目標

- 分類問題を深層学習で解決する
- 回帰問題を深層学習で解決する
- ファインチューニングを理解する
- 転移学習を理解する
- 分類の評価方法を理解する：２クラス分類／多クラス分類
- 領域分割の概念を理解する：U-Net,　U-Net++

分類問題のポイント

- 正解がカテゴリ（クラス）である
- 2分類か，３クラス以上の分類かで評価方法が異なる

良悪性分類：2分類

クラス分類：I, II, III, IV, V

2クラスの画像分類

入

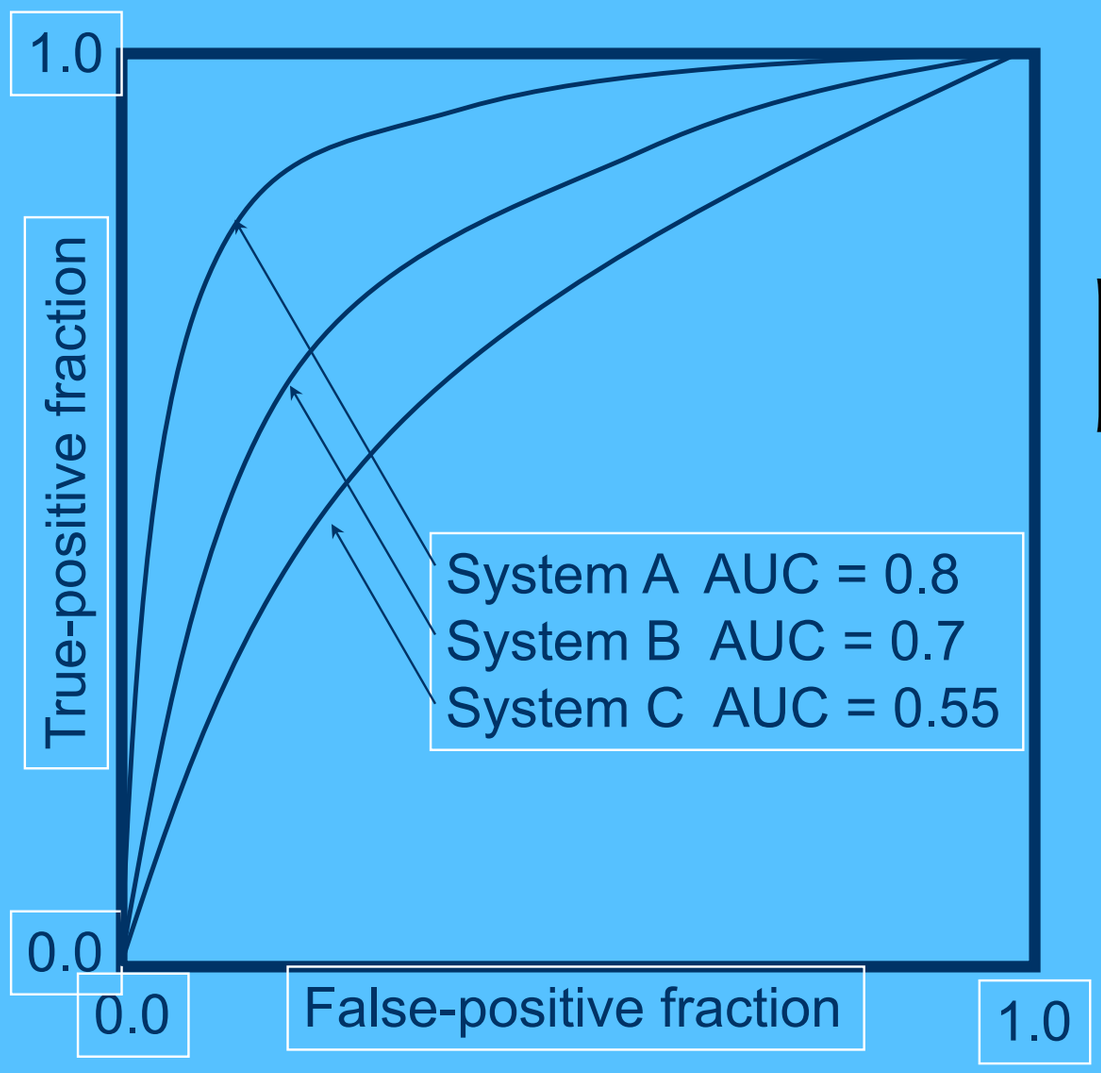
酒

寿司

2クラス分類の例

$x \longrightarrow y$

良性悪性（２分類）



ROC曲線で評価

ROC曲線下面積

カットオフを決める

感度，特異度

論文での取り扱い

JACC: CARDIOVASCULAR IMAGING
VOL. 13, NO. 2, 2020

PUBLISHED BY ELSEVIER

ORIGINAL RESEARCH

A Deep Learning Approach for Assessment of Regional Wall Motion Abnormality From Echocardiographic Images

Kenya Kusunose, MD, PhD,[a] Takashi Abe, MD, PhD,[b] Akihiro Haga, PhD,[c] Daiju Fukuda, MD, PhD,[a] Hirotsugu Yamada, MD, PhD,[a] Masafumi Harada, MD, PhD,[b] Masataka Sata, MD, PhD[a]

ABSTRACT

OBJECTIVES This study investigated whether a deep convolutional neural network (DCNN) could provide improved detection of regional wall motion abnormalities (RWMAs) and differentiate among groups of coronary infarction territories from conventional 2-dimensional echocardiographic images compared with that of cardiologists, sonographers, and resident readers.

BACKGROUND An effective intervention for reduction of misreading of RWMAs is needed. The hypothesis was that a DCNN trained using echocardiographic images would provide improved detection of RWMAs in the clinical setting.

METHODS A total of 300 patients with a history of myocardial infarction were enrolled. From this cohort, 3 groups of 100 patients each had infarctions of the left anterior descending (LAD) artery, the left circumflex (LCX) branch, and the right coronary artery (RCA). A total of 100 age-matched control patients with normal wall motion were selected from a database. Each case contained cardiac ultrasonographs from short-axis views at end-diastolic, mid-systolic, and end-systolic phases. After the DCNN underwent 100 steps of training, diagnostic accuracies were calculated from the test set. Independently, 10 versions of the same model were trained, and ensemble predictions were performed using those versions.

x

超音波画像３枚

y

局所壁運動

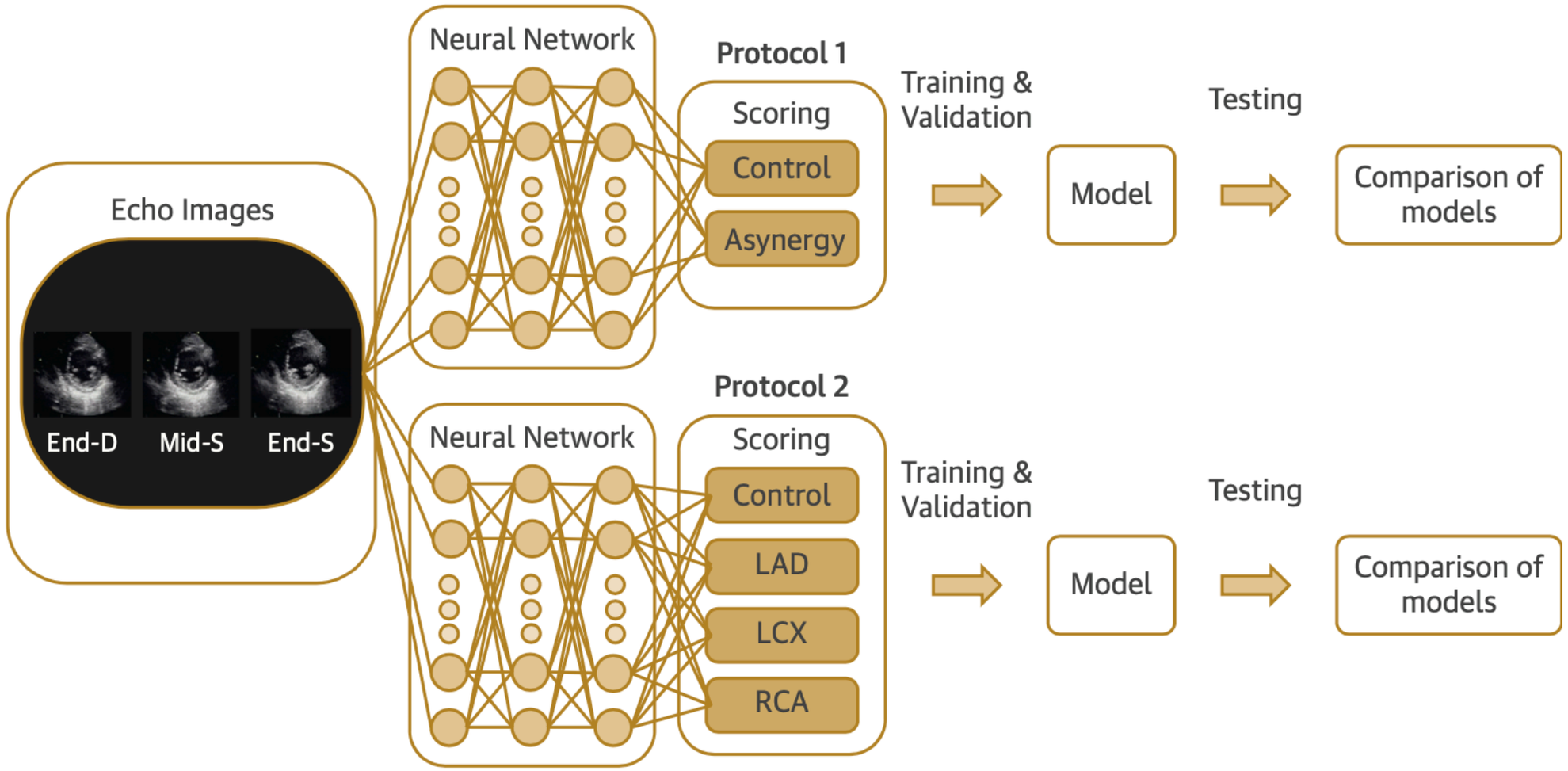
異常の有無

（２分類）

$n = 300$

Kusunose K, Abe T, Haga A, Fukuda D, Yamada H, Harada M, Sata M. A Deep Learning Approach for Assessment of Regional Wall Motion Abnormality From Echocardiographic Images. JACC Cardiovasc Imaging. 2020 Feb;13(2 Pt 1):374-381.
doi: 10.1016/j.jcmg.2019.02.024. Epub 2019 May 15. PMID: 31103590.

CENTRAL ILLUSTRATION **Neural Networks for the Presence of RWMAs and the Territory of RWMAs**

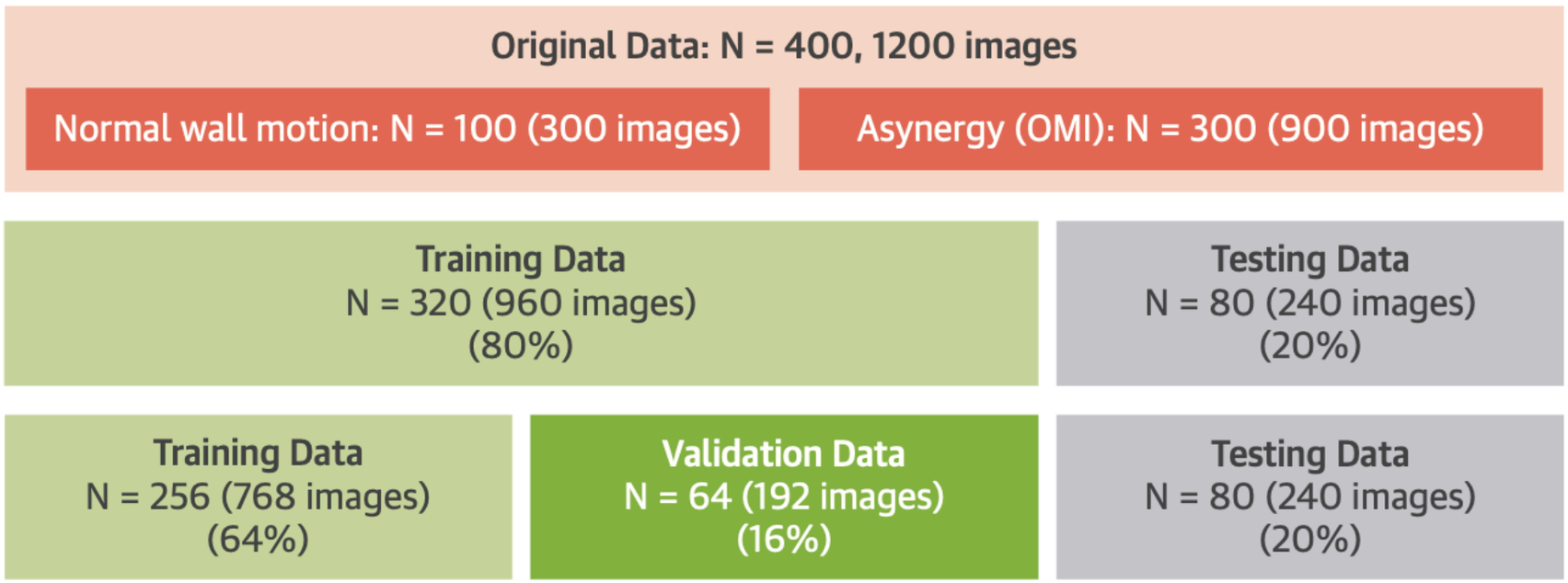


Kusunose, K. et al. J Am Coll Cardiol Img. 2020;13(2):374-81.

The fully connected layers transform the image features into the final scores by adjusting weights for neuron activations during training. Echo = echocardiography; End-D = end diastolic (phase); LAD = left anterior descending artery; LCX = left circumflex artery; Mid/End-S = mid-/end systolic (phases); RCA = right coronary artery; RWMA = regional wall motion abnormality.

Kusunose K, Abe T, Haga A, Fukuda D, Yamada H, Harada M, Sata M. A Deep Learning Approach for Assessment of Regional Wall Motion Abnormality From Echocardiographic Images. JACC Cardiovasc Imaging. 2020 Feb;13(2 Pt 1):374-381. doi: 10.1016/j.jcmg.2019.02.024. Epub 2019 May 15. PMID: 31103590.

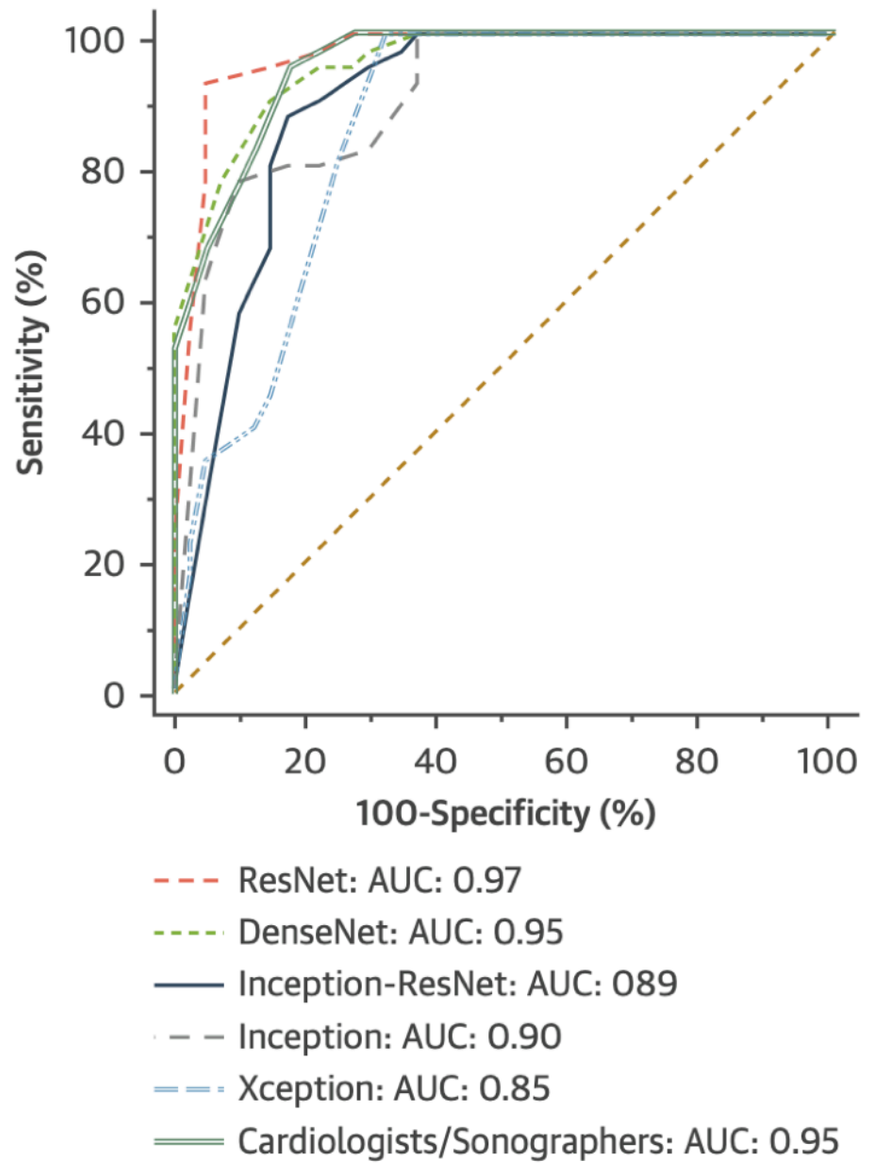
FIGURE 1 **Import Data**



There were a total of 400 cases from which 1,200 images were split with 256 cases (786 images) as the training set, 64 cases (192 images) as the validation set, and 80 cases (240 images) as the test set. OMI = old myocardial infarction.

Kusunose K, Abe T, Haga A, Fukuda D, Yamada H, Harada M, Sata M. A Deep Learning Approach for Assessment of Regional Wall Motion Abnormality From Echocardiographic Images. JACC Cardiovasc Imaging. 2020 Feb;13(2 Pt 1):374-381.
doi: 10.1016/j.jcmg.2019.02.024. Epub 2019 May 15. PMID: 31103590.

FIGURE 3 **Diagnostic Ability to Detect the Presence of RWMAs**



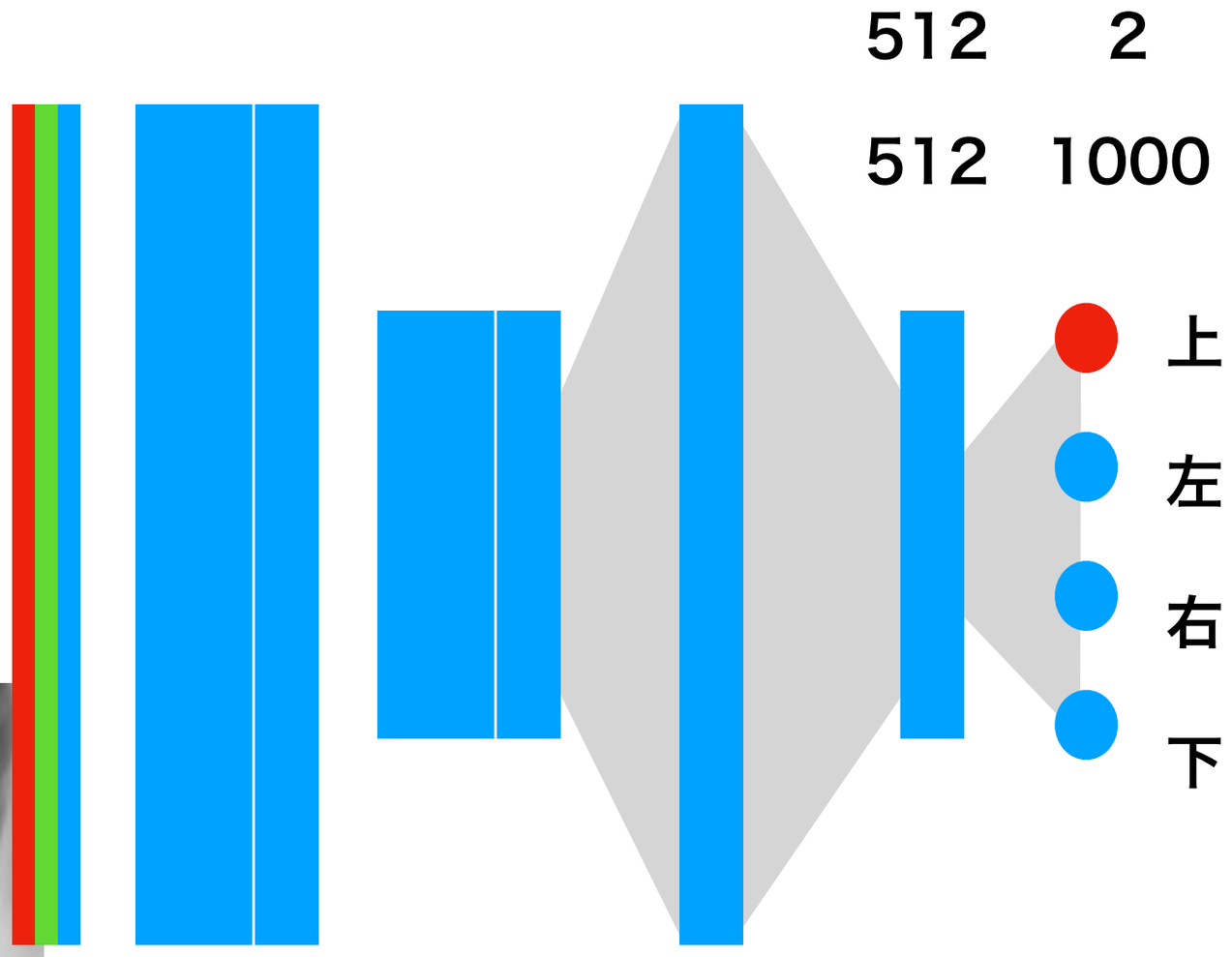
The area under the curves by several deep learning algorithms for detection of territories of wall motion abnormality were good. AUC = area under the curve; ResNet = residual neural network; other abbreviation as in **Figure 2**.

