

データサイエンス講座

Level 01-9

第9章：深層学習と先端技術を体験

第9章：深層学習と先端技術を体験

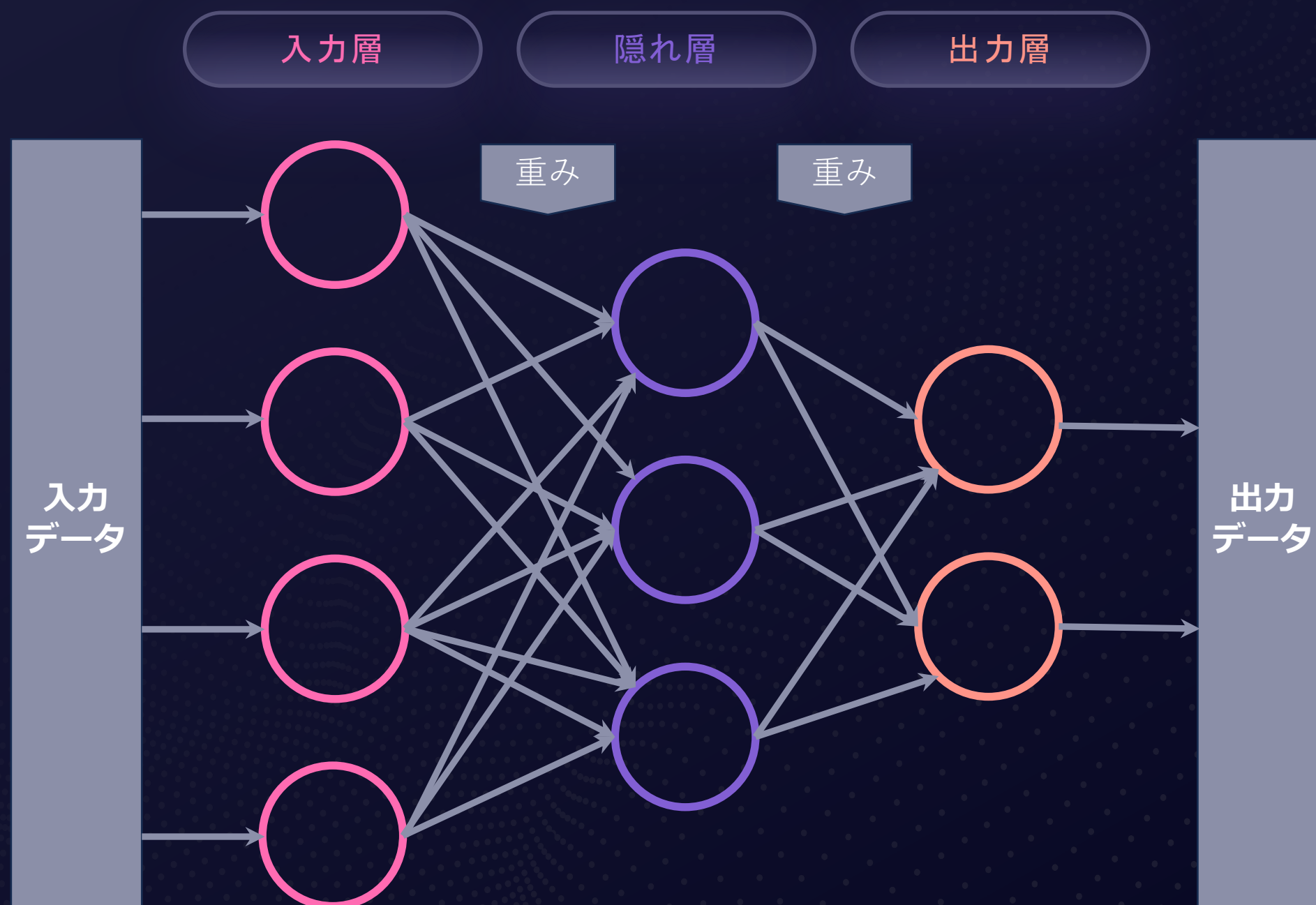
深層学習基礎

ビジネスへの適用と重要性

深層学習は、ニューラルネットワークを活用してパターンを学習し、複雑な問題を解決する高度な機械学習手法で、画像認識、自然言語処理、異常検知などの分野で多大な成功を収めています。その自動特徴抽出能力や高い汎化性能により、従来のルールベースや単純な機械学習では対応できなかった複雑なデータ解析を可能にし、顧客体験の向上、業務効率化、新規事業創出といったビジネスへの適用が急速に進んでいます。特に、医療診断の精度向上や金融リスク管理、チャットボットによるカスタマーサポートの強化など、多岐にわたる分野で深層学習がもたらす影響は計り知れません。これにより、企業が競争優位を確立し、イノベーションを促進する上で不可欠な技術となっています。

ニューラルネットワーク

DS Lv.1-7



ノード

ニューラルネットワークの基本構成要素で、ニューロンとも呼ばれます。各ノードは、入力を受け取り、計算を行い、その結果を次のノードに送ります。

層（レイヤー）

ニューラルネットワークは複数の層で構成されており、それぞれ異なる役割を持ちます。

重み（ウェイト）

ノード間の接続の強さを示す値で、ネットワークが学習するための最も重要なパラメータ。

代表的な深層学習モデル

DS Lv.1-9

CNN (Convolutional Neural Network)

主に画像処理に特化したニューラルネットワーク。画像の特徴（エッジやパターン）を自動で抽出し、分類や認識を行う。

- 画像分類（例: 写真から猫と犬を識別）、物体検出。

RNN (Recurrent Neural Network)

時系列データや連続するデータを扱うために設計されたネットワーク。前の状態を記憶し、次の予測に活用する。

- 音声認識、文章生成、株価予測などの時系列データ解析。

LSTM (Long Short-Term Memory)

RNNの一種で、長期間の依存関係を扱うのが得意。記憶や忘却のメカニズムを持ち、より精度の高い解析を可能にする。

- 長文の文脈解析、翻訳、チャットボット。

Transformer

入力全体のデータ間の関係を一度に捉える革新的なネットワーク。注意機構（Attention）を用いて効率的かつ高精度に処理。

