に基づいているという点で類似しています。しかし、この 2 つのネットワークトポロジは私たちのものよりも複雑で、特徴マップの空間解像度は最初のレイヤーでより積極的に削減され、計算量が減少します。4.5 節で示すように、私たちのモデルは単一ネットワーク分類精度の点で Szegedy ら (2014)のモデルを上回っています。

3. Classification framework

前節では、ネットワーク構成の詳細を紹介した。本節では、分類 ConvNet の学習と評価の詳細について説明します。

3.1. Training

ConvNet の学習手順は，一般的に Krizhevsky et al. (2012) に従います (後述するように，マルチスケールの学習画像から入力画像をサンプリングすることを除けば) ． すなわち、訓練は、モーメンタムミニバッチ勾配降下法 (バックプロパゲーション (LeCun ら、1989) に基づく) を用いた多項ロジスティック回帰目的関数を最適化することによって行われます。バッチサイズは 256、モーメンタムは 0.9 に設定された。学習は、重み減衰 (L2 のペナルティ乗数を 5×10^{-4} に設定) と、最初の 2 つの全結合層のドロップアウト正則化 (ドロップアウト率を 0.5 に設定) によって正則化された。学習率は、最初 10^{-2} に設定され、検証セットの精度が向上しなくなると、10 倍で小さくしました．合計 3 回の学習率の低下を行い、370K 回 (74 エポック) の繰り返しで学習を停止した。我々は、(Krizhevsky et al., 2012)と比較して、より多くのパラメータ数と我々のネット

の深さにもかかわらず、(a)より大きな深さとより小さい conv.フィルタサイズによって課される暗黙の正則化、(b)特定の層の事前初期化のために、ネットが収束するのに必要なエポック数がより少ないと推測している。

ネットワーク重みの初期化は重要であり、初期化が悪いとディープネットのグラジエントの不安定性のために学習が停滞する可能性があるからである。この問題を回避するために、我々は、ランダムな初期化で訓練するのに十分な浅い構成 A (表 1) を訓練することから始めた。次に、より深いアーキテクチャを訓練する際には、最初の 4 つの畳み込み層と最後の 3 つの完全連結層をネット A の層で初期化した (中間層はランダムに初期化した) 。事前に初期化された層の学習率を低下させず、学習中に変更できるようにした。ランダムな初期化のために (該当する場合)、平均がゼロで分散が 10-2 の正規分布から重みをサンプリングした。バイアスはゼロで初期化した。論文投稿後、Glorot & Bengio (2010) のランダム初期化手順を使用することで、事前学習なしで重みを初期化できることがわかりました。

ConvNet 入力画像は、トレーニング画像を固定サイズ 224×224 に再スケーリングした後、ランダムにトリミングしました (SGD の反復ごとに 1 画像につき 1 回トリミング) 。 またトレーニングセットをさらに強化するために、画像にランダム水平フリッピングとランダム RGB カラーシフトを施しました (Krizhevsky ら、2012)。 トレーニングイメージの再スケーリングについては、以下で説明します。

トレーニング画像のサイズ. S を等方的に再スケーリングされた学習画像の最小辺とし，そこから ConvNet の入力分だけ切り取ります (S を学習スケールと呼ぶこともあります). $S = 224$ の場合は，訓練画像の最小辺を完全に覆う全画像の統計量をキャプチャし， $S \gg 224$ の場合は，小さな物体や物体の一部を含む画像の小さな部分に対応します.

トレーニングスケール S を設定するための 2 つのアプローチを検討します。1 つ目は、S を固定することです。これは、単一スケールトレーニングに対応します (サンプリングされた画像内の画像コンテンツは、依然としてマルチスケールの画像統計を表すことができることに注意してください) 。私たちの実験では、$S = 256$ (先行技術で広く使用されている (Krizhevsky ら (2012), Zeiler＆Fergus、2013, Sermanet ら、(2014))) と $S = 384$ の 2 つの固定スケールでトレーニングされたモデルを評価しました。ConvNet の構成が与えられると、最初に $S = 256$ を使用してネットワークをトレーニングしました。 $S = 384$ ネットワークのトレーニングを高速化するために、まず $S = 256$ で事前トレーニングされた重みで初期化し、 $S = 256$ と比べて学習率の初期値をより小さな 10^{-3} を使用しました。

S を設定する 2 番目のアプローチはマルチスケールトレーニングです。各トレーニングイメージは、特定の範囲 $[S_{min}, S_{max}]$ からランダムに S をサンプリングすることによって個別に再スケーリングされます($S_{min} = 256$ および $S_{max} = 512$ を使用しました) 。 画像内のオブジェクトはサイズが異なる可能性があるため、トレーニング中にこれを考慮すると効果的です。 これは、スケールのジッタリングによるトレーニングセットの拡張と見なすこともできます。この場合、単一のモデルが広範囲のスケールでオブジェクトを認識するようにトレーニングされます。 速度上の理由から、シングルスケールモデルのすべてのレイヤーを同じ構成で微調整することにより、マルチスケールモデルをトレーニングしました。固定 $S = 384$ で事前トレーニングしました。

