東京工業大学情報理工学院総合型選抜　活動実績報告書

氏名 (ふりがな):　**前田 恭輝 （ まえだ　たかき ）**

高等学校:　**愛知県立愛知工業高等学校（現：愛知県立愛知総合工科高等学校）**

(2018 年 3 月 ☒卒業・☐卒業見込み)

活動実績概要 (150 字程度):

会社での設備保全業務を通して、感じた課題が多くあり、IT を用いて解決できるようになりたいと思った。そこで、IT について資格を通して学び、実際にアプリを考えて、学習しながら実装を行った。今回は、今後進みたい分野である機械学習も実際に使ってみて、様々な知見を深めることができた。

活動実績の実施状況:

☒志願者が単独で行った

☐教師などからの指導を受けながら志願者が単独で行った

☐共同で行った

☐その他

報告書本体ページ数(表紙を含まない): **4** ページ

注意:

- 報告書本体を 4 ページ以内で作成し、この表紙と一緒に提出すること。
- 報告書本体の形式は自由とするが、文字の大きさは 10 ポイント以上にすること。また内容として活動実績の背景、具体的な内容、活動実績の実施状況の説明、参考にした資料の一覧などを必ず含むこと。
- 報告書本体に、活動実績を志願者が単独で行ったか否か、共同で行った場合は自身の役割、指導を受けた場合はどの部分に対する指導か等の説明を書くこと。
- 報告書本体には氏名、学校名はどうしても必要な場合を除いて書かないこと。

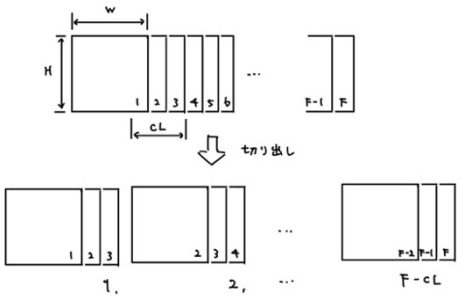
機械学習を用いた動画像分析による
異常検知アプリケーションの実装と改良

１．背景と活動概要

私は、トヨタの製鉄会社にて主に大型クレーンの保全に携わってきて、多くの課題を感じていた。そのうちの一つに、点検時の巻き込まれや転落等の安全性と点検項目の多さがある。そんな中、AI の発展で自動化がより加速し労働者が減っていく一方で、自動化していく機械は変わらず人がメンテナンスをしていく必要がある。それどころか、自動化が進み、安全、品質基準が上がるにつれて、点検項目は増えていく。実際に働いていた現場では、人手不足も深刻化していて、点検が十分に行えていない現状があった。

そこで、私は保全業務を支援できるようなシステムやアプリケーションの提案をしたいと考え、IT について資格を通して学び、近年進化する機械学習を活用できないかと思い至った。中でも、点検頻度が高く、巻き込まれ等のリスクがあり、トラブルの起きやすい駆動部の点検にフォーカスした保全の支援を考えた。具体的には、動画像を機械学習手法を活用して分析し、正常時との比較を行って異常を検知できないかと考えた。

また、各フレームを特定長に切り出したものを評価する“クリップ処理”を行うことで、既存の機械学習を用いた画像分析に加え、正常時より速い、遅い、異なる“動作”を検知できるのではないかと考えた。そして、Python と機械学習を学習しながら、アプリの実装を試みた。



クリップ処理のイメージ

２．活動内容

2.0 活動記録について

今回、スペースの関係上掲載できなかった、ソースコード、テストに使用した映像、測定記録等は Google Drive に公開しており、参考文献欄に掲載している。

2.1 環境構築

動作環境：MacBook Pro 13inc
使用言語：python
使用ライブラリ：Numpy（演算処理）、
OpenCV（動画像処理）、Matplotlib（グラフ描画）、
scikit-learn、Keras、TensorFlow（機械学習）
上記の環境で行った。