Kusunose K, Abe T, Haga A, Fukuda D, Yamada H, Harada M, Sata M. A Deep Learning Approach for Assessment of Regional Wall Motion Abnormality From Echocardiographic Images. JACC Cardiovasc Imaging. 2020 Feb;13(2 Pt 1):374-381. doi: 10.1016/j.jcmg.2019.02.024. Epub 2019 May 15. PMID: 31103590.

４方向の分類問題

4クラスの画像分類


入


512 2
512 1000
上
左
右
下

多クラス分類の例

$x \longrightarrow y$

ステージ（多分類）

反応

真値 \	I	II	III	IV	V	VI
I	26	2	4	6	2	2
II	2	34	5	3	10	2
III	4	5	44	6	2	1
IV	5	2	1	26	4	6
V	11	3	6	12	16	8
VI	6	3	17	2	19	86

混同行列で評価

精度＝（対角成分）／全体

分類ごとの精度

論文での取り扱い

Circulation

ORIGINAL RESEARCH ARTICLE

Fully Automated Echocardiogram Interpretation in Clinical Practice

Feasibility and Diagnostic Accuracy

Editorials, see p 1636 and p 1639

BACKGROUND: Automated cardiac image interpretation has the potential to transform clinical practice in multiple ways, including enabling serial assessment of cardiac function by nonexperts in primary care and rural settings. We hypothesized that advances in computer vision could enable building a fully automated, scalable analysis pipeline for echocardiogram interpretation, including (1) view identification, (2) image segmentation, (3) quantification of structure and function, and (4) disease detection.

METHODS: Using 14035 echocardiograms spanning a 10-year period, we trained and evaluated convolutional neural network models for multiple tasks, including automated identification of 23 viewpoints and segmentation of cardiac chambers across 5 common views. The segmentation output was used to quantify chamber volumes and left ventricular mass, determine ejection fraction, and facilitate automated determination of longitudinal strain through speckle tracking. Results were evaluated through comparison to manual segmentation and measurements from 8666 echocardiograms obtained during the routine clinical workflow. Finally, we developed models to detect 3 diseases: hypertrophic cardiomyopathy, cardiac amyloid, and pulmonary arterial hypertension.

Jeffrey Zhang, BA
Sravani Gajjala, MBBS
Pulkit Agrawal, PhD
Geoffrey H. Tison, MD, MPH
Laura A. Hallock, BS
Lauren Beussink-Nelson, RDCS
Mats H. Lassen, BM
Eugene Fan, MD
Mandar A. Aras, MD, PhD
ChaRandle Jordan, MD, PhD
Kirsten E. Fleischmann, MD, MPH
Michelle Melisko, MD
Atif Qasim, MD, MSCE
Sanjiv J. Shah, MD
Ruzena Bajcsy, PhD
Rahul C. Deo, MD, PhD

x

超音波画像１枚

y

分割画像

多クラス分類

$n = 14035$

Zhang J, Gajjala S, Agrawal P, Tison GH, Hallock LA, Beussink-Nelson L, Lassen MH, Fan E, Aras MA, Jordan C, Fleischmann KE, Melisko M, Qasim A, Shah SJ, Bajcsy R, Deo RC. Fully Automated Echocardiogram Interpretation in Clinical Practice. Circulation. 2018 Oct 16;138(16):1623-1635. doi: 10.1161/CIRCULATIONAHA.118.034338. PMID: 30354459; PMCID: PMC6200386.

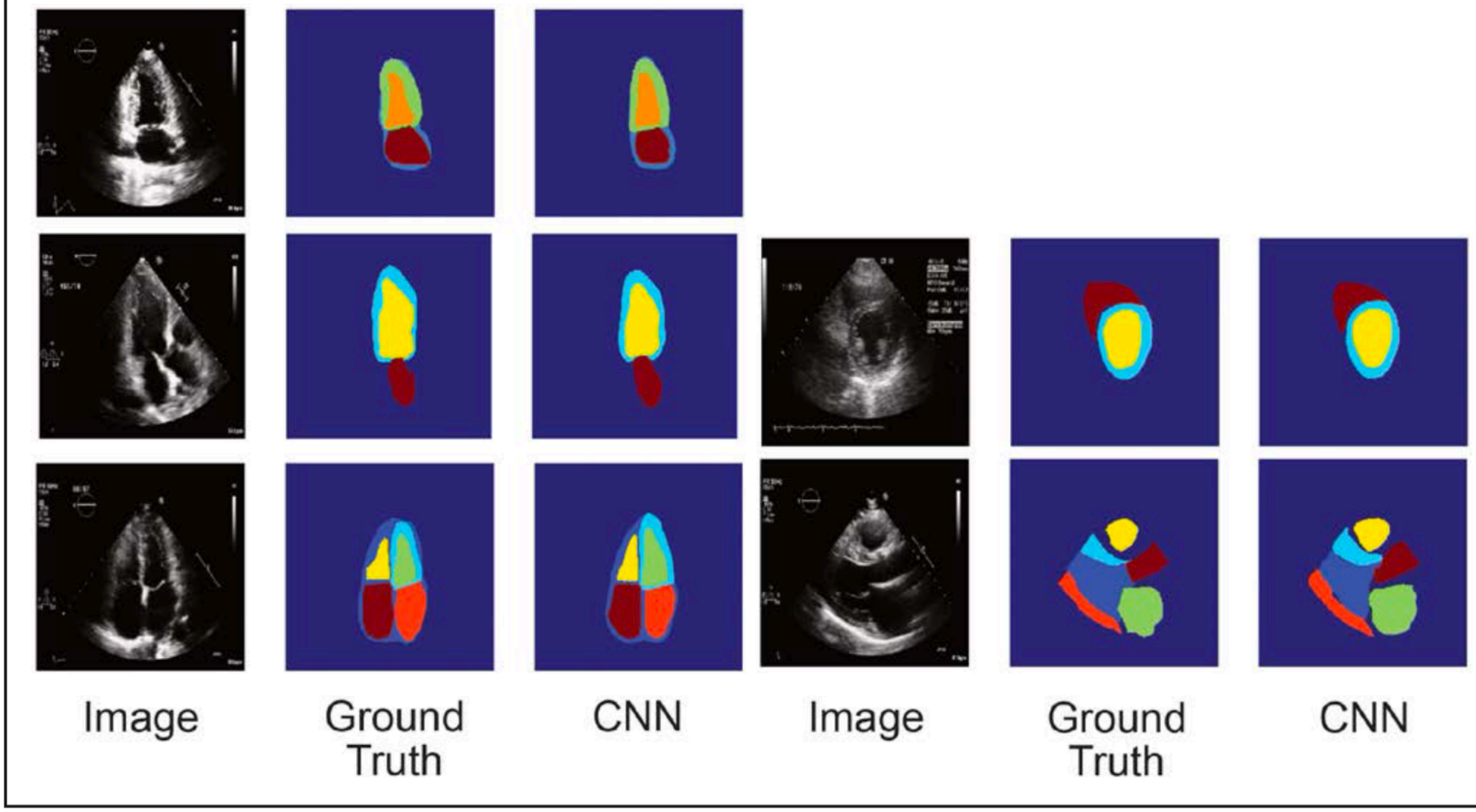


Figure 3. Convolutional neural networks successfully segment cardiac chambers.
We used the U-net algorithm to derive segmentation models for 5 views: A2c, A3c, A4c (left: top, middle, and bottom, respectively), parasternal short axis at the level of the papillary muscle (right, middle), and PLAX (right, bottom). For each view, the trio of images, from left to right, corresponds to the original image, the manually traced image used in training (ground truth), and the automated segmented image (determined as part of the cross-validation process). A2c indicates apical 2-chamber; A3c, apical 3-chamber; A4c, apical 4-chamber; CNN, convolutional neural network; and PLAX, parasternal long axis.

Zhang J, Gajjala S, Agrawal P, Tison GH, Hallock LA, Beussink-Nelson L, Lassen MH, Fan E, Aras MA, Jordan C, Fleischmann KE, Melisko M, Qasim A, Shah SJ, Bajcsy R, Deo RC. Fully Automated Echocardiogram Interpretation in Clinical Practice. Circulation. 2018 Oct 16;138(16):1623-1635. doi: 10.1161/CIRCULATIONAHA.118.034338. PMID: 30354459; PMCID: PMC6200386.

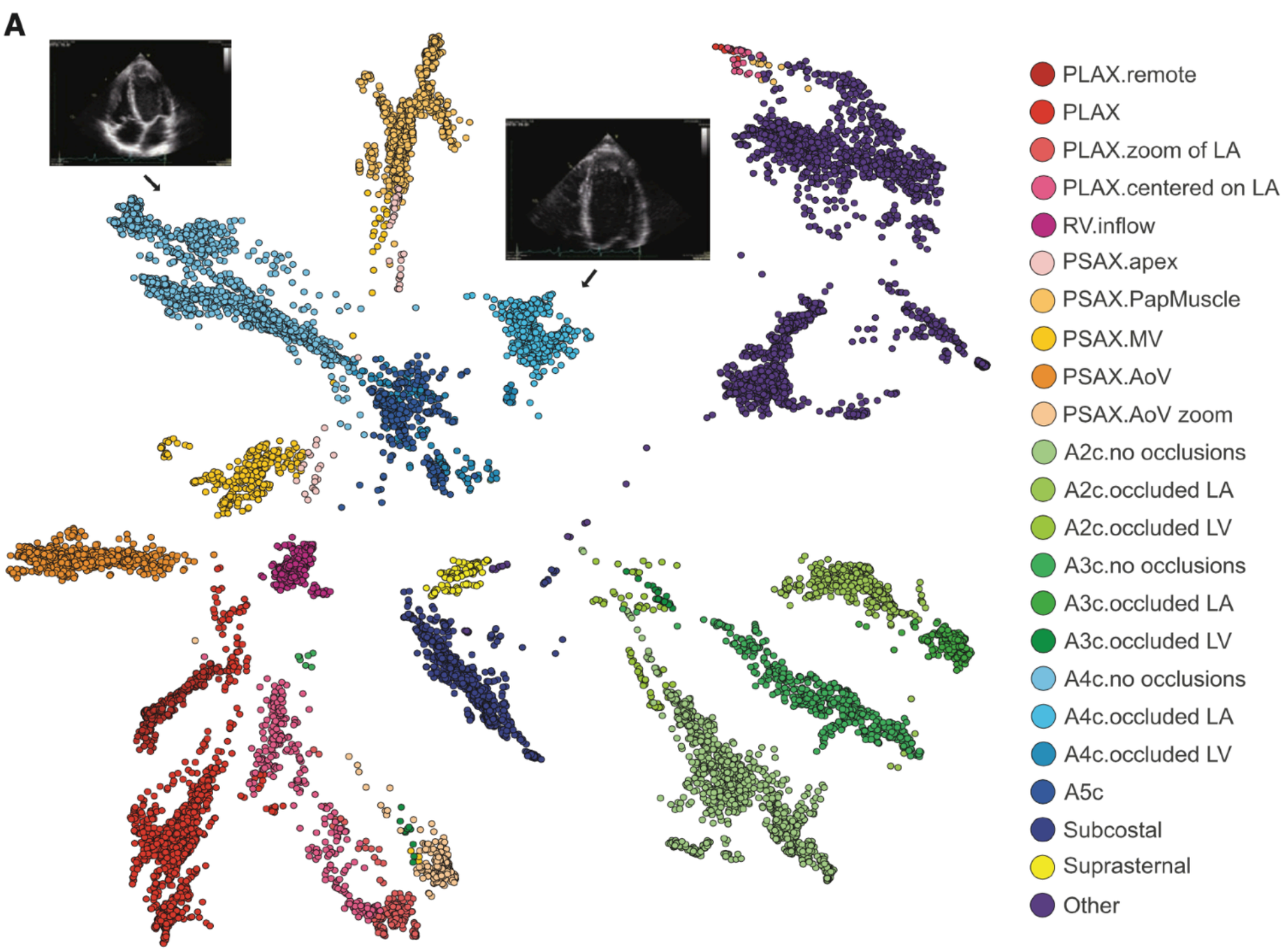


Figure 2. Convolutional neural networks successfully discriminate echocardiographic views.
A, t-Distributed Stochastic Neighbor Embedding (t-SNE) visualization of view classification. t-SNE is an algorithm used to visualize high-dimensional data in lower dimensions. It depicts the successful grouping of test images corresponding to 23 different echocardiographic views. Echocardiographic still images indicate the distinct clustering of images of A4c views without occlusions and those with occlusion of the left atrium. **B**, Confusion matrix demonstrating successful and unsuccessful view classifications within the test data set. Numbers along the diagonal represent successful classifications, whereas off-diagonal entries are misclassifications. A2c indicates apical 2-chamber; A3c, apical 3-chamber; A4c, apical 4-chamber; echo, echocardiogram; LV, left ventricular; and PLAX, parasternal long axis.

Zhang J, Gajjala S, Agrawal P, Tison GH, Hallock LA, Beussink-Nelson L, Lassen MH, Fan E, Aras MA, Jordan C, Fleischmann KE, Melisko M, Qasim A, Shah SJ, Bajcsy R, Deo RC. Fully Automated Echocardiogram Interpretation in Clinical Practice. Circulation. 2018 Oct 16;138(16):1623-1635. doi: 10.1161/CIRCULATIONAHA.118.034338. PMID: 30354459; PMCID: PMC6200386.

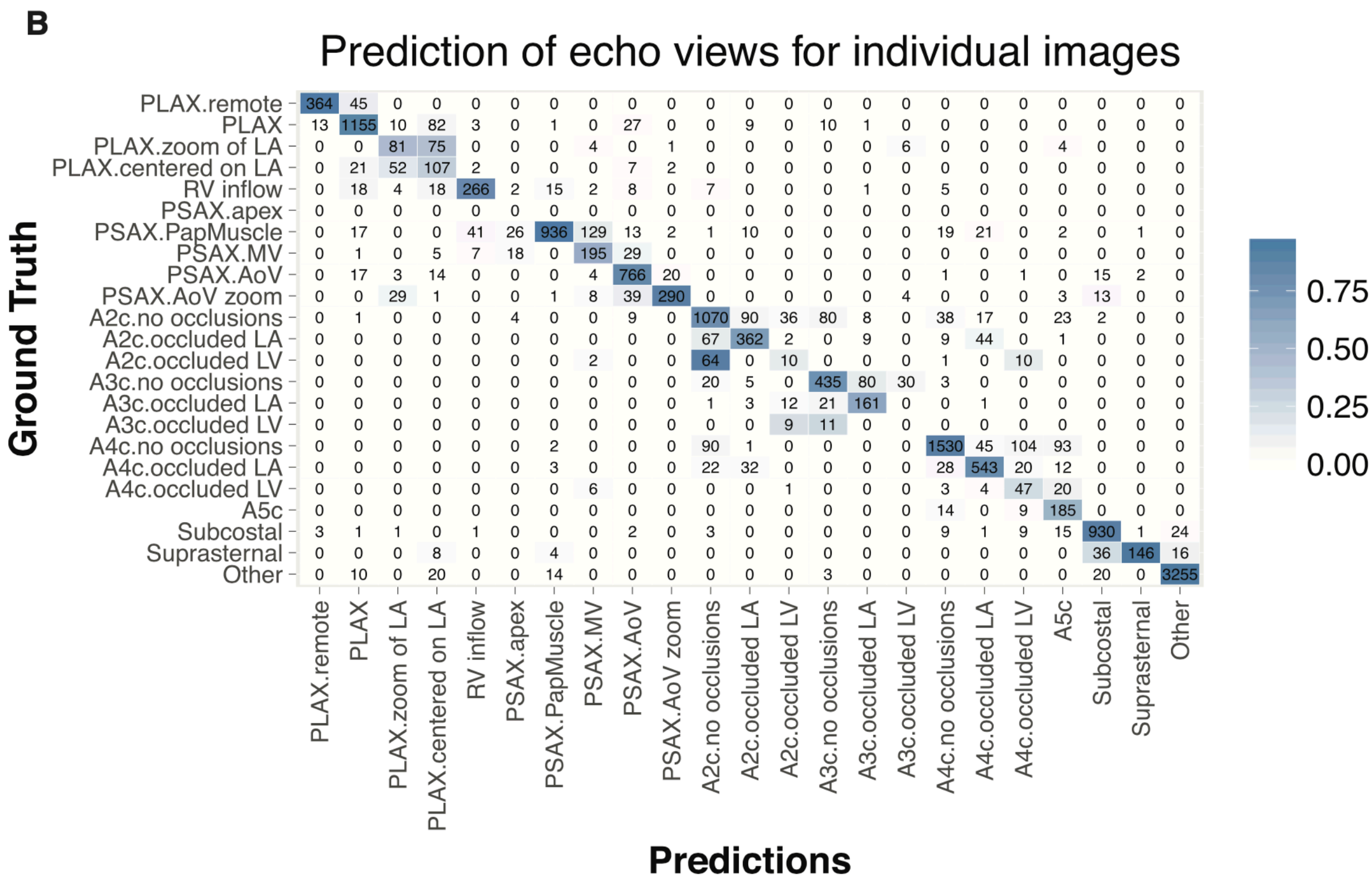


Figure 2. Convolutional neural networks successfully discriminate echocardiographic views.
A, t-Distributed Stochastic Neighbor Embedding (t-SNE) visualization of view classification. t-SNE is an algorithm used to visualize high-dimensional data in lower dimensions. It depicts the successful grouping of test images corresponding to 23 different echocardiographic views. Echocardiographic still images indicate the distinct clustering of images of A4c views without occlusions and those with occlusion of the left atrium. **B**, Confusion matrix demonstrating successful and unsuccessful view classifications within the test data set. Numbers along the diagonal represent successful classifications, whereas off-diagonal entries are misclassifications. A2c indicates apical 2-chamber; A3c, apical 3-chamber; A4c, apical 4-chamber; echo, echocardiogram; LV, left ventricular; and PLAX, parasternal long axis.

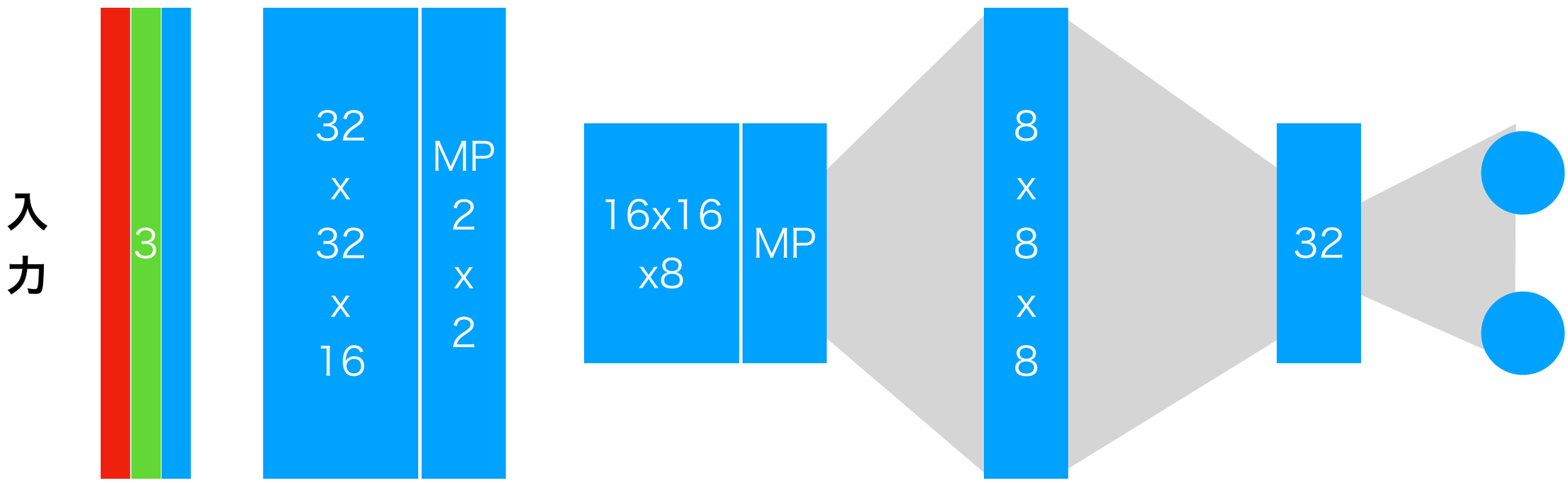
Zhang J, Gajjala S, Agrawal P, Tison GH, Hallock LA, Beussink-Nelson L, Lassen MH, Fan E, Aras MA, Jordan C, Fleischmann KE, Melisko M, Qasim A, Shah SJ, Bajcsy R, Deo RC. Fully Automated Echocardiogram Interpretation in Clinical Practice. Circulation. 2018 Oct 16;138(16):1623-1635. doi: 10.1161/CIRCULATIONAHA.118.034338. PMID: 30354459; PMCID: PMC6200386.