3.2. TESTING

テスト時には，訓練された ConvNet と入力画像が与えられ，以下の方法で分類されます．まず，Q と呼ばれる（テストスケールとも呼ぶ）事前に定義された最小画像側に等方的に再スケーリングされる．Q は必ずしも学習スケール S と等しいとは限らないことに注意する (第 4 節で示すように、各SにいくつかのQ値を使用すると、パフォーマンスが向上します。) 次に、ネットワークは、(Sermanet ら, 2014)と同様の方法で、再スケーリングされたテスト画像上に密に適用される。すなわち、全結合層は、最初に畳み込み層に変換される (最初のＦＣ層は7×7の畳み込み層に、最後の２つのＦＣ層は1×1の畳み込み層

に変換される) 。結果として得られた完全畳み込みネットは，(トリミングされていない) 画像全体に適用されます．その結果、クラスの数に等しいチャンネル数と、入力画像サイズに依存した可変空間分解能を持つクラススコアマップが得られる。最後に、画像のクラススコアの固定サイズベクトルを得るために、クラススコアマップは空間的に平均化されます (合計値がプールされます)。元画像と反転画像に変換後のソフトマックスクラスの分布を平均化して、画像の最終スコアを求めます。

完全畳み込みネットワークは画像全体に適用されるため，テスト時に複数の切り取られた画像をサンプリングする必要がなく(Krizhevsky ら., 2012)，切り取られた画像ごとにネットワークの再計算が必要となるため効率が悪い．一方、Szegedy ら(2014)が行ったように、大規模な切り取られた画像のセットを使用すると、完全畳み込みネットに比べて入力画像のサンプリングが細かくなるため、精度の向上につながる可能性があります。また、複数の切り取られた画像を用いた評価は、畳み込み境界条件が異なるため、密な評価を補完するものです。ConvNet を画像の切り取りに適用する場合、畳み込み特徴マップはゼロでパディングされますが、密な評価の場合、同じ切り取り画像のパディングは (畳み込みと空間プーリングの両方に起因する) ネットワーク全体の受容野が大幅に増加するため、より多くのコンテキストがキャプチャされます。実際には、複数の切り取られた画像による計算時間の増加は、精度の潜在的な向上を正当化するものではないと考えていますが、参考のために、我々のネットワークを 3 つのスケールで各スケール 50 枚の切り取られた画像 (2 フリップの5×5 の規則的なグリッドで)を用いて、合計 150 枚の切り取られた画像を評価しています。これは Szegedy ら (2014) が使用した 4 つのスケールで 144 枚の切り取られた画像に匹敵するものです。

3.3. IMPLEMENTATION DETAILS

私たちの実装は、一般に入手可能な C ++ Caffe ツールボックス（Jia、2013）（2013 年 12 月に分岐）から派生していますが、多数の重要な変更が含まれているため、単一のシステムにインストールされている複数の GPU でトレーニングと評価を実行できます。 フルサイズの（トリミングされていない）画像を複数のスケールでトレーニングおよび評価します（上記のとおり）。 マルチ GPU トレーニングは、データの並列処理を利用し、トレーニング画像の各バッチを複数の GPU バッチに分割し、各 GPU で並列に処理することによって実行されます。 GPU バッチ勾配が計算された後、それらは平均化されて、完全なバッチの勾配が取得されます。 勾配計算は GPU 間で同期しているため、結果は単一の GPU でトレーニングした場合とまったく同じです。

ConvNet トレーニングを高速化するより洗練された方法が最近提案され た(Krizhevsky、2014 年)。この方法は、ネットのさまざまなレイヤーにモデルとデータの並列処理を採用していますが、我々の概念的に非常に単純なスキームでは、単一の GPU を使用する場合と比較して，市販の 4GPU システムを使用した場合、すでに 3.75 倍の高速化が得られることがわかった。4 つの NVIDIA Titan Black GPU を搭載したシステムでは、アーキテクチャに応じて、1 つのネットのトレーニングに 2〜3 週間かかりました。

4. CLASSIFICATION EXPERIMENTS

データセット。 このセクションでは、ILSVRC-2012 データセット（ILSVRC 2012–2014 チャレンジに使用された）で説明されている ConvNet アーキテクチャによって達成された画像分類結果を示します。 データセットには 1000 クラスの画像が含まれており、トレーニング（130 万枚の画像）、検証（5 万枚の画像）、テスト（10 万枚の保留されたクラスラベルの画像）の 3 つのセットに分かれています。 分類のパフォーマンスは、2 つのメジャーを使用して評価されます。上位 1 と上位 5 のエラーです。 前者は、マルチクラスの分類エラーです。つまり、誤って分類された画像の割合です。 後者は ILSVRC で使用される主な評価基準であり、グラウンドトゥルースカテゴリが上位 5 つの予測カテゴリの外側になるような画像の比率として計算されます。

ほとんどの実験では、検証セットをテストセットとして使用しました。 特定の実験もテストセットで実行され、ILSVRC-2014 コンテストへの「VGG」チームエントリとして公式 ILSVRC サーバーに送信されました (Russakovsky ら、2014)。