2.2 プロトタイプの設計と実装

初めに、異常が検知できるかを確認するために、以下のような簡単なモデルを考えて実装してみた。

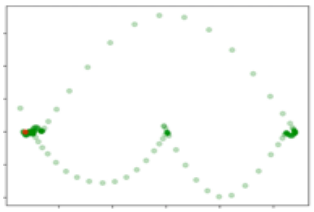
① 動画読込
② 前処理
③ 次元削減
④ クリップ処理
⑤ 次元削減
⑥ k 近傍法で分類

まず、動画像を解析する上でそのまま扱うと莫大なデータ量になってしまうので、次元削減と言われる手法を用いる。これは、データのもつ特徴をより低次元なデータで表す手法で、今回は主成分分析[1]（以下 PCA）と呼ばれる次元削減手法を用いる。これにより、大幅に計算コストを削減することができる。また、この次元削減を行う前に、グレースケール化、低解像度化、二次元配列化、標準化といった前処理を行う。

次に、クリップ処理を行う。この、連続するフレームに対し、更に 2 次元へ次元削減する事で、時間変化の情

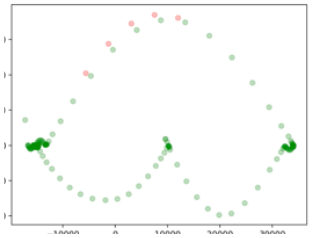
報が含まれ、正常な画像の羅列であっても、速度異常や異常動作が分布として現れると考えたからである。

実際にプロットをしてみると、図のように分布した。(検証データは物体を画面内の３点で手動移動させる、簡単なものを用いた。)

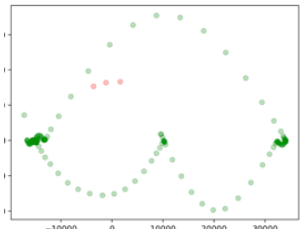


正常データの分布

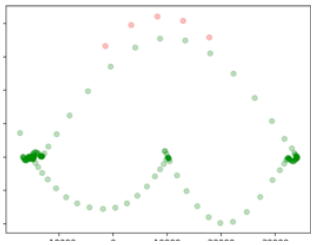
ここで、正常動作と比較し、遅い、速い、異なる動作をしたところ、確かにデータが正常データに対し分離できていることが確認できた。(緑点が正常なデータ、赤点がリアルタイムでプロットしているデータ)



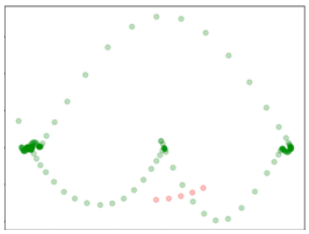
正常動作



遅い動作



速い動作

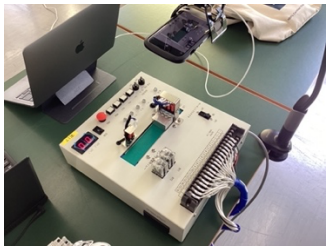


異なる動作

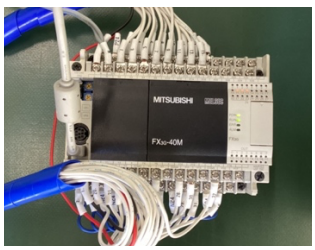
最後に、異常を分類するために k 近傍法[2]と呼ばれる分類手法を用いる。この時、正常データに沿わないものは全て異常とみなす事から、正常データを避ける形でネガティブデータを生成し、それぞれにラベリングをして分類を行った。

2.4 テスト

シーケンサで制御できる装置を用いて、実際の機械のような精密な動きで異常を検知できるかテストした。
(機材協力：愛知総合工科高等学校)