モデルの構築

構築するモデル

```
class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        self.conv1 = nn.Conv2d(3, 16, kernel_size=3, padding=1) # 32x32x3ch -> 16x16x16
        self.conv2 = nn.Conv2d(16, 8, kernel_size=3, padding=1) # 16x16x16 -> 8x8x8
        self.fc1 = nn.Linear(8 * 8 * 8, 32) # 8x8x8 -> 32
        self.fc2 = nn.Linear(32, 2) # 32 -> 2

    def forward(self, x):
        out = F.max_pool2d(torch.relu(self.conv1(x)), 2)
        out = F.max_pool2d(torch.relu(self.conv2(out)), 2)
        out = out.view(-1, 8 * 8 * 8)
        out = torch.tanh(self.fc1(out))
        out = self.fc2(out)
        return out
```



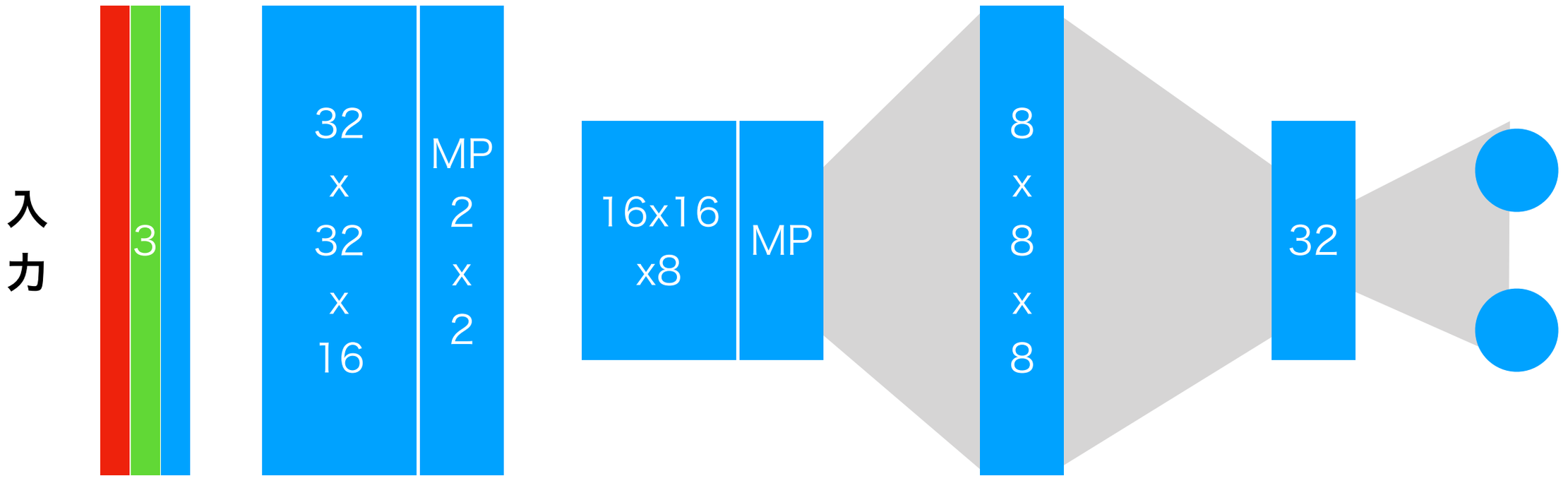
構築するモデル

```
class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        self.conv1 = nn.Conv2d(3, 16, kernel_size=3, padding=1)
        self.conv2 = nn.Conv2d(16, 8, kernel_size=3, padding=1)
        self.fc1 = nn.Linear(8 * 8 * 8, 32)
        self.fc2 = nn.Linear(32, 2)

    def forward(self, x):
        out = F.max_pool2d(torch.relu(self.conv1(x)), 2)
        out = F.max_pool2d(torch.relu(self.conv2(out)), 2)
        out = out.view(-1, 8 * 8 * 8)
        out = torch.tanh(self.fc1(out))
        out = self.fc2(out)
        return out
```

部品を書く部分

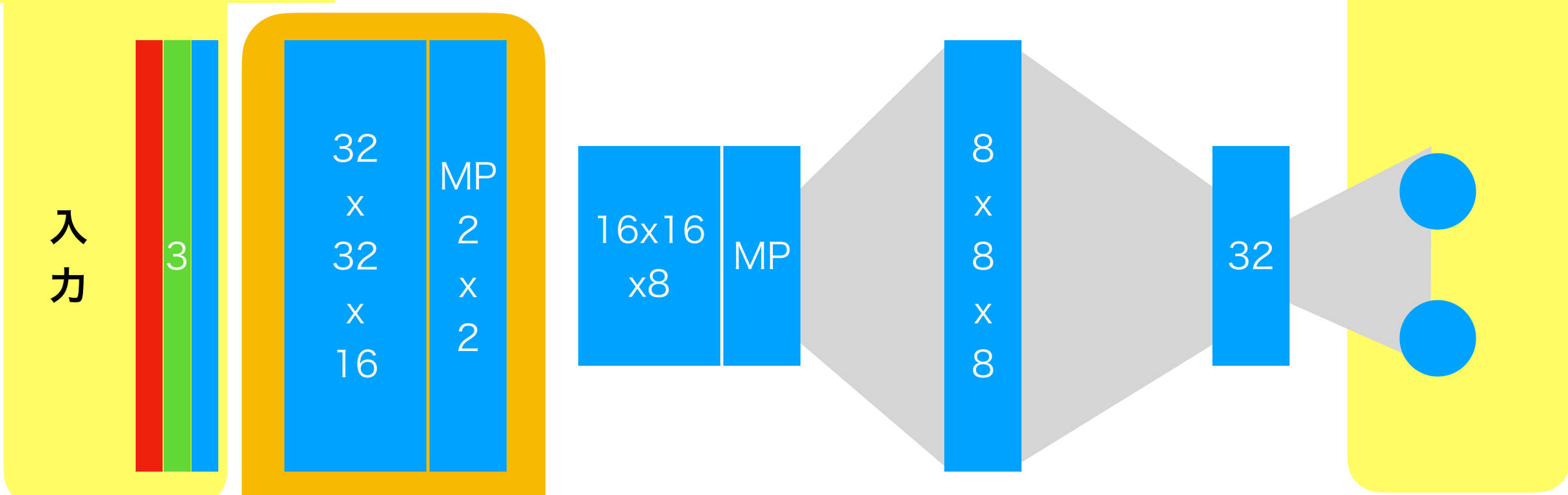
処理順を書く部分



構築するモデル

画像が入力される
32x32x3chにする

2分類なので
出力は２つ



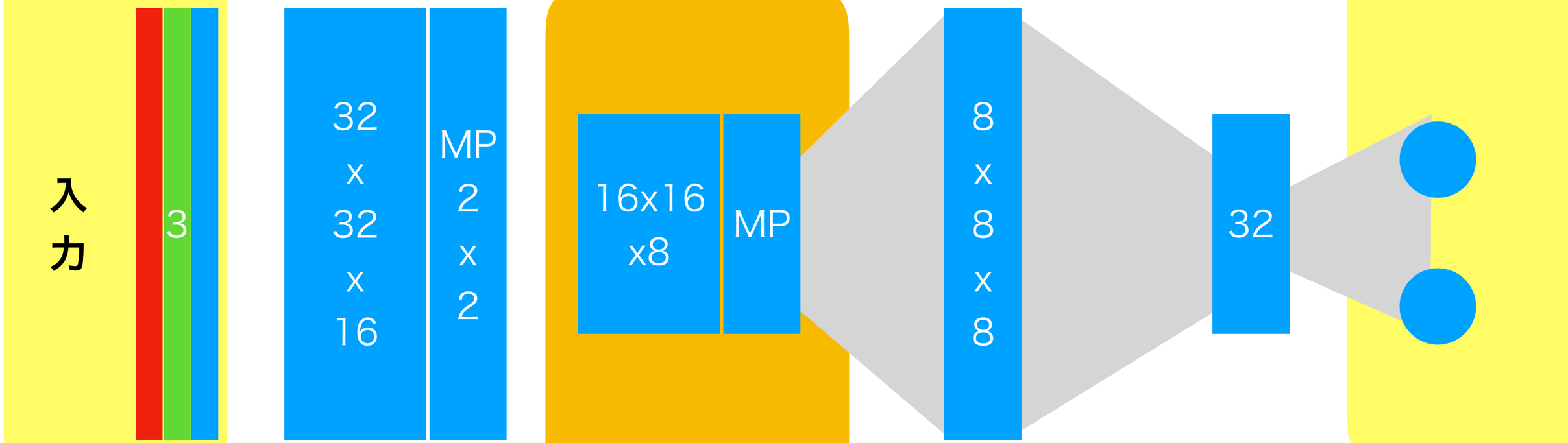
nn.Conv2d(3, 16, kernel_size=3, padding=1)
3chの入力を3x3のカーネルで16chにする
画素数はそのまま
カーネルのパラメタは学習で決定する

そのあとMaxPoolingで2x2画素の最大値を抽出する
2x2x16の中から16種類の最大値が抽出される
その結果16x16画素，16chになる

構築するモデル

画像が入力される
32x32x3chにする

2分類なので
出力は２つ



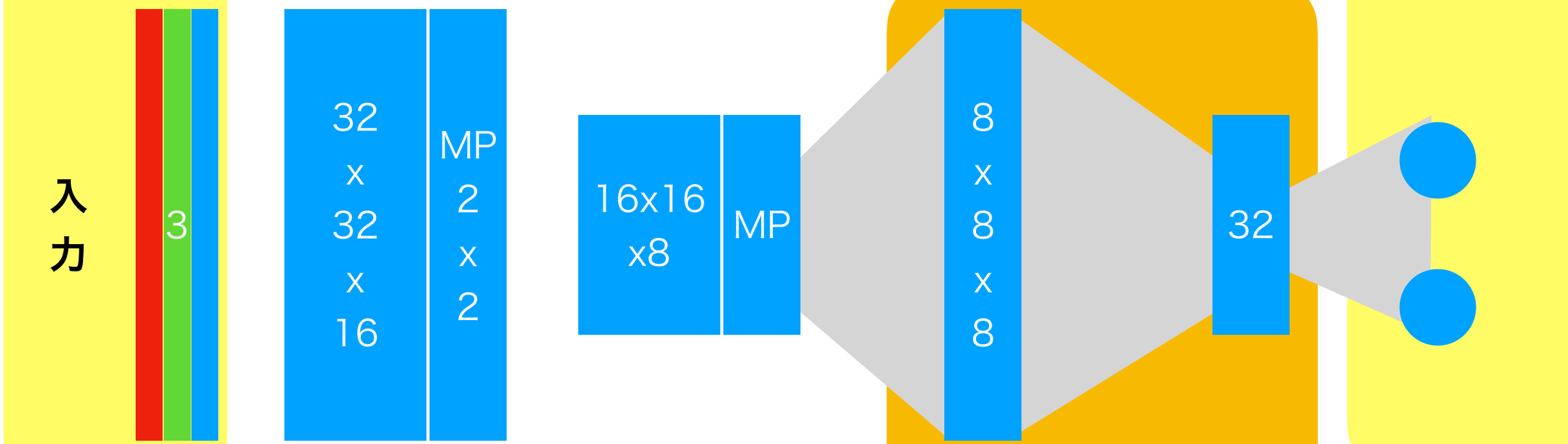
nn.Conv2d(16, 8, kernel_size=3, padding=1)
16chの入力を3x3のカーネルで8chにする
画素数はそのまま
カーネルのパラメタは学習で決定する

そのあとMaxPoolingで2x2画素の最大値を抽出する
2x2x8の中から8種類の最大値が抽出される
その結果8x8画素，8chになる

構築するモデル

画像が入力される
32x32x3chにする

2分類なので
出力は２つ



nn.Linear(8 * 8 * 8, 32)

8x8画素，8chになった画像を平坦に並べる

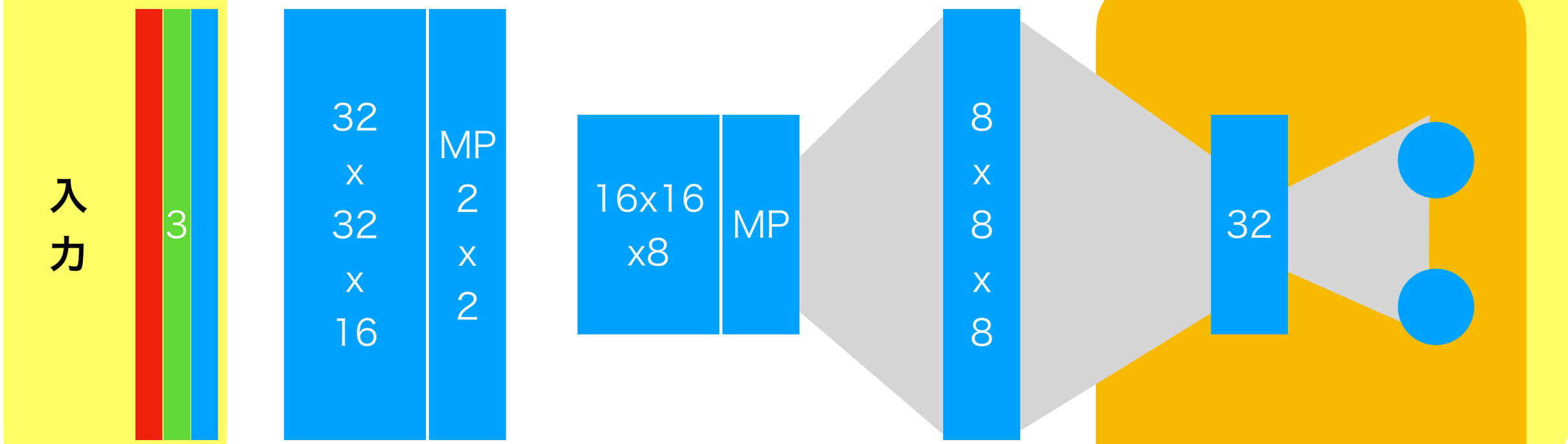
それを全結合型ニューラルネットワークの入力にする

その出力を32ノードに接続する

構築するモデル

画像が入力される
32x32x3chにする

2分類なので
出力は２つ



nn.Linear(32, 2)

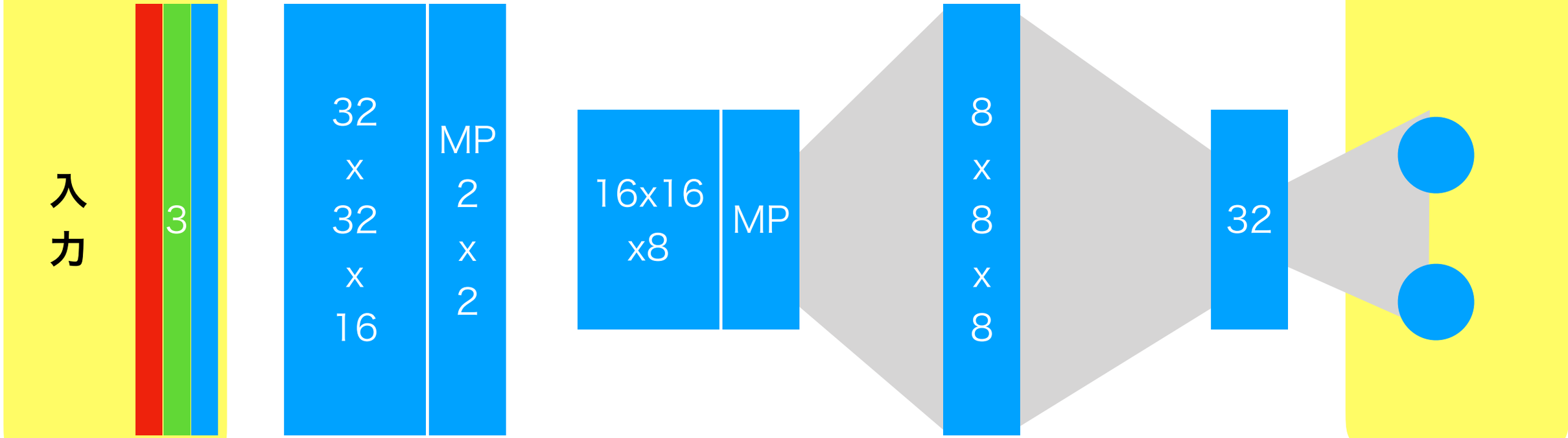
32特徴を
ニューラルネットワークの入力にする

その出力を2ノードに接続する

構築するモデル

画像が入力される
32x32x3chにする

2分類なので
出力は２つ



RGB画像が入力されて

１６のフィルタで
32x32x16特徴に分解されて

MaxPoolingで
16x16x16に削減されて

8のフィルタで
16x16x8特徴に分解されて

MaxPoolingで
8x8x8に削減されて

8x8x8が
平坦に並べられて

32ノードに
接続されて

２ノードで
出力

構築するモデル

```
class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        self.conv1 = nn.Conv2d(3, 16, kernel_size=3, padding=1)
        self.conv2 = nn.Conv2d(16, 8, kernel_size=3, padding=1)
        self.fc1 = nn.Linear(8 * 8 * 8, 32)
        self.fc2 = nn.Linear(32, 2)
```

部品を書く部分

```
    def forward(self, x):
        out = F.max_pool2d(torch.relu(self.conv1(x)), 2)
        out = F.max_pool2d(torch.relu(self.conv2(out)), 2)
        out = out.view(-1, 8 * 8 * 8)
        out = torch.tanh(self.fc1(out))
        out = self.fc2(out)
        return out
```

処理順を書く部分

学習済みモデルの利用

- 大規模な画像データベースを利用して学習済みのモデルを利用する
- 「学習済み」とは，モデルの重みが決定されていること
- 大規模なデータでの学習は時間がかかるので重みが公開されている場合が多い

利用できる学習済みモデル

モデル名	学習データ	入力画像	パラメータ数	ファイル容量（MB）	事前学習モデルの有無	PyTorchでの利用可否
VGG16	ImageNet	224x224, RGB	約138M	約528 MB	あり	✓
VGG19	ImageNet	224x224, RGB	約144M	約548 MB	あり	✓
ResNet50	ImageNet	224x224, RGB	約25.6M	約98 MB	あり	✓
ResNet101	ImageNet	224x224, RGB	約44.5M	約171 MB	あり	✓
InceptionV3	ImageNet	299x299, RGB	約23.9M	約92 MB	あり	✓
MobileNetV2	ImageNet	224x224, RGB	約3.4M	約14 MB	あり	✓
DenseNet121	ImageNet	224x224, RGB	約8M	約32 MB	あり	✓
DenseNet169	ImageNet	224x224, RGB	約14.3M	約56 MB	あり	✓
EfficientNetB0	ImageNet	224x224, RGB	約5.3M	約20 MB	あり	✓
EfficientNetB7	ImageNet	600x600, RGB	約66M	約255 MB	あり	✓
EfficientNetB8	ImageNet	672x672, RGB	約87M	約335 MB	あり	✓（追加ライブラリ）
AlexNet	ImageNet	227x227, RGB	約61M	約233 MB	あり	✓
SqueezeNet	ImageNet	224x224, RGB	約1.24M	約4.8 MB	あり	✓
NASNet-A-Mobile	ImageNet	224x224, RGB	約5.3M	約19 MB	あり	✓（追加ライブラリ）
NASNet-A-Large	ImageNet	331x331, RGB	約88.9M	約343 MB	あり	✓（追加ライブラリ）
Vision Transformer (ViT)	ImageNet	調整可能（384x384など）	約86M	約330 MB	あり	✓（追加ライブラリ）
SENet	ImageNet	調整可能	約115M	約450 MB	あり	✓
Swin Transformer	ImageNet	調整可能（384x384など）	約87M	約340 MB	あり	✓
ConvNeXt	ImageNet	調整可能（384x384など）	約90M	約350 MB	あり	✓

学習済みモデルの利用
（常にうまくいくとはかぎりません！）

- **転移学習**

  別の分野で学習されたデータを積極的に利用する方法
  例：自然画像で学習されたモデルを医用画像に適用する
  モデルの構造の出力層を変更する
  出力層に近い重みを更新して実現

