Mask R-CNNとその関連技術

物体検出

物体検出は特徴を抽出し、対象物の領域を切り出し、クラスを認識し….
とたくさんのことをやっている

- 物体検出の流れ
  ①物体の同定（identification）：画像の中で物体がどこにあるのか？
  ↓
  ②物体認識/分類（classification）：何の物体であるか？

- **物体検出**は**バウンディングボックス**を使用
  長方形で関心領域を切り出す
  ４座標を予測する回帰問題

Ross, Girshick, "Fast r-cnn." in ICCV2015

セマンティックセグメンテーションとは

- **物体領域を画素単位で切り出し、各画素にクラスを割り当てる手法**
- 工業検査や医療画像解析など、**精密な領域分割**に応用される
- 重要な学習データセットに、 VOC2012とMSCOCOがある

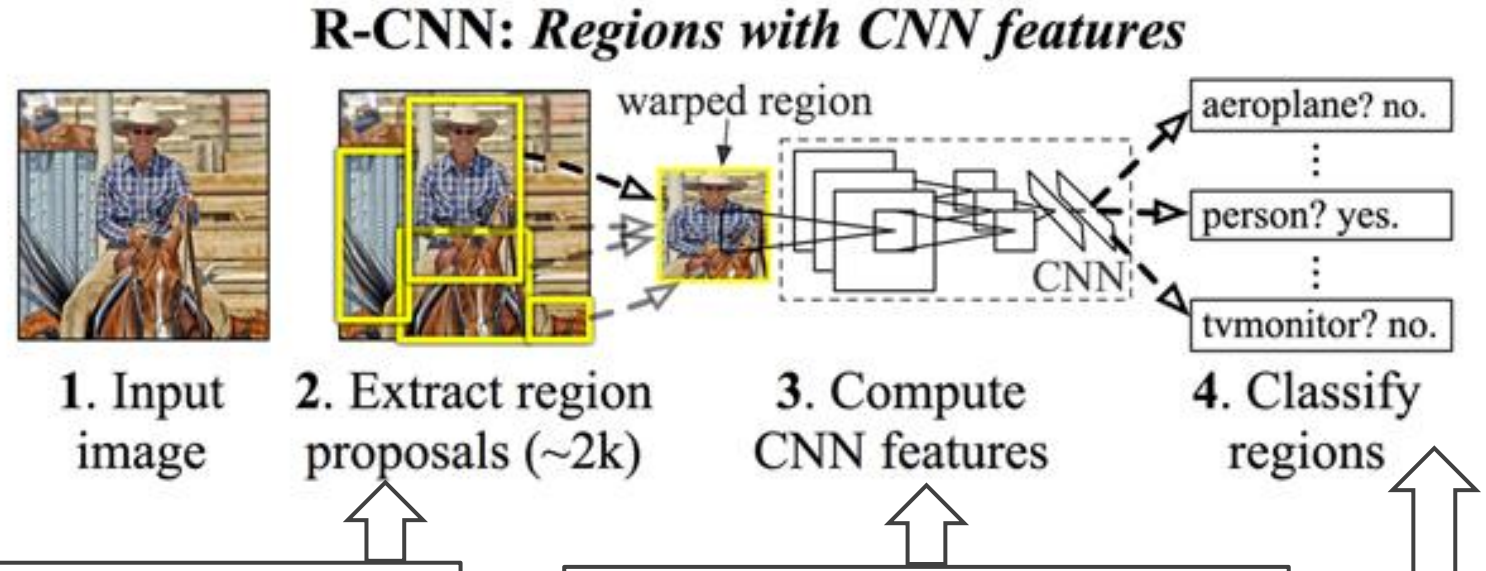
（左図）
バイクと乗車している人間のそれぞれの境界線 / 輪郭線を描くためには、画素単位の高密度な予測をが必要

- 主要手法は、**完全畳み込みネットワーク（FNC; Fully Convolutional Network）**
  - **全ての層が畳み込み層**、全結合層を有しない
  - **画素ごとにラベル付した教師データを与えて学習する**→ 出力ノードが多数
  - 未知画像も画素単位でカテゴリを予測する
  - **入力画像のサイズは可変で良い**

R-CNN（Regional CNN）

- 物体検出+物体認識のアルゴリズムの原形は **R-CNN（Regional CNN）**
- 物体検出タスクと物体認識タスクを順次に行う

Ross, Girshick, "Fast r-cnn." in ICCV2015



関心領域（ROI;Region of Interest）
を切り出す
類似する領域をグルーピング

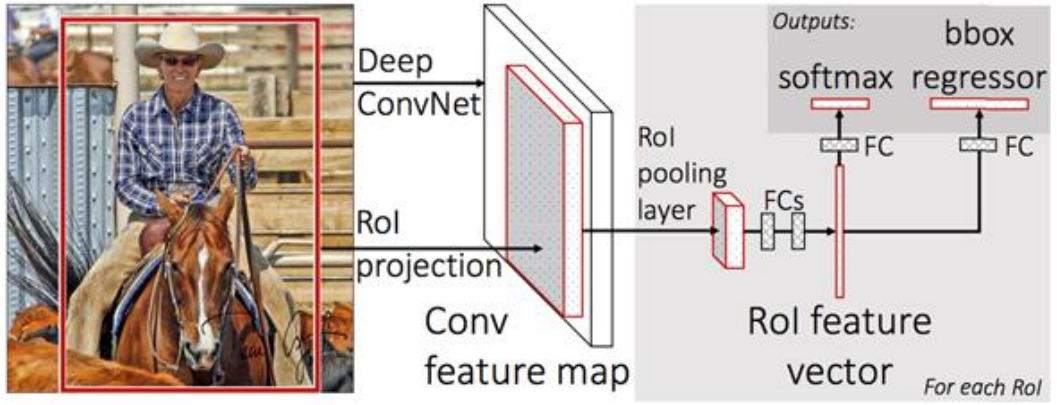
候補領域の画像の大きさを揃える
CNNにより特徴量を求める

CNNで求めた特徴量をSVMで学習
未知画像も特徴量をCNNで求め、
学習済みSVMで分類

R-CNNの発展版

- R-CNNは多数の物体領域に対し複雑な処理を行うため、処理が**重くて遅い**のが課題

- 改良版の **高速R-CNN（Fast R-CNN）**では、関心領域ごとに畳み込み層に通すのではなく、画像ごとに一回の畳み込み操作を行う ➡ **計算量を大幅に減少**



高速R-CNN（Fast R-CNN）の模式図

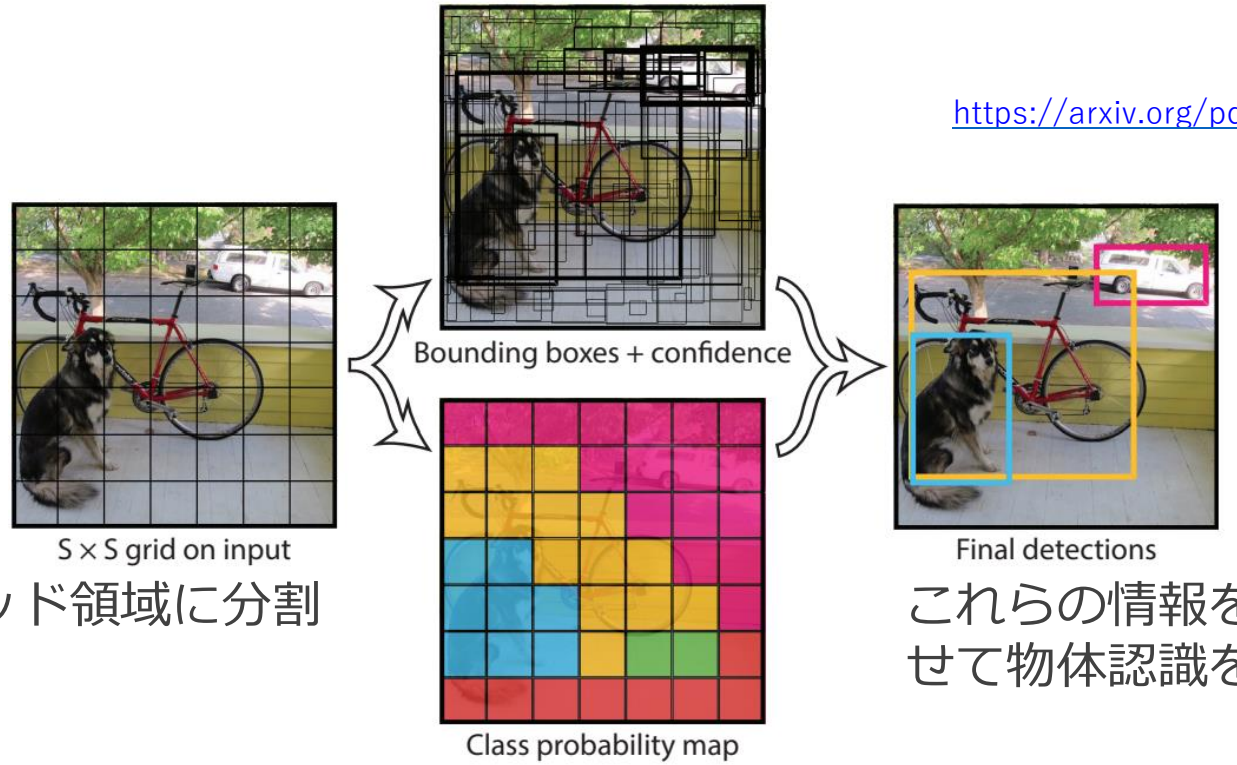
Ross, Girshick, "Fast r-cnn." in ICCV2015

- その後、さらに **Faster R-CNN** も開発された
  - **関心領域の切り出しもCNNで行う**
  - ほぼリアルタイムで動作し、動画認識への応用も
- 他の発展版、**YOLO**（You Only Look Once）や **SSD**（Single Shot Detector）も**領域の切り出しと物体認識を同時に行う**

（参考）YOLO

バウンディングボックスで候補領域を抽出すると同時に、物体なのか、背景なのかの確率を表す信頼度スコアを算出

https://arxiv.org/pdf/1506.02640.pdf



入力画像をグリッド領域に分割

これらの情報を組み合わせて物体認識を行う

各グリッドでクラス分類を行う

- メリット
  - 処理が速い（アルゴリズムは**１つのCNNで完結し、領域推定と分類を同時に行う**）
  - 画像全体を見て予測することができるため、誤検出が「Fast R-CNN」の半分以下

インスタンス・セグメンテーション

- インスタンスセグメンテーションとは
- 画像のピクセルをどのクラスに属するかに分類するのに加えて、検出したそれぞれの物体を見分け、クラスIDに加えインスタンスID(それぞれの物体のID)を予測する

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とインスタンスセグメンテーションの違い
- セマンティックセグメンテーションはピクセルをクラス分類するが、同じクラスの複数の物体があっても区別できない
- インスタンスセグメンテーションは、同じクラスであっても個々の物体を区別することができる

インスタンス・セグメンテーション

- インスタンスセグメンテーションのための有名なアプローチはYOLACT
- YOLOはリアルタイムにワンステップで物体検出を行うアルゴリズムであるのと同様に、**YOLACT**はワンステップでインスタンスセグメンテーションを行う
- もう１つの代表的なアルゴリズムは”**Mask R-CNN**”; 2017年に米Facebook AI Research所属のKaiming Heらが提案　（参考）https://arxiv.org/abs/1703.06870

Mask R-CNNとは

- Mask R-CNNは、**Faster R-CNNを拡張**したアルゴリズム
  - Mask R-CNNはインスタンスセグメンテーションに対応するので、Faster R-CNNの物体検出機能にセグメンテーションの機能を付加したイメージ
  - バウンディングボックス内の画素単位でクラス分類を行うため、物体の形も推定可能

（原論文） https://arxiv.org/abs/1703.06870

- Mask R-CNN は ICCV2017 のBest Paper に選出

※ ICCV （International Conference on Computer Vision）はコンピュータービジョンの最高峰のカンファレンス

Mask R-CNNの特徴

※一部、インスタンスセグメンテーションの説明と重複

画像中の物体らしき領域とその領域にあるクラスを検出

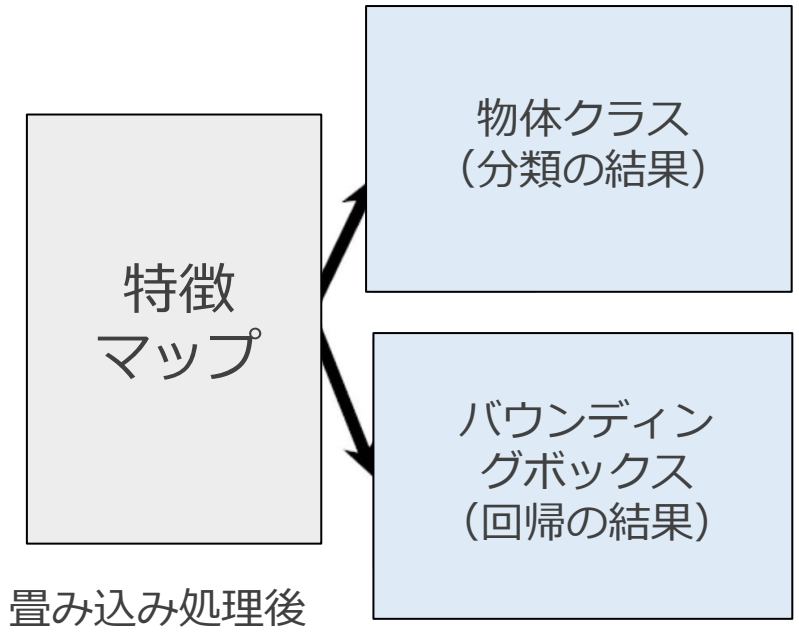
- セグメンテーションとは画像中の画素ごとにクラスを検出すること
- Mask R-CNNは、画像全体ではなく、**物体検出の結果として得られた領域についてのみセグメンテーションを行うことで効率アップ**
- 「物体らしさ」が閾値以上の領域にのみ絞り、領域毎に最も確率が高いクラスを採用