- 自然言語処理（例: ChatGPTや翻訳モデル）、画像生成など。

演習：MNIST 手書き数字分類



学習のポイント

1. 深層学習の基本構造: CNNモデルを使用して画像分類を実現する基本を学ぶ。
2. 画像データの前処理: 正規化や形状変換を行い、モデル入力に適した形に整える。
3. モデルの構築と訓練: 畳み込み層、プーリング層、全結合層の役割を理解。
4. モデル評価と視覚化: 学習済みモデルの精度を確認し、実際の予測結果を可視化。



動作の流れ

1. データの読み込み: MNISTデータセットをダウンロードし、トレーニングデータとテストデータに分割。
2. データの前処理: 画像データを正規化（0～1の範囲にスケーリング）。
3. モデルの構築: CNNの畳み込み層、プーリング層、全結合層を順に設計。
4. モデルのコンパイルと訓練: 損失関数と最適化手法を設定し、モデルをトレーニング。
5. モデルの評価: テストデータでモデルを評価し、精度を出力。
6. 予測の可視化: 実際の手書き数字画像と予測結果を表示。

9-01.py

```
import tensorflow as tf
from tensorflow.keras import layers, models
from tensorflow.keras.datasets import mnist
import matplotlib.pyplot as plt

# Load MNIST dataset
(X_train, y_train), (X_test, y_test) = mnist.load_data()

# Normalize the data
X_train = X_train / 255.0
X_test = X_test / 255.0

# Reshape for the model
X_train = X_train.reshape(-1, 28, 28, 1)
X_test = X_test.reshape(-1, 28, 28, 1)

# Build a simple CNN model
model = models.Sequential([
    layers.Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=(28,
28, 1)),
    layers.MaxPooling2D(pool_size=(2, 2)),
    layers.Flatten(),
    . . .
```