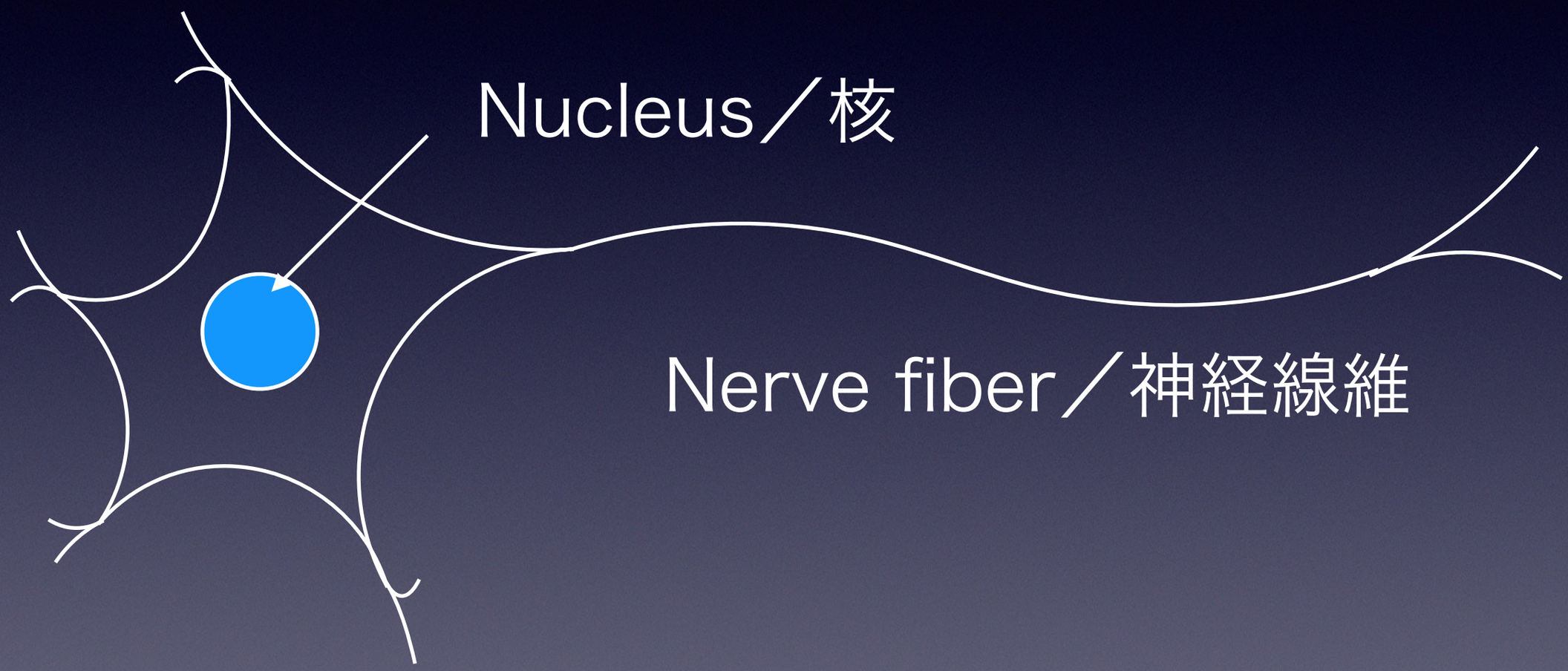
- **ファインチューニング**

  別の分野で学習されたデータを初期値として利用する方法
  例：A社製画像装置で学習したモデルをB社製に適用する
  モデルの構造の出力層を変更する
  出力層に近い重みのみならず全体を更新して実現