（右図）
学習済みのMask R-CNNモデルで行われたセグメントの例：
人物の姿勢の推定、関節等のキーポイントまで検出可能

Mask R-CNNはFaster R-CNNと構造に類似点が多い

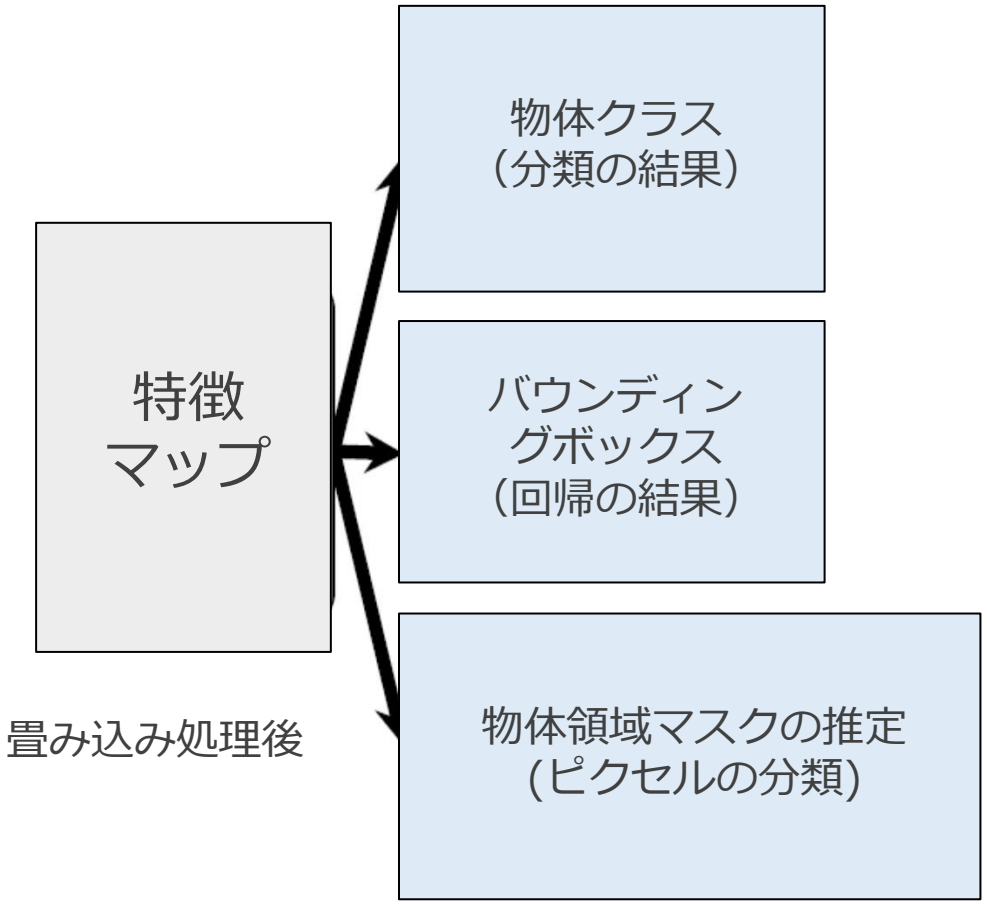
Faster R-CNN

ネットワークが2つに分岐し、2種の結果を出力



Mask R-CNN（セグメンテーションベース）

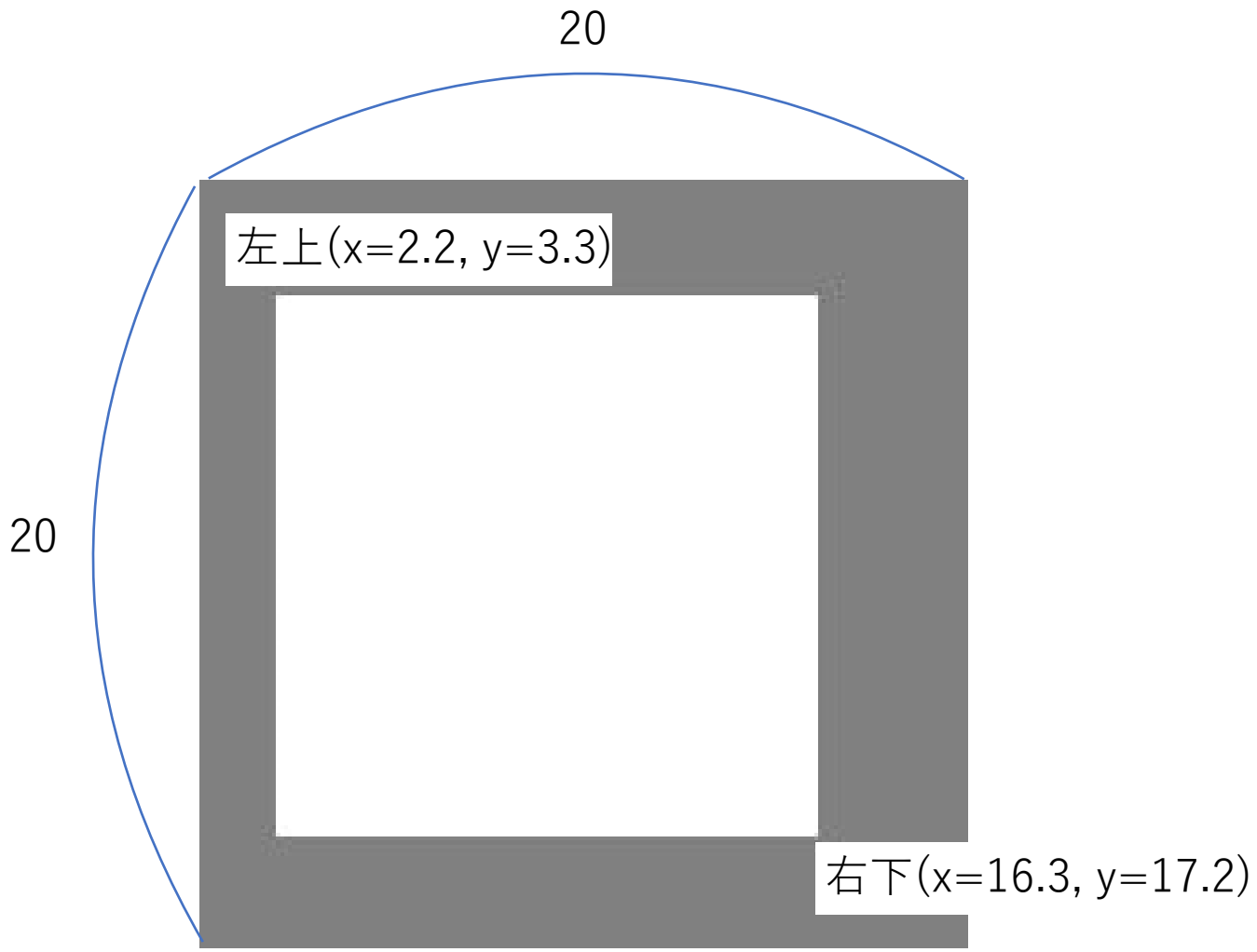
もう1つ分岐がある



RoI Pooling

- Fast/Faster R-CNNでは**RoI Pooling** を使用
- 畳み込み処理後の特徴マップから、region proposal領域を「**固定サイズの特徴マップ**」として抽出
  （※前段の畳込み処理を共通化できて、高速化を実現した要因の１つ）
- 特徴マップ上で固定サイズの特徴ベクトルを切り出す際に、小数を切り捨てるなどの操作をし、情報が粗く離散化されて欠落してしまうため、高精度の推定が難しい

