深層学習を用いた気象状況判断プログラムの試作

Prototyping of weather condition judgment program using deep learning

技術開発部 生産・加工科　鈴木健司 近野裕太 柿崎正貴 清野若菜 山田昌幸 尾形直秀

応募企業 旭日産業株式会社

降雪・積雪状況に応じて融雪装置の発停を制御するため、カメラ画像から降雪・積雪の有無を判断するための手法について検討した。ディープラーニングを用いて、降雪時や積雪時の画像を学習し、適切に判断できるかを評価した。

Key words: AI、ディープラーニング、NVIDIA DIGITS

1．緒言

応募企業の旭日産業株式会社は、降雪時や積雪時にカメラ画像から融雪装置の発停を制御するシステムを検討している。現在は、赤外線の画像を用いて昼夜を問わず状況を判断させているが、雨や水たまりなどを雪として誤認識してしまうことがある。しかし、安全のため閾値を下げ、誤認識している可能性が高くても雪があるものと判断している。そのため、雪がない状況でも運転をしていることがあり、無駄となっている。

そこで、本研究では、ディープラーニング を用いて降雪時、積雪時、降雪・積雪がない時の３パターンの画像を用いて学習を行い、カメラ画像の気象状況を判断するための手法について検討し、カメラ映像からリアルタイムに気象状況を判断するサンプルプログラムを作成した。

2．実験と結果

本研究では、分類する気象状況を、降雪あり・積雪あり、降雪なし・積雪あり、降雪なし・積雪なしに設定することとした。画像の取得には表１に示す IP カメラを用い、学習には表２に示す計算機を用いた。IP カメラは２箇所に設置し、定期的に映像を保存するプログラムを実行し、データを収集した。IP カメラには、光度センサーがついており、昼間はカラー映像に、夜間は赤外線によるモノクロ映像に自動的に切り替えて撮影を行った。

表１　データ収集に用いた IP カメラ

メーカー/型番	H.View / HV-400E1
画素数	400 万画素
フレームレート	20[fps]
視野確度	100[°]
その他	PoE 給電、赤外線、防水 IP66

表２　学習に用いた計算機

OS	Ubuntu18.04LTS
メモリ	16[GB]
CPU	Intel Core i7 7700K
GPU	NVIDIA GeForce 1080
GPGPU プラットフォーム	CUDA V10.0
学習モデル	AlexNet
実装	NVIDIA DIGITS

2．1．NVIDIA DIGITS について

ディープラーニングの学習には、GPU などを製造している半導体メーカーの NVIDIA 社が開発した DIGITS[1)] を用いた。DIGITS は、対話型のディープラーニングを学習するための Web アプリケーションである。図１に示すように、データの前処理やニューラルネットワークの設定、学習の推移、結果の可視化などを GUI 操作で行うことができる。また、学習済みのモデルを使って、推論を行うことも可能であるほか、複数の画像を一度に推論させることもできるため、学習の結果を効率的に評価できる。

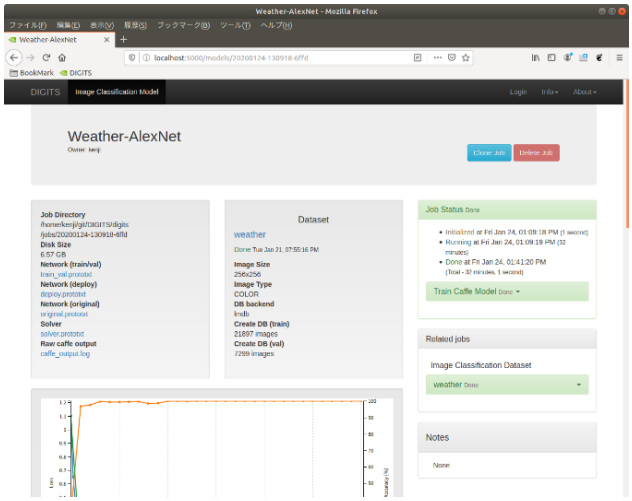


図１　DIGITS の操作画面

事業課題名「深層学習を用いた気象状況判断手法の試作」
[令和元年度][新製品・新技術開発促進事業]