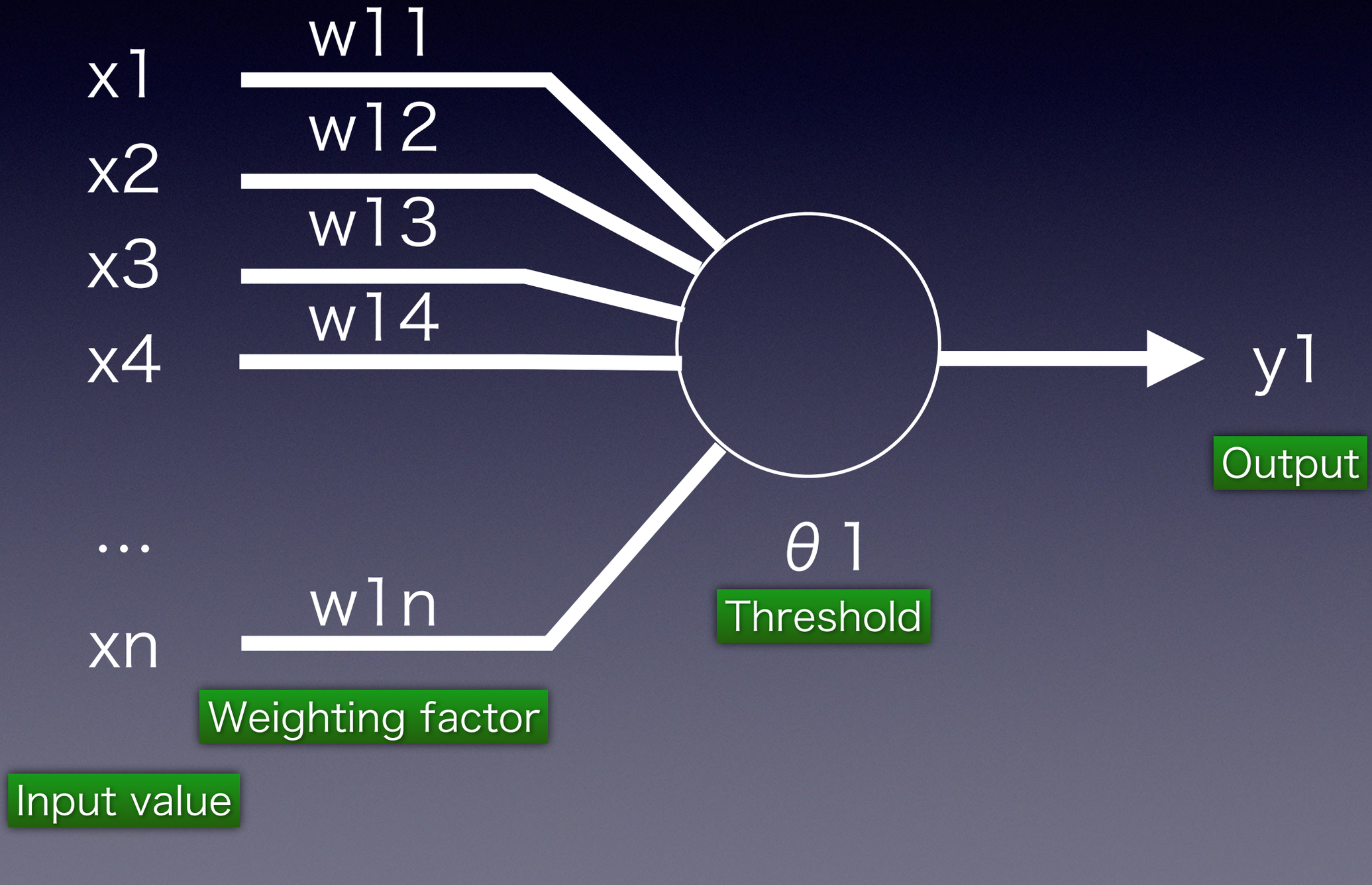
- **知識蒸留**

  すでにある学習済みモデルと同じ出力が得られるように
  元よりも小さなモデルを作ること（今回は取り扱いません）

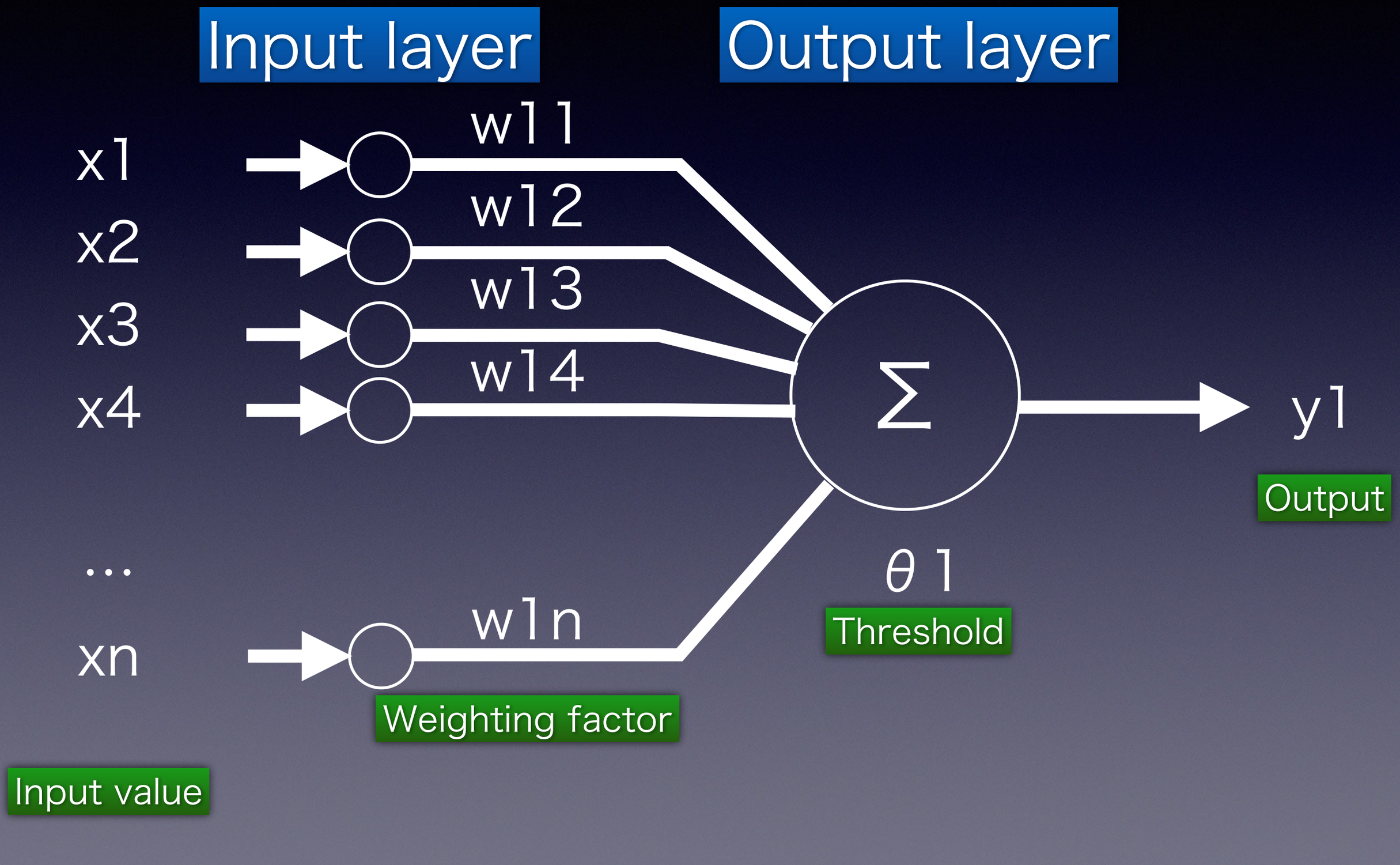
Perceptron



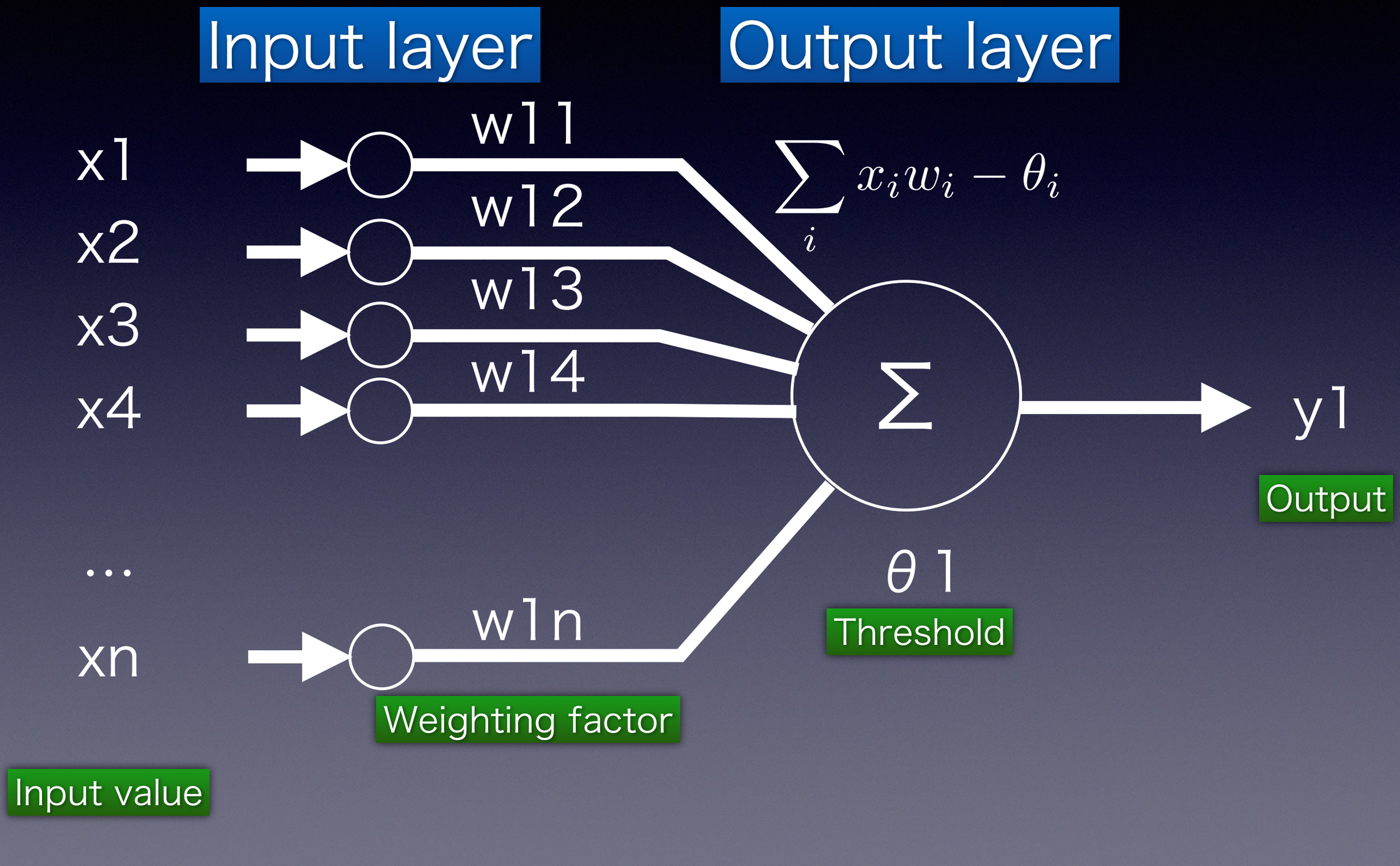
Neuron Model



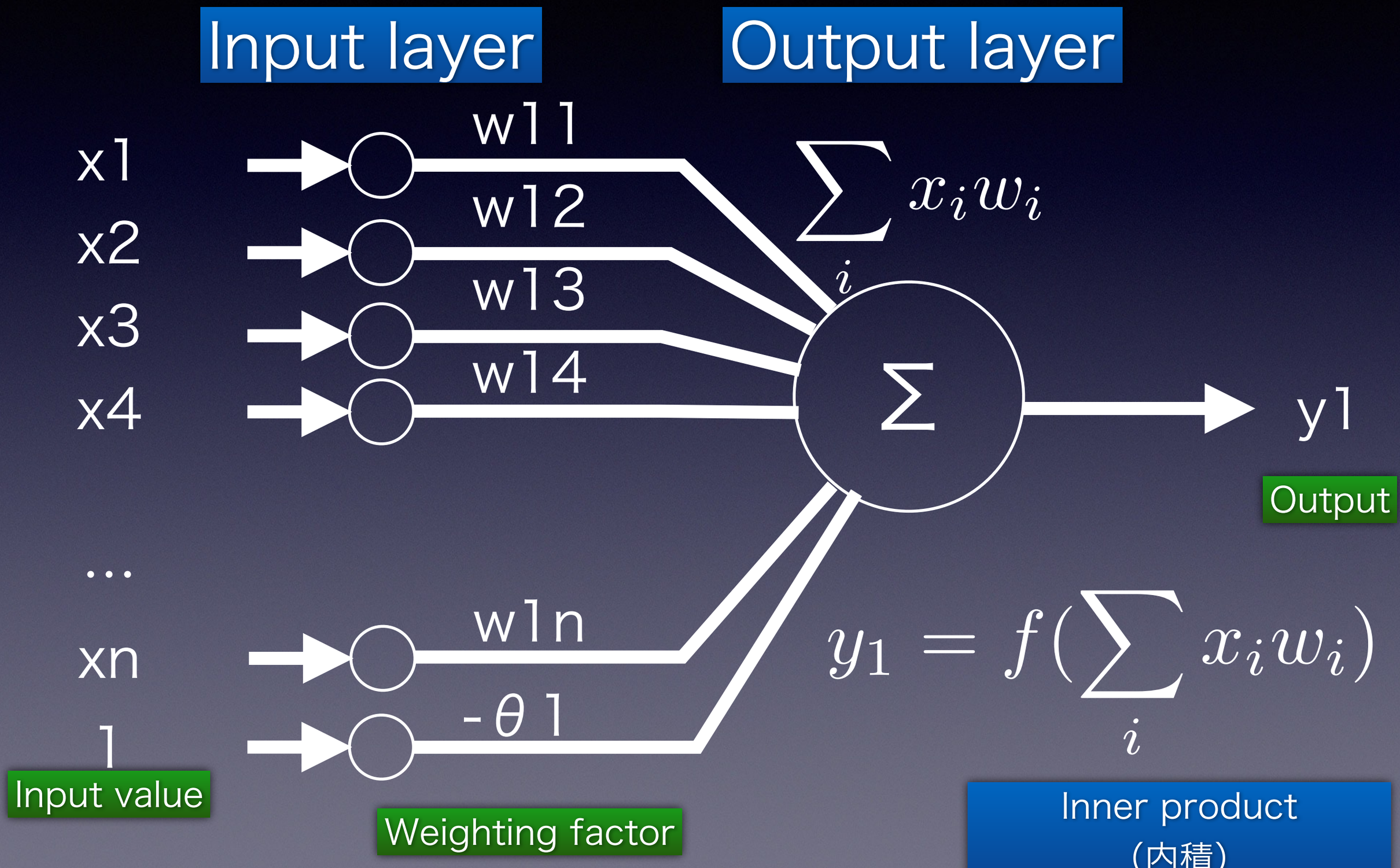
Simple Perceptron

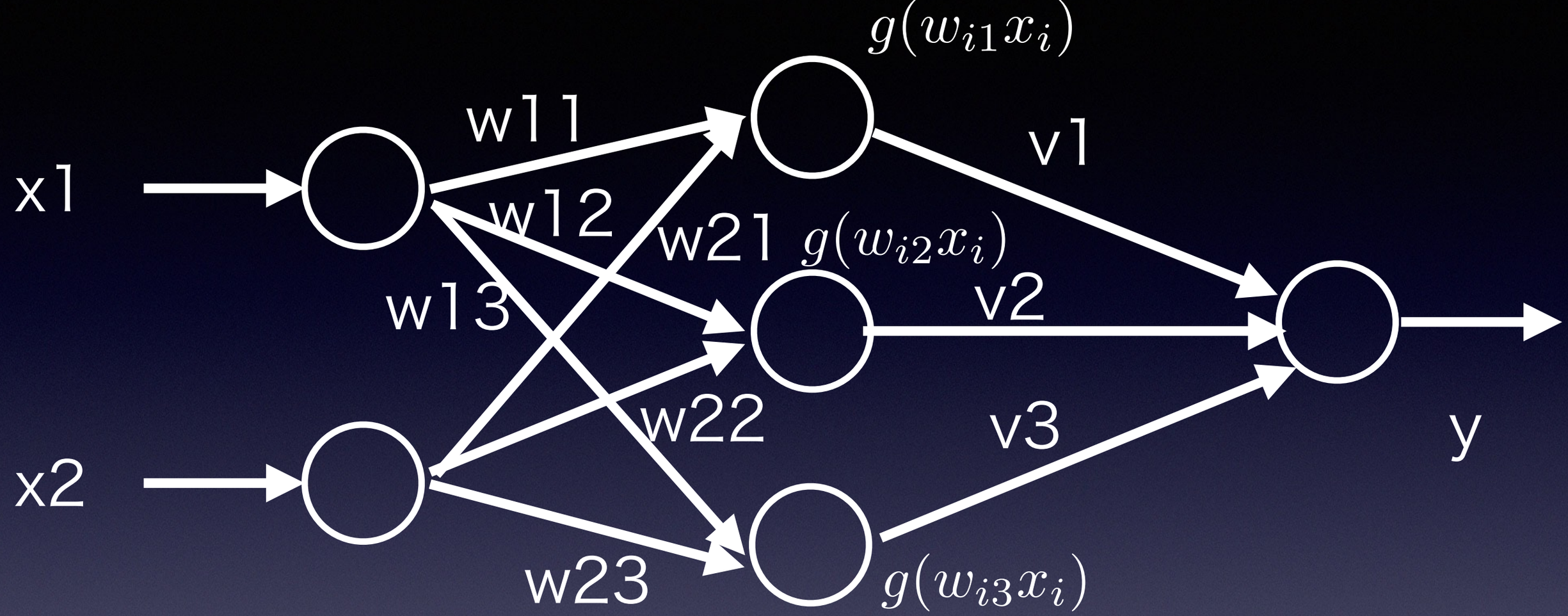


Simple Perceptron



Simple Perceptron





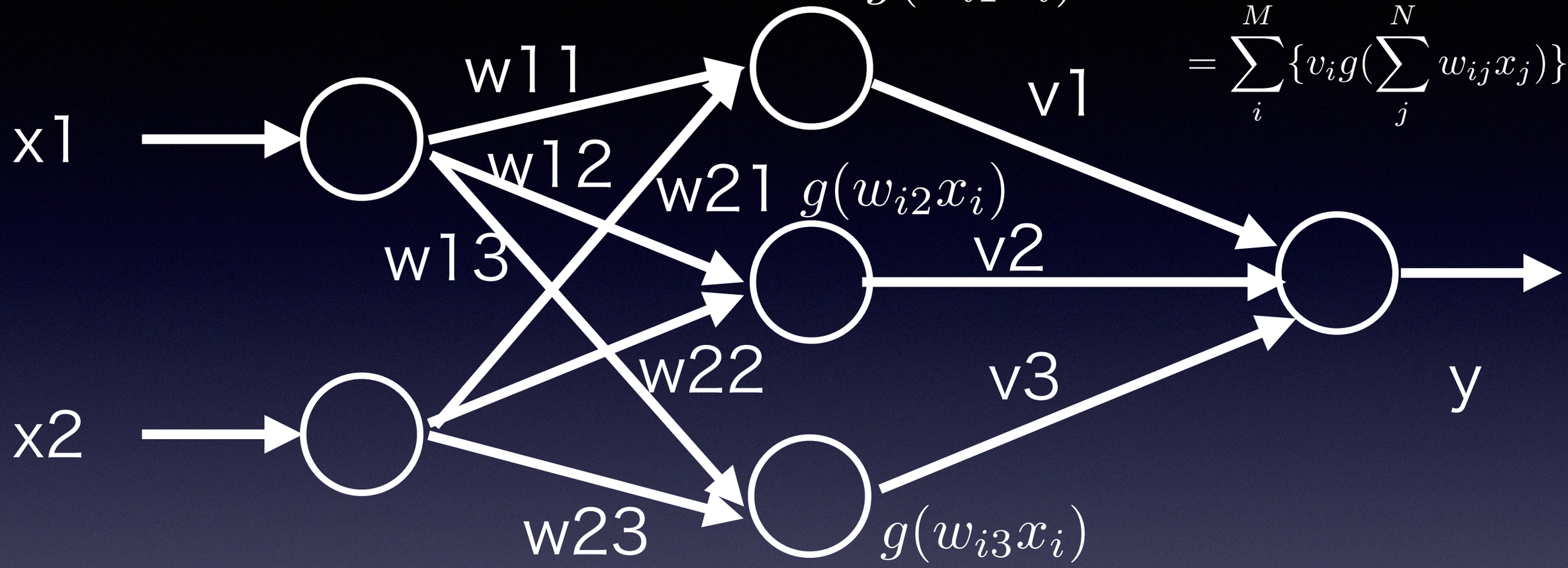
$$g(wx) = \frac{1}{1+\exp(-\beta wx)}$$

Sigmoid Function

$$\neq w_1x_1 + w_1x_1 + \cdots + w_nx_n$$

$$y = \sum_i^3 \{v_i g(\sum_j^2 w_{ij} x_j)\}$$

$$= \sum_i^M \{v_i g(\sum_j^N w_{ij} x_j)\}$$



$$g(wx) = \frac{1}{1 + \exp(-\beta wx)}$$

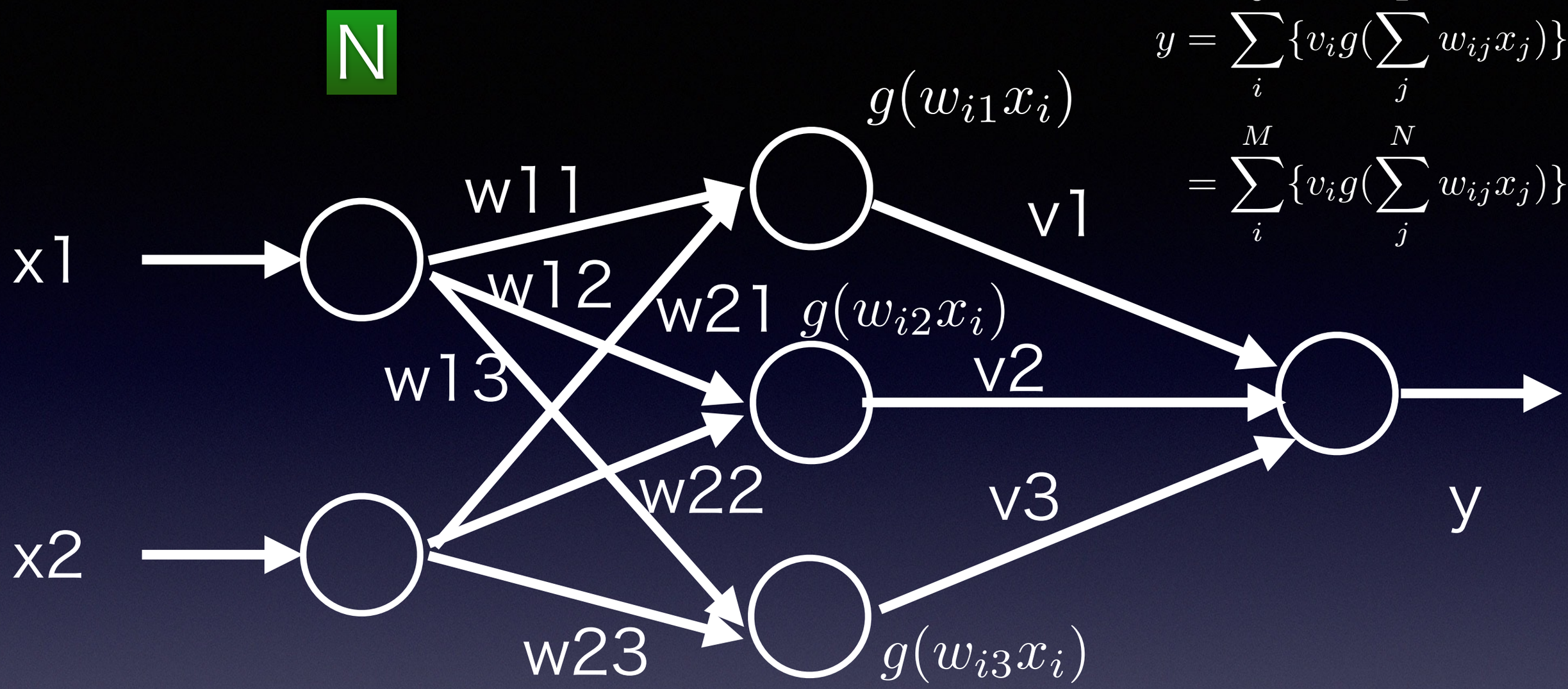
Sigmoid Function

$$\neq w_1 x_1 + w_1 x_1 + \cdots + w_n x_n$$

N

$$y = \sum_i^3 \{v_i g(\sum_j^2 w_{ij} x_j)\}$$

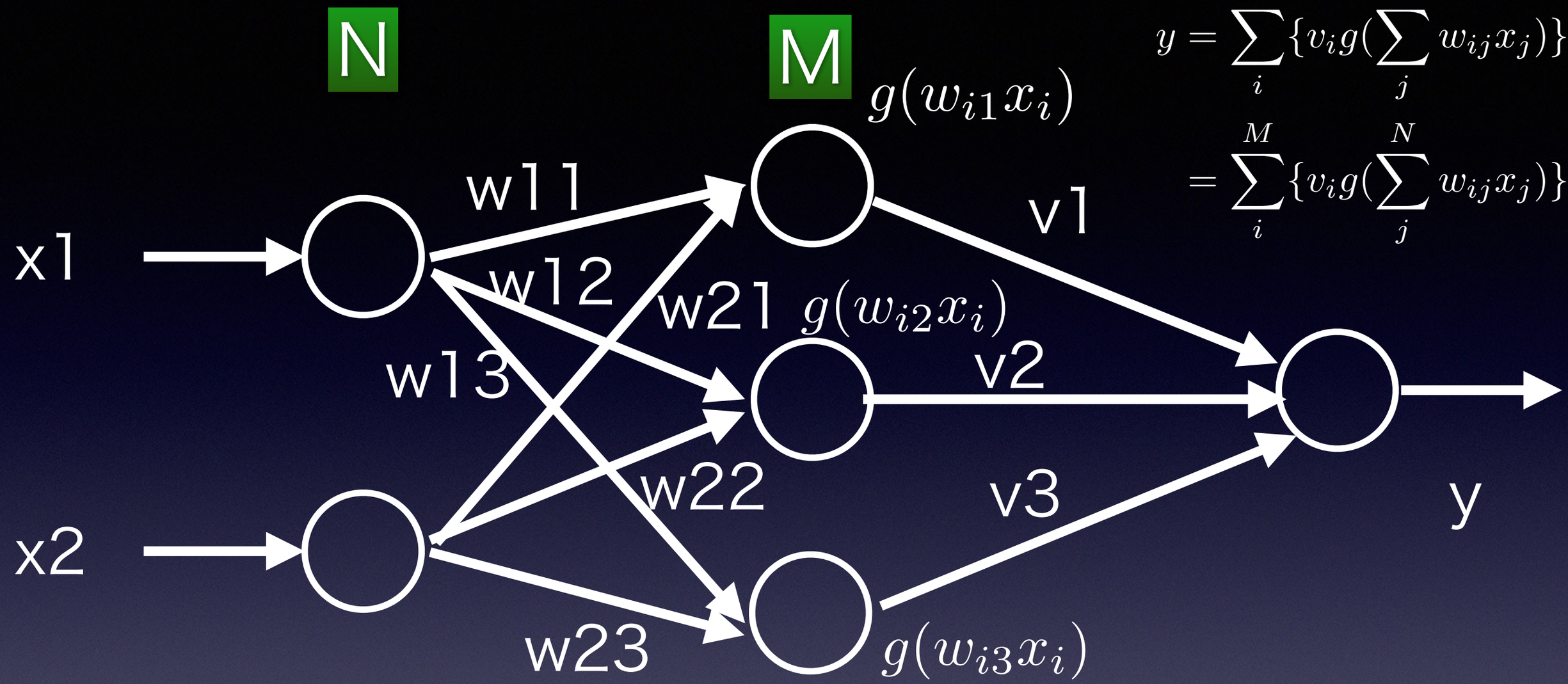
$$= \sum_i^M \{v_i g(\sum_j^N w_{ij} x_j)\}$$



$$g(wx) = \frac{1}{1 + \exp(-\beta wx)}$$

Sigmoid Function

$$\neq w_1 x_1 + w_1 x_1 + \cdots + w_n x_n$$



$$y = \sum_i^3 \{v_i g(\sum_j^2 w_{ij}x_j)\}$$

$$= \sum_i^M \{v_i g(\sum_j^N w_{ij}x_j)\}$$

$$g(wx) = \frac{1}{1+\exp(-\beta wx)}$$

Sigmoid Function

$$\neq w_1x_1 + w_1x_1 + \cdots + w_nx_n$$

Fine Tuningの
2クラス問題を
やってみましょう

```python
#    学習を行うためには requires_grad = Trueである必要があります
#    このままですべて学習ができます

for name, param in model_ft.named_parameters():
#    print(name)
#    print(param)
    print(name, param.requires_grad)
```

```
conv1.weight True
bn1.weight True
bn1.bias True
layer1.0.conv1.weight True
layer1.0.bn1.weight True
layer1.0.bn1.bias True
layer1.0.conv2.weight True
layer1.0.bn2.weight True
layer1.0.bn2.bias True
```

Transfer Learningの
2クラス問題を
やってみましょう

```
#  この２つだけが重要で  あとはＦａｌｓｅにしたいよね
print(model_ft.fc.weight.requires_grad)
print(model_ft.fc.bias.requires_grad)
```

```
True
True
```

```
#  面倒なので全部Falseにします
for name, param in model_ft.named_parameters():
    param.requires_grad = False
```

```
#  ぜんぶFalseになったようです
for name, param in model_ft.named_parameters():
    print(name, param.requires_grad)
```

```
conv1.weight False
bn1.weight False
bn1.bias False
layer1.0.conv1.weight False
layer1.0.bn1.weight False
layer1.0.bn1.bias False
layer1.0.conv2.weight False
```

```
#  この２つだけが重要ですがFalseのようです
print(model_ft.fc.weight.requires_grad)
print(model_ft.fc.bias.requires_grad)
```

```
False
False
```

```
#  なのでTrueにしてあげます
model_ft.fc.weight.requires_grad=True
model_ft.fc.bias.requires_grad=True

```

```
#  無事にTrueになりました

print(model_ft.fc.weight.requires_grad)
print(model_ft.fc.bias.requires_grad)
```

```
True
True
```

領域分割

ディープラーニングを評価

$$x \longrightarrow y$$

図形で評価

検出（箱型）

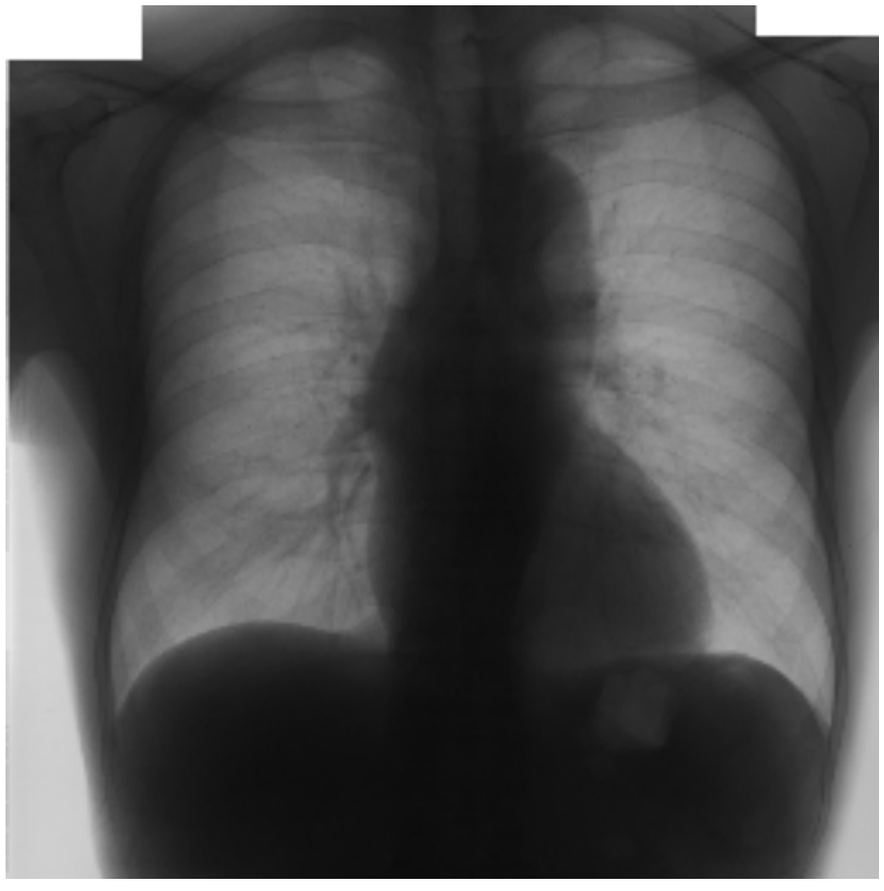
領域抽出（精密抽出）

この２つは
どのくらい重なる？

一致率を計測する

演習　画像の領域分割

肺野領域抽出



U-Net: Convolutional Networks for Biomedical Image Segmentation

Olaf Ronneberger, Philipp Fischer, and Thomas Brox

Computer Science Department and BIOSS Centre for Biological Signalling Studies,
University of Freiburg, Germany
ronneber@informatik.uni-freiburg.de,
WWW home page: http://lmb.informatik.uni-freiburg.de/

Abstract. There is large consent that successful training of deep networks requires many thousand annotated training samples. In this paper, we present a network and training strategy that relies on the strong use of data augmentation to use the available annotated samples more efficiently. The architecture consists of a contracting path to capture

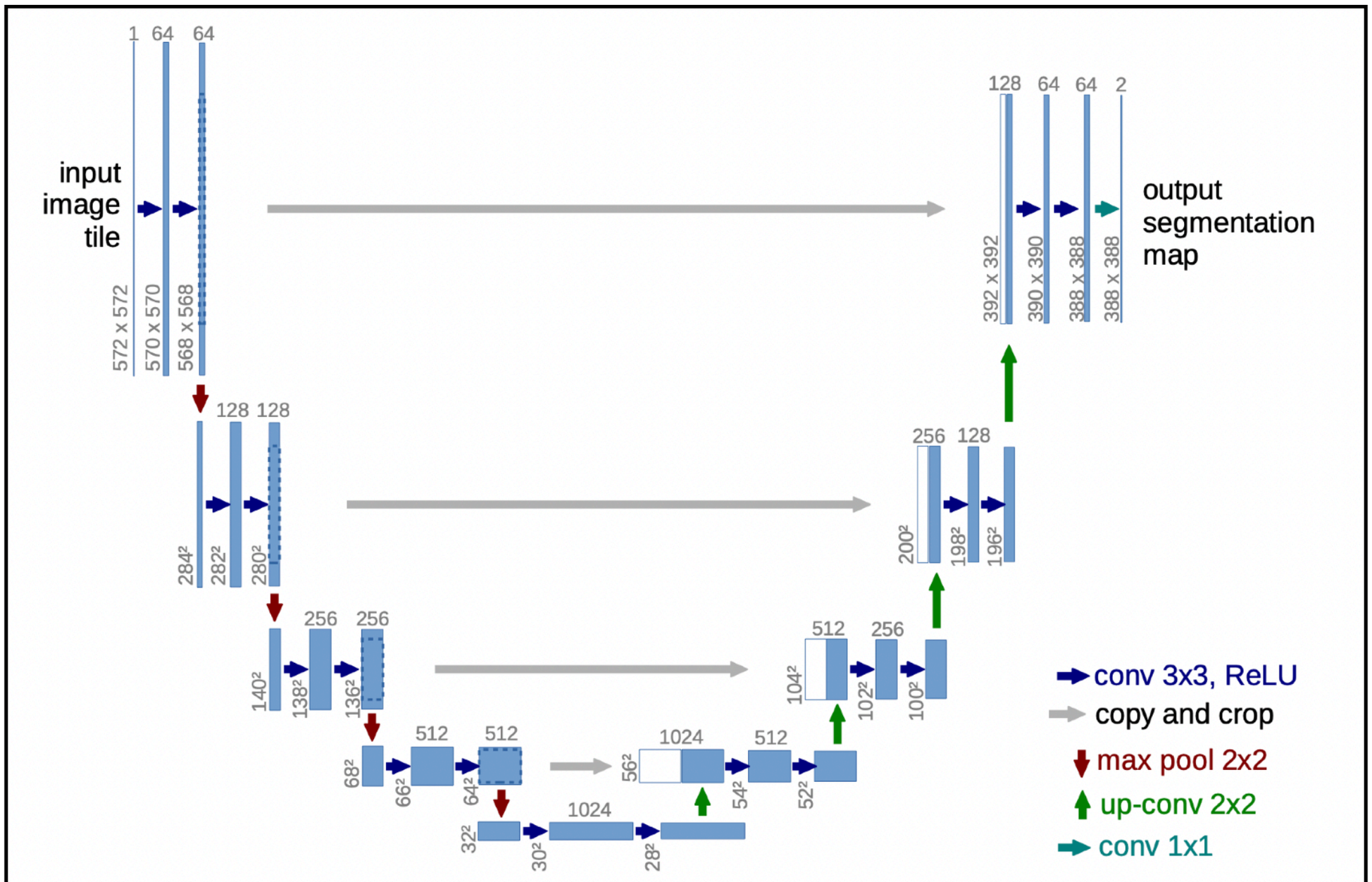


Fig. 1. U-net architecture (example for 32x32 pixels in the lowest resolution). Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations.

