**２０２１　年度　　卒業論文**

卒業論文題目

|  |
| --- |
| 画像検索のための画像特徴ベクトルの次元数に着目した認識精度と計算コストの関係性の評価・実験 |

　指導教員　　　　　　鷹野　孝典　教授

神奈川工科大学　情報工学科

　　　　　学籍番号　　　　1821005

　　　　　学生氏名　　　　　 吉岡　拓郎

　提出日　２０２１年　１２月２３日　指導教員　　印

　受理日　２０２１年　１２月２３日　情報工学科長　　印

# **論文要旨**

内容

[**論文要旨** 2](#_Toc89256446)

[**1.** **まえがき** 1](#_Toc89256447)

[1.1 はじめに 1](#_Toc89256448)

[1.2 本論文の構成 1](#_Toc89256449)

[**2.** **関連研究** 2](#_Toc89256450)

[**3.** **提案方式** 3](#_Toc89256451)

[3.1 提案方式の概要 3](#_Toc89256452)

[3.2 提案方式の手順 3](#_Toc89256453)

[3.2.1 複数モデルの作成について 4](#_Toc89256454)

[3.2.2 特徴ベクトルの抽出 5](#_Toc89256455)

[3.2.3 評価・実験について 5](#_Toc89256456)

[**4.** **実装** 6](#_Toc89256457)

[4.1 実装システム概要 6](#_Toc89256458)

[4.2 実装環境 6](#_Toc89256459)

[4.3 モデルの作成方法 6](#_Toc89256460)

[4.4 特徴ベクトルの抽出方法 7](#_Toc89256461)

[4.5 評価・実験の方法について 8](#_Toc89256462)

[**5.** **むすび** 10](#_Toc89256463)

[**参考文献** 11](#_Toc89256464)

[その他 12](#_Toc89256465)

# **まえがき**

## はじめに

TwitterやInstagram等のソーシャルネットワーキングの普及に伴い，写真や画像の投稿が盛んに行われるようになり，大量の画像や写真が蓄積されている．そこで，ユーザが目的の写真や画像にアクセスする手段として，画像検索機能の重要性が増している．又，画像認識の性能向上のため，多くの研究が推進されている．

画像検索機能の画像認識において，画像の特徴量が高次元になるほど，認識性能が高くなる．しかし，特徴量が高次元になると特徴ベクトルの算出コスト，新式処理の計算コストが大きく増加してしまい，解析が困難になってしまう．この現象を次元の呪いという．次元の呪いとは，データの次元数が大きくなり過ぎると，そのデータで表現できる組み合わせが飛躍的に多くなってしまい，その結果，手元にある有限なサンプルデータでは十分な学習結果が得られなくなる.計算コストが莫大となるだけでなく，十分な学習結果が得られず、未知のデータに適切に対応出来なくなる等の不具合が発生することである．また，低次元だと，意味情報が損失し，認識性能が低くなるが，処理時間が減少するといったことが起こる．

上記のことを念頭に入れて，画像認識の画像処理が有効に行える範囲を調査する．ここで記載する意味情報とは，画像を認識する際に，その判断材料となる情報のことを指す．

画像認識の性能を向上させるために，画像認識に有効な特徴ベクトルとどれくらいの計算コストで処理できるのか計測する．それらの結果を踏まえて評価・考察をする．計算コストを抑えた画像認識手法を調査することで，画像検索システムを成り立たせるうえで，認識性能が正常に扱える有効な範囲はどこなのか明確にする．

## 本論文の構成

本論文は全６章で構成されている。本章では研究の背景や動機，目的について述べる．第２章では，モデルの作成手法，それらにまつわる関連研究について述べる．第３章では，本研究の提案方式とモデルの内容，学習データ，実験環境について述べる。第４章では，提案方式を構築した本システムの設計及び実験システムの実装について述べる．第５章では，Alexnetをもとに作成したモデルを使用して，性能について比較実験をする．実験の概要と，比較実験後の結果を述べる．第６章では，結論と今後の展望について述べ