２．２．学習データの準備

学習データは、屋外２箇所に設置した IP カメラの映像を定期的に動画ファイルに保存する Python プログラムを用いて収集した。学習に用いるデータは画像データであるので、以下の ffmpeg[2]コマンドにより動画ファイルを画像ファイルへ変換した。

```
$ ffmpeg -i input.mp4 -r 20 img%04d.png
```

このコマンドにより、input.mp4 の動画ファイルを img(通し番号).png の画像に変換することができる。r オプションは、動画１秒あたり何枚の画像に変換するかの値であり、上記の場合は、1 秒あたり 20 枚の画像を出力する。

このようにして画像を収集し、気象状態に合わせて、降雪あり・積雪あり、降雪なし・積雪あり、降雪なし・積雪なしの３つの状況に分類した。DIGITS では、ディレクトリ名が分類する正解ラベルに対応しており、画像データは気象状況に応じて、それぞれの名前のディレクトリへ保存する。画像枚数は、それぞれの状態につき約１万枚準備した。図２に３つの気象状況の画像データ例を示す。

図２　気象状況の例　（降雪あり・積雪あり（上）、降雪なし・積雪あり(中)、降雪なし・積雪なし(下)）

２．３．AlexNet による学習と判定結果

DIGITS を用いて学習を行い、学習モデルには AlexNet[3]を用いた。AlexNet は現在飛躍的に精度を高めている様々なニューラルネットワークの基礎的なモデルである。今回の分類数は３つと少数となっており、基礎的なモデルでも十分対応可能であると考え、採用した。

図３に学習結果のグラフを示す。epoch は学習の進捗、loss は損失関数の値となっており、epoch が進むにつれ loss の数値が小さくなっていき、学習がうまく進んでいることを示している。また、正答率 accuracy は学習が進むにつれ、数値が大きくなっており、正しく気象状況を判定できていることを示している。

train は学習データ、val は学習に用いていない検証用のデータで数値を算出したことを示している。したがって、学習済みのモデルが正しく判定できるかを判断するためには、val の数値が重要となってくる。例えば、loss(train)が小さくなっているのに対し、loss(val)が大きくなるような場合は、学習データでは正しく判定できるが、それ以外の未学習データは正しく判定できない過学習が起きていると考えられる。今回の学習結果のグラフは、loss(train)、loss(val)ともに値が小さくなっていることから、過学習は起こっておらず、正しく学習できていると考えられる。

図４は未学習のデータを学習済みモデルで判定させたときの結果である。画像の解像度は 1920x1080 であるが、学習モデルの入力サイズは 256x256 であるため、画像を縮小して入力する。気象状況は降雪あり・積雪ありであるが、正しく判定されていることが確認できた。

この他、DIGITS では、一括で画像判定をすることもできる。それぞれの気象状況について、10 枚ずつの未学習画像データ、合計 30 枚を判定させた。結果は、30 枚中 29 枚が正しく気象状況を判断していることが確認できた。

