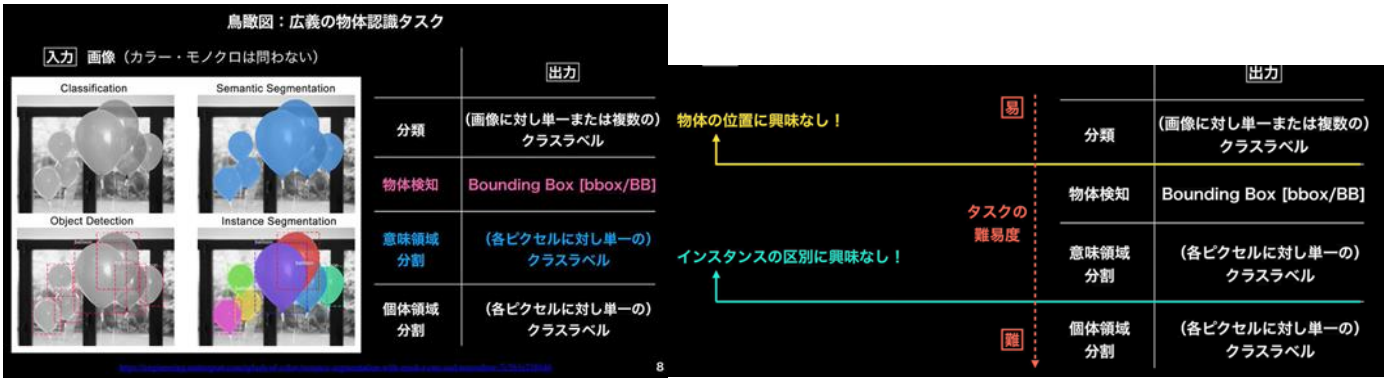
●Section6：物体検知・セグメンテーション



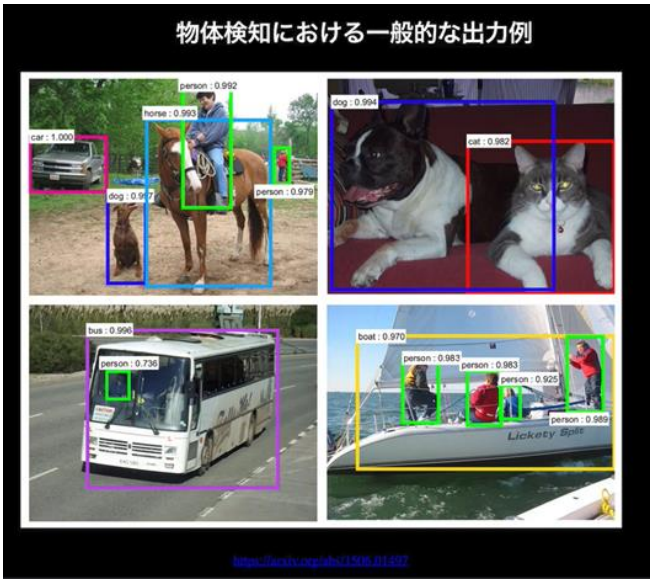
入力は画像。出力は、

・分類：クラスラベル

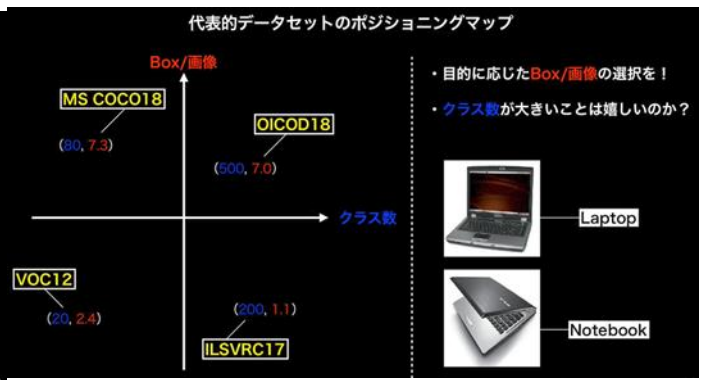
・物体検知：バウンディングボックス

・意味領域分割

・個体領域分割



◇代表的データセット

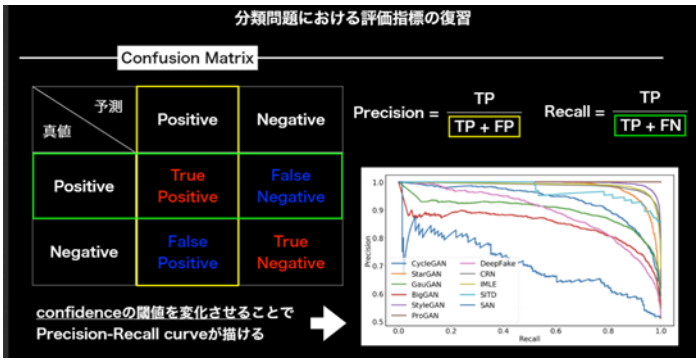


いづれも物体検出コンペティションで用いられたデータセット（誤分類や別名表記もあり批判もある）。

※他には、CIFAR-10,CIFAR-100,Food-101,楽天データ(文字画像)など。

◇評価指標

・混同行列（Confusion Matrix）



・Recall (再現率)：実測が Positive であるもののうち、Positive と予測した割合を示す。

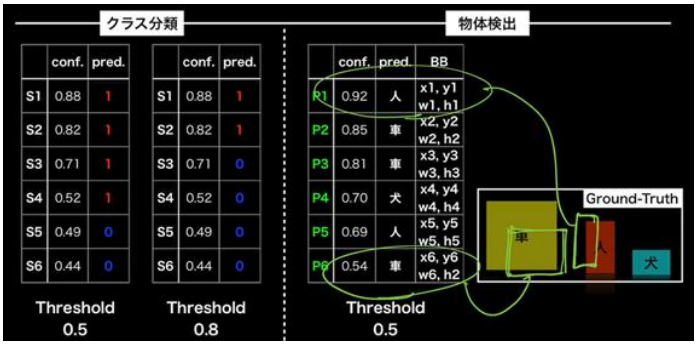
・Precision (適合率)：Positive と予測したもののうち、実測が Positive である割合を示す。

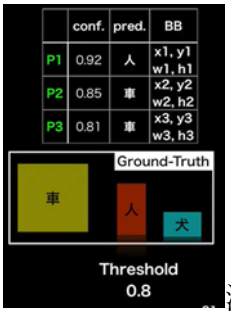
$$※\text{Confidence} = \Pr(\text{Object}) * \text{IoU}^{\text{truth}}_{\text{pred}}$$

前項：$Pr(Object)$は BBox の中に Object（物体）が含まれていたら、その BBox は正確だと判断するためです。（もし BBox の中に Object が存在しなければ Confidence 全体も 0 になります.）

後項：$\text{IoU}^{\text{truth}}_{\text{pred}}$は予測と正解の IoU の値が大きければ正確だと判断するためのものです.