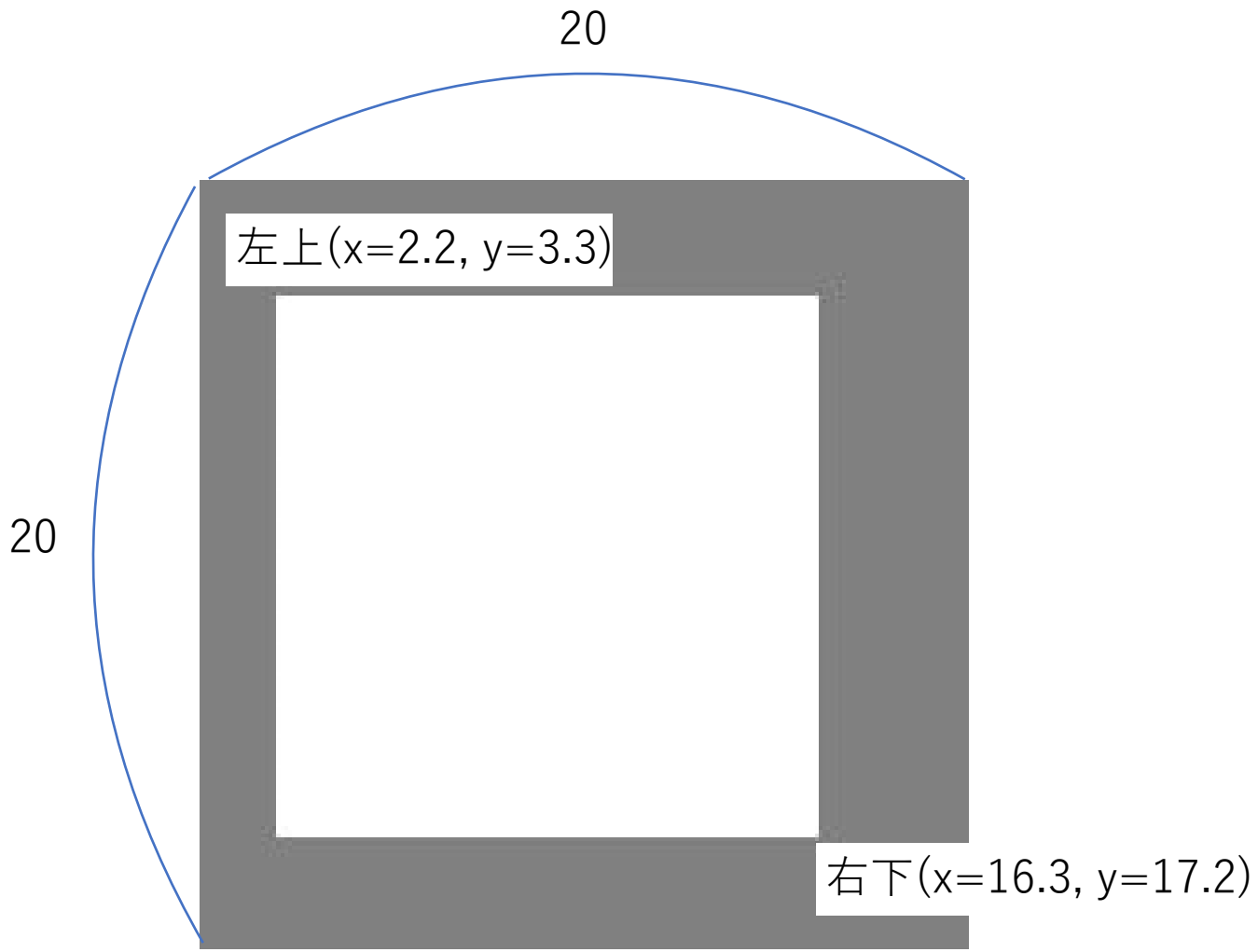
RoI Pooling



灰色の正方形の大きさが
特徴マップの大きさ

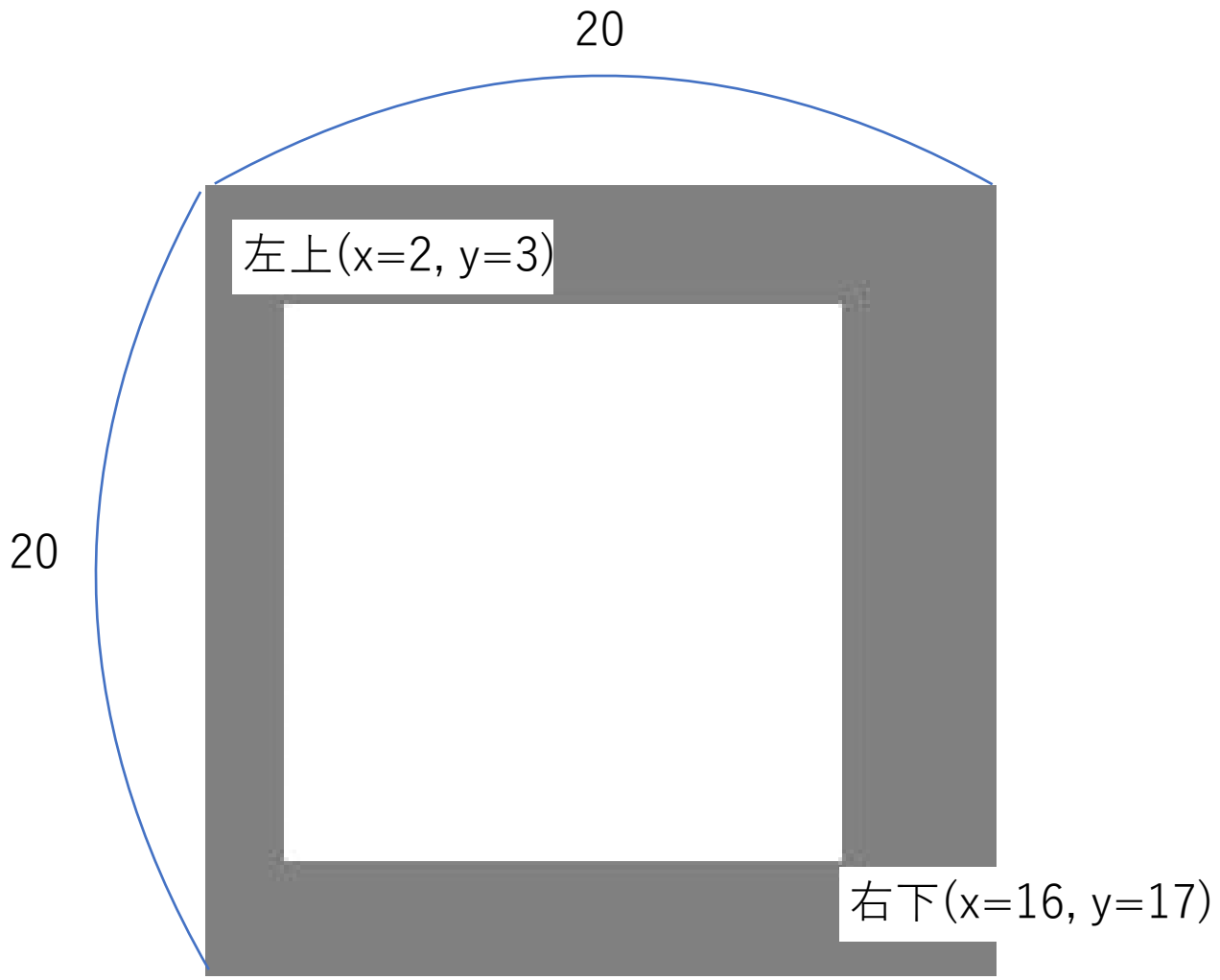
白い正方形が
提案された領域とする

RoI Pooling



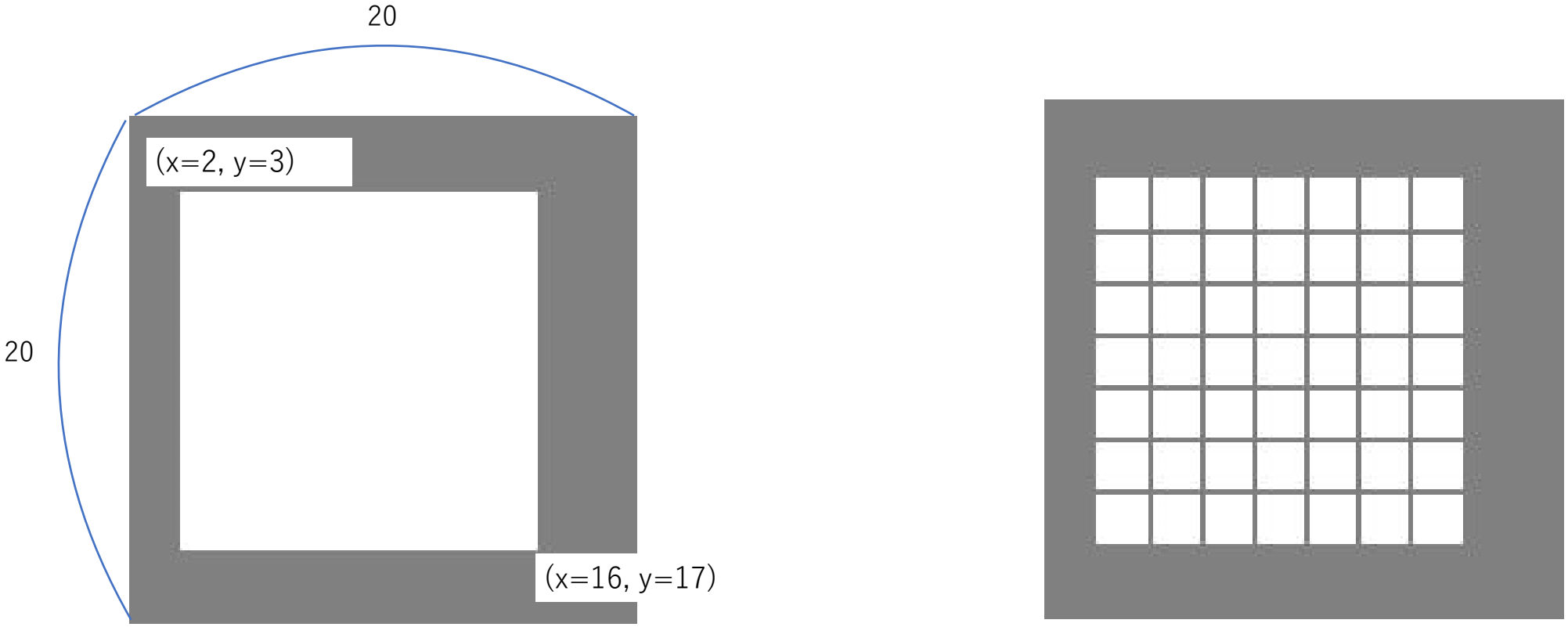
7x7の特徴マップを
を得たいとする

RoI Pooling



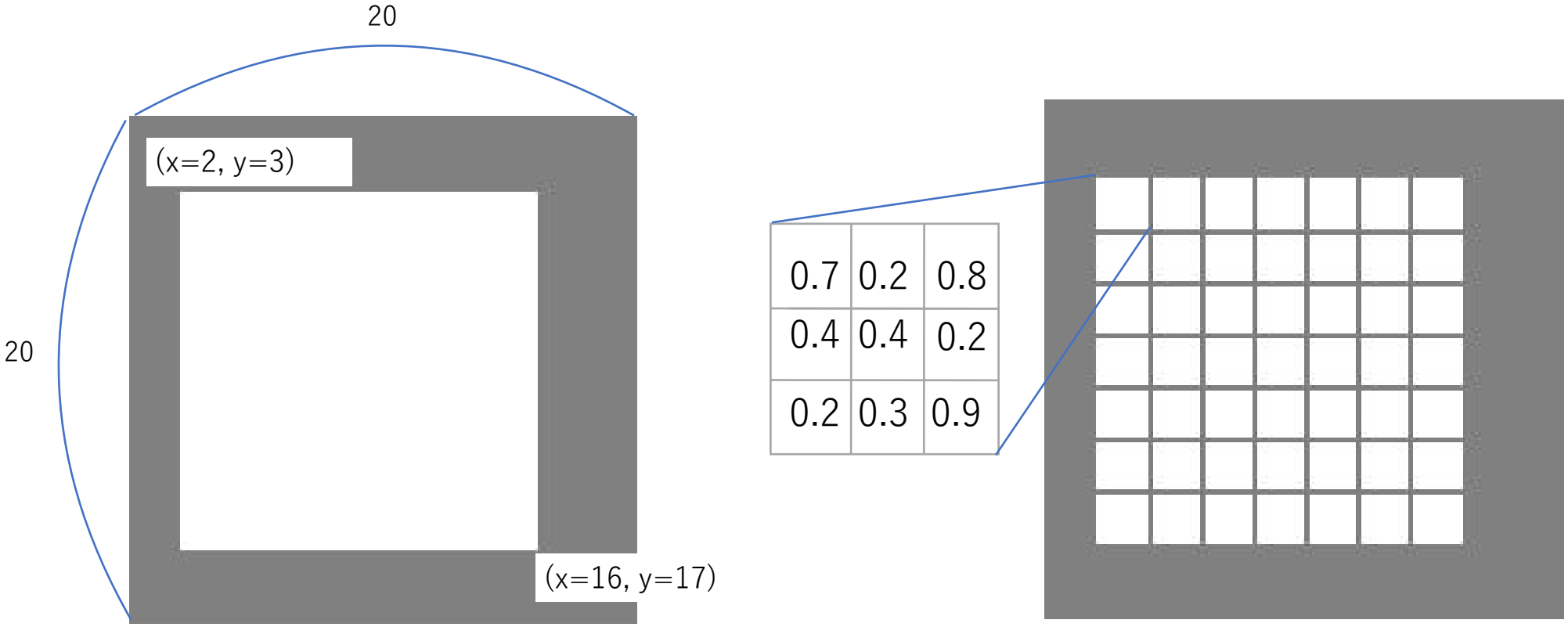
まず、region proposalの座標を整数に丸める

RoI Pooling



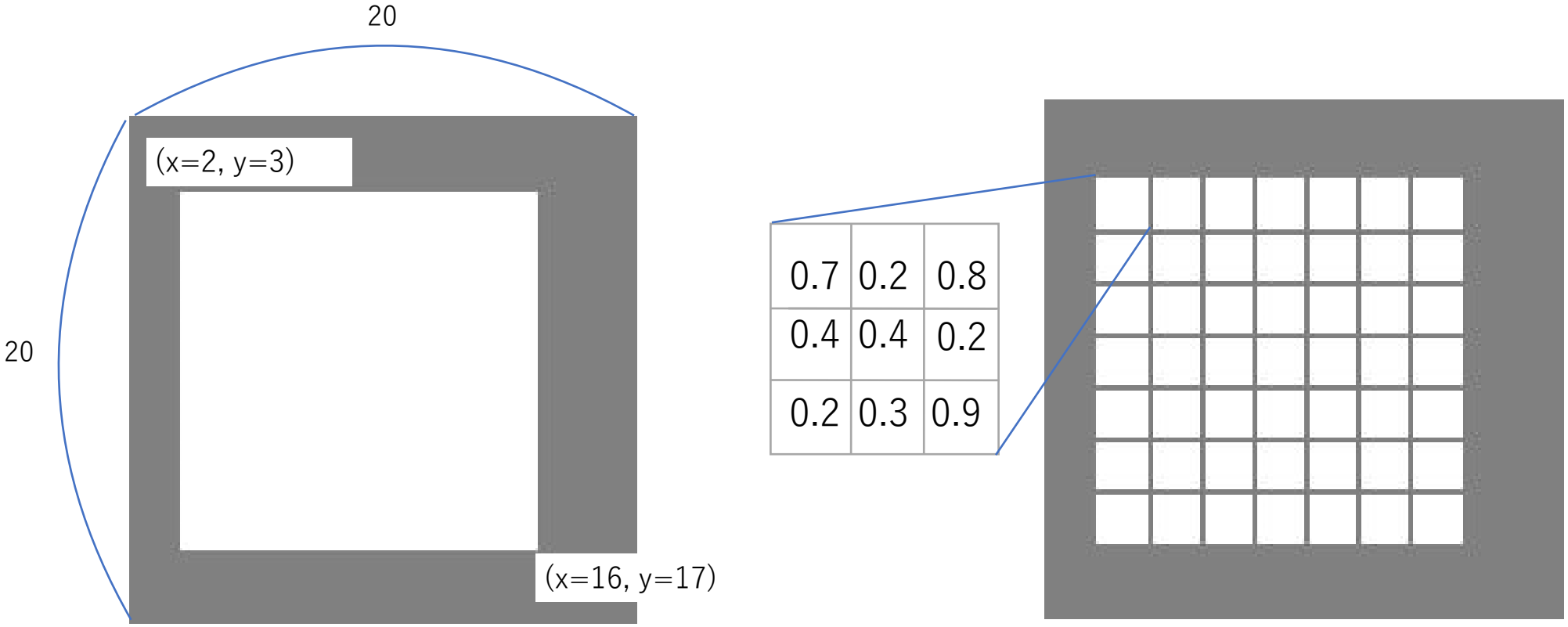
region proposalを7x7に分割する

RoI Pooling



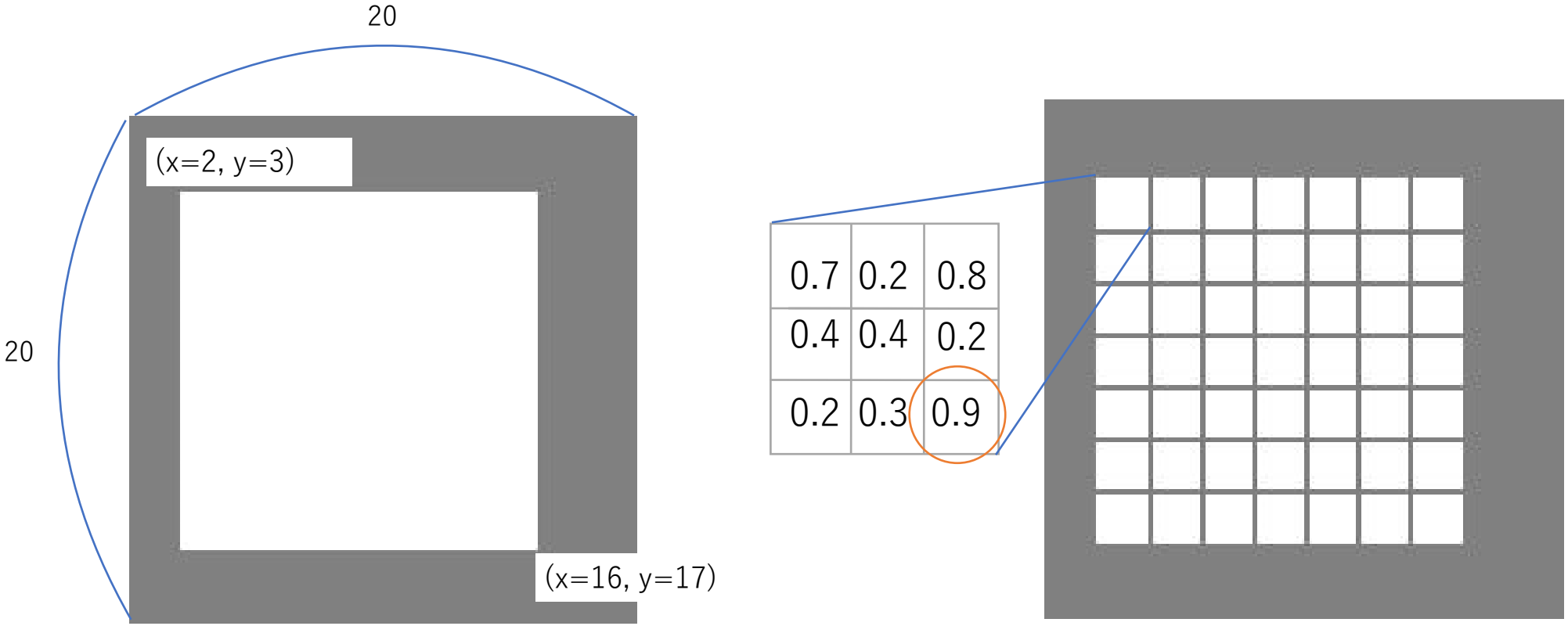
7x7の領域の一つ一つは、複数のピクセルを持っている

RoI Pooling



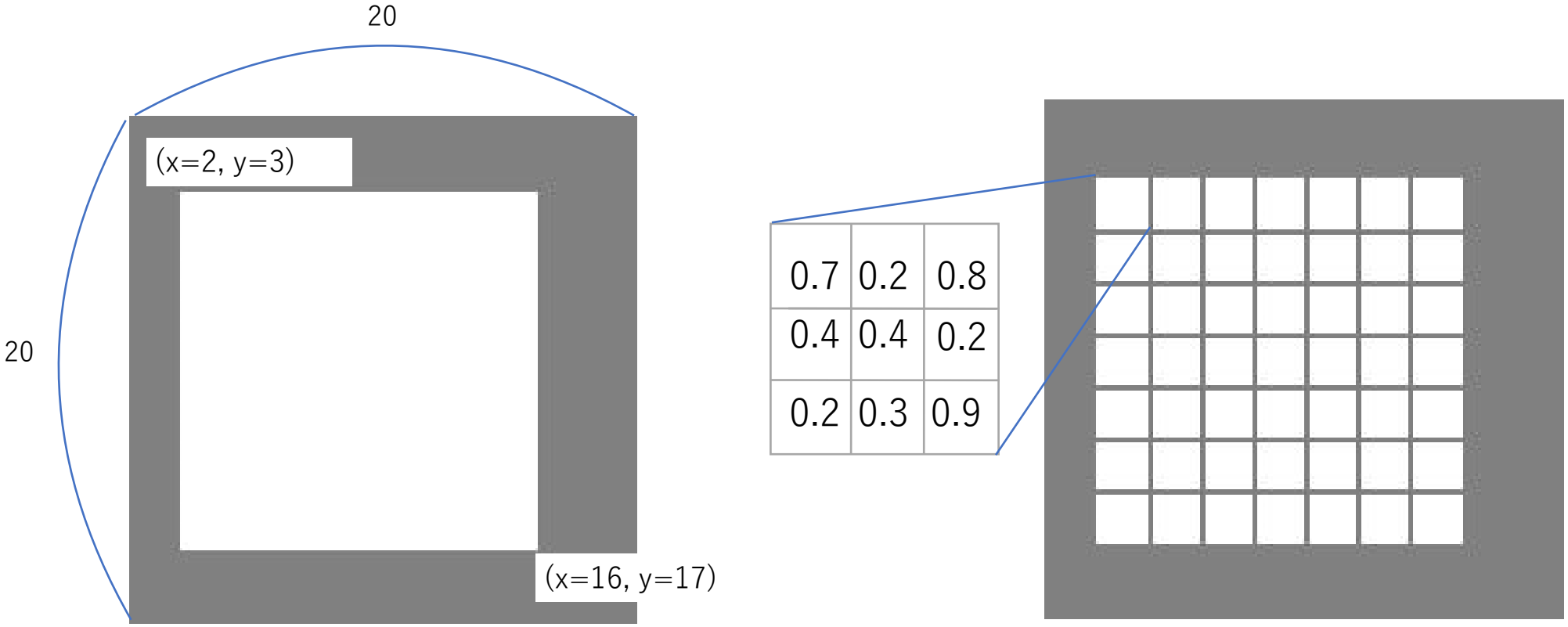
7x7の領域の一つ一つについて、
複数のピクセルを一つにまとめる

RoI Pooling



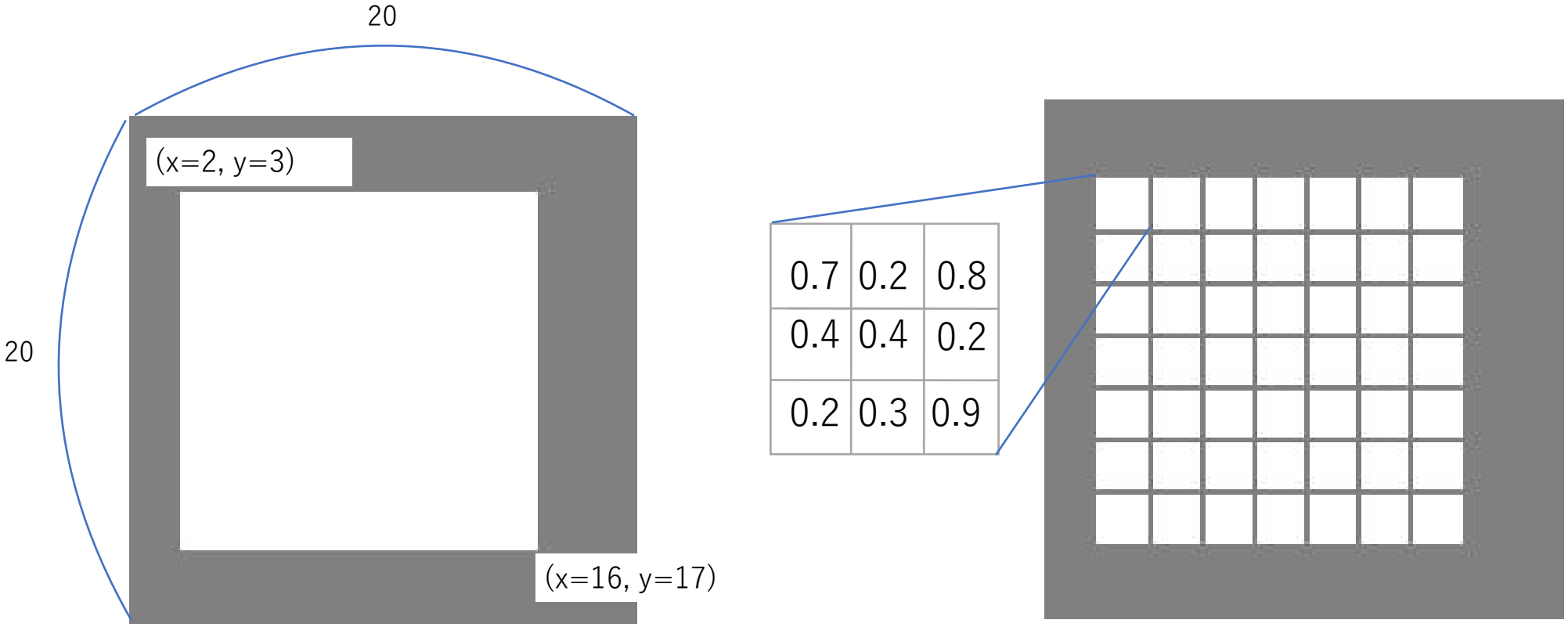
どうやって複数のピクセルを一つにするのか
複数のピクセルの最大値（ここでは0.9）
