畳み込みニューラルネットワークによる曲がり角の画像の識別

M2018SC012 吉井健人
指導教員：大石泰章

1　はじめに

文献 [1] のディープビリーフネットワークの研究がきっかけとなって，ニューラルネットワークが注目されるようになった．同時に，ニューラルネットワークの画像認識への有用性も注目され，画像認識技術は大きく発展した．ニューラルネットワークを用いた画像認識技術は，自動車における歩行者や障害物の検知や，スマートフォンの顔認証，手書き文字の識別など，世の中のいたるところで使われており，これからも今まで以上に研究され必要とされる技術である．

本研究の最終的な目的は，道路における曲がり角の識別である．画像認識によって曲がり角が近くにあるかどうかを判断させ，曲がり角の危険を見つける．直線道路と比べて対向車線の車と事故を起こしやすく，歩行者が横切る可能性も高い曲がり角は，自動車を運転するにあたって特に危険な場所である．この研究によって曲がり角を判別し自動車を減速させれば，事故が起きる可能性を大きく減らすことができる．

本研究では，集めた道路の画像を用いて独自のデータセットを作成し，それをニューラルネットワークに学習させることで曲がり角が近いかどうかを識別させる．

2　畳み込みニューラルネットワーク

本研究で扱う畳み込みニューラルネットワークとは，畳み込み層とプーリング層を含むニューラルネットワークのことを指す．

まず，畳み込み層について説明する．畳み込み層とは，元の画像にフィルタをかけることで画像の特徴を見つけるためのものである．文献 [2] にあるように，畳み込み層に入力される画像のサイズを，画素数を単位として $W \times W$，チャンネル数を K とする．入力画像においてチャンネルとは画像の色に対応し，赤色，緑色，青色の 3 つの色を表すチャンネルがある．チャンネルの画像は各画素の信号が強い場合は白色，弱い場合には黒色で表現されたものになる．位置 $(i,j)\,(0 \leq i \leq W-1, 0 \leq j \leq W-1)$，チャンネル $k\,(0 \leq k \leq K-1)$ における画素値を z_{ijk} と書く．畳み込み層の出力において設定されたチャンネルの数 M はフィルタの数に等しい．フィルタの画像サイズを $H \times H$ とする．このときフィルタの性質は $h_{pqkm}(0 \leq p \leq H-1, 0 \leq q \leq H-1, 0 \leq k \leq K-1, 0 \leq m \leq M-1)$ で表され，畳み込み層の出力は

$$u_{ijm} = \sum_{k=0}^{K-1} \sum_{p=0}^{H-1} \sum_{q=0}^{H-1} z_{si+p,sj+q,k} h_{pqkm} \tag{1}$$

によって表される．ただし，s はストライドと呼ばれ，フィルタの適用位置の間隔を表す．すなわち，s 画素ずつフィルタをずらしながら，出力画像を計算していく．

畳み込み層の出力 u_{ijm} を活性化関数に入力する．活性化関数とは次の層に渡す値を整えるためのものである．本論文では ReLU 関数という活性化関数を使用する．ReLU 関数の入力値を x，出力値を y とすると，

$$y = \begin{cases} 0 & (x<0), \\ x & (x \geq 0) \end{cases} \tag{2}$$

の関係がある．

次にプーリング層について説明する．プーリング層は畳み込み層で抽出された特徴の位置感度を低下させることで，画像内における微小な位置の変化に対して影響を受けにくくするためのものである．また，本研究においてプーリング層では最大プーリングというやり方を用いる．画像サイズ $W \times W \times K$ の入力画像 z_{ijk} において画素 (i,j) を中心とした $G \times G$ の範囲に含まれる画素の集合を P_{ij} とする．出力値を u_{ijk} とすると，

$$u_{ijk} = \max_{(p,q) \in P_{ij}} z_{pqk} \tag{3}$$

となる．

多層のニューラルネットワークであるほど，画像データは，畳み込み層やプーリング層に通すことによって，画像サイズが減少していき，いずれ出力される画像サイズは 1 となってしまう．そこで，画像データの周囲に画素値 0 の画素を 1 画素分埋め込むパディングを行い，畳み込み層やプーリング層の出力サイズが減少しないようにする．