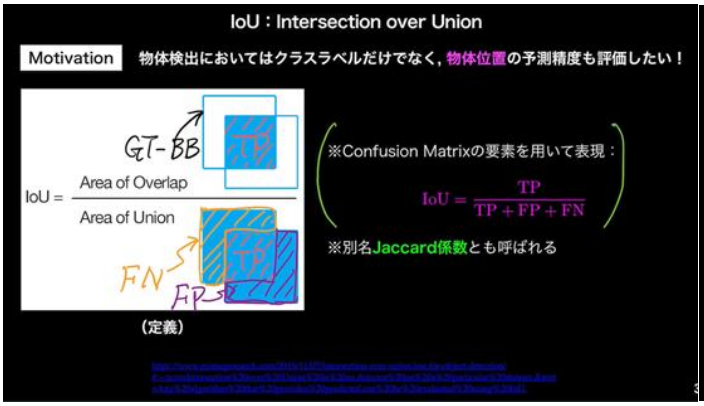
・閾値(Threshold)の変化に対する振る舞い

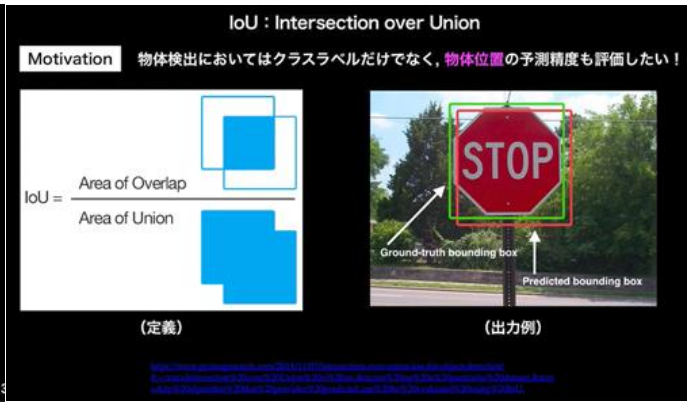




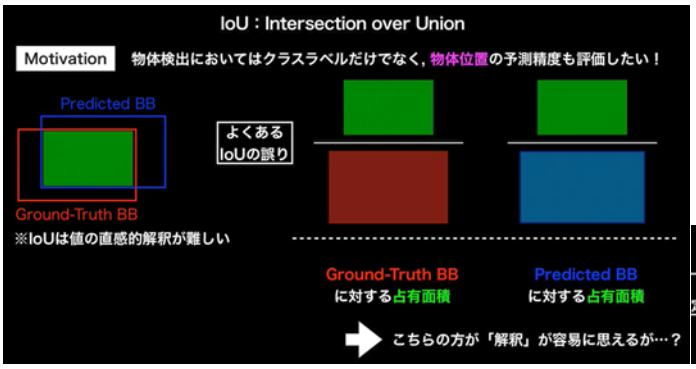
注)閾値により評価数が変化する

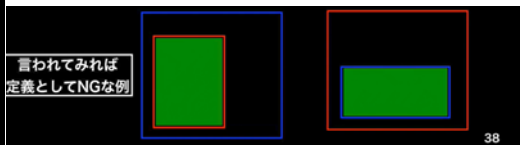
・ラベル分類だけではなく、物体の位置の予測精度評価指標

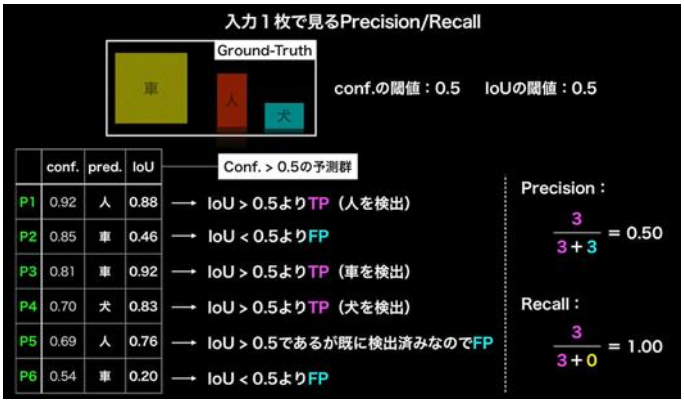


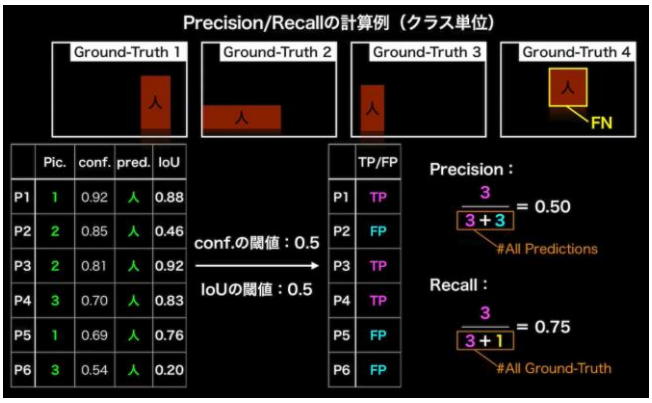


IoU の解釈において、よくある間違い　⇒バウンディングボックス内の占有面積ではない。







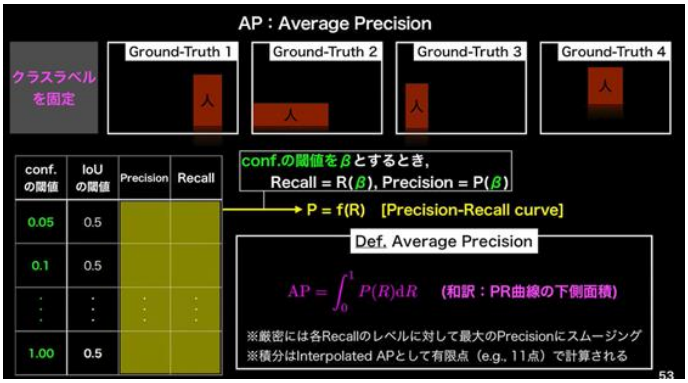


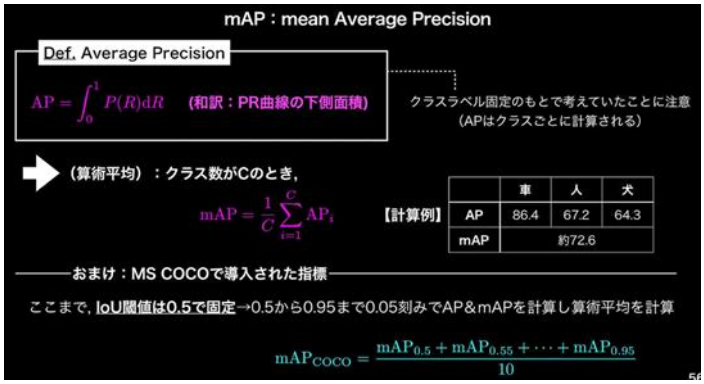
※P5…閾値を超えているが、P1 で既に検出しているので、P5 は FP とされる。
※Recall の+1 分は、検出されなかった Pic4 を指している。

・AP（Average Precision）と mAP（mean Average Precision）

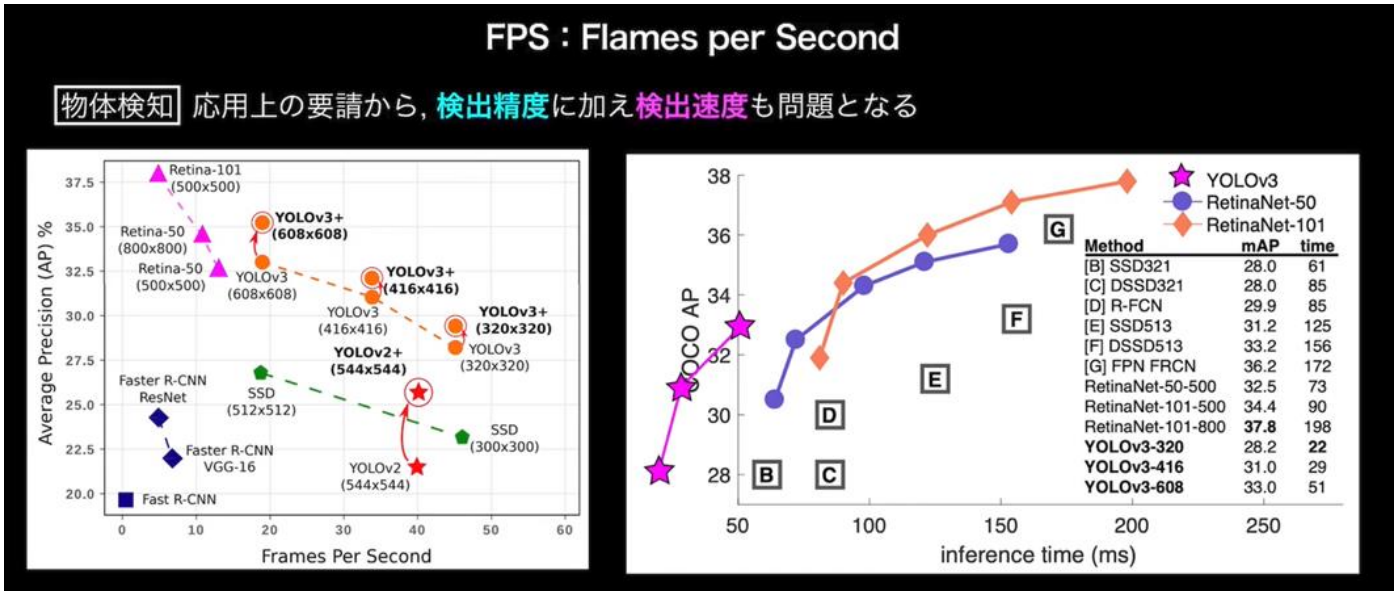
$$AP = \int_0^1 P(R)dR \quad ・・・PR曲線の下側面積$$

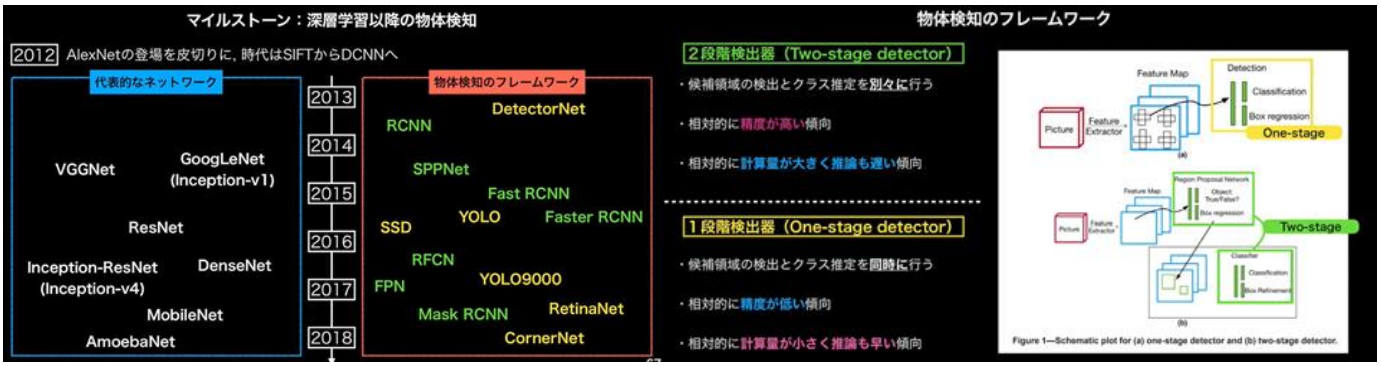
$$mAP = \frac{1}{C}\sum_{i=1}^{C} AP_i$$