# **関連研究**

# **提案方式**

## 提案方式の概要

本研究では，画像検索システムにおいて，画像処理が有効に扱える範囲を調査するための画像検索評価システムを提案する．学習モデルを自作し，次元数を操作して画像検索システムとして機能するかどうか実験・評価をする．

CNN　(Convolutional Neural Network)　を用いて抽出した画像の特徴ベクトルは，画像の意味情報が保持されていると仮定する．ここでの意味情報とは，画像を認識する際に，その判断材料となる情報のことを指す．特徴ベクトルの生成に，CNNを利用する．CNNを用いた特徴ベクトルの生成方法としては，識別層の一つ二つ手前の全結合層を用いる．また，次元数を変化させる層は，特徴ベクトルを生成する層と同じく，識別層から一つ二つ手前の全結合層の値を変化させる．

## 提案方式の手順

　この提案方式では，画像認識の認識性能を高く保ち，計算コストを抑える最適な画像特徴ベクトルの次元数を調査する．画像集合を用意し，CNNを用いて特徴ベクトルを生成する．本研究では，cifar10データセット内のテストデータ一万件を画像集合として用意する．

初めに，学習済みの分類モデルを作成する．本研究では，cifar10のデータセットをモデルの学習に用いる．作成した学習済みモデルを使用して，特徴ベクトルの抽出を行う．抽出した特徴ベクトルはpickleファイルに保存する．その後，抽出した一万件の特徴ベクトルを用いて，特徴ベクトル間の距離を計算する．そして，距離が近い順に類似度が高いとして，評価を行う．そのときの計算コストと精度について評価する．具体的な流れを下記に記入する．また，提案システム図を図3-1に示す．