演習：CIFAR-10 画像分類



学習のポイント

1. 多クラス分類の基礎: CIFAR-10のような小規模データセットで、複数のクラスを正確に分類する方法を学ぶ。
2. CNNの応用: カラーデータ（RGB画像）を処理するためのモデル設計を体験。
3. 学習曲線とモデル評価: トレーニングと検証データの結果を比較し、過学習や未学習を検出。
4. 視覚的な結果分析: 画像に対する予測結果をラベルで可視化し、モデルの直感的な理解を深める。



動作の流れ

1. データの読み込み: CIFAR-10データセットをダウンロードし、トレーニングデータとテストデータに分割。
2. データの前処理: 画像データを正規化（0～1の範囲にスケーリング）。
3. モデルの構築: CNNの畳み込み層（特徴抽出）、プーリング層（次元削減）、全結合層（分類）を設計。
4. モデルのコンパイルと訓練: トレーニングデータを用いてモデルを学習し、検証データで性能を確認。
5. モデルの評価: テストデータを用いてモデルの精度を測定し、評価結果を出力。
6. 予測の可視化: 数枚の画像を表示し、モデルの予測ラベルと実際のラベルを比較。

9-02.py

```
import tensorflow as tf
from tensorflow.keras import layers, models
from tensorflow.keras.datasets import cifar10
import matplotlib.pyplot as plt

# Load CIFAR-10 dataset
(X_train, y_train), (X_test, y_test) = cifar10.load_data()

# Normalize the data
X_train = X_train / 255.0
X_test = X_test / 255.0

# Build a simple CNN model
model = models.Sequential([
    layers.Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=(32, 32, 3)),
    layers.MaxPooling2D(pool_size=(2, 2)),
    layers.Flatten(),
    layers.Dense(128, activation='relu'),
    layers.Dense(10, activation='softmax')
])

. . .
```


演習：簡易テキスト分類



学習のポイント

1. テキストデータの数値化: Bag of Words (BoW) モデルを用いて、テキストを数値ベクトルに変換。
2. ナイーブベイズモデルの適用: シンプルな分類モデルを使用して、テキスト分類の基本を学ぶ。
3. トレーニングと評価: モデルの訓練後、精度を測定し分類結果を解釈する。
4. カスタムデータの活用: 自分で用意した文章に対してモデルを適用し、予測を行う楽しさを体感。



動作の流れ

1. テキストデータの準備: サンプルのテキストデータとラベル（例: ポジティブ/ネガティブ）を定義。
2. テキストデータの数値化:
3. CountVectorizerを使用して、テキストをBag of Words形式の数値ベクトルに変換。
4. データ分割: トレーニングデータとテストデータに分割。
5. モデルの訓練:
6. ナイーブベイズモデルを使用して、トレーニングデータから分類モデルを学習。
7. モデルの評価: テストデータを使って精度を評価し、正解率を出力。
8. カスタム予測: 自作の文章をモデルに入力し、分類結果を出力。

9-03.py

```
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score

# Sample data
texts = [
    "I love this product", "This is the worst experience",
    "Amazing quality!", "Not worth the money", "Best purchase ever!",
    "Terrible service", "Very satisfied", "Would not recommend",
    "Absolutely fantastic", "Horrible experience"
]
labels = [1, 0, 1, 0, 1, 0, 1, 0, 1, 0] # 1 = Positive, 0 = Negative

# Convert text to feature vectors
vectorizer = CountVectorizer()
X = vectorizer.fit_transform(texts)

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, labels, test_size=0.3,
                                                    random_state=42)

. . .
```


DS Lv.1-9

お疲れ様でした

info@mjin.space

mjin.space