・検出速度





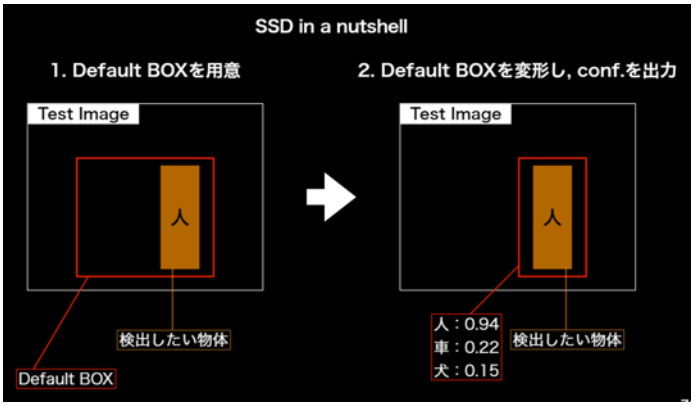
◇SSD（Single Shot Detector）

物体検出の技術を使えば画像中の複数の物体の位置を特定して矩形(バウンディングボックス)で囲み、更にそれぞれの矩形について物体の識別を行うことが可能。

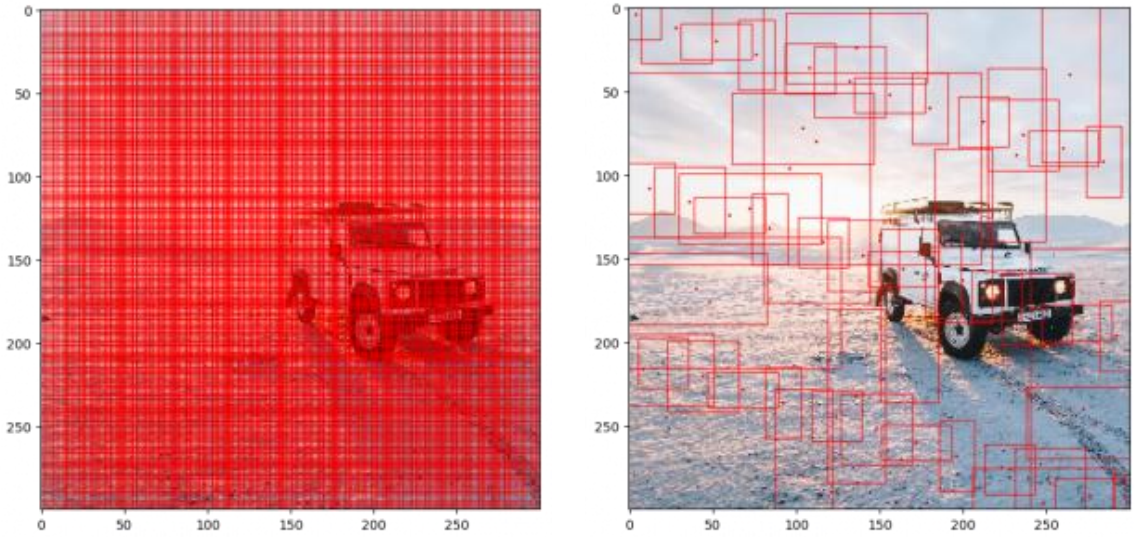
物体検出モデルの歴史は深く R-CNN から始まり Fast R-CNN、Faster R-CNN と精度と処理速度が改善されてきた。

これらの手法は基本的に以下の動画のようにバウンディングボックスを画像内で色々と動かして物体が検出される良い場所を見つけ出そうというもの。

その後、精度と処理速度とともに Faster R-CNN を上回る SSD(Single Shot Multibox Detector)が提案された。



SSD の肝となるのは、「**デフォルトボックス（default boxes）**」という長方形の「枠」です。
一枚の画像を SSD に読ませ、その中のどこに何があるのか予測させるとき、SSD は画像上に大きさや形の異なるデフォルトボックスを **8732 個**乗せ、その枠ごとに予測値を計算します。



左図：8732 個のデフォルトボックスを描画

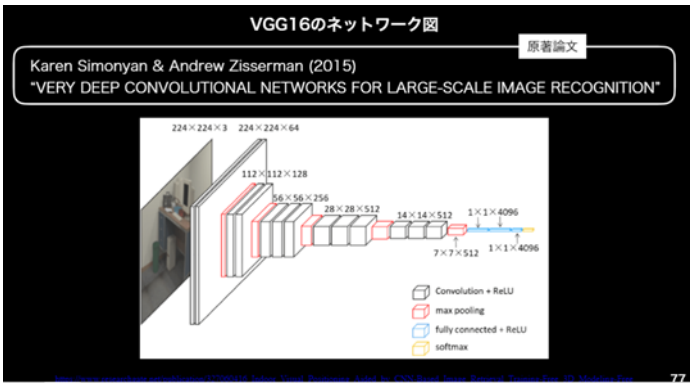
右図：8732 個のデフォルトボックスのうち、数十個をランダムに描画

このデフォルトボックスの役割は、それぞれが、
1. **位置の予測　⇒自身が物体からどのくらい離れていて、どのくらい大きさが異なるのか？**
2. **クラスの予測　⇒そこには何があるのか？**
の 2 つを予測する事です。

参考サイト：https://avinton.com/blog/2018/03/single-shot-multibox-detector-explained1/

SSD は、VGG16 がベースとなっている。

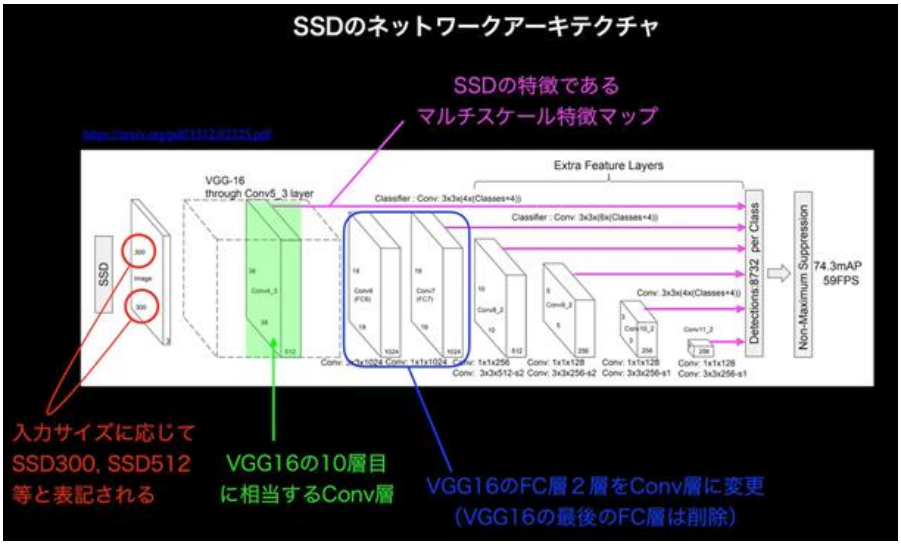
（Maxpooring,Softmax 層をカウントしない 16 層の構造）



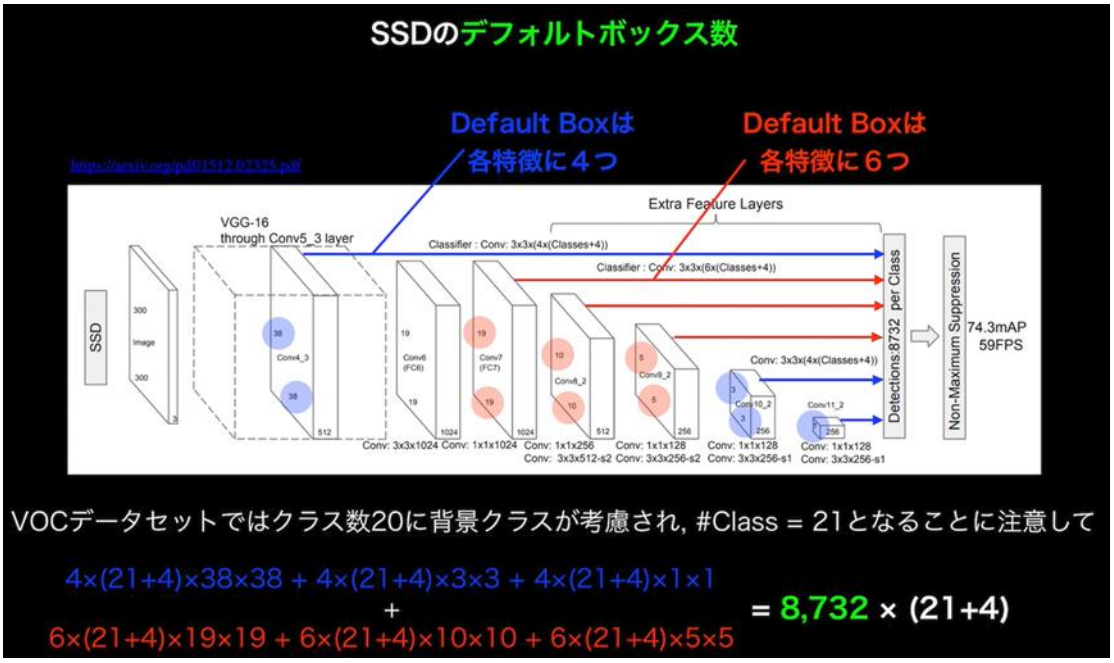
R-CNN ではバウンディングボックスを色々と動かしてそのたびに CNN による演算を行っていたので、１枚の画像から物体検出を行うのにかなりの処理時間がかかっていた・・・。

SSD では"Single Shot"という名前が暗示しているように、１度の CNN 演算で物体の「領域候補検出」と「クラス分類」の両方を行います。これにより物体検出処理の高速化が可能になった。

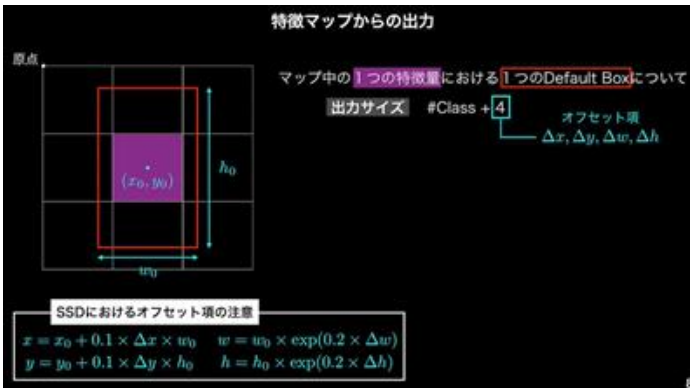
・SSD ネットワーク アーキテクチャ

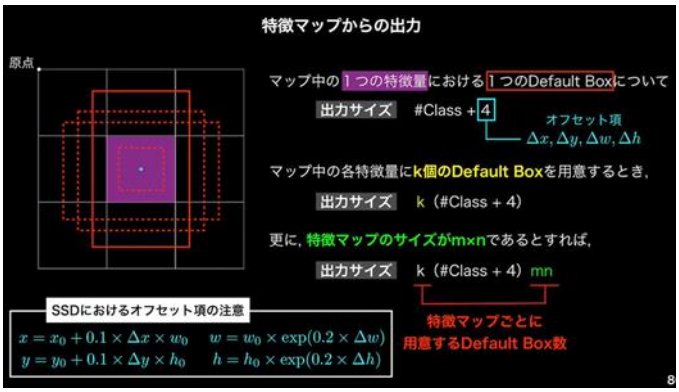


SSD ネットワークは、最初のレイヤー(ベースネットワーク)に画像分類に使用されるモデルを用いている（論文では VGG-16 をベースネットワークとしている）。ベースネットワークの全結合層を切り取り、上のように畳み込み層を追加したものが SSD の構造となる。