STEP-1: 学習済みのモデルを作成し，それを用いて評価をするための特徴ベクトルを抽出する．

STEP-2: STEP-1にて抽出した特徴ベクトルをpickle形式で保存する．

STEP-3: STEP-2で特徴ベクトルを保存したpickleファイルロードをして，別のプログラムでも使用できるようにする．

STEP-4: STEP-3でロードを行った特徴ベクトルについて，実験・評価を行う．遠隔式の講義によって提供されたスライド講義動画をユーザが視聴する．

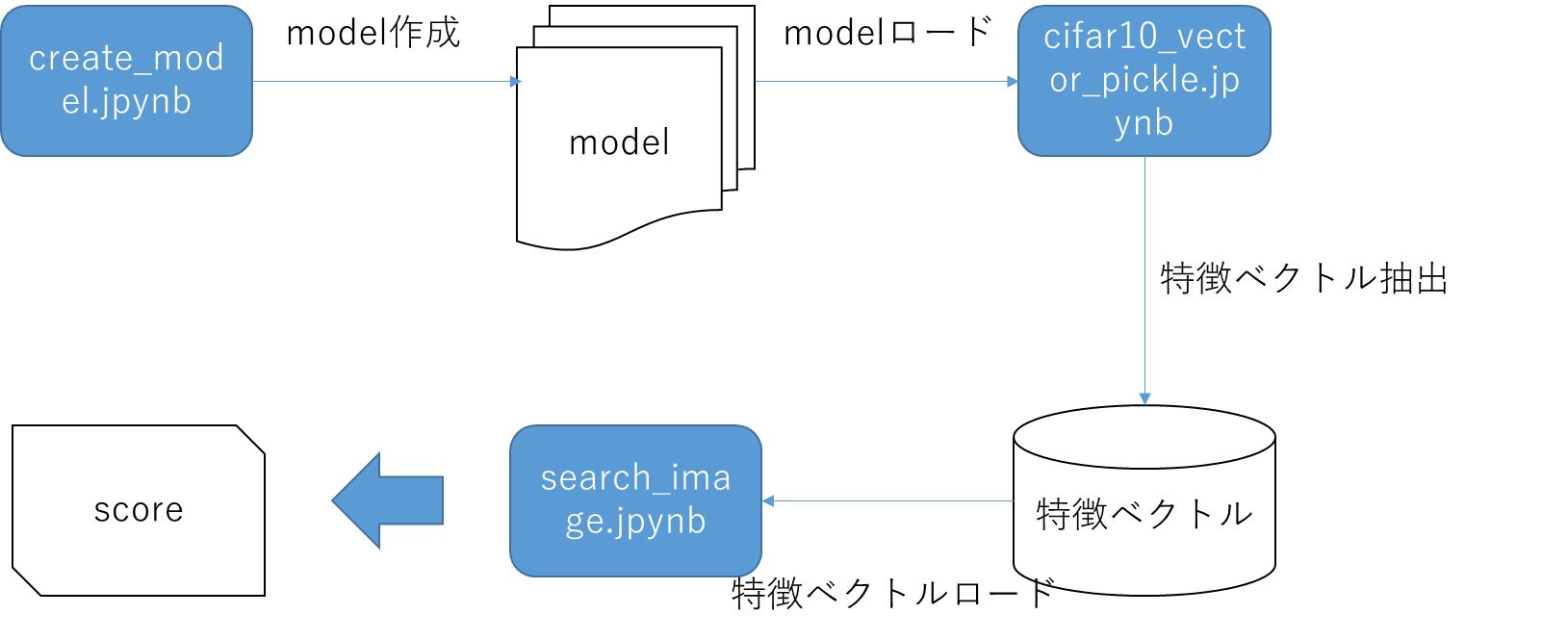


図3-1　提案システム図

### 複数モデルの作成について

本研究では，次元数を変化させて最適な画像検索システムを成り立たせるうえで，認識精度に有効な範囲を調査する．そのため，次元数を変化させたモデルを複数作成する必要がある．本研究では，識別層(softmax)の一つ二つ手前の全結合層の次元数の値を変化させえてモデルを作成する．(図3-2)