をとるか、平均値をとる

RoI Pooling



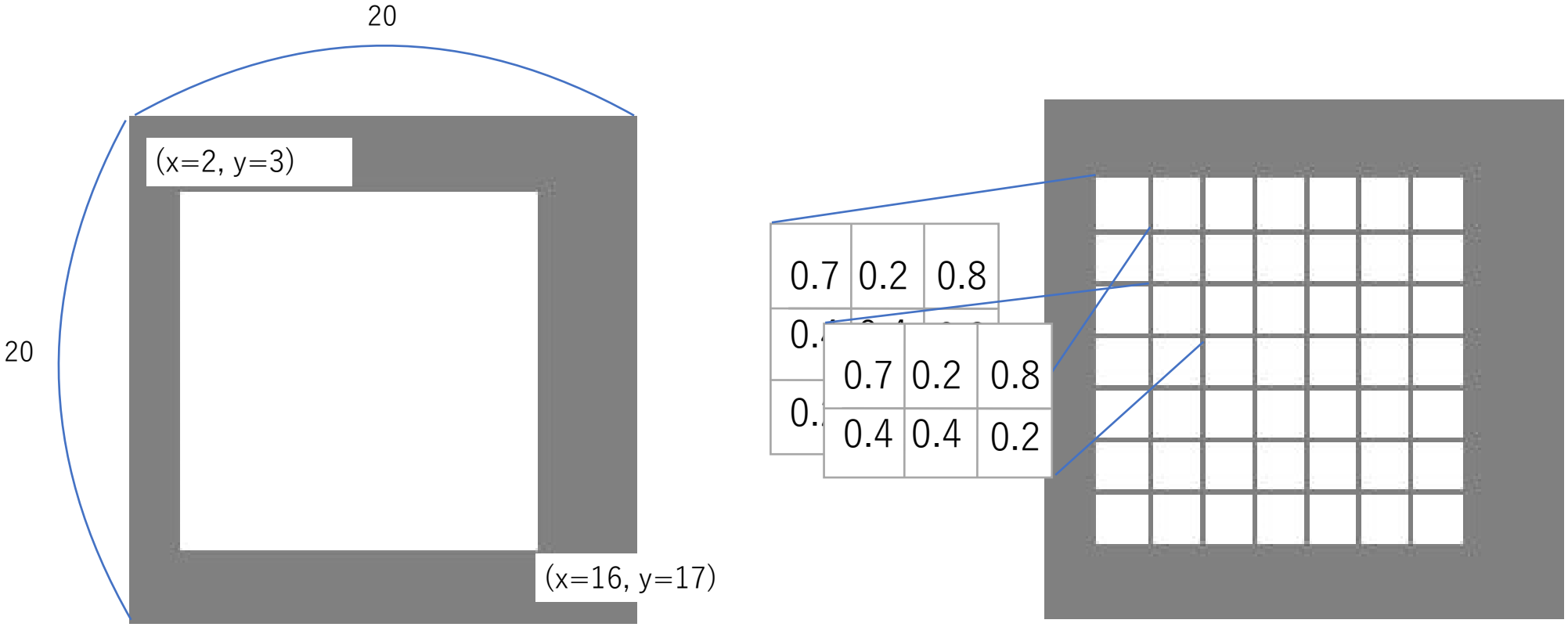
その値を7x7の領域のうちの一つを代表する値とする

RoI Pooling



7x7の領域一つずつその値を計算し、最終的に
7x7の固定サイズの特徴マップとして切り出すことができる

RoI Pooling

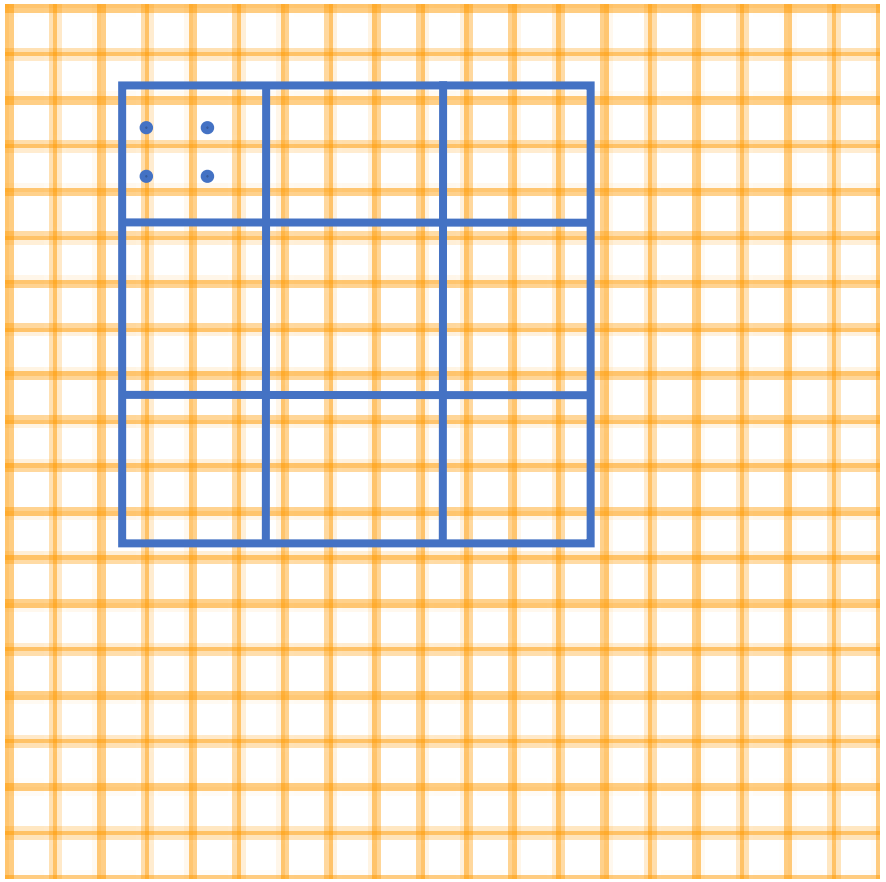


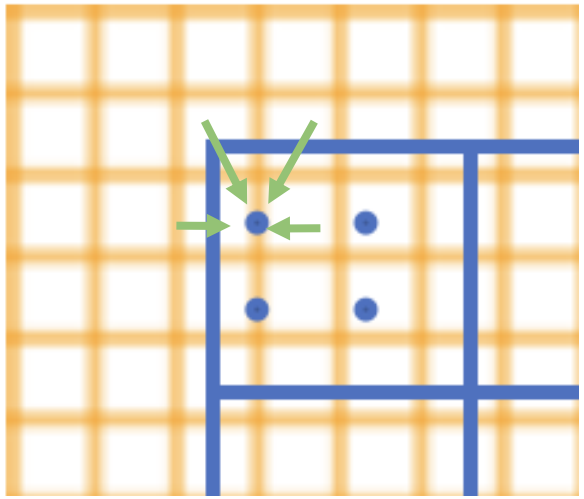
しかし、図のように
領域によってサイズが異なる場合があり、情報が
少ないところがあったりして荒いまとめ方になることも

RoI PoolingとRoI Align

- RoI Pooingの代わりに、 Mask R-CNNでは新しい手法 **RoI Align** を導入
- Alignment（位置合わせ）を重視したRoI (Region of Interest：関心領域)特徴の作成がポイント
- 特徴マップをただ間引くのではなく、**補間処理**によって、より多くのピクセルの情報を使うことで推定の精度を上げられた
- （具体的に）RoIをN×Nのグリッドに分割し、グリッドの各点の値を特徴マップの4ヶ所からサンプリングした値を補間法（ bilinear interpolation など）で算出する → その結果を最大/平均poolingして固定サイズのRoI特徴ベクトルとする