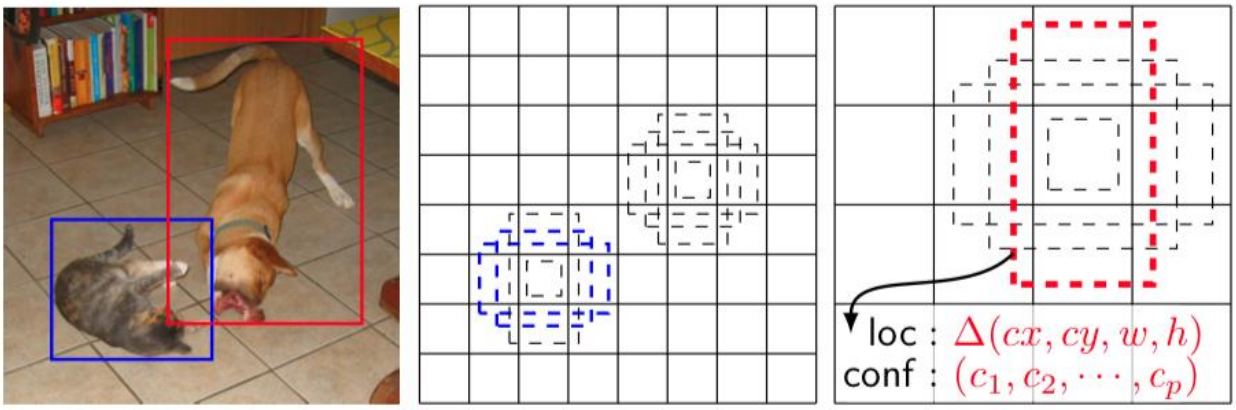


予測の際は、それぞれのレイヤーから特徴マップを抽出して物体検出を行います。具体的にはそれぞれの特徴レイヤーに 3×3 の畳み込みフィルタを適用してクラス特徴と位置特徴を抽出する。





※特徴マップ…線形フィルターで入力画像を畳み込み，バイアス項を加えて非線形フィルタを適用することによって得られる

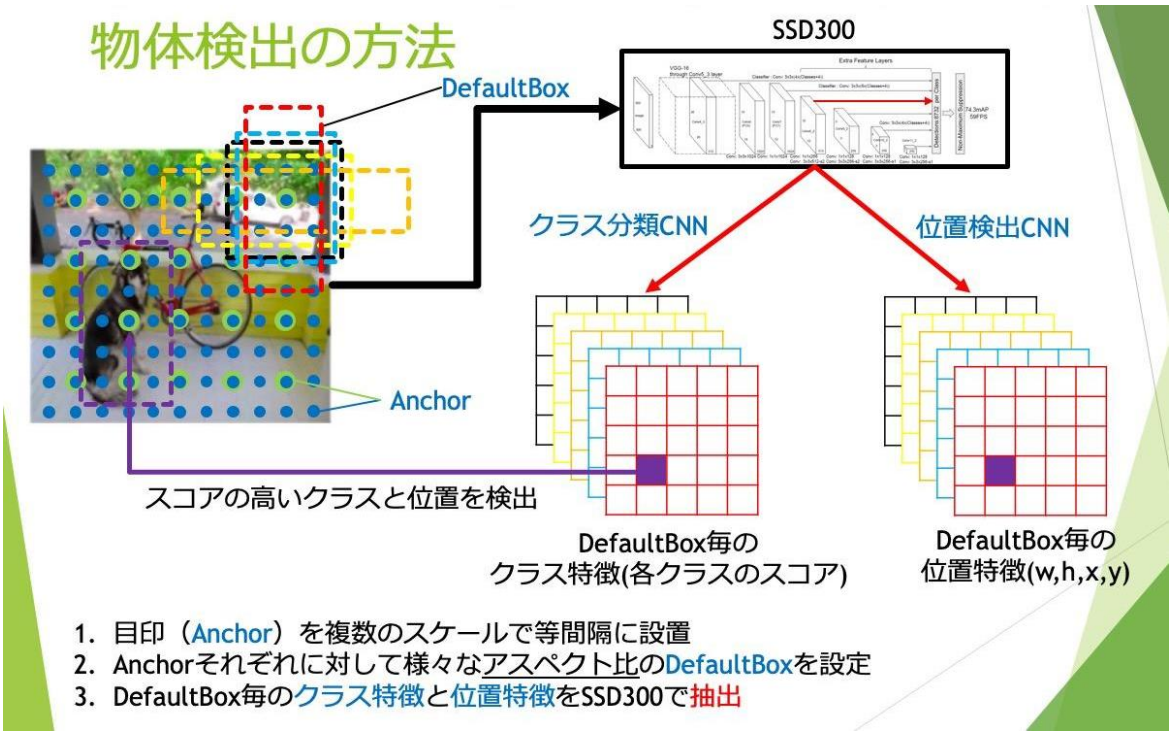


(a) Image with GT boxes (b) 8×8 feature map (c) 4×4 feature map

(a)が入力画像と各物体の正解ボックスです。

(b)と(c)のマス目は特徴マップの位置を表しており、各位置においてデフォルトボックスと呼ばれる異なるアスペクト比の矩形を複数設定します。各位置の各デフォルトボックスについてスコア(confidence)の高いクラスを検出します。

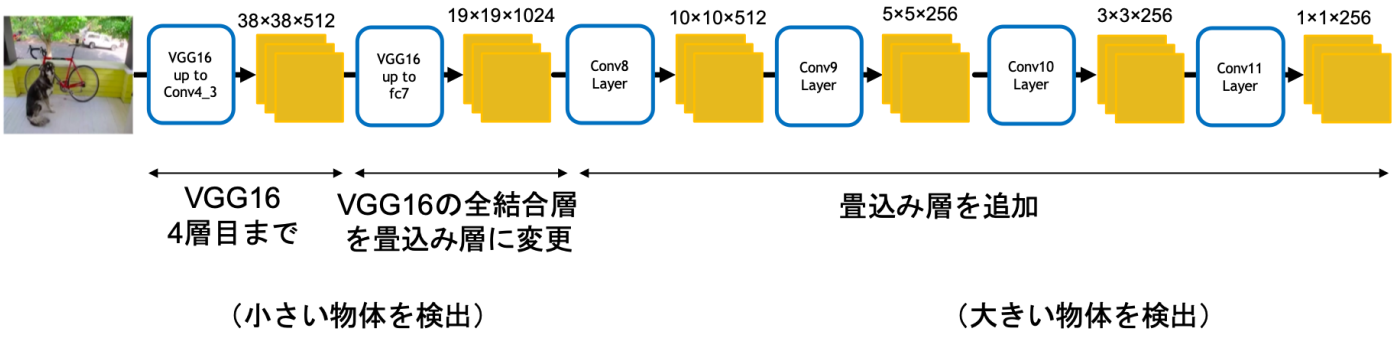
訓練時には各クラスの誤差と、デフォルトボックスと正解ボックスの位置の誤差を元にモデルの学習を行います。



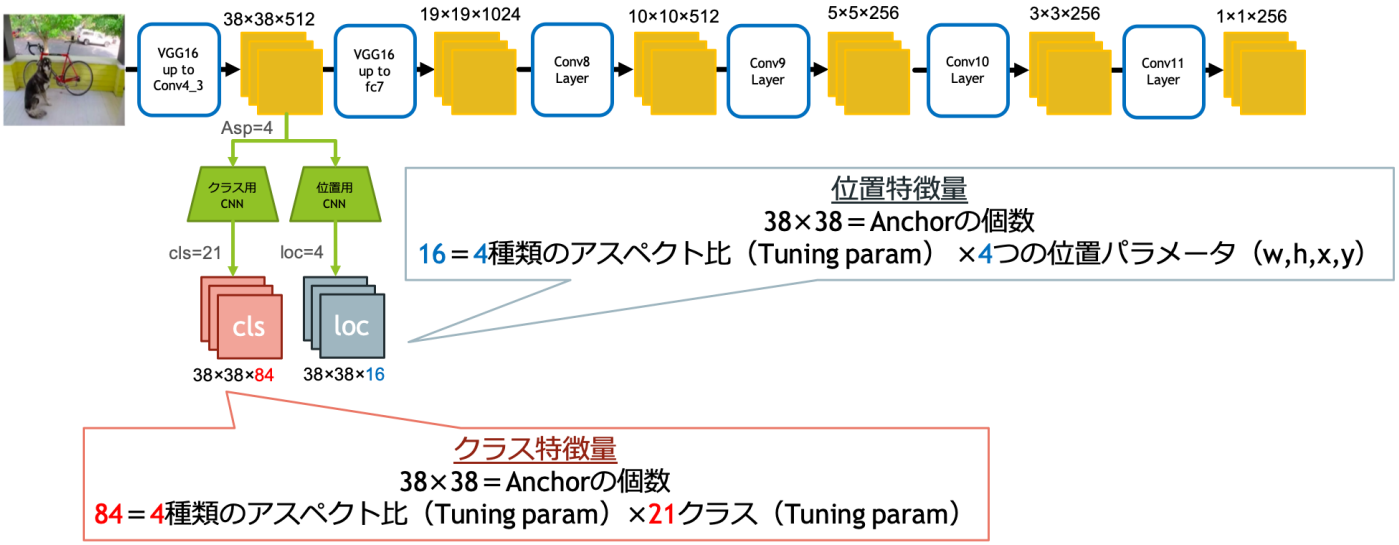
検出の仕組みを示した上の図で(b)と(c)を見ると 4×4 の特徴マップの方がデフォルトボックスが