https://arxiv.org/pdf/1505.04597.pdf

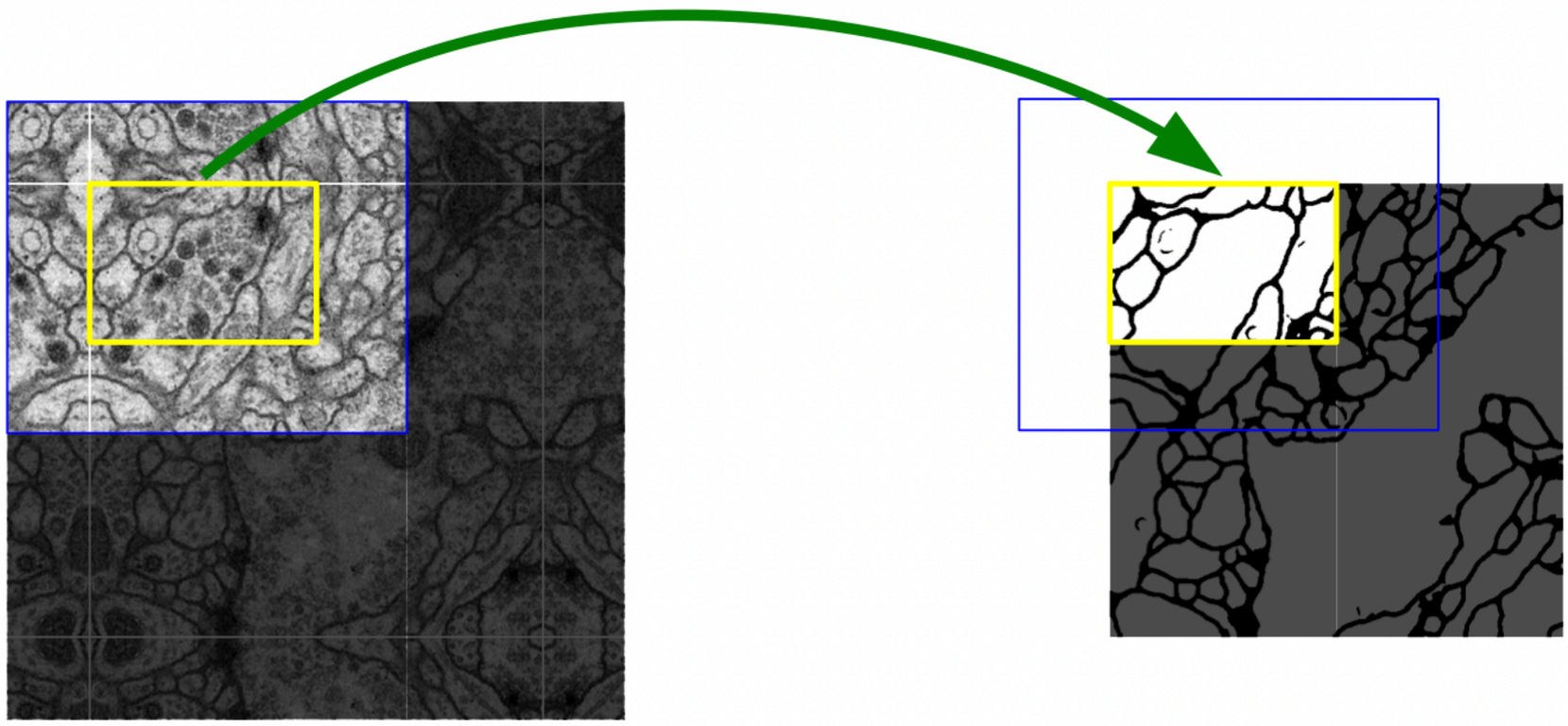


Fig. 2. Overlap-tile strategy for seamless segmentation of arbitrary large images (here segmentation of neuronal structures in EM stacks). Prediction of the segmentation in the yellow area, requires image data within the blue area as input. Missing input data is extrapolated by mirroring

https://arxiv.org/pdf/1505.04597.pdf

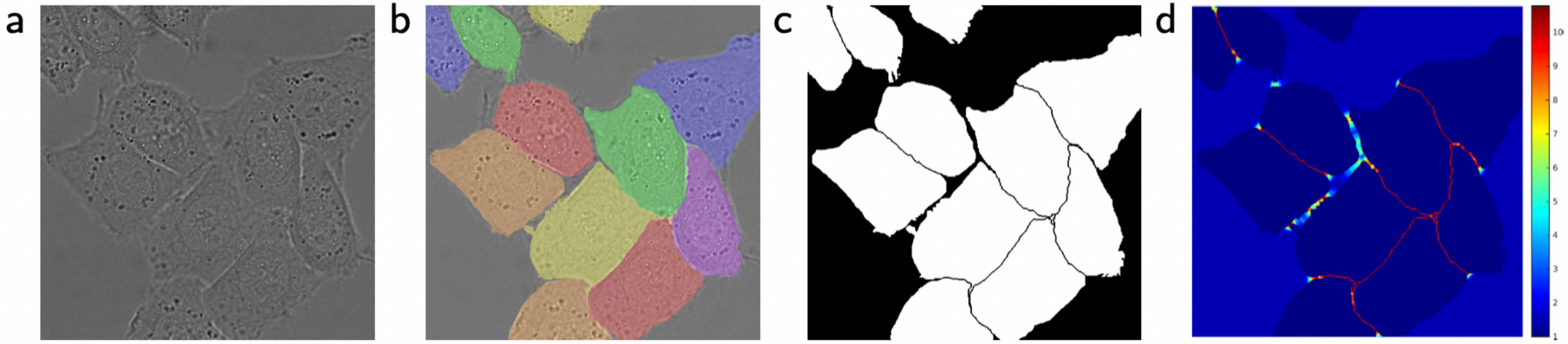


Fig. 3. HeLa cells on glass recorded with DIC (differential interference contrast) microscopy. (**a**) raw image. (**b**) overlay with ground truth segmentation. Different colors indicate different instances of the HeLa cells. (**c**) generated segmentation mask (white: foreground, black: background). (**d**) map with a pixel-wise loss weight to force the network to learn the border pixels.

https://arxiv.org/pdf/1505.04597.pdf

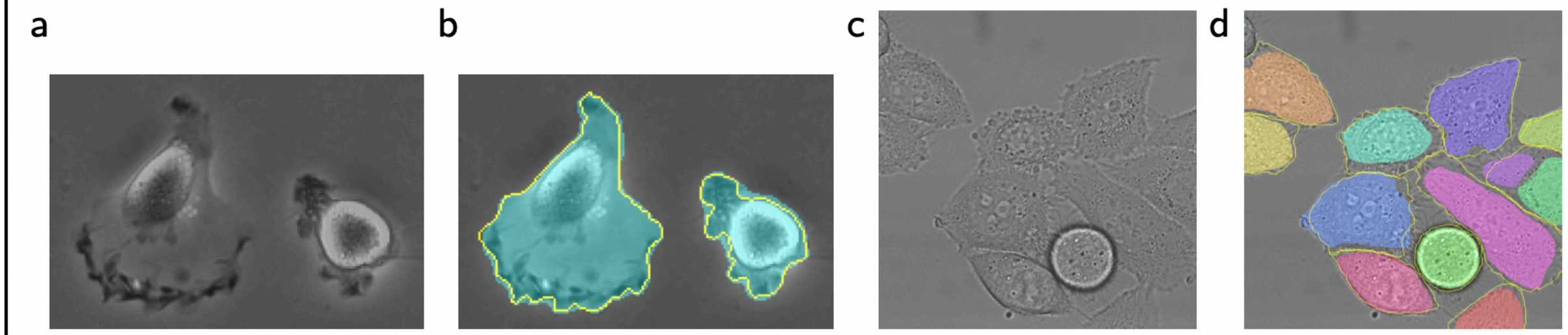


Fig. 4. Result on the ISBI cell tracking challenge. (**a**) part of an input image of the "PhC-U373" data set. (**b**) Segmentation result (cyan mask) with manual ground truth (yellow border) (**c**) input image of the "DIC-HeLa" data set. (**d**) Segmentation result (random colored masks) with manual ground truth (yellow border).

https://arxiv.org/pdf/1505.04597.pdf

実際に動かしてみましょう

Image Input

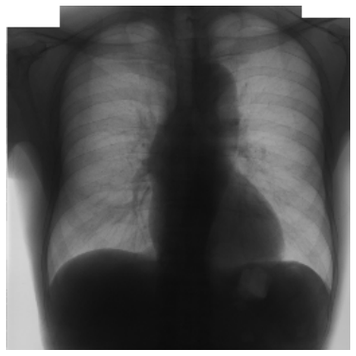


Image Output

conv2d_23 (Conv2D)	(None, 32, 32, 128)	147584	batch_normalization_21[0][0]
batch_normalization_22 (BatchNo	(None, 32, 32, 128)	512	conv2d_23[0][0]
up_sampling2d_5 (UpSampling2D)	(None, 64, 64, 128)	0	batch_normalization_22[0][0]
concatenate_5 (Concatenate)	(None, 64, 64, 192)	0	up_sampling2d_5[0][0] batch_normalization_8[0][0]
conv2d_24 (Conv2D)	(None, 64, 64, 64)	110656	concatenate_5[0][0]
batch_normalization_23 (BatchNo	(None, 64, 64, 64)	256	conv2d_24[0][0]
conv2d_25 (Conv2D)	(None, 64, 64, 64)	36928	batch_normalization_23[0][0]
batch_normalization_24 (BatchNo	(None, 64, 64, 64)	256	conv2d_25[0][0]
conv2d_26 (Conv2D)	(None, 64, 64, 1)	65	batch_normalization_24[0][0]

Total params: 31,401,345
Trainable params: 31,389,569
Non-trainable params: 11,776

Conv 2D

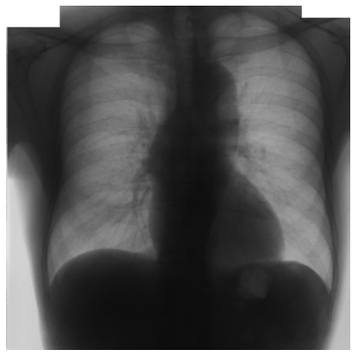
Batch Normalization

Max pooling

Concatinate

Up-sampling

ge Input



conv2d_23 (Conv2D)	(None, 32, 32, 128)	147584	batch_normalization_21[0][0]
batch_normalization_22 (BatchNo	(None, 32, 32, 128)	512	conv2d_23[0][0]
up_sampling2d_5 (UpSampling2D)	(None, 64, 64, 128)	0	batch_normalization_22[0][0]
concatenate_5 (Concatenate)	(None, 64, 64, 192)	0	up_sampling2d_5[0][0] batch_normalization_8[0][0]
conv2d_24 (Conv2D)	(None, 64, 64, 64)	110656	concatenate_5[0][0]
batch_normalization_23 (BatchNo	(None, 64, 64, 64)	256	conv2d_24[0][0]
conv2d_25 (Conv2D)	(None, 64, 64, 64)	36928	batch_normalization_23[0][0]
batch_normalization_24 (BatchNo	(None, 64, 64, 64)	256	conv2d_25[0][0]
conv2d_26 (Conv2D)	(None, 64, 64, 1)	65	batch_normalization_24[0][0]

Total params: 31,401,345
Trainable params: 31,389,569
Non-trainable params: 11,776

Conv 2D

Batch Normalization

Max pooling

Concatinate

Up-sampling

正解の作成方法

２値画像を作る／多値画像にする

ラベル番号に意味があるので可逆圧縮を使うこと

VoTT

https://github.com/Microsoft/VoTT/releases

PLUTO

http://www.suenaga.cse.nagoya-u.ac.jp/wiki/index.php?PLUTO%A5%D7%A5%E9%A5%B0%A5%A4%A5%F3

特集論文／CADの新展開

PLUTO：医用画像診断支援共通プラットフォーム
PLUTO: A common Platform for Computer-aided Diagnosis

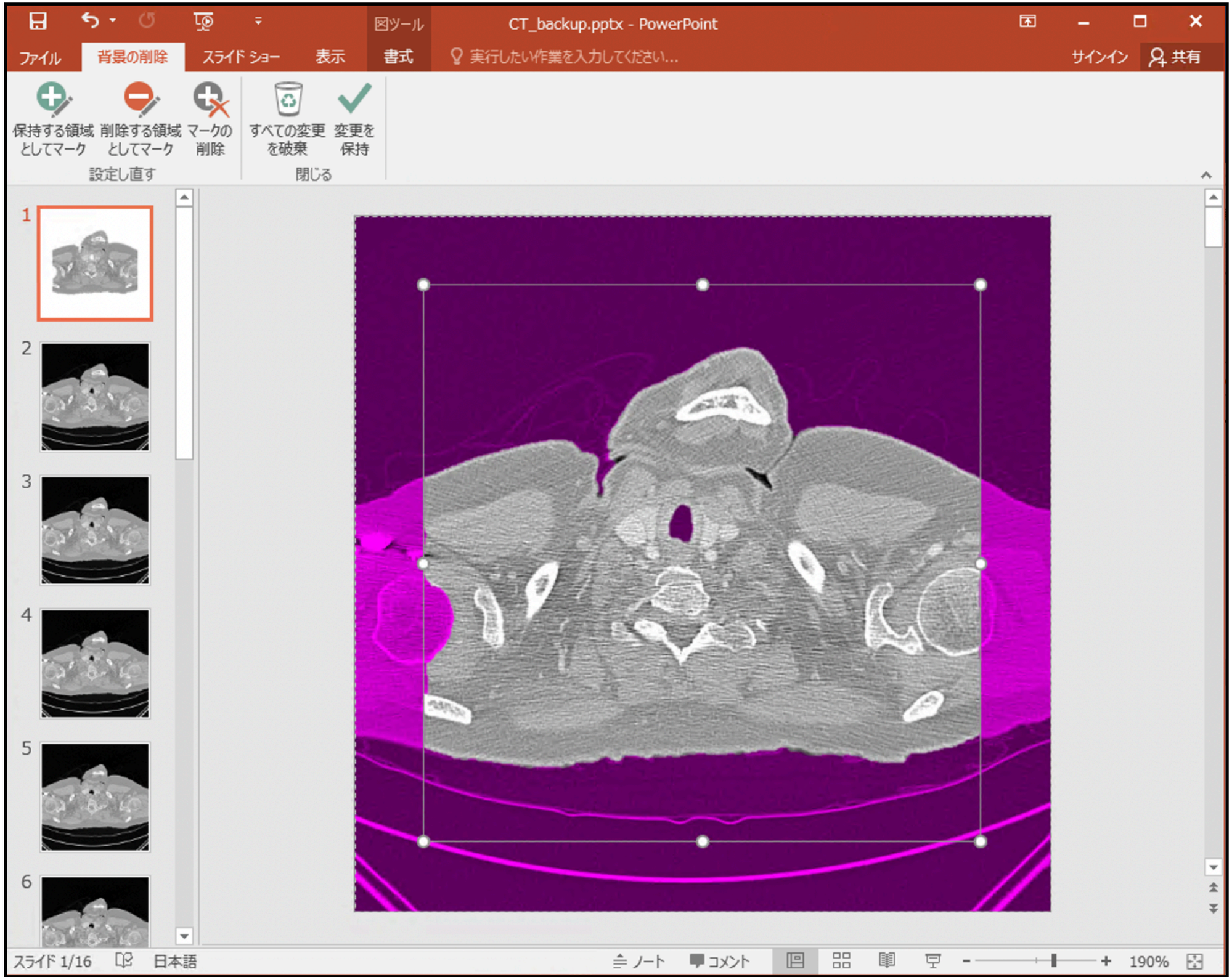
二村 幸孝[*1] 出口 大輔[*2] 北坂 孝幸[*1]
Yukitaka NIMURA Daisuke DEGUCHI Takayuki KITASAKA
森 健策[*1] 末永 康仁[*1]
Kensaku MORI Yasuhito SUENAGA

要 旨

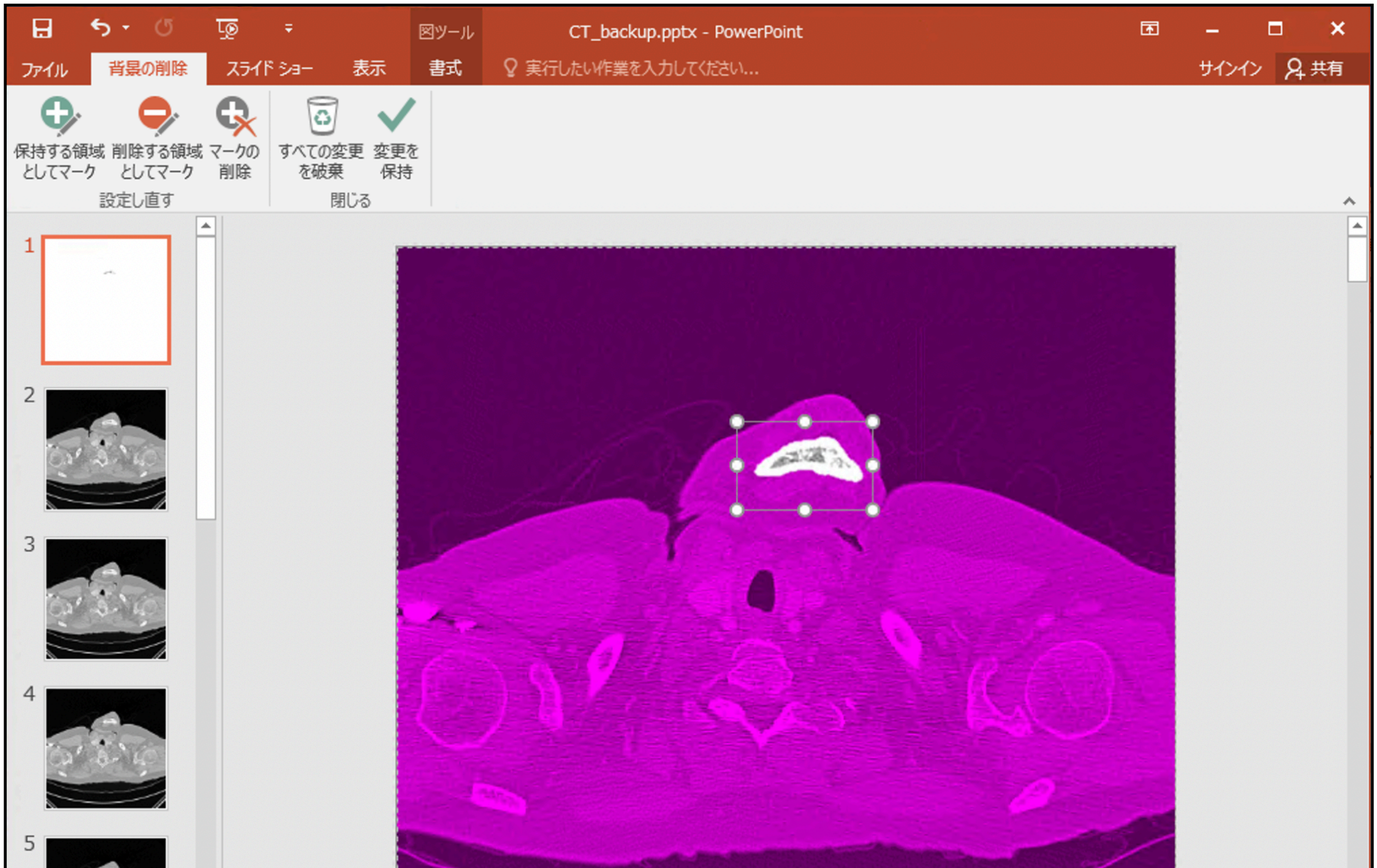
本稿では文部科学省科学研究費補助金特定領域研究「多次元医用画像の知的診断支援」において，多臓器・多疾病計算機支援医用画像診断システムを実現するための共通開発基盤および診断支援アルゴリズムの共有化のために開発された，医用画像診断支援共通プラットフォーム「PLUTO」について述べる．PLUTOは，診断支援システムを開発するための共通プラットフォームであり，多臓器・多疾病計算機支援医用画像診断システムが必要とするさまざまな診断支援機能を容易に実現できるように設計されている．ここでは，このPLUTOのソフトウェア構成とこれまでに実現されている機能について紹介する．