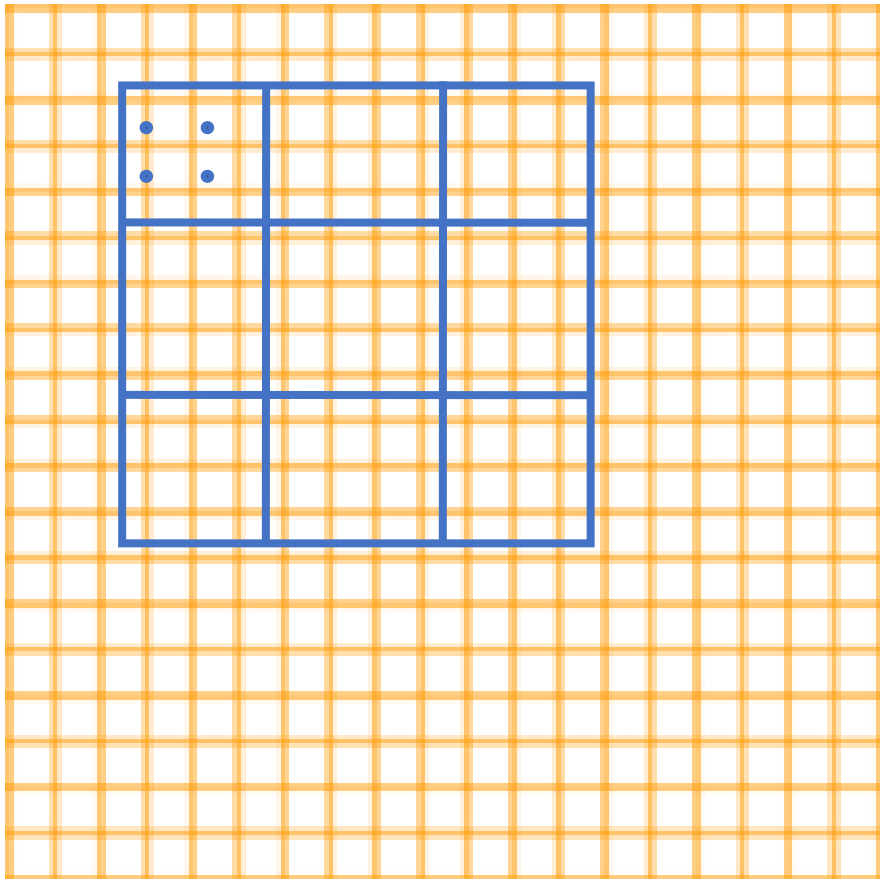
RoI Align

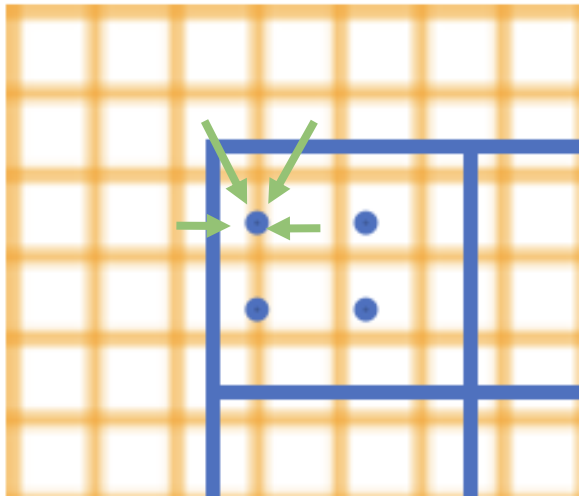




3x3の固定された
特徴マップを得たいとする

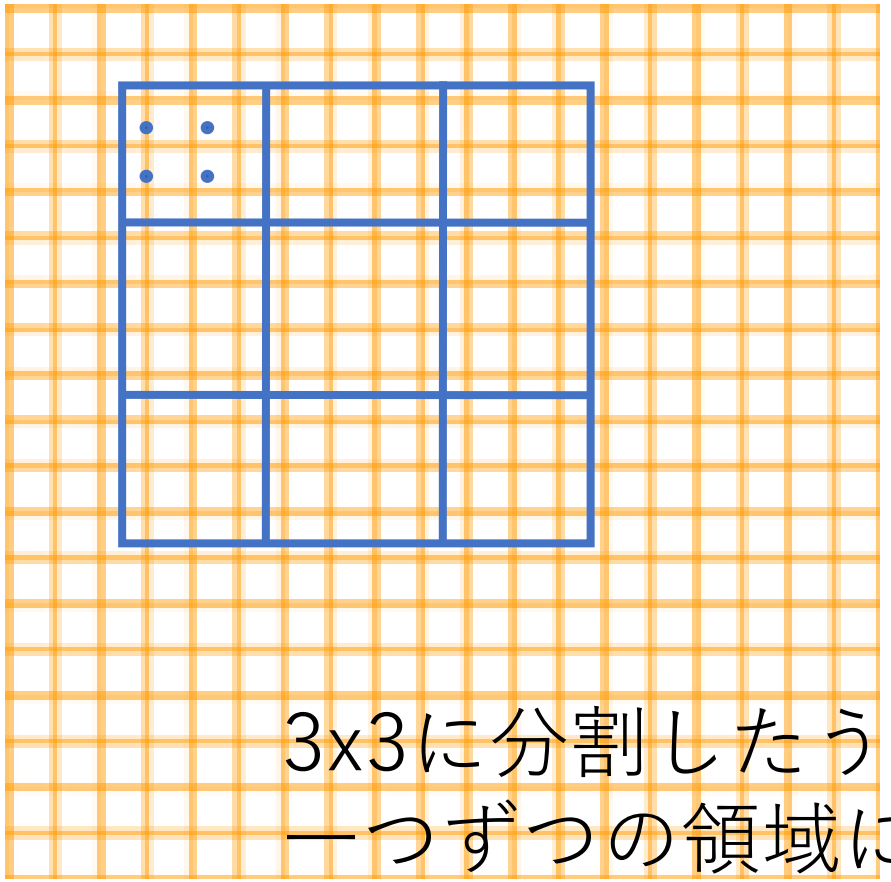
RoI Align

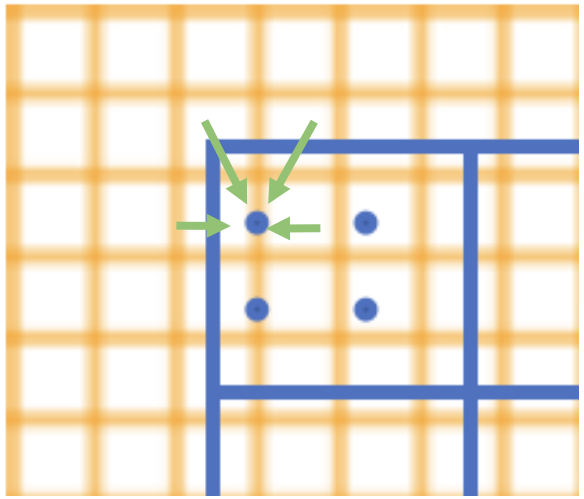




整数に丸めない

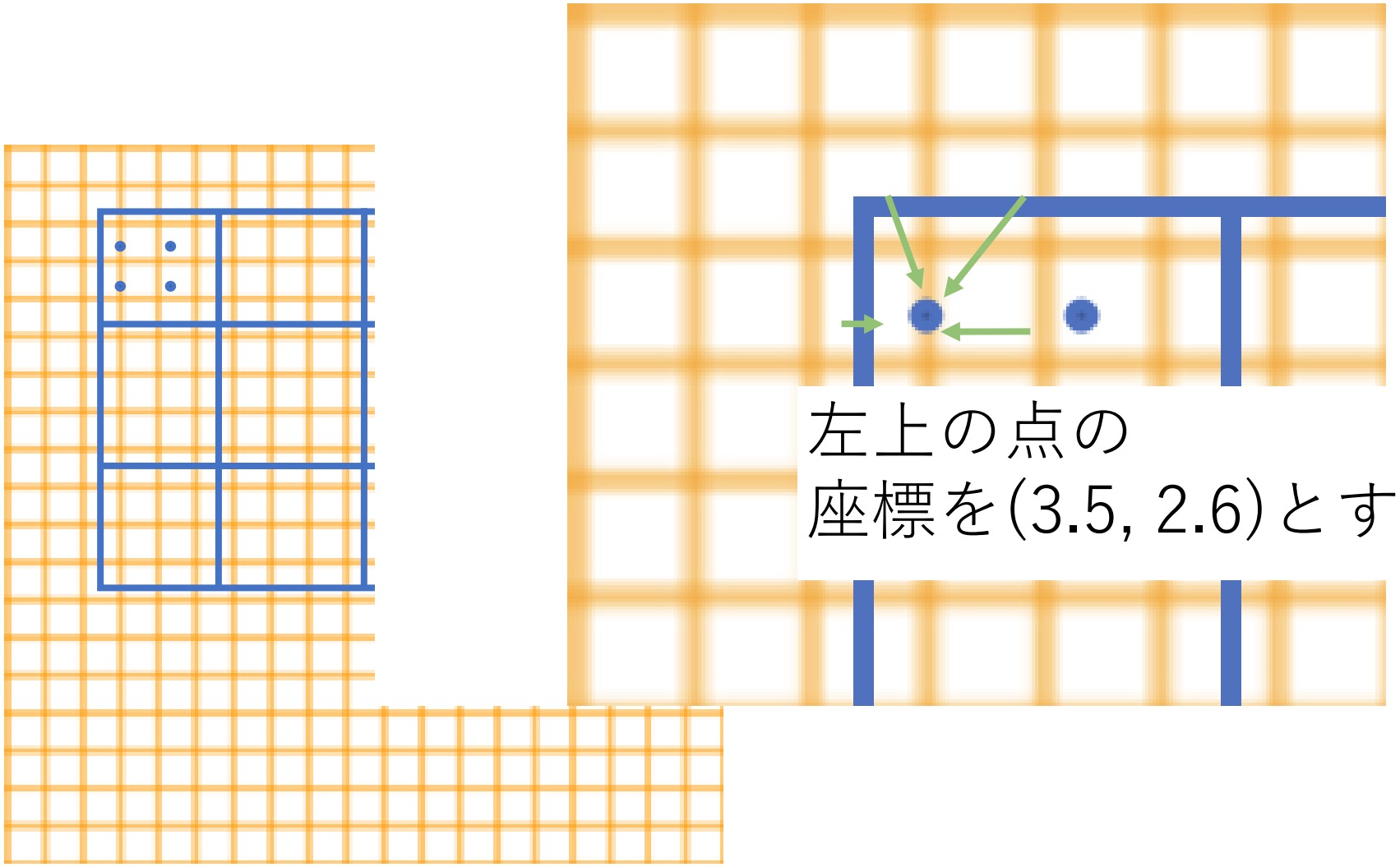
RoI Align



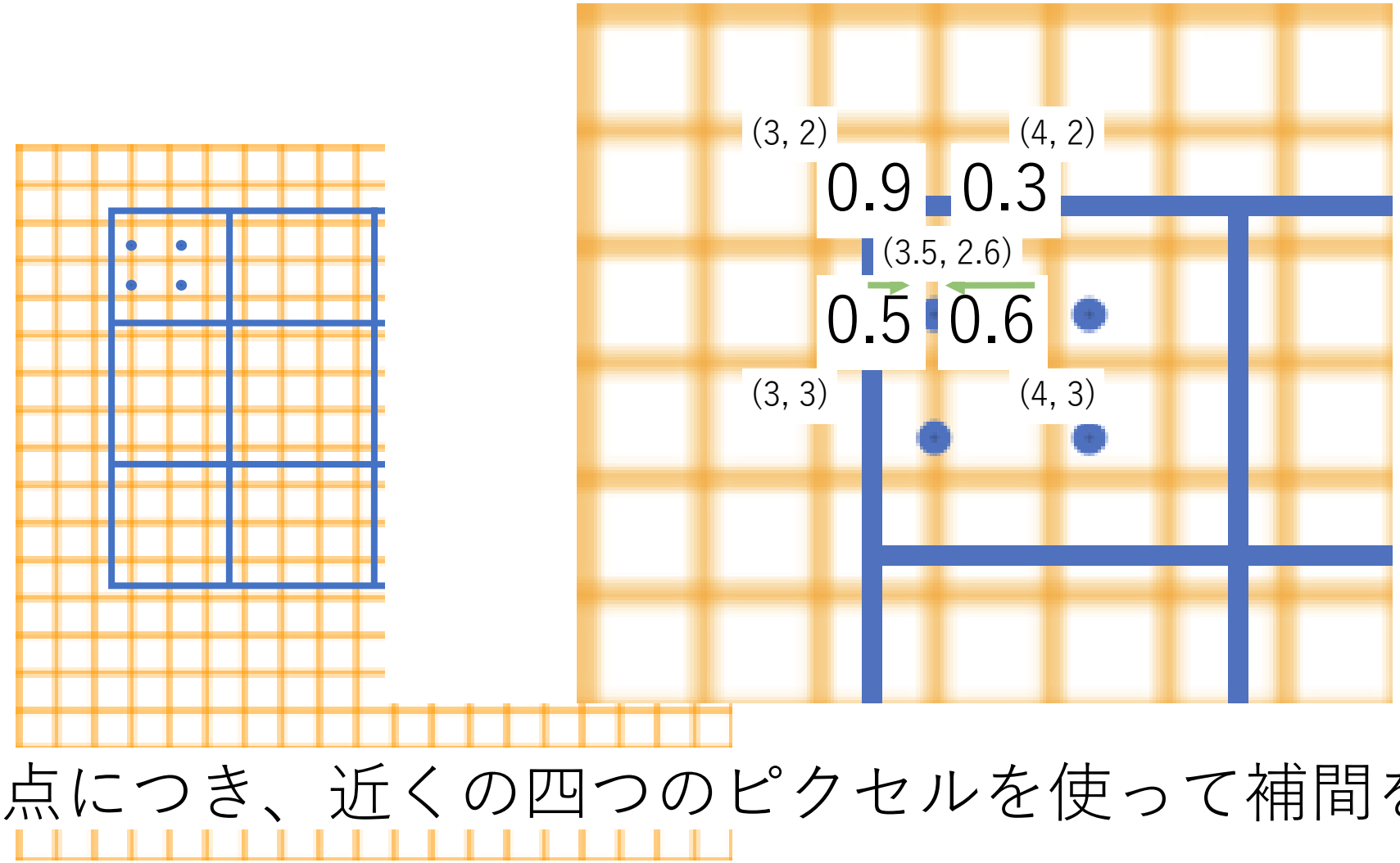