大きく、各特徴マップではデフォルトボックスの大きさが異なることがわかります。

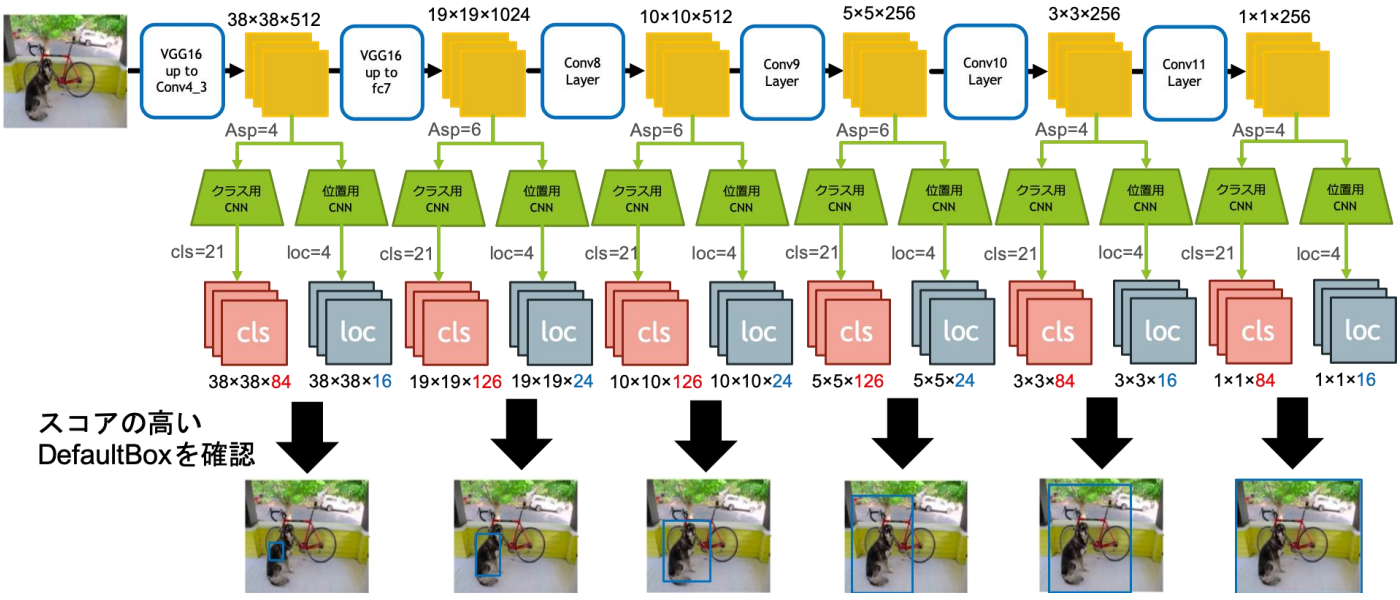
SSD では、サイズの違う畳み込み層をベースネットワークの後に追加することで様々なスケールで物体を検出することができます。



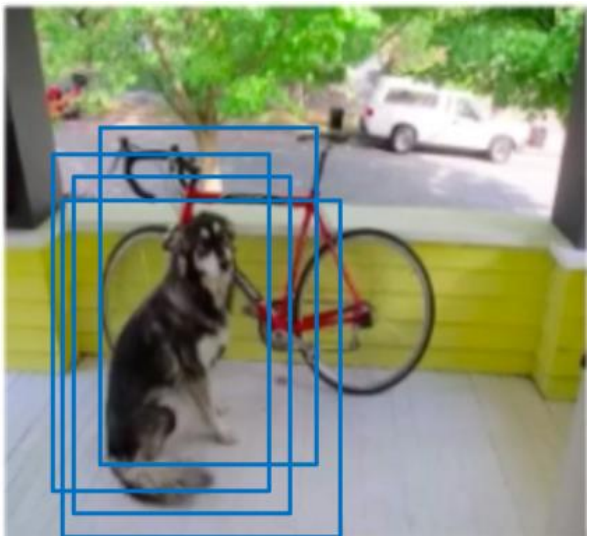
以下のように各特徴マップにおいてクラス特徴量と位置特徴量を算出します。

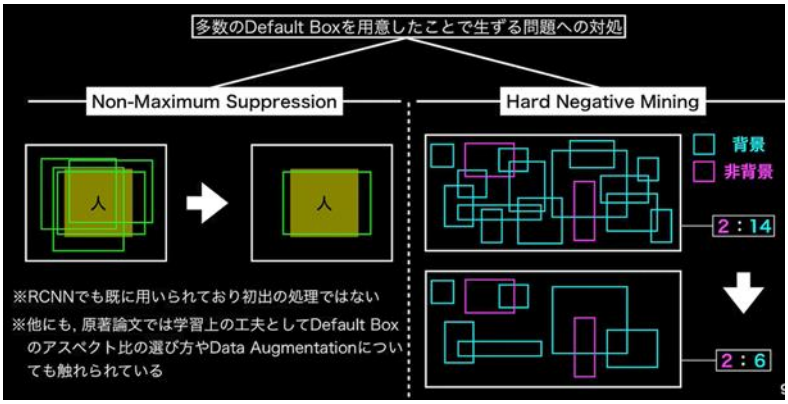


そして各畳み込み層で違うスケールで物体検出を行います。



これにより各層からの検出結果が得られるので次の図のように重複が生じてしまいます。





そこで次のような手順を踏んで重複を除去します。

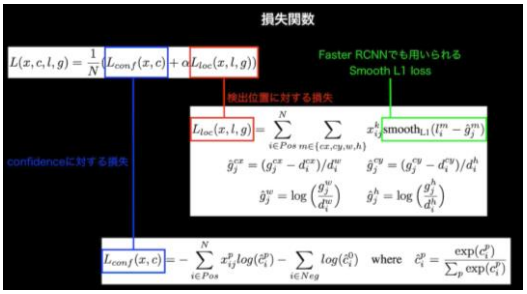
1. スコアが高いデフォルトボックスのみ抽出。
2. 各クラスごとにデフォルトボックスの重なり率 IoU を計算
3. IoU が高い場合はスコアの低いデフォルトボックスを除去

この処理の結果、１つの物体に複数のバウンディングボックスが付与されることを回避できます。

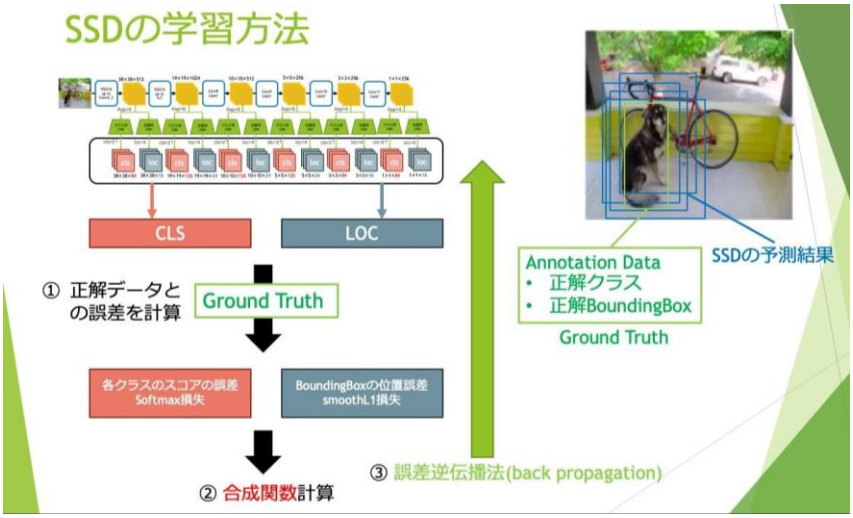
・学習の仕組み

各特徴マップについて各クラスのスコアの誤差と、デフォルトボックスの位置誤差との合成関数から正解データとの誤差を計算します。クラス誤差と位置誤差それぞれの具体的な計算は論文を参照していただきたいですが、最終的な損失関数は 2 つの損失関数を重み付けしたものになります。

$$L(x, c, l, g) = \frac{1}{N}(L_{CLS}(x, c) + \alpha L_{LOC}(x, l, g))$$



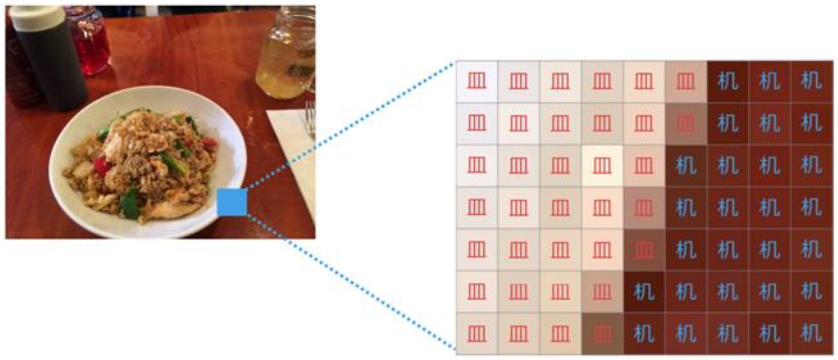
この計算結果を元に誤差逆伝播法によりモデルの重みを更新します。



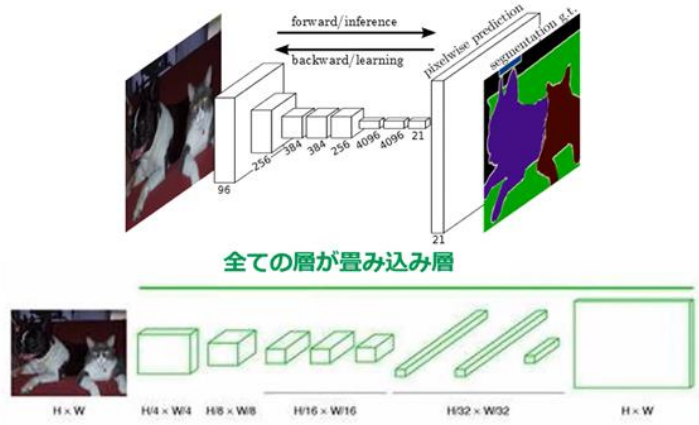
参考サイト：https://www.acceluniverse.com/blog/developers/2020/02/SSD.html