また，訓練データに対して過度に対応してしまい，未知のデータに対して対応できないという過学習を抑制するために，畳み込み層とプーリング層の他にバッチノーマライゼーション層を追加する．バッチノーマライゼーション層は，入力された画像データをデータの分布が平均 0 分散 1 になるように正規化することで，データの分布の偏りを減らすためのものである．バッチノーマライゼーション層の利点は，過学習の抑制のほかに，学習が高速になること，初期値への依存性が少なくなることがある．

3　使用する環境

google colaboratory はクラウド上で python を使用することができるサービスである．主な利点は機械学習に必要なライブラリがある程度そろっているという点，linux のコマンドを windows 上でも使えるという点，そして，強力な GPU を使用することができるので，学習を短時間で行うことができ，特定の GPU でしか使えないライブラリも使用することができるという点である．また，クラウド上でプログラムを実行するので，初めから，ある程度環境が整っており，さらに，PC 環境の影響を受けな

図 1　データセットの画像

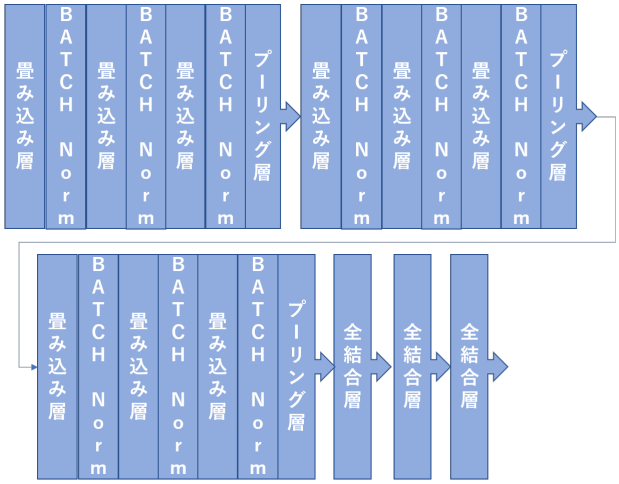

図 2　畳み込みニューラルネットワークの構造

いといった利点もある．本研究では google colaboratory を使い，python3 を GPU で動かす．

4　使用するデータセット

カリフォルニア大学バークレー校が公開した「Berkeley DeepDrive 100K」と呼ばれるデータセットを用いる．これは，自律自動車に搭載されたカメラの録画した映像 10 万本やそれらから抽出された画像からなる．特に，データセット中の画像の一部を，曲がり角が近くにある画像と，曲がり角がないか遠くにある画像に分けて独自のデータセットを作成し，使用する．作成したデータセットの枚数は曲がり角のない直線道路の訓練データが 519 枚，テストデータが 150 枚で，曲がり角の訓練データが 489 枚，テストデータが 150 枚である．また，画像サイズはもともと 1280×720 であるが，これを変換し，320×180 にしたものを入力データとする．データセットの画像の例を図 1 に示す．画像のラベルは左から曲がり角，直線道路，曲がり角，直線道路である．

5　画像認識

図 2 に用いる畳み込みニューラルネットワークの構造（ReLU 関数は省略）を示す．畳み込み層のフィルターのサイズは 3×3 でストライドは 1，プーリング層のフィルターのサイズは 2×2 でストライドは 2 である．さらにそれらの層の後に，畳み込み層やプーリング層によって特徴部分が取り出された画像を結合するための 3 層の全結合層がある．最後の全結合層が出力層であり，出力数は 2 である．出力層以外の畳み込み層と全結合層の出力は活性化関数である ReLU 関数によって変換される．