3x3に分割したうちの
一つずつの領域に4つ点を取る

RoI Align

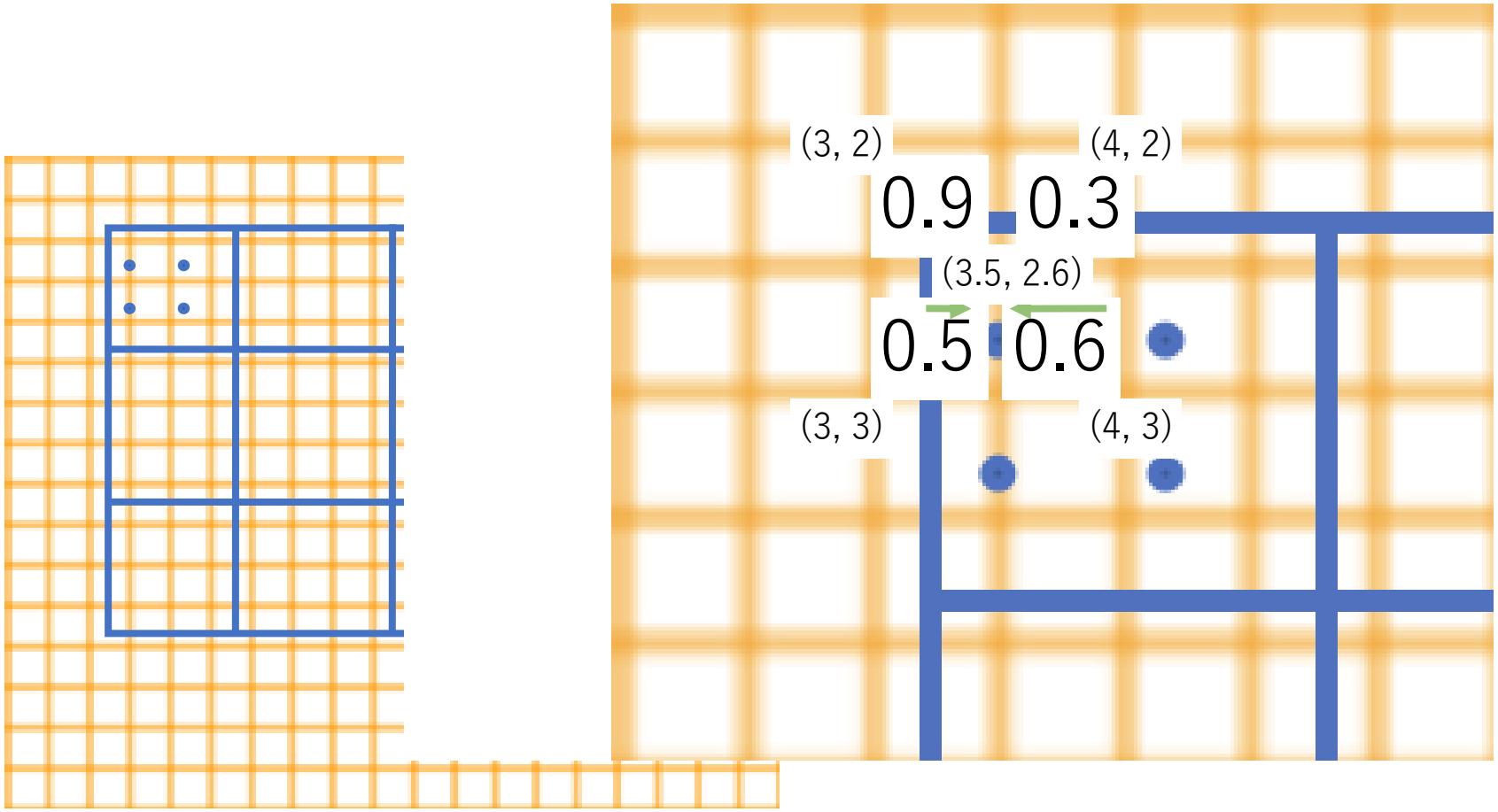


RoI Align



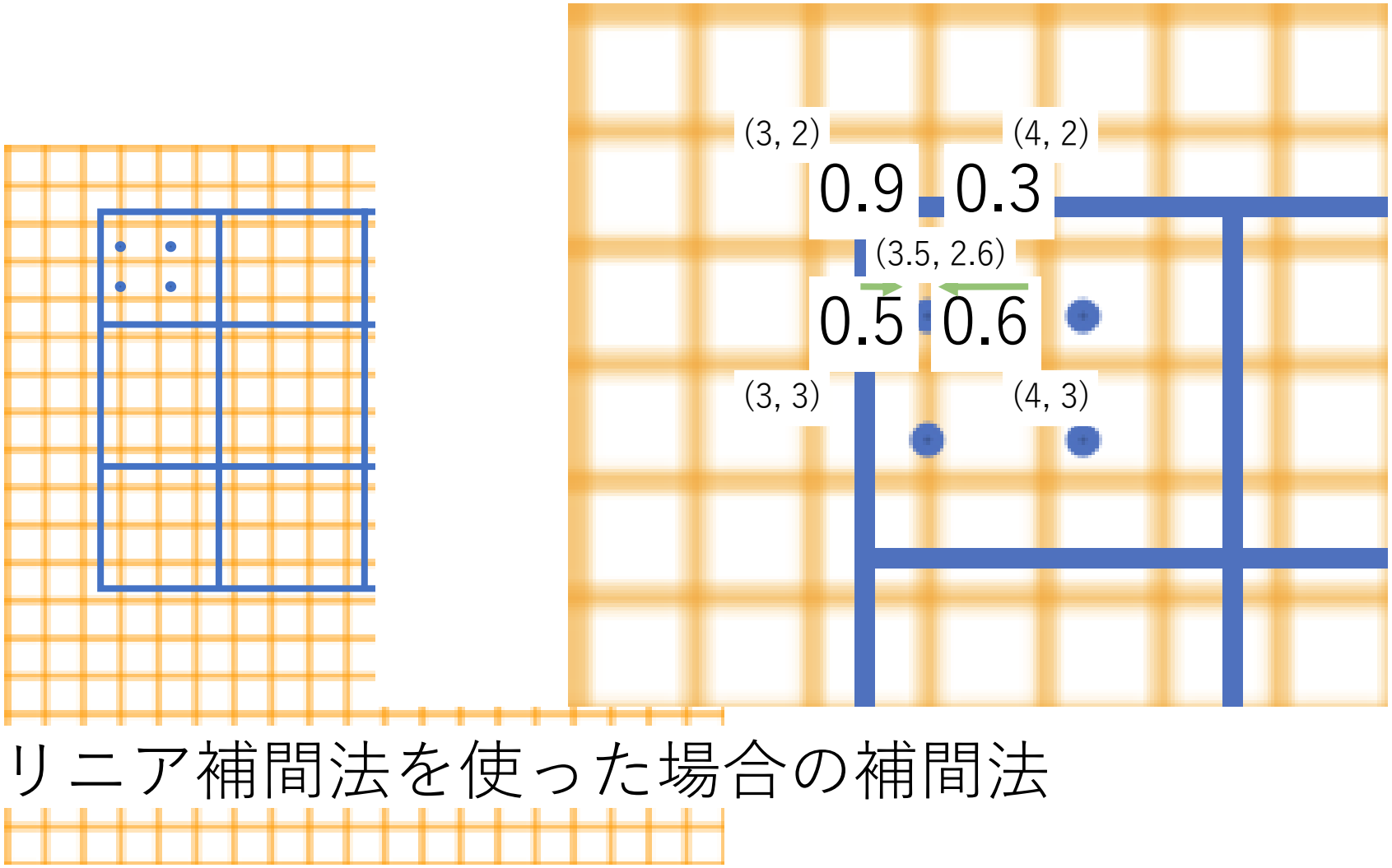
一つの点につき、近くの四つのピクセルを使って補間を行う

RoI Align



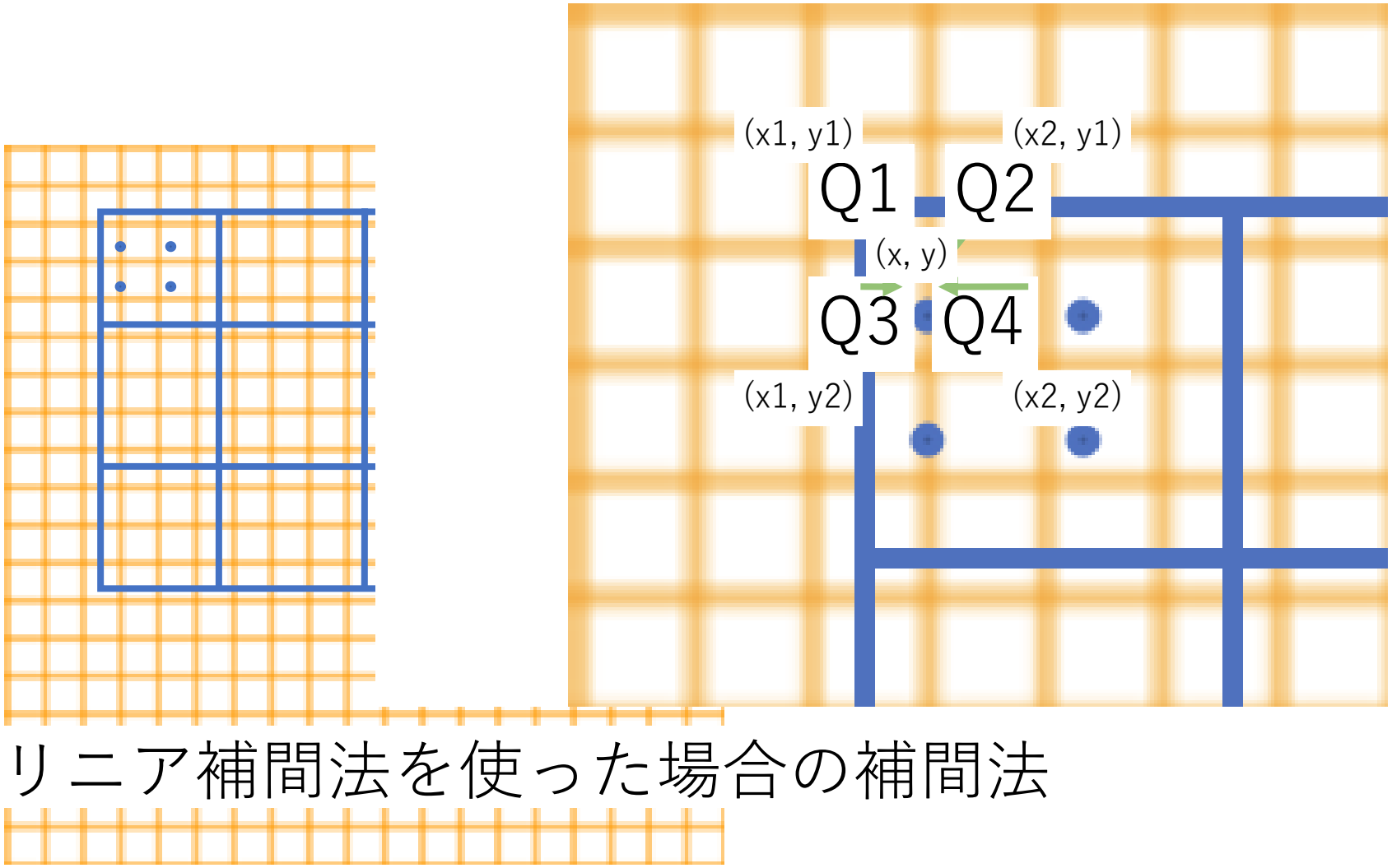
0.9, 0.3, 0.5, 0.6がピクセルの値を示す
(3, 2) (4, 2) (3, 3) (4, 3)がピクセルの座標