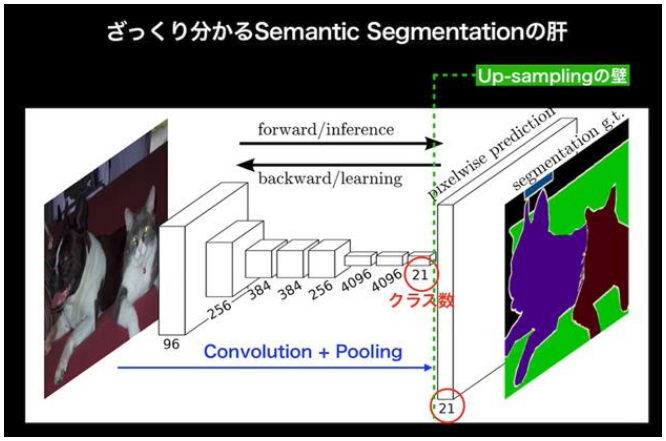
◇Semantic Segmentation

各ピクセルをその意味（周辺のピクセルの情報）に基づいて、カテゴリ分類する手法。矩形領域を切り出すのではなく、ピクセル単位での詳細な領域分割が得られる。



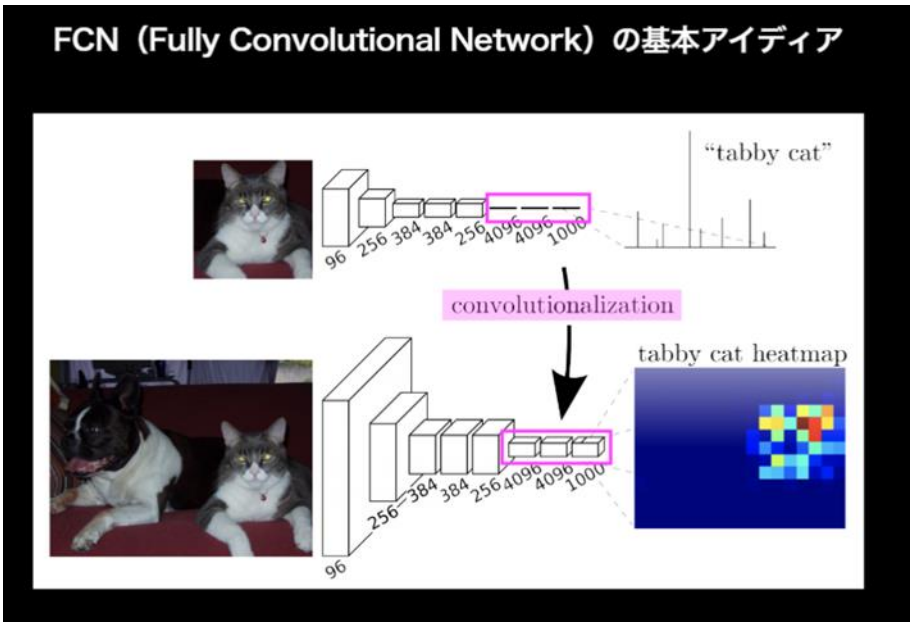
・全畳み込みネットワーク（Fully Convolutional Network:FCN）概略図





一般的な CNN では畳み込み層と全結合層がありますが、FCN はこの全結合層を畳み込み層に置き換えたもので、「物体がなにであるか」という出力から「物体がどこにあるか」という出力になる。

画像上部分が画像全体のクラス分類で、画像下部分はヒートマップで猫がどのあたりにいるかネットワークが把握していることがわかります。



FCN では、入力画像の画素数だけ出力層が必要になる。すなわち各画素が各々どのカテゴリーに属するのか？を出力する為に、出力層には縦画素数×横画素数×カテゴリー数の出力ニューロンを用意する必要がある。

（右図では、識別すべきカテゴリー数が２０＆どのカテゴリーにも属さない（背景）の計２１カテゴリーの分類を行っている。）

・Up-sampling

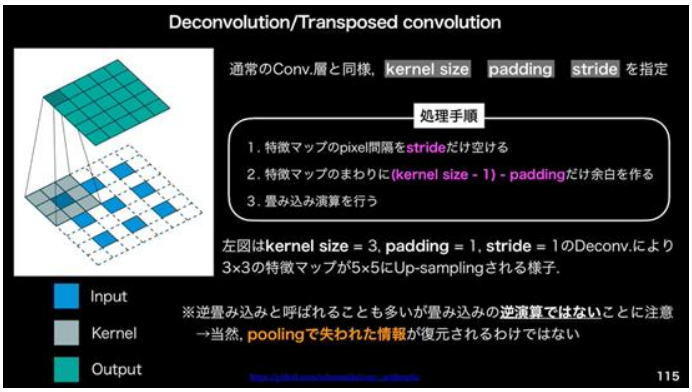
CCN では、畳み込みによって画像サイズが小さくなっていく。

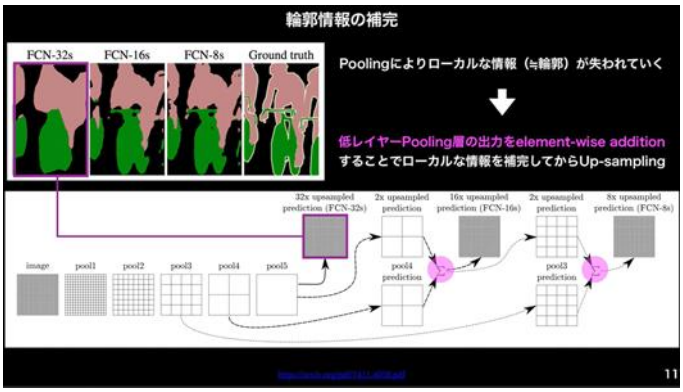
⇒最終出力層に入力層と同じ画素数を得るには、畳み込みと反対方向の解像度を細かくする必要がある。

Up-sampling は、解像度を細かくする方法の 1 つで、畳み込みによって小さくなった画像サイズを、入力画素と等しい解像度の出力を得る事が可能。

Up-sampling の中でも、deconvolution（Transposed convolution）が有名で、逆畳み込み（または転置畳み込み）と呼ばれる。

注）逆畳み込みは、畳み込みの逆演算ではない。

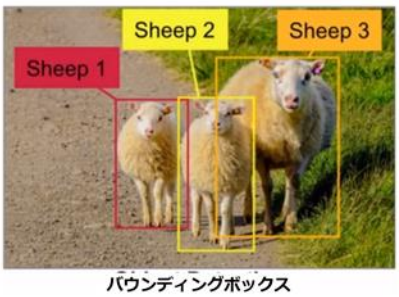




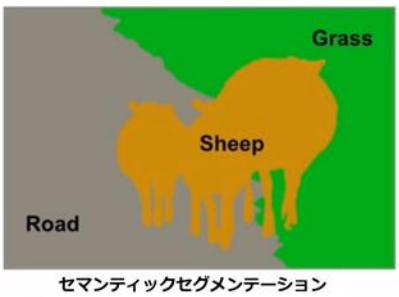
参考サイト：https://github.com/vdumoulin/conv_arithmetic

・インスタンス セグメンテーション

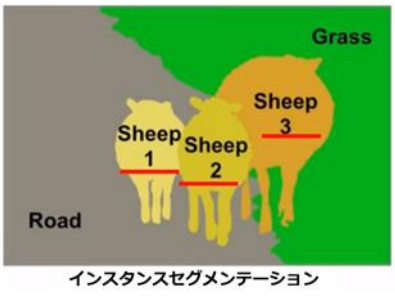
セマンティック セグメンテーションが、各画素がどのカテゴリーに属するか？を求める手法に対し、個々の物体毎に認識させる技術を。インスタンス セグメンテーションと言う。



バウンディングボックス



セマンティックセグメンテーション



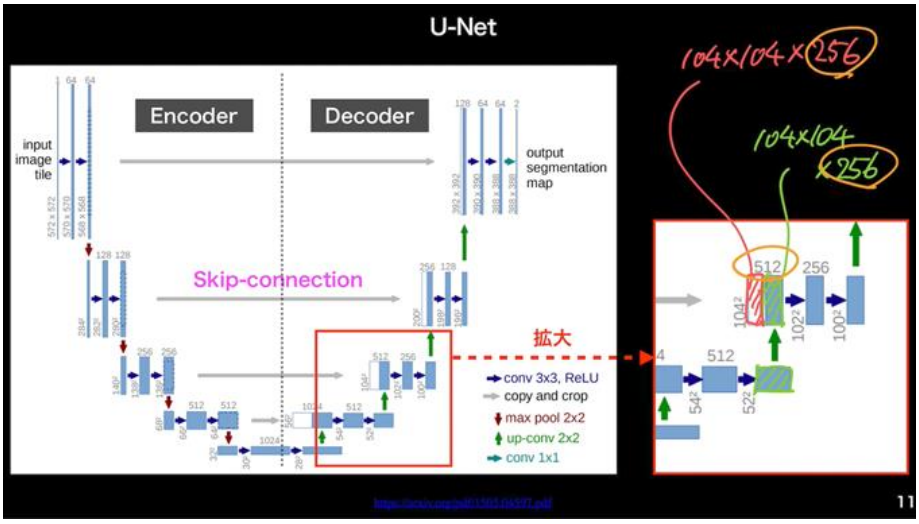
インスタンスセグメンテーション

・U-Net

U-net は FCN(fully convolution network)の 1 つであり、画像のセグメンテーション（物体がどこにあるか）を推定するためのネットワーク。

生物医科学(biomedical）の画像のセグメンテーションを行うために 2015 年に発表された。

参考サイト：https://www.acceluniverse.com/blog/developers/2019/11/u-net.html



fully convolution network(FCN)、deconvolution、skip-conection が使われており、左右対象でアルファベットの「U」に似ていることから、「U-net」と呼ばれている。

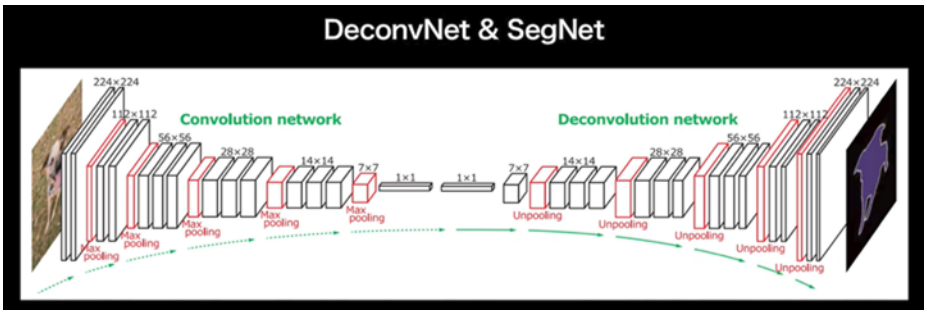
左側の処理では画像の畳み込みを行い、右側では逆畳み込みを skip-conection（中央矢印）からの情報を参考に行なっている。