訓練データを学習させる回数を 20 回としたときの訓練データ及び，テストデータに対する出力をラベルと比べたときの正答率（精度）の変化を図 3 に示す．精度は 1 に近い方が畳み込みニューラルネットワークの学習ができている．

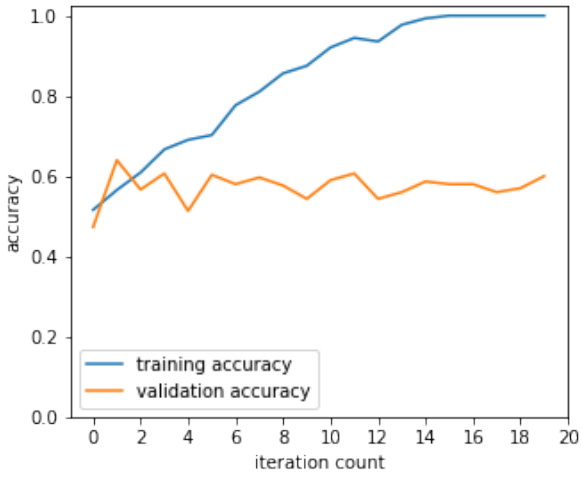

図 3　畳み込みニューラルネットワークにおける訓練データとテストデータの精度

図 3 から認識精度は 5,6 割程度であり，十分な学習ができていないことが分かる．この原因として学習に用いた画像の枚数が少ないことが考えられる．学習にかかった時間は 11 分 31 秒である．

6　転移学習

6.1　転移学習の概要

転移学習とは、あるデータセットを学習した学習済みのニューラルネットワークを，別のデータセットに適用することで効率的に学習させる方法である．転移学習を行うことによって，少ないデータでも高い精度を出すことができ，学習時間を短縮できる．本研究では ResNet の学習済みモデルを使用する．

6.2　ResNet

ResNet は 2015 年に考案されたニューラルネットワークのモデルである [3]．基本的に畳み込みニューラルネットワークは層を増やすほど性能は上がるが，単純に層を増やすと劣化問題（層が深いモデルの学習において，精度が層が浅いモデルより劣化する現象）が起こってしまう．ResNet ではそういった問題を大きく改善している．

図 4 は ResNet の最初と最後の層及び，最初から 2 つ分のブロックの構造（ReLU 関数は省略）を示す．ResNet の特徴は，通常の層に加えてブロックの入力から出力をつなぐショートコネクションが存在することである．ブロックの入力を x，出力を $F(x)$ とすると，ショートコネクションも含めた全体の出力は $H(x) = F(x) + x$ となる．学習によって整形されるブロックの出力 $F(x)$ は $H(x)$ と x の差であるので，このブロックを残差ブロックという．このようなモデルにすることによって ResNet は劣化問題を改善し，通常の畳み込みニューラルネットワークよ