RoI Align



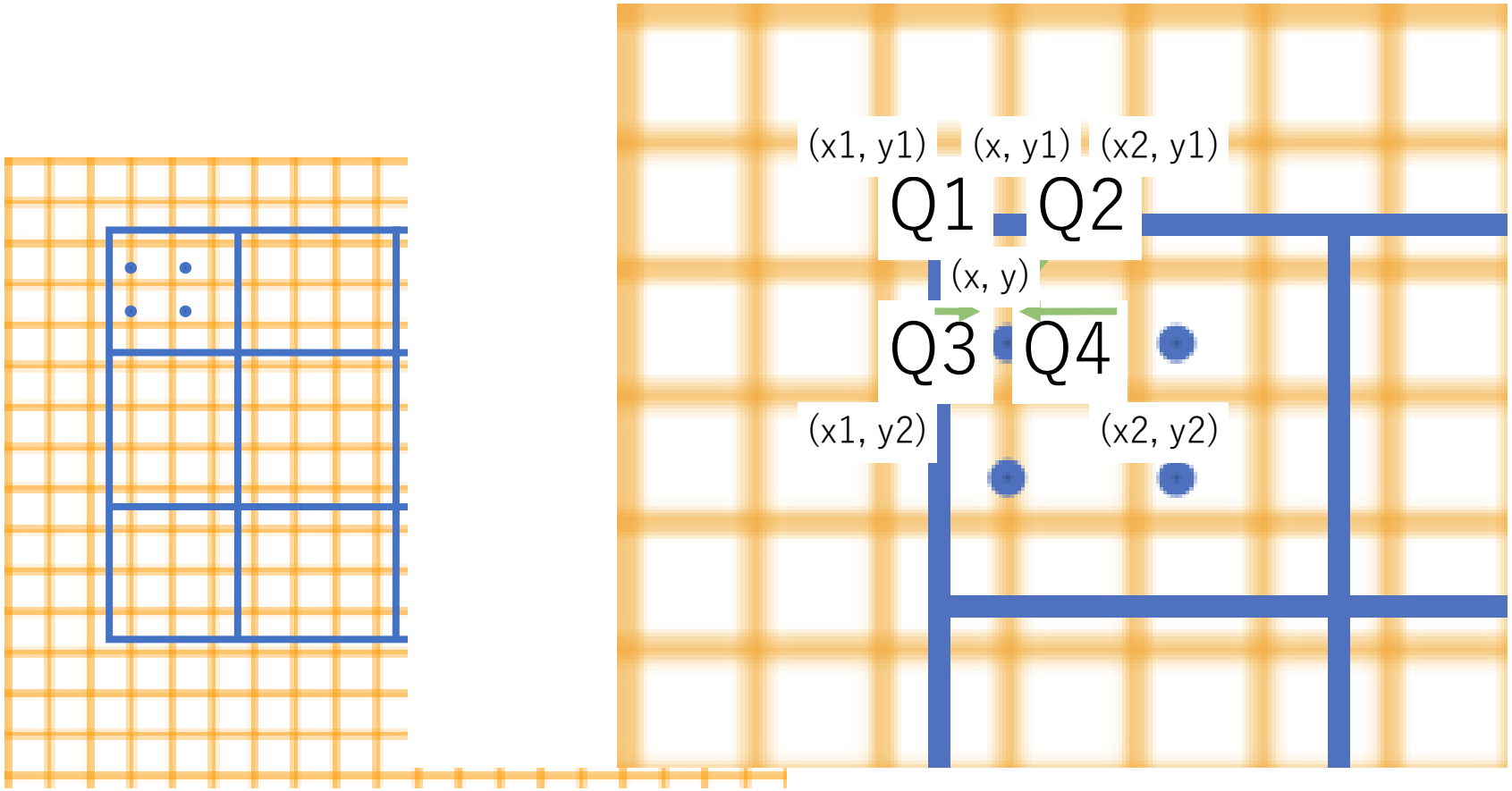
バイリニア補間法を使った場合の補間法

RoI Align



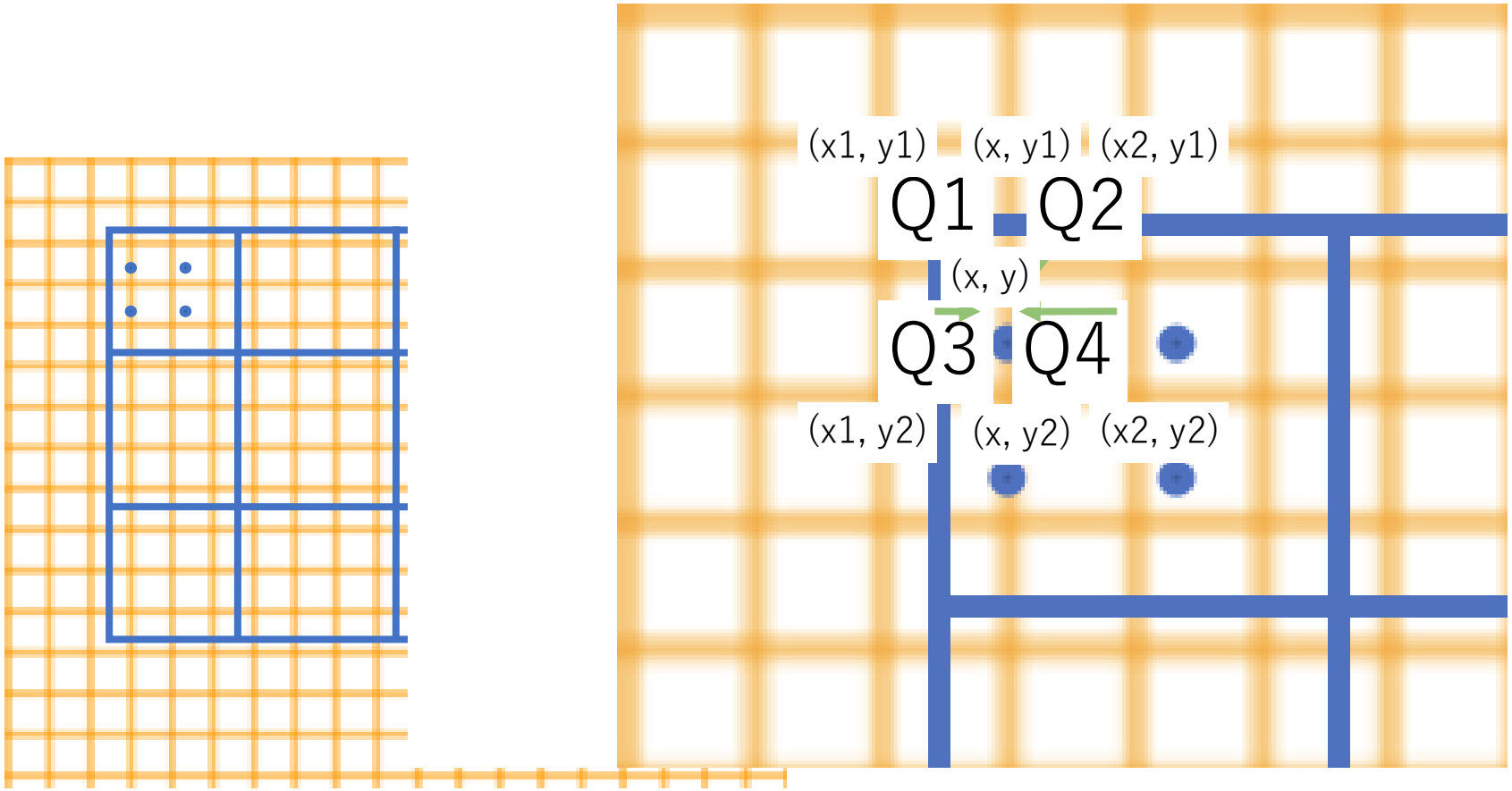
バイリニア補間法を使った場合の補間法

RoI Align



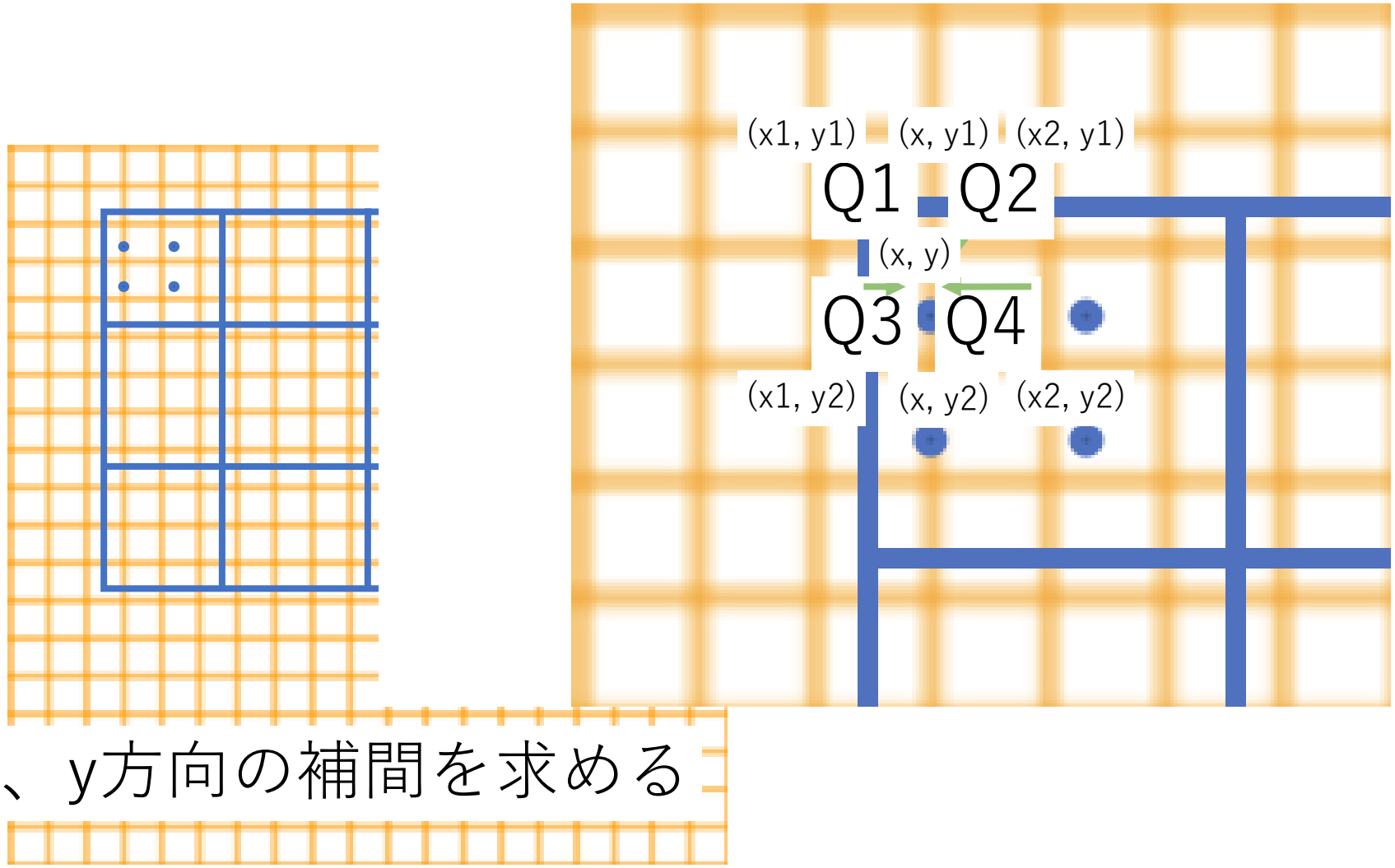
まず、x方向の補間を求める
(x2 - x) / (x2 - x1) * Q1 + (x - x1) / (x2 - x1) * Q2 で
(x, y1) の推定値を求めることができる ＝ f(x, y1)

RoI Align



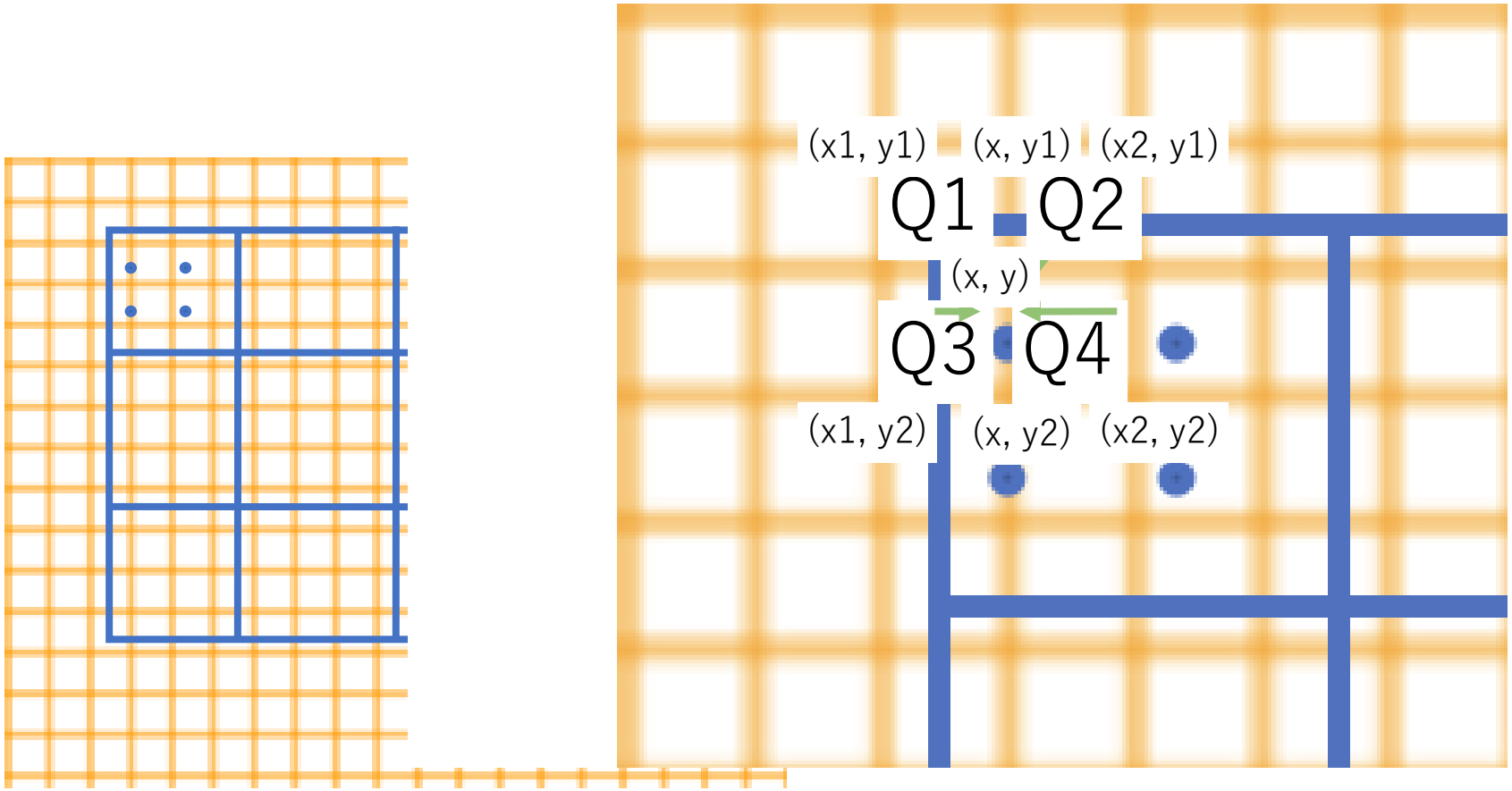
まず、x方向の補間を求める
(x2 - x) / (x2 - x1) * Q3 + (x - x1) / (x2 - x1) * Q4 で
(x, y2) の推定値を求めることができる ＝ f(x, y2)

RoI Align



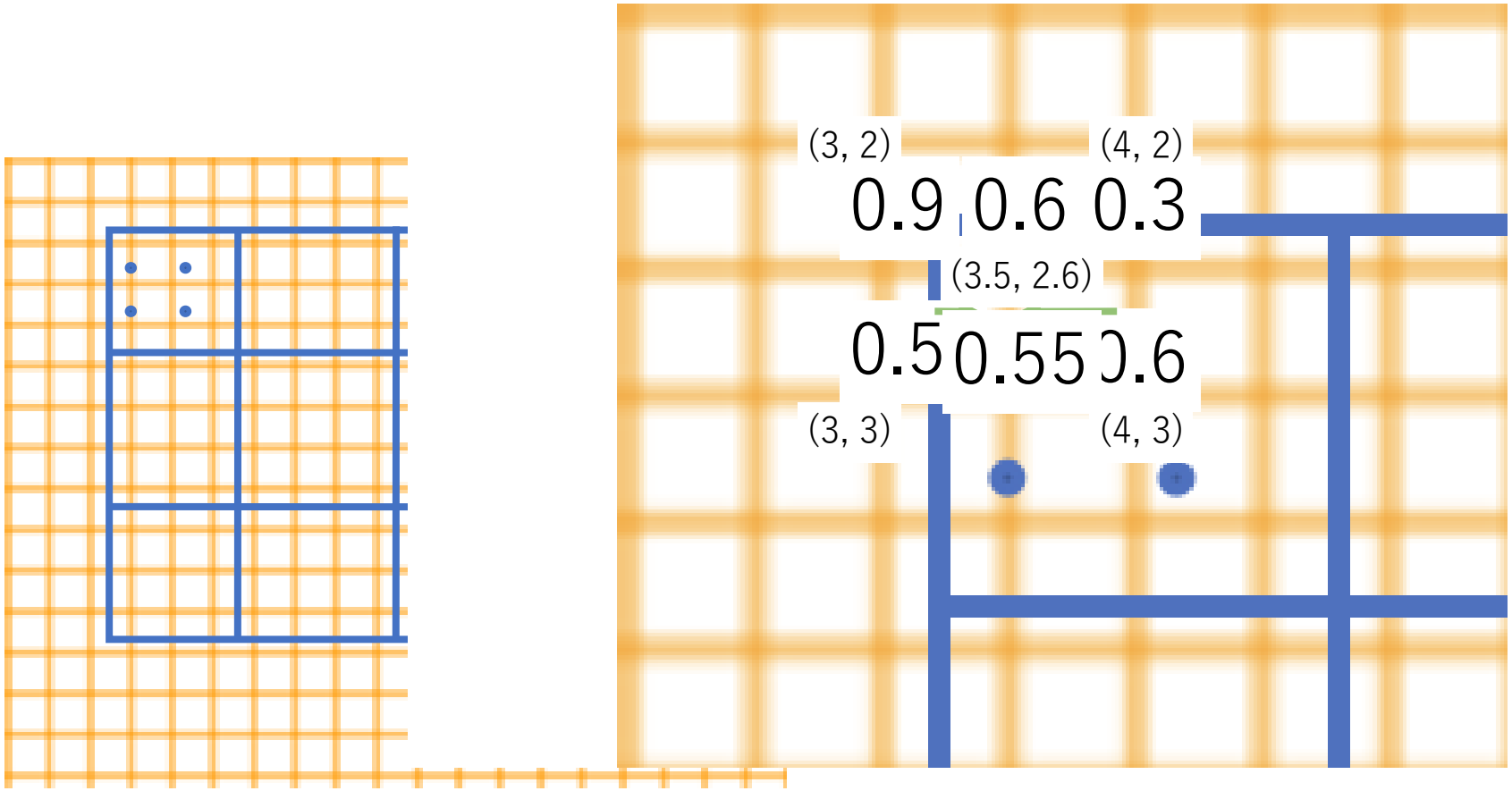
次に、y方向の補間を求める

RoI Align



(y2 - y) / (y2 - y1) * f(x, y1) + (y - y1) / (y2 - y1) * f(x, y2
で 点の推定値 f(x, y)を 求めることができる