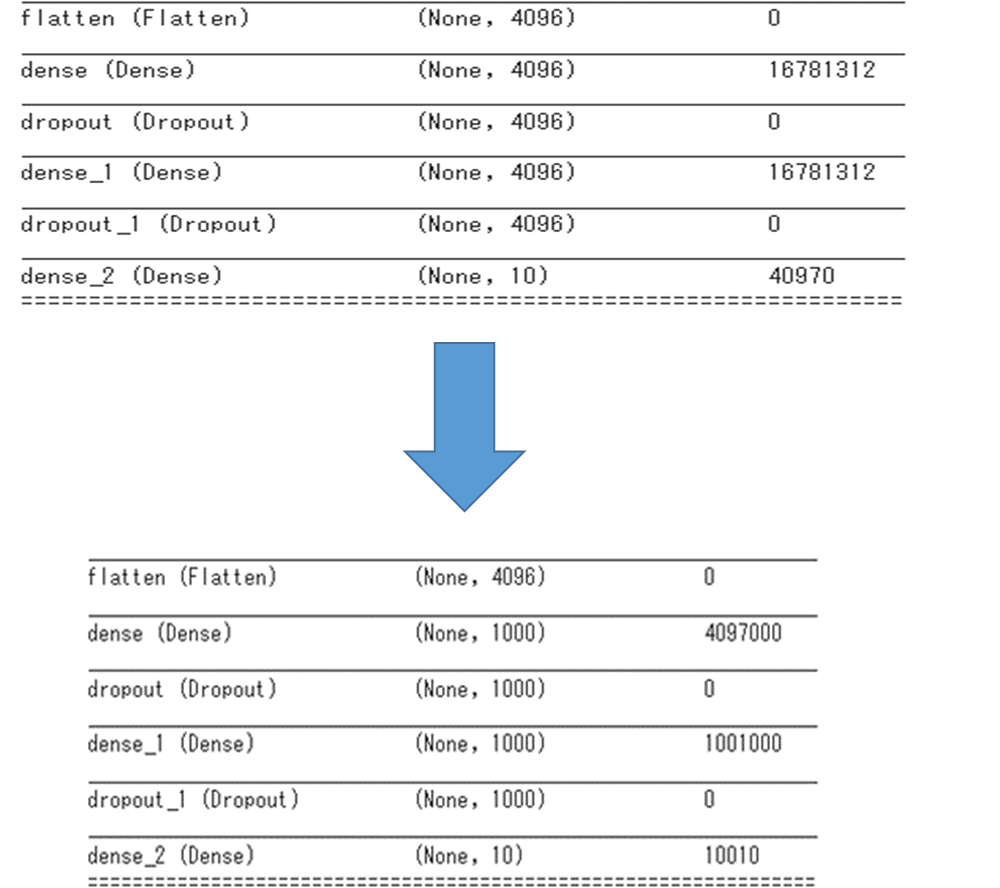


図3-2　モデル変更箇所の例

### 特徴ベクトルの抽出

本研究では，17層のモデルを使用して，特徴ベクトルの抽出を行う．実際に，特徴ベクトルを抽出する層は，15層目のDense層から特徴ベクトルを抽出する．例として，15層目の次元数が4096の場合の特徴ベクトルを表3-1に示す．

表3-1　次元数4096の特徴ベクトル一部

|  |
| --- |
| array([[ 0.90278023, 0.0702083 , 0.67032325, ..., 0.58752143,  0.8521021 , 0.9570551 ],  [ 0.94941396, 0.26129064, 0.85260475, ..., 0.46636236,  0.7725632 , 0.8928799 ],  [ 0.962252 , 0.26241592, 0.78311104, ..., 0.56774634,  0.88178414, 0.88763523],  ...,  [ 0.89410365, 0.8571504 , 0.87597483, ..., 0.42534053,  0.6453284 , 0.79393625],  [ 0.89593625, 0.62443584, -0.02818177, ..., 0.6658653 ,  0.7585609 , 0.9019706 ],  [ 0.950409 , 0.57382387, 0.6239227 , ..., 0.3752147 ,  0.6969327 , 0.9168432 ]], dtype=float32) |

### 評価・実験について

ここまでの提案で，モデルの作成，特徴ベクトルの抽出について説明を行った．ここから本研究の画像検索における認識性能について評価，実験を行う．具体的には，特徴ベクトル間の距離を測り，それをもとに距離が近い順に類似度が高いとして，列挙する．そして，評価する際は，画像のラベルを確認して，元の画像と同じ場合は正解としてscoreを１ずつ足していく．そのscoreの合計と正答率の割合を求めて，どれくらいの認識性能が出せているのかを確認する．

# **実装**

## 実装システム概要

実装システムでは，画像検索において，画像処理が有効に扱える範囲を調査するために，GoogleCorabolatoryの仮想環境を使用して，pythonを用いて実装をした．検索の対象は，cifar10データセット内のテストデータ1万件を使用して，特徴ベクトルの抽出を行う．

## 実装環境

## モデルの作成方法

モデルの作成では，GoogleCorabolatryを利用し，pythonでシステムの開発を行う．ここでは5との次元数の異なるモデルを作成する．それぞれ13層と15層のdence層の部の次元数を変化させてモデルを作成していく．以下に作成したモデルについて表に示す．