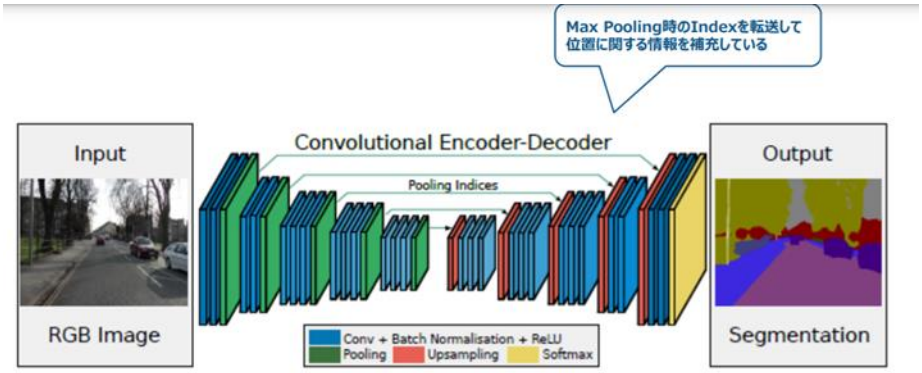
・skip-conection

畳み込み処理を加えていくと、ネットワークが「物体が何であるか」についての特徴を抽出していくが、pooling の影響で「物体がどこにあるか」についての特徴は失われていき、逆畳み込みを行なっても物体の位置情報は満足に復元できない場合がる。

⇒畳み込みを行なった後、特徴マップを保持しておいて、後で逆畳み込みをする画像に足し合わせる処理を skip-conection と言い、推定する領域を精密（シャープ）にする事が可能。

・その他セマンティック セグメンテーションのモデル





Badrinarayanan, V., A. Kendall, and R. Cipolla. "Segnet: A deep convolutional encoder-decoder architecture for image segmentation." arXiv. Preprint arXiv: 1511.0051, 2015.

Segnet とは、セマンティックセグメンテーション手法の一つで、Encoder-Decoder 構造を持っている。Encoder は画像の特徴を抽出する部分、Decoder というのは抽出された特徴から、抽出したことによって抽象的になってしまった情報から、高解像度な画像を再構築する部分になっている。

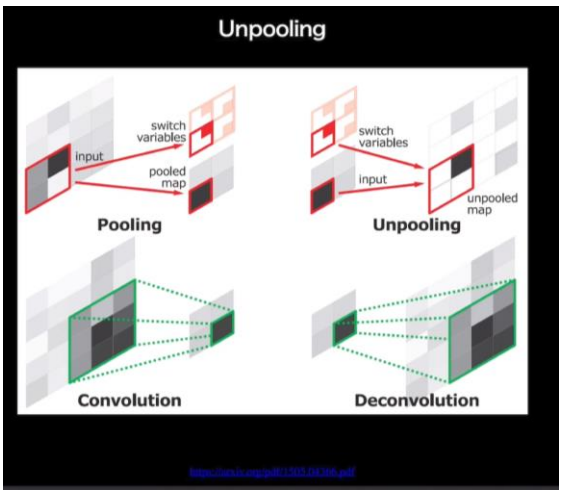
U-Net との違いは、

①U-Net では Deconvolution を使用しているのに対し、Segnet では学習の不要な Unpooling によって、計算の高速化を狙いつつ解像度を上げている。

②U-Net にはエンコーダブロックの途中の層からデコーダ構造にスキップする構造を持つ（この狙いは、畳み込みを繰り返すことによって、完全に抽象化された特徴から高解像度な画像を復元するのではなく、もっと浅い層の高解像度な情報をスキップしてやることで、解像度を向上させる）が、Segnet にはスキップ構造が無い。

・Unpooling

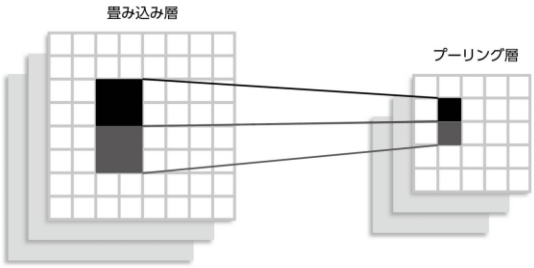
Up-sampling の一つ。Encoder 層の Maxpooling 時に、最大であった場所を保持しておき、Unpooling では、その部分を復元し、それ以外の部分を 0 にすることにより実現している。メリットとしては、学習するパラメータがないので、メモリや計算コストの節約になっている。



畳み込みネットワークは 2 つの細胞の働きを模倣するように考案されており、単純型細胞に対応する「畳み込み層」、複雑型細胞に対応する「プーリング層」というコンポーネントが用意されている。

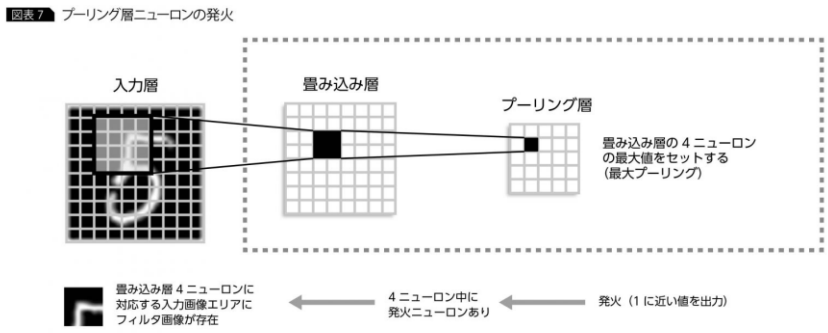
プーリング層は、複雑型細胞をモデル化したもので、入力画像におけるフィルタ形状の位置ずれを吸収するように機能する。その仕組みは、畳み込み層と比較するとかなり単純である。 プーリング層の個々のニューロンは、畳み込み層の一定領域のニューロンと結合する。

図表 6 畳み込み層とプーリング層の結合



ただし、この畳み込み層の領域同士は重複せず、1つの畳み込み層ニューロンは、ただ 1 つのプーリング層ニューロンのみと結合する。プーリング層の各ニューロンの出力値は、結合する畳み込み層のニューロンの出力値のなかで最大となる値を設定する。これにより、フィルタ形状の位置ずれを吸収することが可能。

プーリング層のあるニューロンの発火は、対応する畳み込み層のニューロンのどこかが発火したことを意味する。これを入力層に戻して考えると、畳み込み層の複数ニューロンに対応する局所受容野の領域を合成した部分にフィルタ形状が存在することを意味するので、畳み込み層の 1 ニューロンでカバーする局所受容野からずれた範囲での形状検出が可能となる

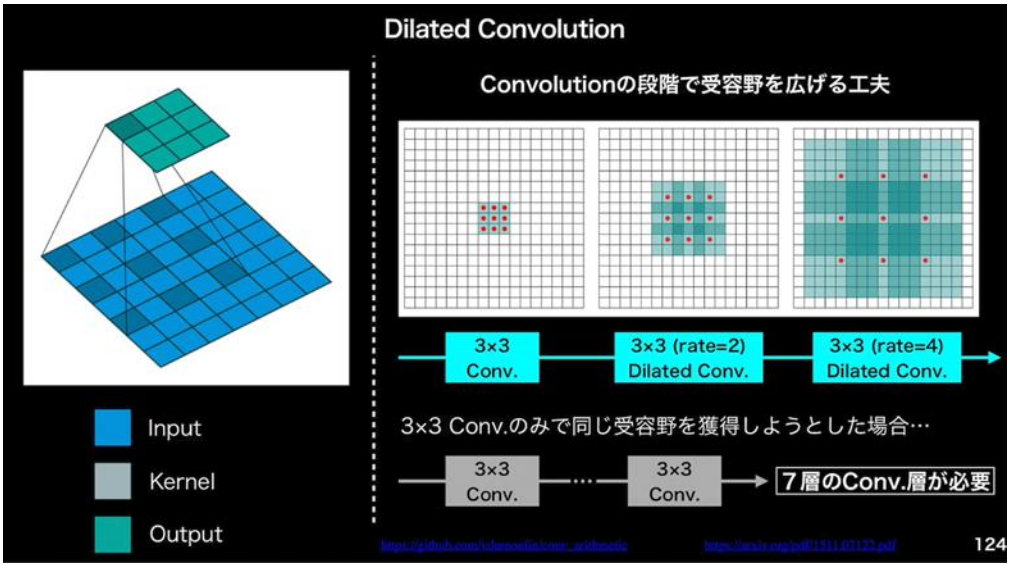


・Dilated Convolution

下記の図で示すように隙間の空いた歯抜けのフィルタを畳み込む手法。

dilation_rate を大きくしていけば pooling を使わずとも、小さなフィルターサイズで長距離の畳み込みが可能。

これを使えば pooling を使わないため画像サイズが小さくならない。。



ニューロンは、目の網膜の様々な部位と接続しており、ニューロンに影響を与える網膜上の範囲のことを「受容野」と言う。
畳み込みネットワーク中のニューロン間の結合パターンは、動物の視覚野の構成から着想を得ている。個々の皮質ニューロンは受容野（視野の限定された領域）における刺激にのみ応答する。

参考サイト：畳み込みネットワークの「基礎の基礎」を理解する

●Section7：DCGAN（Deep Convolutional GAN）

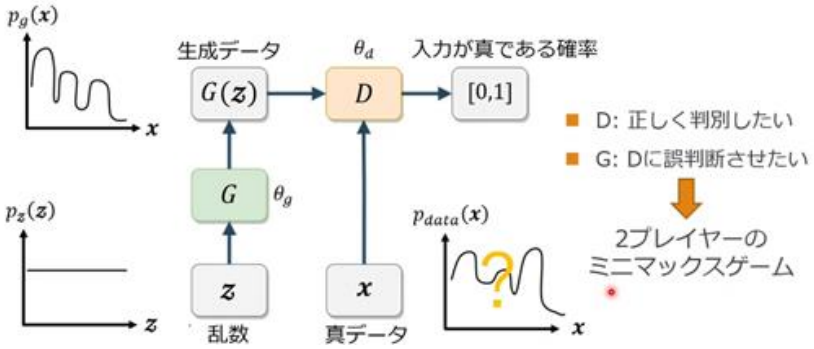
目次

- GANについて
  - GANの構造
  - ミニマックスゲームと価値関数
  - GANの最適化方法
  - 本物のようなデータを生成できる理由
- DCGANについて
  - 具体的なネットワーク構造
- 応用技術の紹介
  - 概要