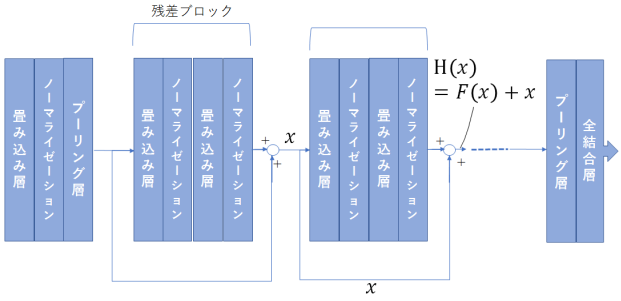

図 4　ResNet の構造

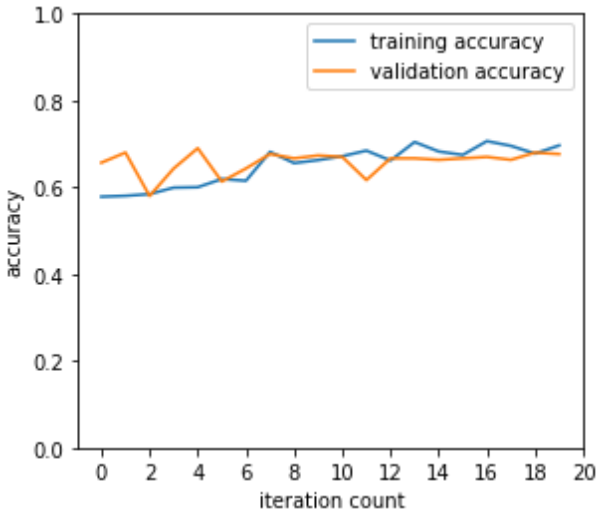

図 5　ResNet における訓練データとテストデータの精度

りも層が深くても，精度を上げることができる．

図 4 の最後のプーリング層はグローバルアベレージプーリング層といい，入力データの各チャンネルで画素の平均を求め，それらをベクトルとして出力する．利点は全結合層の代わりにするため，モデルのパラメータが減り過学習を軽減できる点である．また，最後の全結合層が出力層であり，出力数は 2 である．最初のバッチノーマライゼーション層と残差ブロックの一つ目のバッチノーマライゼーション層の出力は活性化関数である ReLU 関数によって変換される．

使用する ResNet の畳み込み層と全結合層の合計は 18 層である．まず，ResNet の性能を確かめるため，転移学習を行わずに ResNet に訓練データを学習させる．ResNet の入力データにはもともとの画像サイズ 1280×720 を使用する．訓練データを学習させる回数を 20 回としたときの訓練データ及び，テストデータに対する出力をラベルと比べたときの正答率（精度）の関係を図 5 に示す．

図 5 から，精度は 6,7 割程度であり，図 2 の畳み込みニューラルネットワークに比べて精度は上がっていることが分かる．学習にかかった時間は 18 分 5 秒である．

ここで，図 2 の畳み込みニューラルネットワークの畳み込み層を 6 層増やし，図 4 の ResNet と同じ様に，畳み込み層と全結合層の合計を 18 層としたときの，訓練データ及び，テストデータに対する出力をラベルと比べたときの正答率（精度）の関係を図 6 に示す．