図 4-1　作成したモデル一覧表

　作成したモデルの内容について，下記の表4-1に示す．

|  |
| --- |
| Model: "sequential"  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Layer (type) Output Shape Param #  =================================================================  conv2d (Conv2D) (None, 32, 32, 96) 2688    max\_pooling2d (MaxPooling2D (None, 16, 16, 96) 0  )    batch\_normalization (BatchN (None, 16, 16, 96) 384  ormalization)    conv2d\_1 (Conv2D) (None, 16, 16, 256) 614656    max\_pooling2d\_1 (MaxPooling (None, 8, 8, 256) 0  2D)    batch\_normalization\_1 (Batc (None, 8, 8, 256) 1024  hNormalization)    conv2d\_2 (Conv2D) (None, 8, 8, 384) 885120    conv2d\_3 (Conv2D) (None, 8, 8, 384) 1327488    conv2d\_4 (Conv2D) (None, 8, 8, 256) 884992    max\_pooling2d\_2 (MaxPooling (None, 4, 4, 256) 0  2D)    batch\_normalization\_2 (Batc (None, 4, 4, 256) 1024  hNormalization)    flatten (Flatten) (None, 4096) 0    dense (Dense) (None, 4096) 16781312    dropout (Dropout) (None, 4096) 0    dense\_1 (Dense) (None, 4096) 16781312    dropout\_1 (Dropout) (None, 4096) 0    dense\_2 (Dense) (None, 10) 40970    =================================================================  Total params: 37,320,970  Trainable params: 37,319,754  Non-trainable params: 1,216  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

## 特徴ベクトルの抽出方法

特徴ベクトルを抽出するのは，cifar10\_vector\_pickle.jpynbによって行う．このプログラムでは，作成済みの学習済みモデルをロードする必要がある．

　以下のコードでモデルをロードすることができる．

|  |
| --- |
| model = load\_model('alexnet\_cifar10\_2000.h5')  model.summary() |

　cifar10データセット内のテストデータ一万件を特徴ベクトル抽出に用いるため，cifar10データセットをロードします．以下にコードを示す．

|  |
| --- |
| from keras.datasets import cifar10  (x\_train, y\_train), (x\_test, y\_test) = cifar10.load\_data() |

　ここまでで，特徴ベクトルを取得できる準備は整っている．以下に特徴ベクトルを取得するコードを示す．

|  |
| --- |
| dense =models.Model(inputs=model.input, outputs=model.layers[-3].output)  dense\_out = dense.predict(x\_test) |

一番上の行で，特徴ベクトルを抽出したい層を指定している．本研究では，上から15番目の層から特徴ベクトルの抽出を行う．次の行の所で，特徴ベクトルの抽出を行っている．これで，一万件の特徴ベクトルの抽出をすることができる．抽出した特徴ベクトルの保存については，pickleを用いる．理由としては，numpy配列として扱うことができるため，画像検索評価システムに落とし込みやすいからである．又，保存方法としてcsvも考えていたが，文字列として保存されてしまうので，扱える形に合うるのが少々手間である観点から使用していない．しかし，一万件のデータから一つ一つ取り出すといった作業は，pickleではできないため，必要に応じて使い分ける必要がある．以下に示すコードは，pickleでの保存，ロードの方法である．

|  |
| --- |
| with open('cifar10\_vecs2000.pickle', 'wb') as f:    pickle.dump(dense\_out, f) |
| with open('cifar10\_vecs2000.pickle', 'rb') as f:    b = pickle.load(f) |

　上記のように，簡単なコードで一万件の特徴ベクトルを保持することができる．例として，4096次元で抽出した特徴ベクトルを以下に示す．

|  |
| --- |
| array([[ 0.90278023, 0.0702083 , 0.67032325, ..., 0.58752143,  0.8521021 , 0.9570551 ],  [ 0.94941396, 0.26129064, 0.85260475, ..., 0.46636236,  0.7725632 , 0.8928799 ],  [ 0.962252 , 0.26241592, 0.78311104, ..., 0.56774634,  0.88178414, 0.88763523],  ...,  [ 0.89410365, 0.8571504 , 0.87597483, ..., 0.42534053,  0.6453284 , 0.79393625],  [ 0.89593625, 0.62443584, -0.02818177, ..., 0.6658653 ,  0.7585609 , 0.9019706 ],  [ 0.950409 , 0.57382387, 0.6239227 , ..., 0.3752147 ,  0.6969327 , 0.9168432 ]], dtype=float32) |

## 評価・実験の方法について

　評価・実験をするにあたって必要となる特徴ベクトルを抽出して，保存することができた．ここから本研究の本題である画像検索における，認識性能について評価・実験を行う．

# **むすび**

# **参考文献**

# その他