◇GAN（Generative Adversarial Nets）：生成器と識別器を競わせて学習するモデル

・Generator…乱数からデータを生成

・Discriminator…入力データが真データであるか？を識別



2プレイヤーのミニマックスゲームとは？

・1人が自分の勝利する確率を最大化する作戦を取る

・もう一人は相手が勝利する確率を最小化する作戦を取る

- GANでは価値関数Vに対し，Dが最大化，Gが最小化を行う．

$$\min_G \max_D V(D,G)$$

$$V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}\left[\log\left(1 - D\left(G(z)\right)\right)\right]$$

- バイナリークロスエントロピーと似ている？

$$L = -\sum y \log \hat{y} + (1-y)\log(1-\hat{y})$$

GANの価値関数はバイナリークロスエントロピー

- 単一データのバイナリークロスエントロピー

$$L = -y \log \hat{y} + (1-y) \log(1-\hat{y})$$

y: 真値（ラベル）
$\hat{y}$: 予測値（確率）

- 真データを扱う時: $y = 1,\ \hat{y} = D(x)$ ➡ $L = -\log[D(x)]$
- 生成データを扱う時: $y = 0,\ \hat{y} = D(G(z))$ ➡ $L = -\log[1 - D(G(z))]$

- 2つを足し合わせる

$$L = -\left(\log[D(x)] + \log[1 - D(G(z))]\right)$$

$$V(D,G) = \mathbb{E}_{x \sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}\left[\log\left(1 - D(G(z))\right)\right]$$

- 複数データを扱うために期待値を取る
- 期待値: 何度も試行する際の平均的な結果値 $\sum xp(x)$

・最適化方法

① Generator のパラメータ θ_gを固定

真データと生成データをm個づつサンプル

θ_dを勾配上昇法（Gradient Ascent）で更新

$$\frac{\partial}{\partial \theta_d} \frac{1}{m} \left[log[D(x)] + log\left[1 - D(G(z))\right]\right] \quad ※\theta_d \text{を} k \text{回更新}$$

② Discriminator のパラメータ θ_dを固定

生成データをm個サンプル

θ_gを勾配降下法（Gradient Descent）で更新

$$\frac{\partial}{\partial \theta_g} \frac{1}{m} \left[log\left[1 - D(G(z))\right]\right] \quad ※\theta_g \text{を} 1 \text{回更新}$$

〈疑問〉なぜ、Generator は本物のようなデータを生成するのか？

⇒生成データが本物とそっくりな状況とは、$p_g = p_{data}$

⇒価値関数が $p_g = p_{data}$ の時に、最適化されている事を示せばよい。

$$\min_G \max_D V(D,G)$$

$$V(D,G) = \mathbb{E}_{x \sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}\left[\log\left(1 - D(G(z))\right)\right]$$

これを２つのステップにより確認する。

Step1. G を固定し、価値関数が最大値を取る時の$D(x)$を算出

ステップ1: 価値関数を最大化する$D(x)$の値は?

➢Generatorを固定

$$V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}\left[\log\left(1 - D\left(G(z)\right)\right)\right]$$
$$= \int_x p_{data}(x)\log\left(D(x)\right)dx + \int_z p_z(z)\log\left(1 - D\left(G(z)\right)\right)dz$$
$$= \int_x p_{data}(x)\log\left(D(x)\right) + p_g(x)\log\left(1 - D(x)\right)dx$$

$y = D(x), a = p_{data}(x), b = p_g(x)$ と置けば

$$a\log(y) + b\log(1-y)$$

➢$a\log(y) + b\log(1-y)$の極値を求めよう

➢$a\log(y) + b\log(1-y)$をyで微分

$$\frac{a}{y} + \frac{b}{1-y}(-1) = 0$$

$$\frac{a}{y} = \frac{b}{1-y}$$

$$a - ay = by$$

$$a = (a+b)y$$

$$y = \frac{a}{a+b}$$

➢$y = D(x), a = p_{data}(x), b = p_g(x)$なので

$$D(x) = \frac{p_{data}(x)}{p_{data}(x) + p_g(x)}$$

➢価値関数が最大値を取るときの$D(x)$が判明

Step2. 上記の$D(x)$を価値関数に代入し、G が価値関数を最小化する条件を算出

ステップ2: 価値関数はいつ最小化するか?

➢価値関数の$D(x)$を$\frac{p_{data}(x)}{p_{data}(x)+p_g(x)}$で置き換え

$$V = \mathbb{E}_{x\sim p_{data}}\log\left[\frac{p_{data}(x)}{p_{data}(x)+p_g(x)}\right] + \mathbb{E}_{x\sim p_g}\log\left[1 - \frac{p_{data}(x)}{p_{data}(x)+p_g(x)}\right]$$
$$= \mathbb{E}_{x\sim p_{data}}\log\left[\frac{p_{data}(x)}{p_{data}(x)+p_g(x)}\right] + \mathbb{E}_{x\sim p_g}\log\left[\frac{p_g}{p_{data}(x)+p_g(x)}\right]$$

➢二つの確率分布がどれぐらい近いのか調べる必要がある

■有名な指標としてJSダイバージェンスがある

■$JS(p_1 \parallel p_2) = \frac{1}{2}\left(\mathbb{E}_{x\sim p_1}\log\left(\frac{2p_1}{p_1+p_2}\right) + \mathbb{E}_{x\sim p_2}\log\left(\frac{2p_2}{p_1+p_2}\right)\right)$

■JSダイバージェンスは非負で、分布が一致する時のみ0の値を取る

➢価値関数を変形

$$V = \mathbb{E}_{x\sim p_{data}} \log\left[\frac{p_{data}(x)}{p_{data}(x)+p_g(x)}\right] + \mathbb{E}_{x\sim p_g} \log\left[\frac{p_g}{p_{data}(x)+p_g(x)}\right]$$

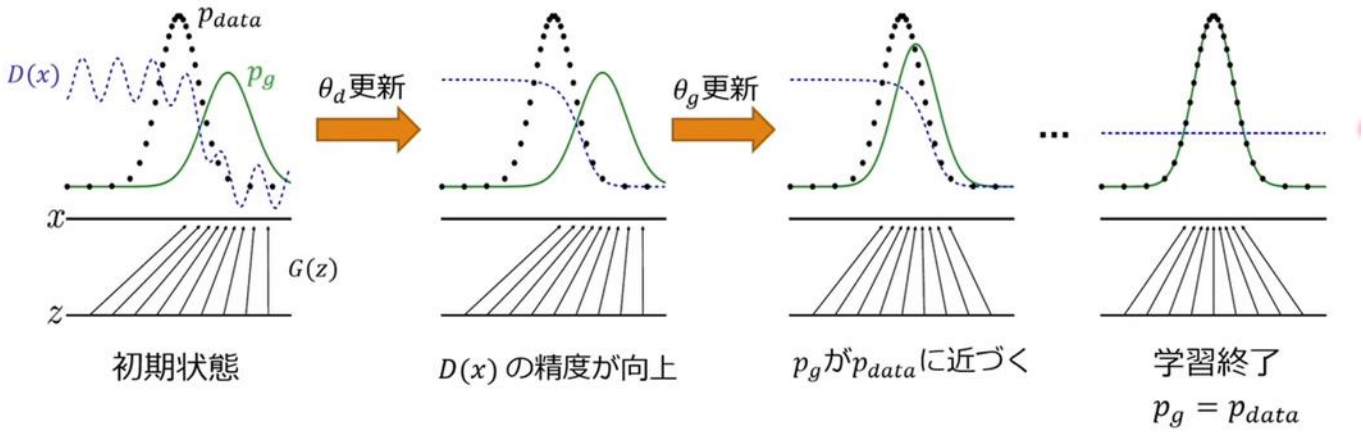
$$= \mathbb{E}_{x\sim p_{data}} \log\left[\frac{2p_{data}(x)}{p_{data}(x)+p_g(x)}\right] + \mathbb{E}_{x\sim p_g} \log\left[\frac{2p_g}{p_{data}(x)+p_g(x)}\right] - 2\log 2$$

$$= 2JS(p_{data} \parallel p_g) - 2\log 2$$

$\min_G V$は、$p_{data} = p_g$の時に最小値となる$(-2\log 2 \approx -1.386)$

GAN の学習により、Generator は本物の様なデータを生成できる

学習ステップの可視化



◇DCGAN

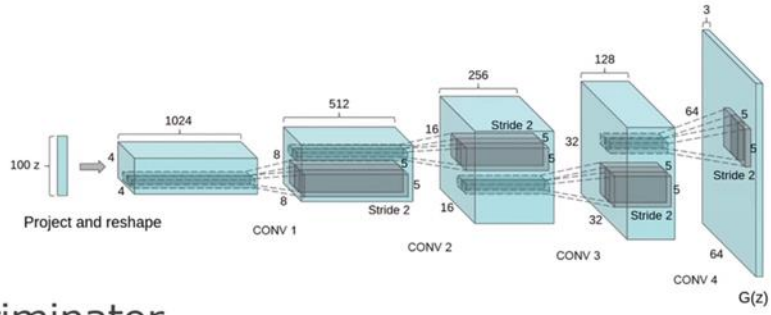
・GAN を利用した画像生成モデル

・いくつかの構造制約により生成品質を向上

DCGANのネットワーク構造

➢Generator

■転置畳み込み層により乱数を画像にアップサンプリング



➢Discriminator

■畳み込み層により画像から特徴量を抽出し、最終層をsigmoid関数で活性化

・Generator：

Pooling 層の代わりに転置畳み込み層を使用し、最終層は tanh、その他は ReLU 関数で活性化

・Discriminator:

Pooling 層の代わりに畳み込み層を使用し、Leaky ReLU 関数で活性化

・共通事項

中間層に全結合層を使わない

バッチ正則化を適用する

ネットワーク構造

➢輪郭画像と低周波動画像を別々に生成する