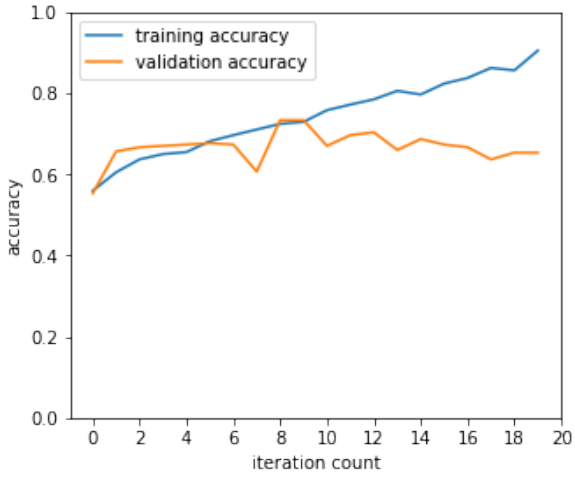

図 6　18 層の畳み込みニューラルネットワークの訓練データとテストデータの精度

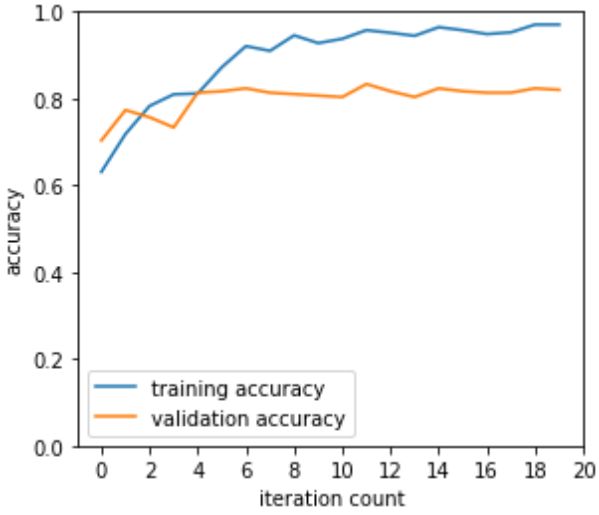

図 7　転移学習における訓練データとテストデータの精度

図 6 から，18 層に変えた畳み込みニューラルネットワークの精度は 6,7 割程度であり，図 5 とほとんど変わらない．学習にかかった時間は 12 分 4 秒である．このことから，ResNet の精度が 5 章の畳み込みニューラルネットワークより上がった理由は ResNet を使用したことよりも，入力データの大きさと層の多さによるものであると予想できる．

6.3　転移学習を用いた画像認識

ResNet を用いて転移学習を行う．事前学習には ImageNet という巨大なデータセットを用いる．文献 [4] より，python ライブラリの一つである pytorch を用いることによって，簡単に ResNet による転移学習を実装できる．その後，本研究で用意した訓練データを学習させる．訓練データを学習させる回数を 20 回としたときの訓練データ及び，テストデータに対する出力をラベルと比べたときの正答率（精度）の関係を図 7 に示す．

図 8　畳み込みニューラルネットワークによる画像認識

図 9　ResNet による画像認識

図 7 から学習精度は 8 割程度となり，5 章の畳み込みニューラルネットワークと比べ精度が向上したことが分かる．また，学習にかかった時間は 18 分 26 秒である．6.2 節の ResNet を事前学習なしで使ったときよりも高い精度を出すことができた．また，18 層よりも層の数が多い ResNet の転移学習も行ったが，精度はほとんど変わらなかった．

7　未知の画像に対しての画像認識

テスト画像から 10 枚の画像を取り出し，5 章で学習を行った畳み込みニューラルネットワークを用いて，画像の推定を行う．その結果を図 8 に示す．10 枚中 5 枚が正しく認識された．

同じ 10 枚の画像に対して，6.2 節で学習を行った ResNet を用いて，画像の推定を行う．その結果を図 9 に示す．図 9 より 10 枚中 7 枚が正しく認識された．

同じ 10 枚の画像に対して，6.3 節で学習を行った ResNet を用いて，画像の推定を行う．その結果を図 10 に示す．

図 10　転移学習を用いた ResNet による画像認識

図 10 より 10 枚中 10 枚が正しく認識された．

このように，曲がり角の画像と直線道路の画像を一定程度識別することができ，特に転移学習を行った ResNet によって，より多くの画像を正しく識別できることが分かった．

8　まとめ

独自のデータセットを作成し，曲がり角が近い画像と直線道路の画像の判別を行った．転移学習を行うことで少ないデータセットでも一定の精度を出すことができた．しかし，それでも精度は 8 割程度であった．その原因として，人や自動車といった障害物が存在することや，直線道路も曲がり角もどちらも道路であり，曲がり角の認識が難しい問題であることが考えられる．また，自分で撮影した画像の推定を行った結果，曲がり角が近くにあるかどうかを正しく識別できることが確認できた．

参考文献

[1] G. E. Hinton，S. Osindero，and Y.-W. Teh : A fast learning algorithm for deep belief nets. Neural Computation，18, 1527-1554 (2006)

[2] 岡谷 貴之： 深層学習 (機械学習プロフェッショナルシリーズ)， 講談社 (2015)

[3] K. He，X. Zhang，S. Ren, and J. Sun：Deep residual learning for image recognition. Proceedings of the IEEE Conference on Computer vision and Pattern Recognition，770-778 (2016)

[4] Transfer Learning for Computer Vision Tutorial — PyTorch Tutorials 1.4.0 documentation https://pytorch.org/tutorials/beginner/transfer_learning_tutorial.html