キーワード：計算機支援医用画像診断，共通プラットフォーム，多臓器，多疾病

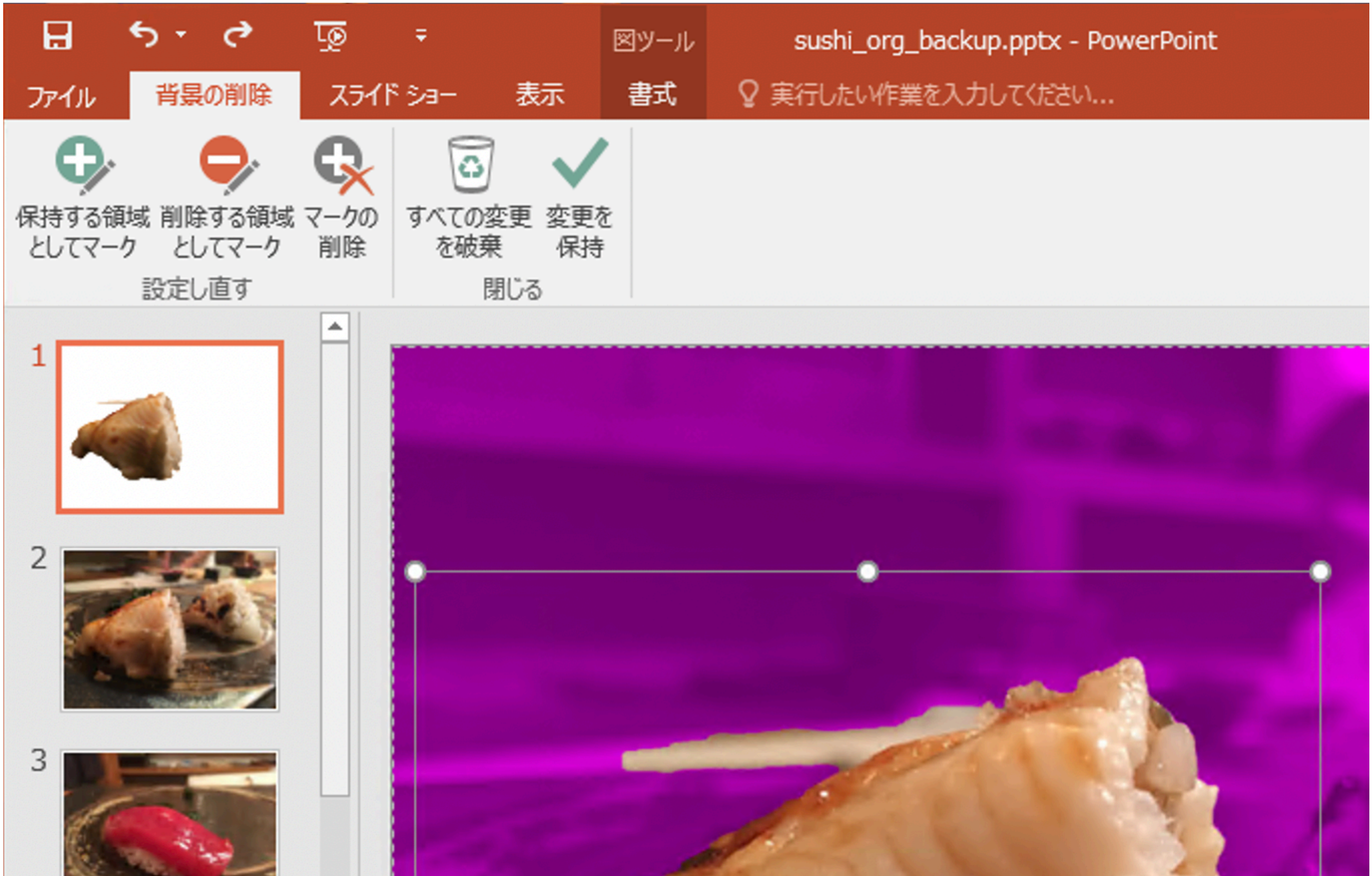
PowerPointも使える！



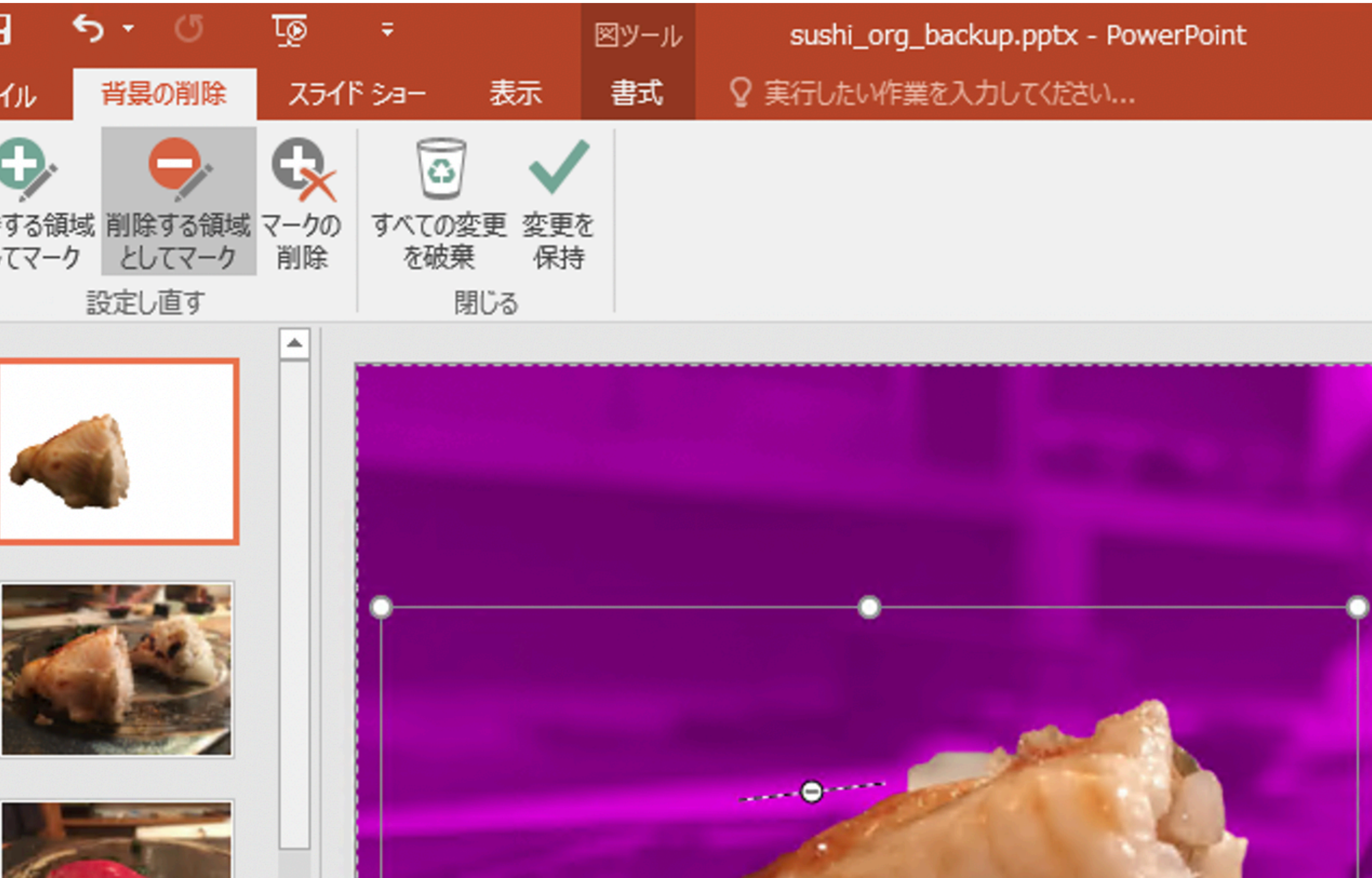
PowerPointも使える！



PowerPointも使える！



PowerPointも使える！



背景の削除　スライド ショー　表示　書式　実行したい作業を入力してください...

域
ク

削除する領域としてマーク

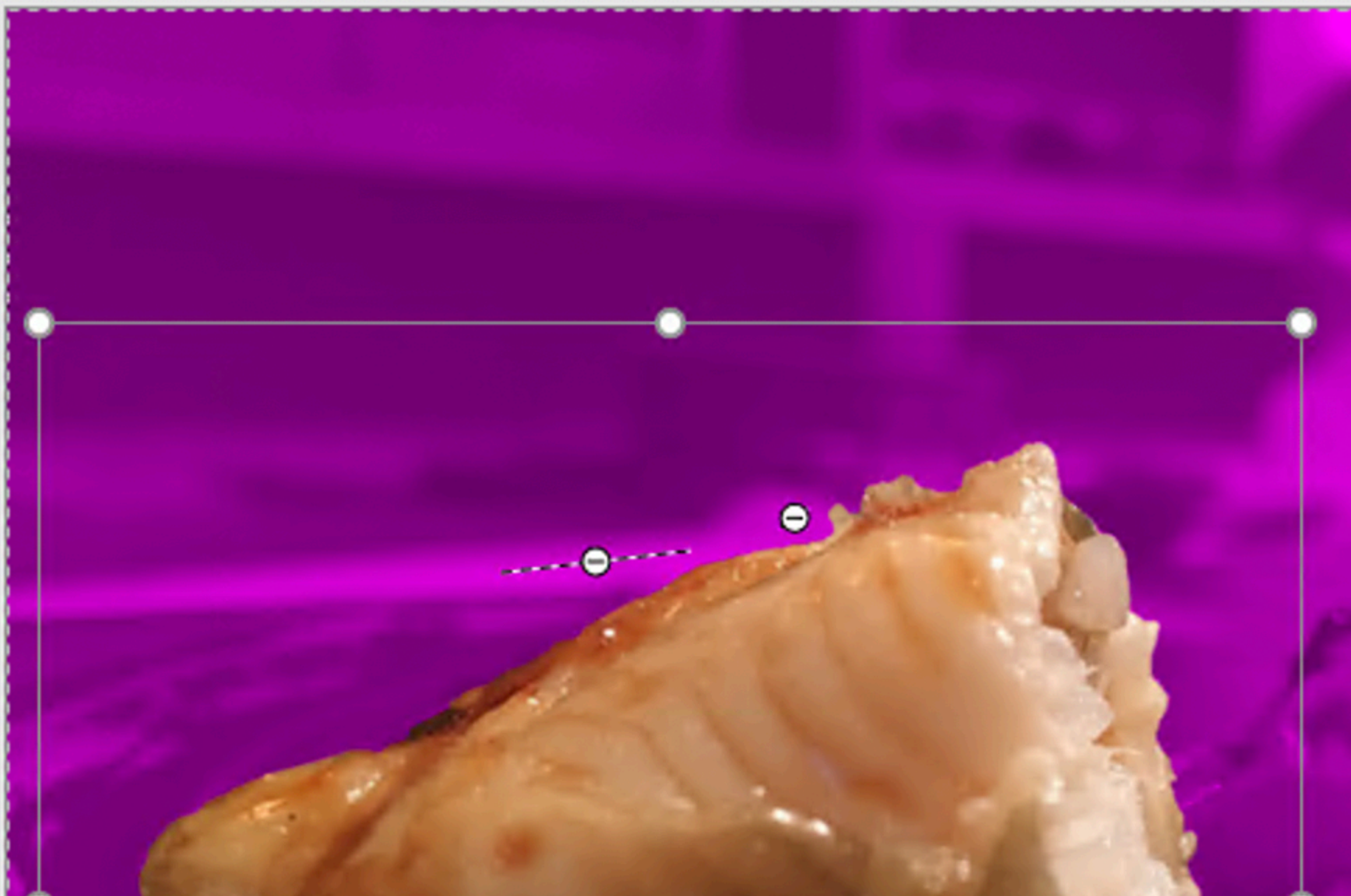
マークの削除

設定し直す

すべての変更を破棄

変更を保持

閉じる



“GrabCut” — Interactive Foreground Extraction using Iterated Graph Cuts

Carsten Rother* Vladimir Kolmogorov† Andrew Blake‡
Microsoft Research Cambridge, UK

Figure 1: **Three examples of GrabCut .** The user drags a rectangle loosely around an object. The object is then extracted automatically.

Abstract

The problem of efficient, interactive foreground/background segmentation in still images is of great practical importance in image editing. Classical image segmentation tools use either texture (colour) information, e.g. Magic Wand, or edge (contrast) information, e.g. Intelligent Scissors. Recently, an approach based on optimization by graph-cut has been developed which successfully combines both types of information. In this paper we extend the graph-cut approach in three respects. First, we have developed a more powerful, iterative version of the optimisation. Secondly, the power of the iterative algorithm is used to simplify substantially the

free of colour bleeding from the source background. In general, degrees of interactive effort range from editing individual pixels, at the labour-intensive extreme, to merely touching foreground and/or background in a few locations.

1.1 Previous approaches to interactive matting

In the following we describe briefly and compare several state of the art interactive tools for segmentation: Magic Wand, Intelligent Scissors, Graph Cut and Level Sets and for matting: Bayes Matting and Knockout. Fig. 2 shows their results on a matting task, together with degree of user interaction required to achieve those results.

https://dl.acm.org/doi/10.1145/1186562.1015720

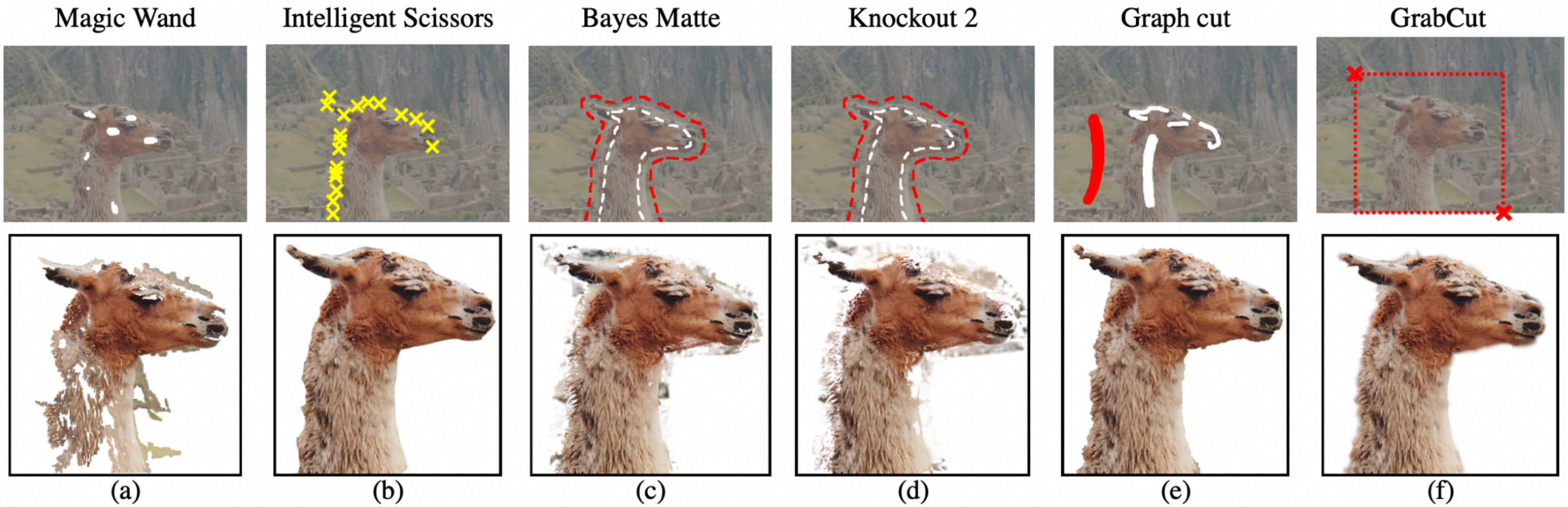


Figure 2: **Comparison of some matting and segmentation tools.** The top row shows the user interaction required to complete the segmentation or matting process: white brush/lasso (foreground), red brush/lasso (background), yellow crosses (boundary). The bottom row illustrates the resulting segmentation. GrabCut appears to outperform the other approaches both in terms of the simplicity of user input and the quality of results. Original images on the top row are displayed with reduced intensity to facilitate overlay; see fig. 1. for original. Note that our implementation of Graph Cut [Boykov and Jolly 2001] uses colour mixture models instead of grey value histograms.

https://dl.acm.org/doi/10.1145/1186562.1015720

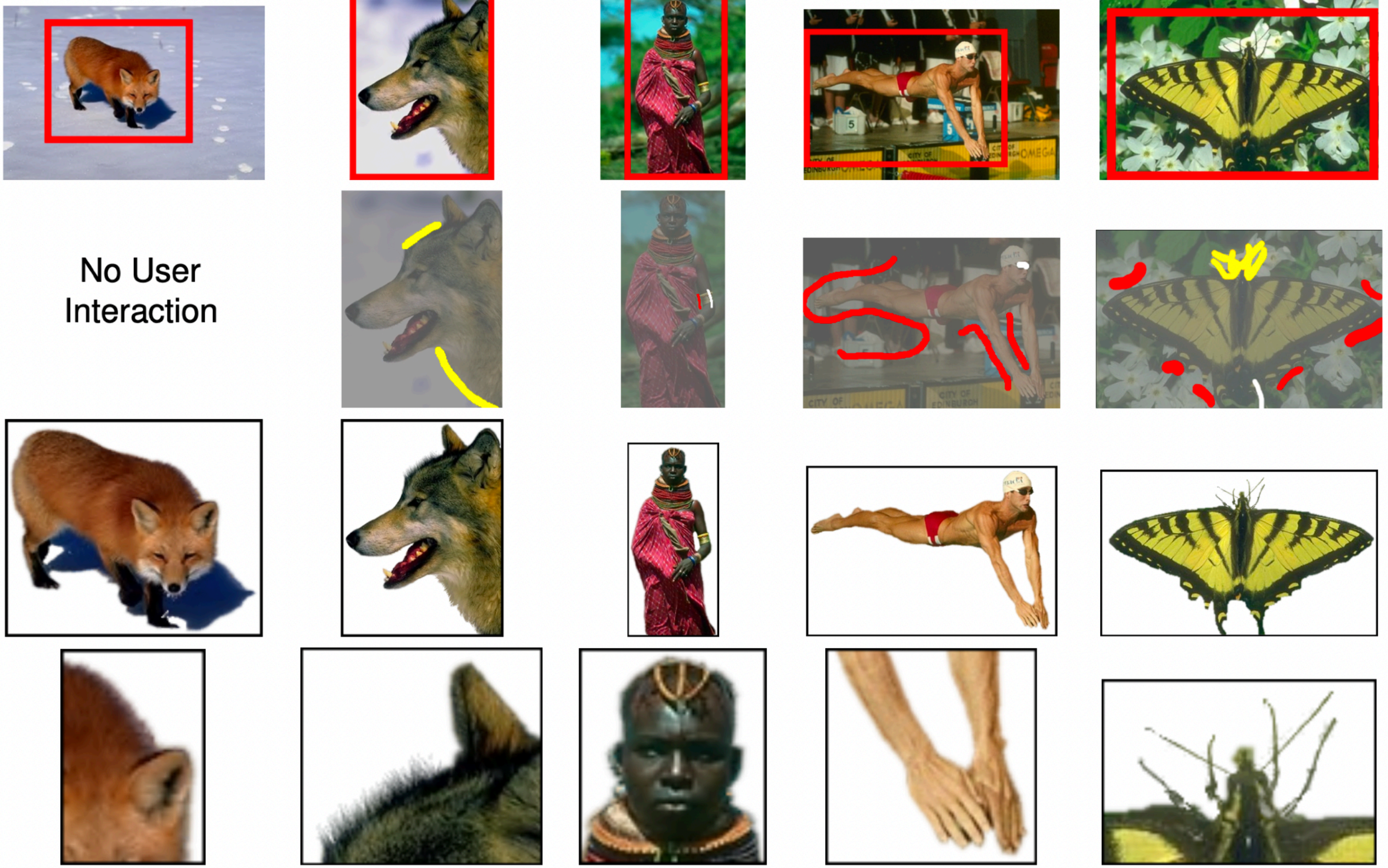


Figure 8: **Results using GrabCut .** The first row shows the original images with superimposed user input (red rectangle). The second row displays all user interactions: red (background brush), white (foreground brush) and yellow (matting brush). The degree of user interaction increases from left to right. The results obtained by GrabCut are visualized in the third row. The last row shows zoomed portions of the respective result which documents that the recovered alpha mattes are smooth and free of background bleeding.

https://dl.acm.org/doi/10.1145/1186562.1015720

評価方法

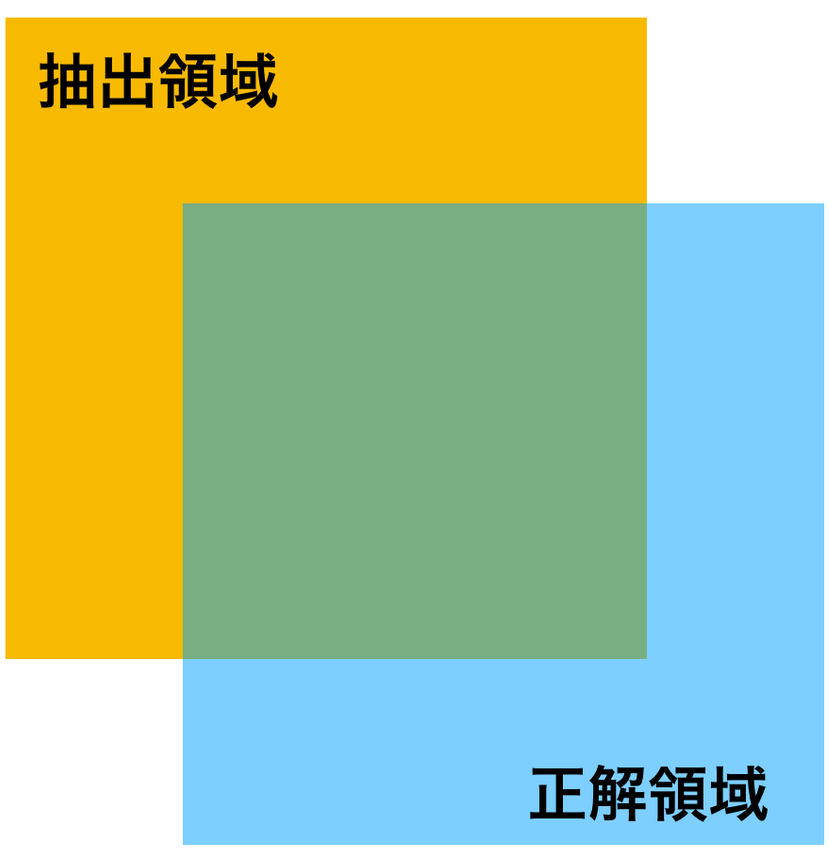
IoU

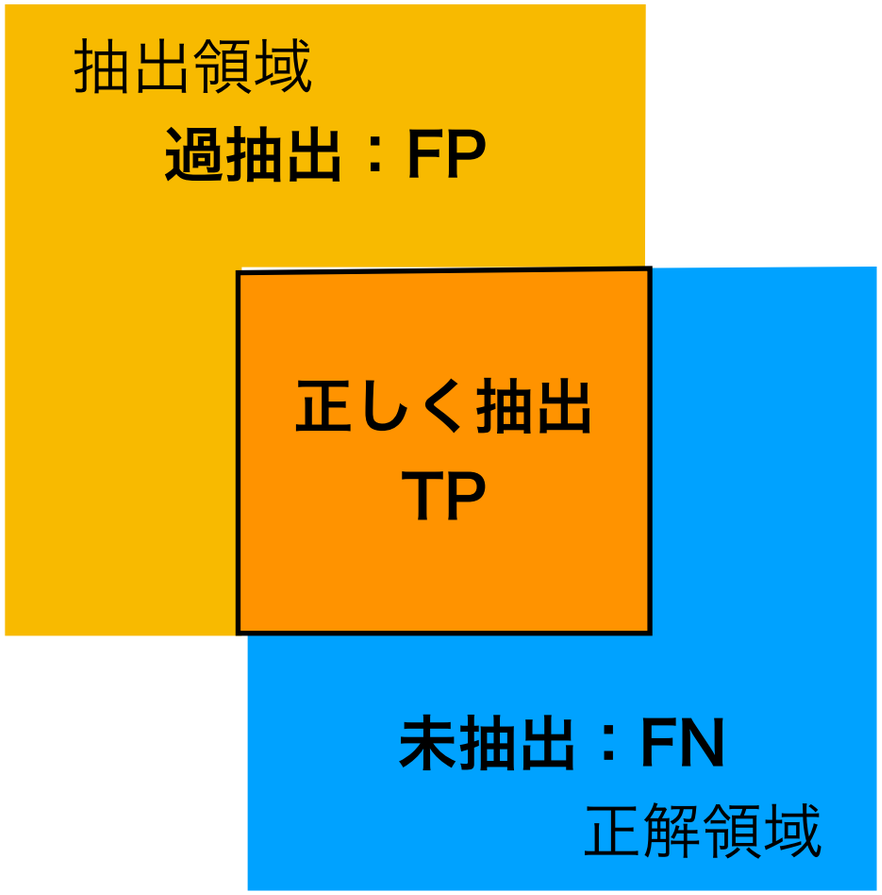
Precision

Recall

F1

評価方法：領域の評価





IoU (Jaccard) = TP/(TP+FP+FN)