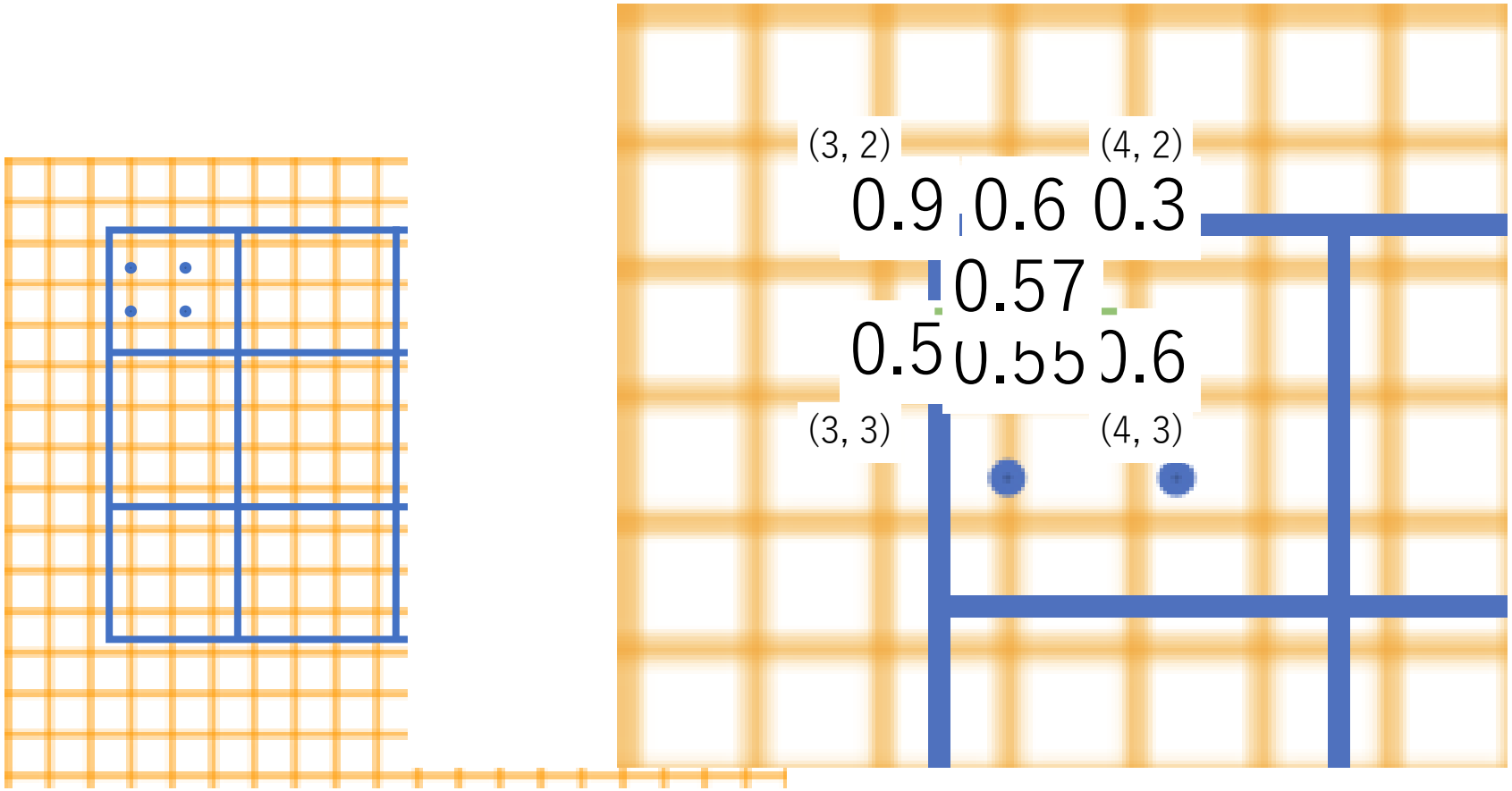
RoI Align



$$(4 - 3.5) / (4 - 3) * 0.9 + (3.5 - 3) / (4 - 3) * 0.3 = 0.6$$

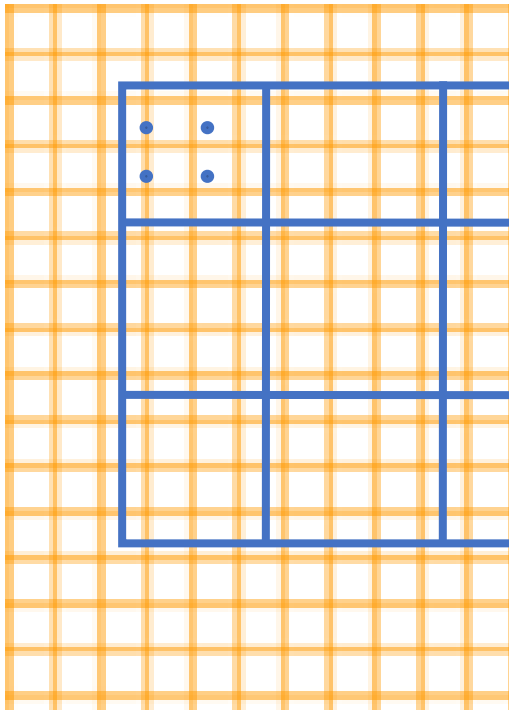
$$(4 - 3.5) / (4 - 3) * 0.5 + (3.5 - 3) / (4 - 3) * 0.6 = 0.55$$

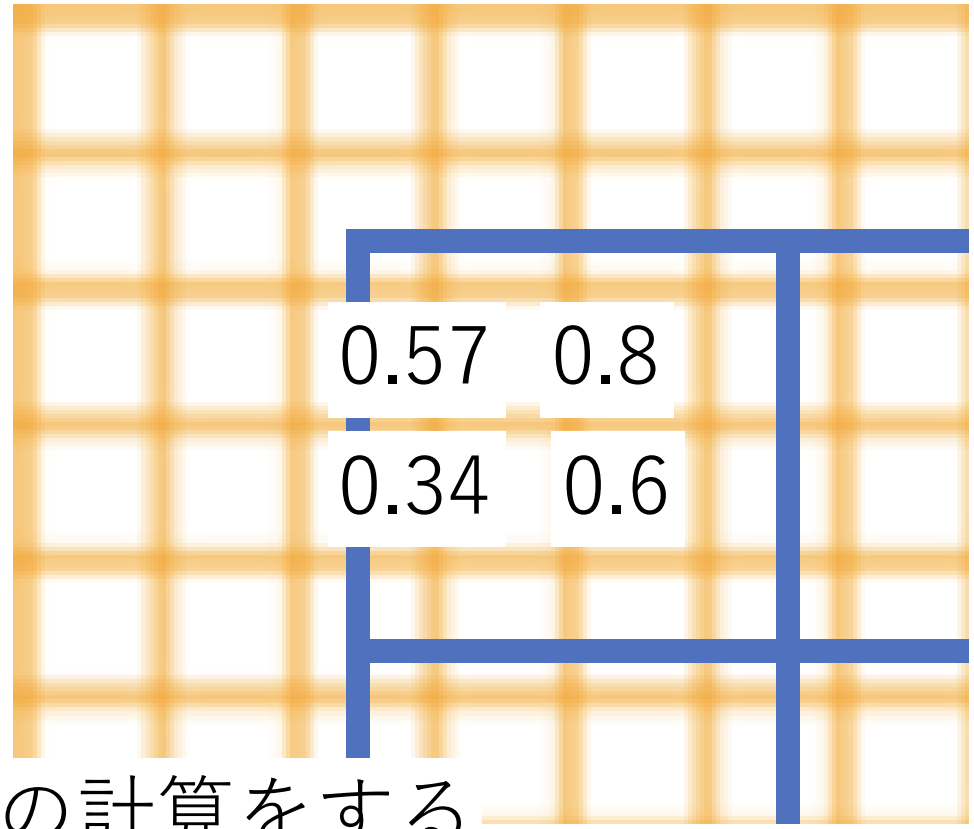
RoI Align



(3 - 2.6) / (3 - 2) * 0.6 + (2.6 - 2) / (3 - 2) * 0.55 = 0.57

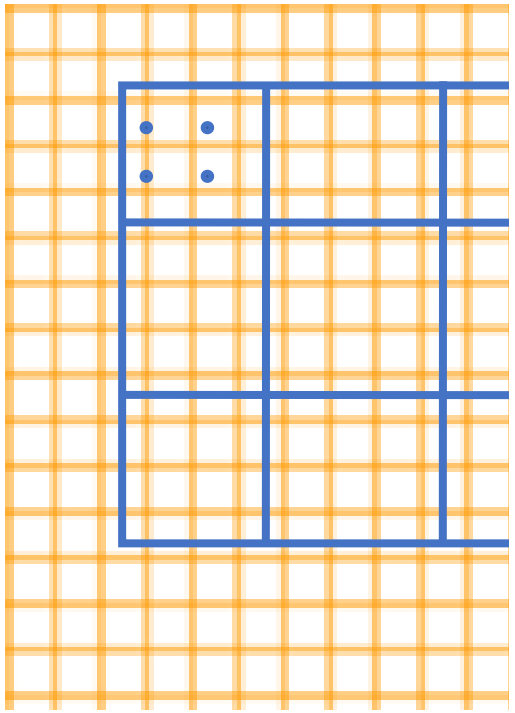
RoI Align

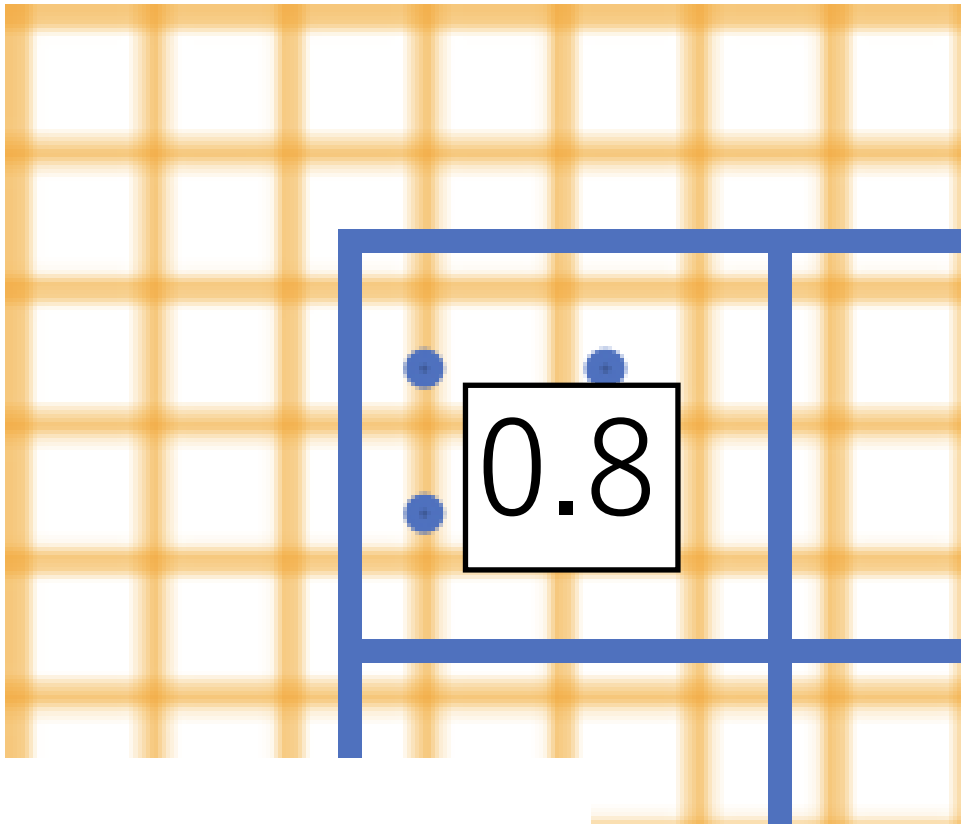




それぞれの点についてその計算をする

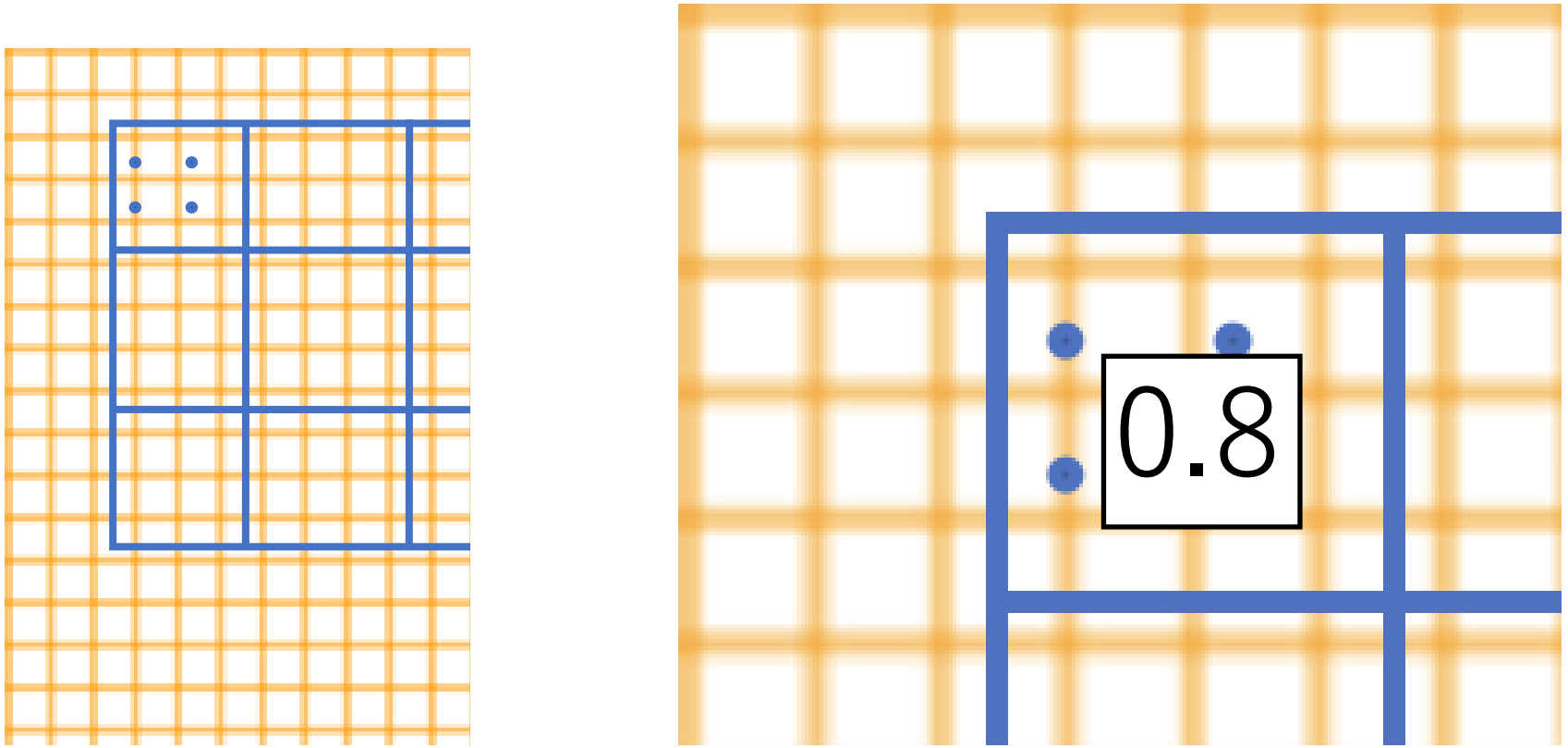
RoI Align





最大値か平均をとる
最大値の場合は0.8　これが領域の代表値

RoI Align



3x3のサイズにしたいであればこれを
それぞれの領域で求め、最終的に9個の代表値ができる

RoI Align

手順まとめ

1. Ｎｘ Ｎの特徴マップにしたい場合、Ｎｘ Ｎの領域に分割する
2. その領域一つ一つについて4つの点を打つ
3. 一つ一つの点について、周りの四つのピクセルを使い、何らかの補間法で点の値を求める
4. 四つの点を一つにまとめる