撮影風景



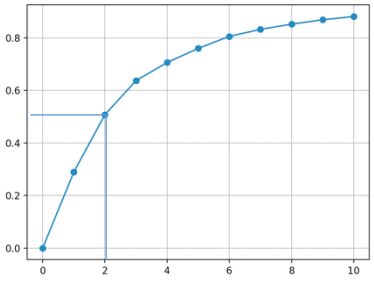
シーケンサ

すると、明らかな速度異常や動作異常でないと検知できないことがわかった。また、ランプの異常点灯や指の写り込み等の異常が検知できないことがわかった。

加えて、正常データでピントズレが起きた際、分布が大幅に変化して、正しく異常検知できなかった。

2.5 原因調査

まず、次元削減による情報ロスが考えられたので、削減後の次元数に応じた寄与度（表現できている元の情報量）を測定した所、図のような結果であった。(縦軸は寄与度、横軸は次元数)



これによると、２次元への次元削減では半分程度の情報量しか保持できておらず、ランプ等の小さな画素変化が表現しきれていないことが考えられた。

加えて、分散が最大になるように主成分を合成することから、学習データにおける変化をよく表現できる反面、学習データに含まれない変化はうまく表現することができず、特に今回のような低次元の場合、異常画像であっても、正常画像に近い表現をしてしまって、手の写り込みといった異常が検知できなかったと考えられる。

また、正常データに正常時から大幅にずれた(ピントズレ等)ノイズが含まれると、そのデータを表現できるよう主成分を合成してしまい、異常データの検知ができなくなっていたと考えられる。

2.6 対策

まず、削減後の次元数を寄与度が 90%以上となるように、次元数を設定する。その際、距離を用いた分類手法である現手法は、球面集中現象[4]と呼ばれる、次元数が増えるにつれて球の体積が表面に集中していく現象によってうまく分類できなくなることがわかった。その為、これとは異なる手法を用いる必要がある。加えて、現手法のような教師あり学習は異常データが確保できているときに有効な手法であり、今回の方法は無理やり教師あり学習の手法を適用している形であることがわかった。

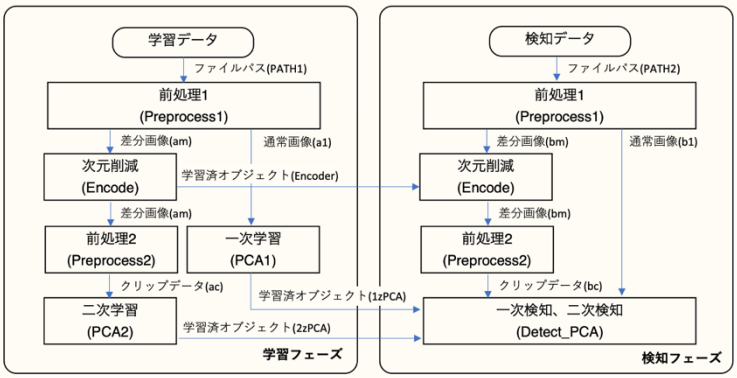
次に、PCA では学習していないデータを次元削減すると、正しく復元できない。この事を利用し、復元前後で差分を取ることで異常度を測る教師なし学習手法があ

り、その手法であれば、ランプや指の写り込み等の異常検知も期待できることがわかった。

以上のことから、PCA を用いた教師なし学習による異常検知モデルを考えることにした。

2.7 アプリケーション改良案と実装

図のようなモデルを考えた。ここで、削減する次元数は寄与度が 90%以上になるように次元数を設定した。（括弧内はソースコードにおけるオブジェクト名）



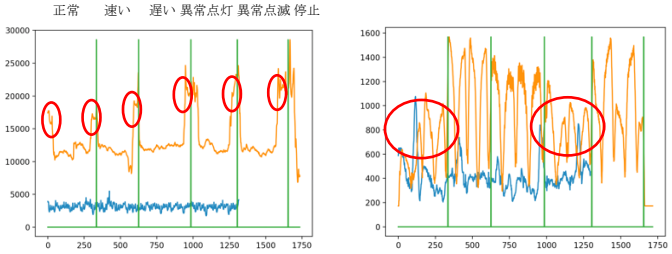
まず、異常検知を二段階に分ける事を考えた。これは、異常画像は 1 フレームのみ、異常動作は複数フレームの変化のある部分のみを評価したいからである。（以下、前者を一次検知、後者を二次検知と呼ぶ。）よって、前処理 1 では 2.2 と同様に処理したものと、変化のある部分のみを取り出す為に、フレーム前後の差分抽出と二値化を行なったものとで分けて出力する。