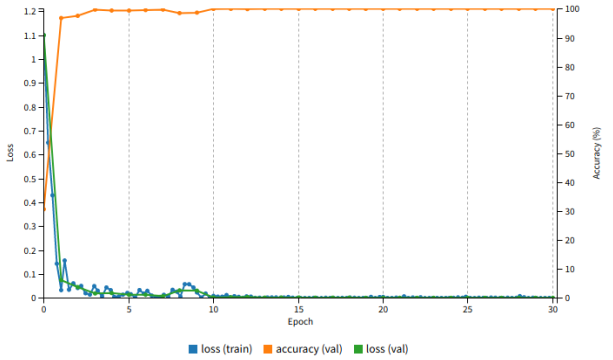


図３　正答率 accuracy と損失関数の値 loss の結果

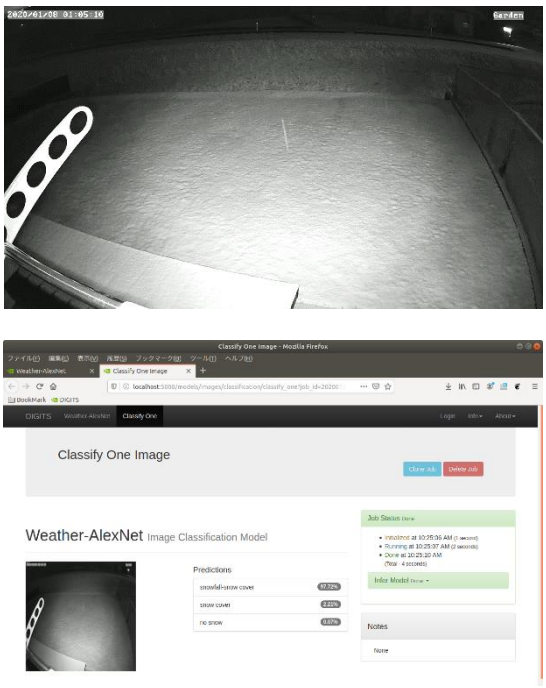


図４　降雪あり・積雪ありの未学習データ（上）とその認識結果（下）

２．４．リアルタイム気象判定のサンプルプログラム

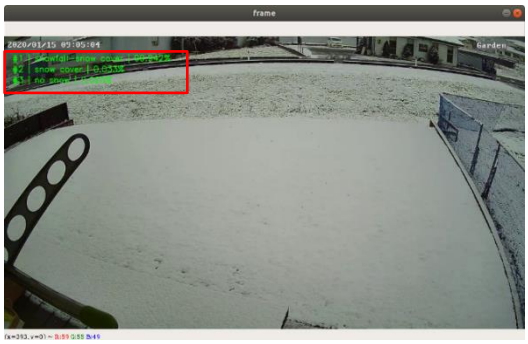
学習済みモデルを使ったアプリケーションの例として、IP カメラに映っている気象状況をリアルタイムに判定させるサンプルプログラムを作成した。

DIGITS で学習を行うと caffemodel ファイルが作成されるが、これが学習済みモデルになる。caffemodel ファイルは、python などの言語で、AI 技術を取り入れたプログラムにポーティングして使用できる。

caffemodel を用いて正しく判定させるためには、他にモデル構造を定義した deploy.prototxt ファイルと学習データの平均値である mean.binaryproto が必要になる。このため、mean.binaryproto はプログラム中で使用できるように npy 形式に変換して mean.npy としておく。

判定の流れは、まず deploy.prototxt を読み込んでネットワークモデルを構築し、caffemodel により重み付けをする。次に入力データから mean.npy の値を引き、そのデータを学習済みの重み付けをされたネットワークに入力すれば結果が出力される。

図５に作成した気象状況判定プログラムの実行画面を示す。IP カメラにアクセスし、左上に学習済みモデルで判定した気象状況を赤枠内に表示している。図５の画像は降雪あり・積雪ありのものであるが、約 99[%]の確率で正しく判断していることが確認できた。



#1 | snowfall-snow cover | 99.942%（降雪あり・積雪あり）
#2 | snow cover | 0.033%（降雪なし・積雪あり）
#3 | no snow | 0.025%（降雪なし・積雪なし）

図５　学習済みの caffemodel を用いたリアルタイム気象状況判定のサンプルプログラム

３．結言

本研究では、IP カメラの画像を使って、学習データセットを作成し、降雪あり・積雪あり、降雪なし・積雪あり、降雪なし・積雪なしの３つの状況に分類するディープラーニングの学習を行った。その結果、過学習になることなく、気象状況を正しく判断するモデルを作成することができた。

また、学習済みモデルを用いて IP カメラの映像からリアルタイムに気象状況を判定するサンプルプログラムを作成した。これにより、DIGITS による学習結果を用いて独自にアプリケーションを作成する手法を示すことができた。

これらの結果より、降雪などの気象状況に応じて機器を制御するような状況に対して、解決するための手法を示すことができた。

今後の課題は、今回学習したモデルが、他の場所に設置したカメラの映像でも正しく判定できるか検証を行い、必要があれば各場所に応じた学習の調整方法を検討することである。また、朝方や夕方の時間帯、晴天や雨天時の様々な状況に対応できるかも考えなければならない。

参考文献

１） nVIDIA. “CUDA-X AI”. NVIDIA DIGITS | NVIDIA Developer. https://developer.nvidia.com/digits. (参照 2020-01-31).

２） FFmpeg. “FFmpeg”. http://www.ffmpeg.org/. (参照 2020-01-31).

３） A Krizhevsky, I Sutskever, GE Hinton, "ImageNet Classification with Deep Convolutional Neural Networks." Advances in neural information processing systems. 2012. p. 1097-1105.