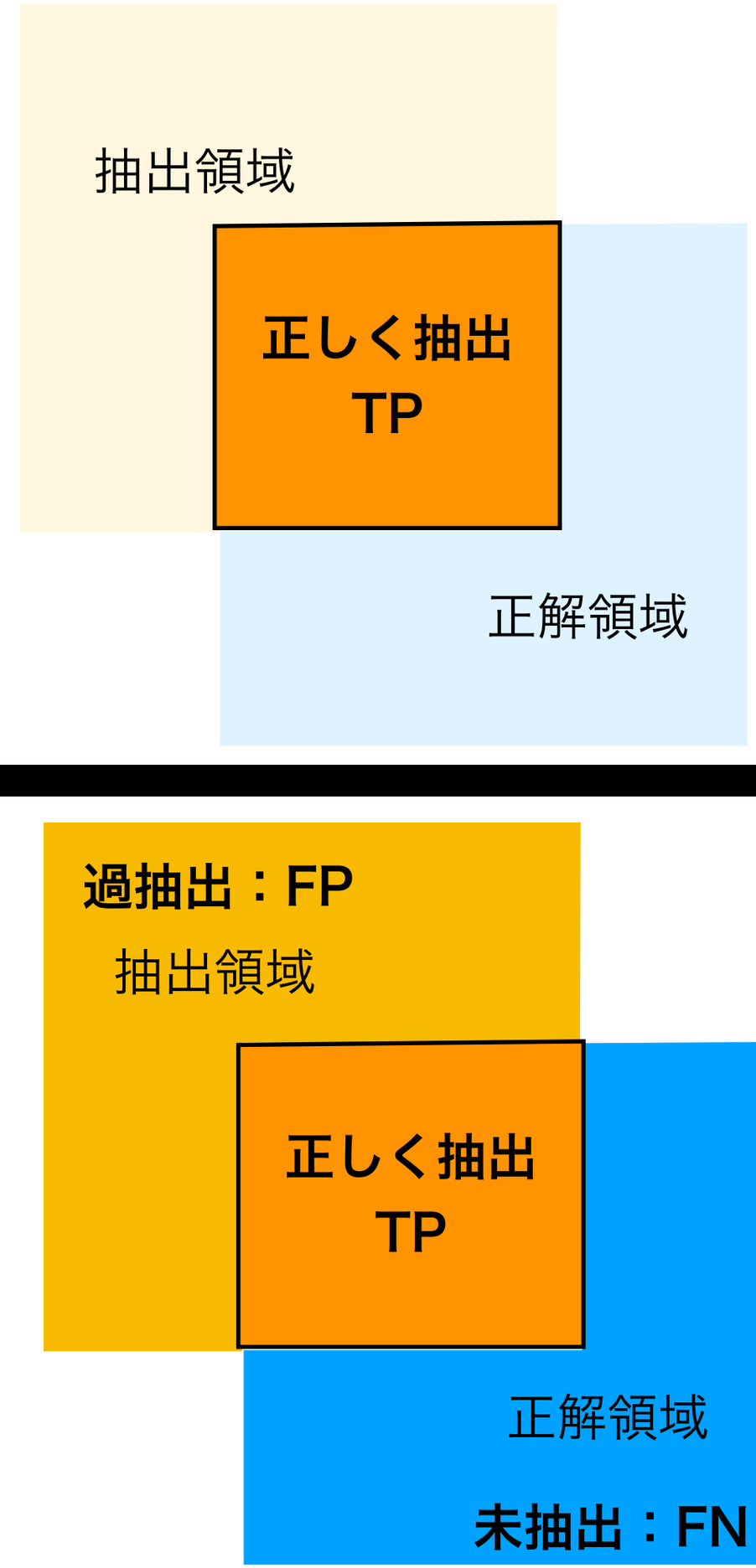
Precision = TP/(TP+FP)

Recall = TP/(TP+TN)

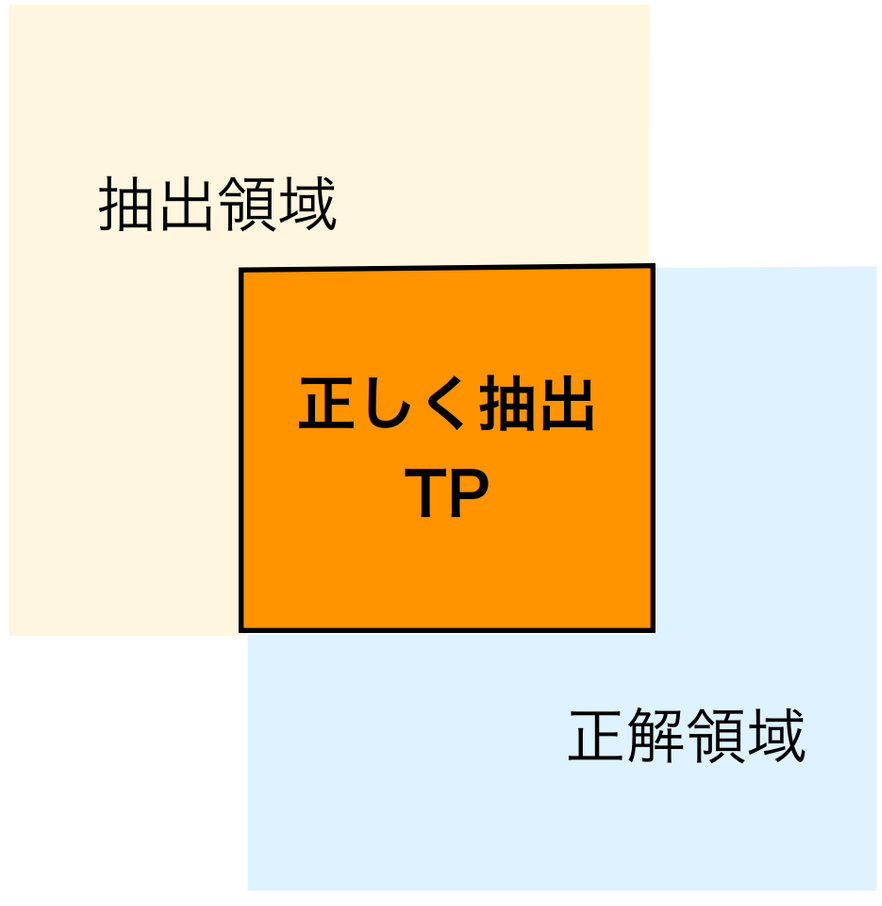
F1(Dice) = 2/{1/Precision + 1/ Recall} = 2TP/(2TP+FP+FN)

IoU

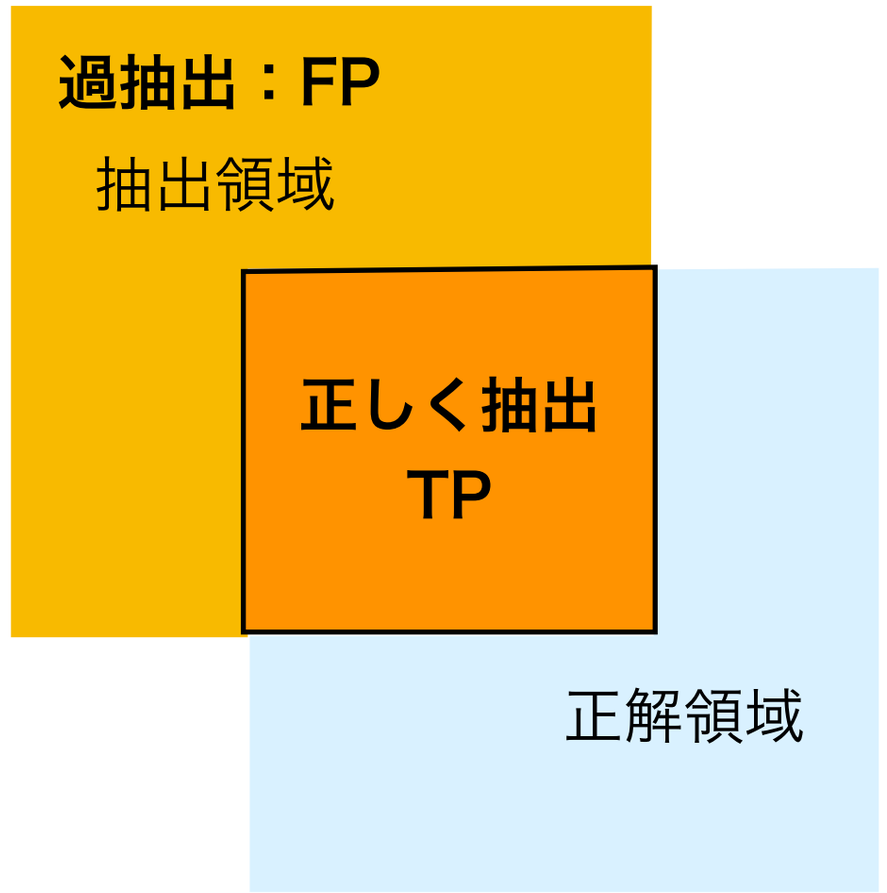
IoU (Jaccard) = TP/(TP+FP+FN)



Precision

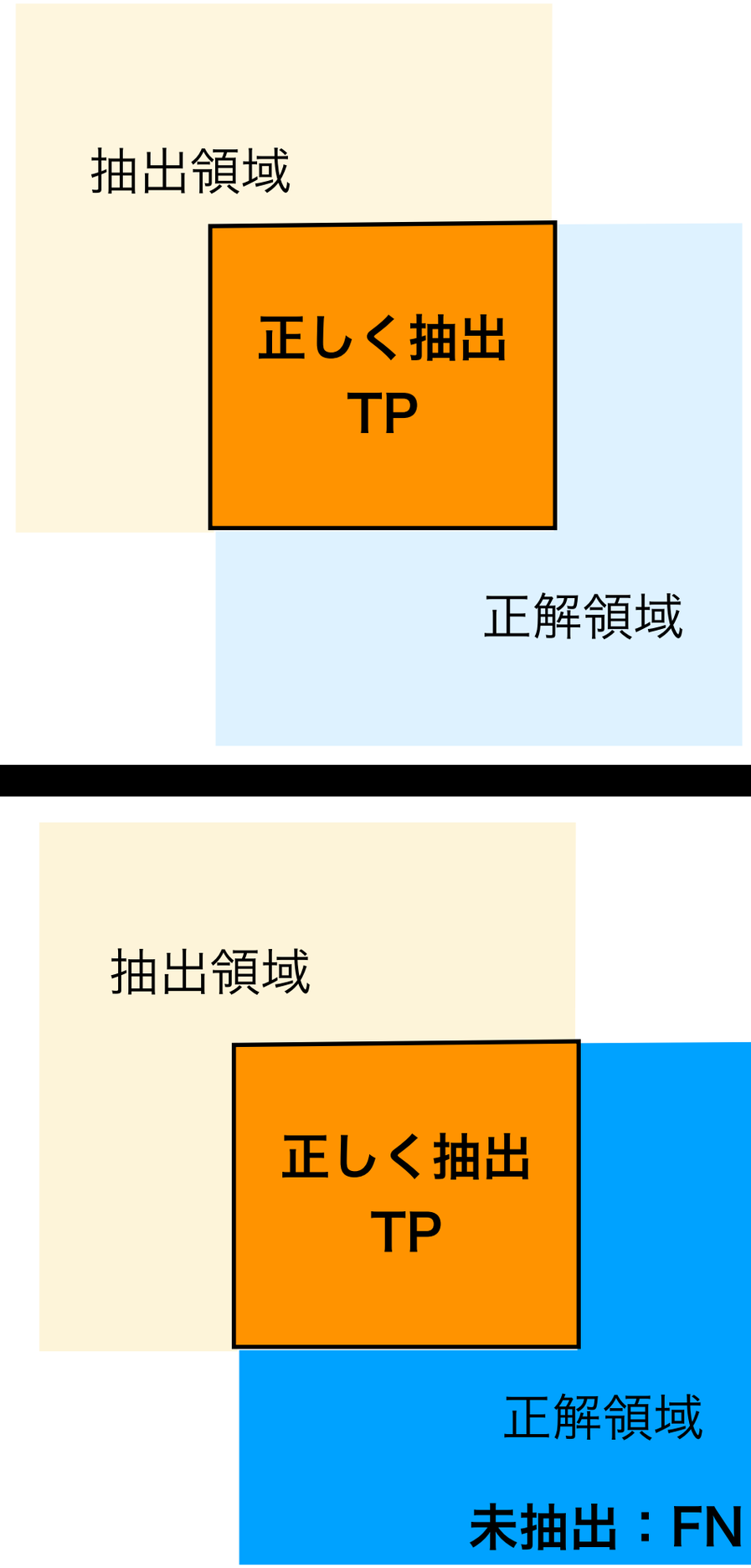


Precision = TP/(TP+FP)

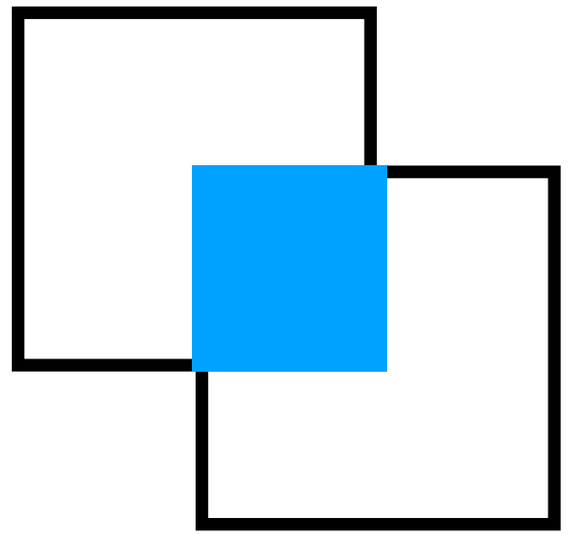


Recall

Recall = TP/(TP+TN)



評価方法：領域の評価



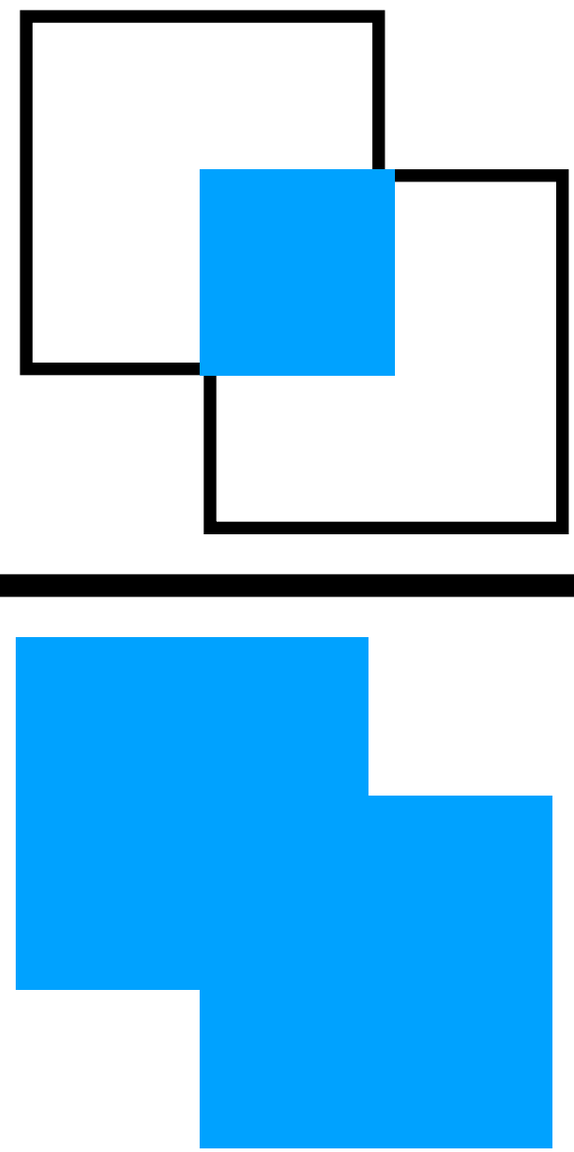
Intersection-over-Union (IoU) =

(Jaccard index)

評価方法：領域の評価

Intersection-over-Union (IoU) =

$$\frac{|A \cap B|}{|A \cup B|}$$



A, Bは集合.

|A|は，Aの要素数を表すよ.

画像の場合には画素数を用いるよ.

(Jaccard index)

評価方法：領域の評価

Intersection-over-Union (IoU) = (Jaccard index)

$$\frac{|A \cap B|}{|A \cup B|}$$

Sørensen–Dice index =

$$\frac{2|A \cap B|}{|A| + |B|}$$

Simpson coefficient = (Overlap coefficient)

$$\frac{|A \cap B|}{\min\{|A|, |B|\}}$$

min{a, b}は，a, bの小さい方を表すよ

評価方法：領域の評価

２つの集合において，一部が共通要素の場合

IoU < Dice < Simpson

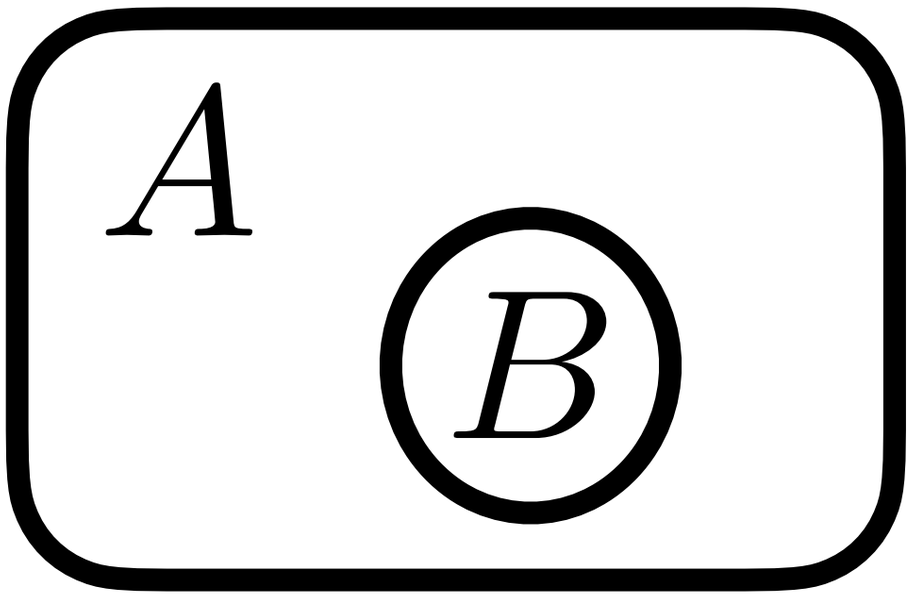
２つの集合が，空集合の場合 $|A \cup B| = 0$

IoU = Dice = Simpson = 0

２つの集合において，一つが他に完全に含まれる（真部分集合）の場合

IoU ≠ 1, Dice ≠ 1, Simpson = 1

評価方法：領域の評価



$$|A| = 100$$
$$|B| = 10$$

Intersection-over-Union (IoU) (Jaccard index)

$$= \frac{|A \cap B|}{|A \cup B|} = \frac{10}{100} = 0.1$$

２つを似ていないと評価したいとき

Sørensen–Dice index

$$= \frac{2|A \cap B|}{|A| + |B|} = \frac{2 \cdot 10}{100 + 10} = \frac{20}{110} = 0.182$$

２つを似ていると評価したいとき

Simpson coefficient

$$= \frac{|A \cap B|}{\min\{|A|, |B|\}} = \frac{10}{\min\{100, 10\}} = \frac{10}{10} = 1$$

課題に取り組みましょう

画像分類のポイント

- 分類問題を深層学習で解決する：**出力層の作り方**
- ファインチューニングを理解する：**モデルの改造**
- 転移学習を理解する：**モデルの選択と重みの固定**
- 評価方法：**２クラス分類／多クラス分類**

領域分割のポイント

- 領域分割のモデルを作る：**U-Net, U-Net++**
- 結果を可視化する：**ラベル番号をカラーで表示**
- 必要な領域を抽出する：**出力の整数化**
- 評価方法：**IoUなど**