次に、一次検知とは別に、次元削減の為に後者のデータを PCA で学習し、次元削減を行う。その後、そのデータを前処理 2 でクリップ処理、二次元配列化、標準化を行う。

学習フェーズでは、上記の流れで処理したデータを学習し、学習済オブジェクトを出力する。

検知フェーズでは、学習済オブジェクトを使って、評価する動画像を同様の流れで次元削減し復元する。これと、次元削減前のデータの差分をとり、全画素の絶対値の総和を各フレームの異常度とする事で、異常の検知を試みた。

すると、正常データ（青）と異常データ（橙）は以下のようになった。ここで、縦軸は異常度、横軸がフレーム番号となっており、異常データの異常モードは左から、正常、高速、低速、ランプ点灯異常（異常画像）、ランプ点滅異常（異常動作）となっていて、緑線によってモードの区切りを示している。また、異常画像と異常動作は異常な画素が小さく（全体の 1.96%）、速度の異常度は小さい（正常時の+6.3 %と-15.0%の速度）ものを使った。



一次異常検知　　二次異常検知

想定では、一次では異常画像のみ、二次では異常画像以外の異常モードで異常を示すはずであるが、この結果によると全体的に異常度を高く示しており、一次では異常画像に加えモードの切り替わりで異常を示していた。

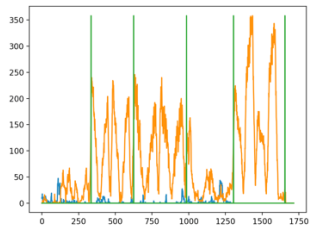
2.8 原因調査

ここで、異常度を示すデータを画像に復元し確認してみた。すると、装置左上の黒色のスイッチ部が常に異常を示していた。また、制御装置の操作時、異常度が大きく変化していることがわかった。このことから、環境光の変化を検知し全体的に異常度を高く示していたと考えられる。

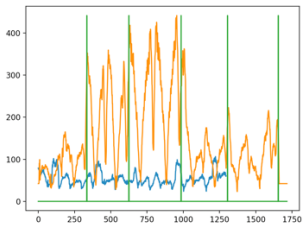
2.9 対策

環境光の変化なので、二次検知においての 1 画素あたりの異常度自体は大きくなく、占める画素が広範囲であるために、総和で算出した異常度に大きく影響していると考えられる。その為、異常画素の異常度は大きいものとして考え、異常度が閾値以下のものを無視して算出することで、実際の異常のみを抽出できると思われる。

以上より、異常度のフィルタリングを行った結果以下のような結果を得ることができ、確かに異常を検知することができた。また、次元削減の次元数をあえて落とす事で、2.5 で見られた特徴から、環境光変化の影響を低減し、速度異常を集中して検知できないか試みた。すると、概ね狙い通りの結果が出たが、異常動作の異常度は下がってしまった。これは、次元数を落とした影響で異常画素の小さいランプ異常の情報が落とされてしまったことが考えられる。



フィルタリング後



次元数低下後

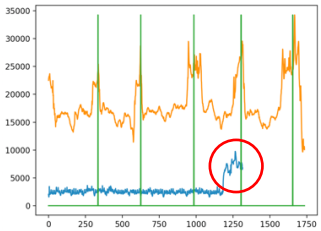
2.11 プロトタイプの改良＋α

ここで、PCA 以外の機械学習手法を用いたモデルを考えて、実装、比較してみた。今回は、自己符号化器[3]（以下 AE）を実装してみた。理由として、ノイズに強く、PCA の時に大きく影響していた一瞬の大きい画素変化（ピントずれ等）に対し強くなることが期待できることと、ニューラルネットワークによる機械学習手法で、知見を深めることができると考えたからである。

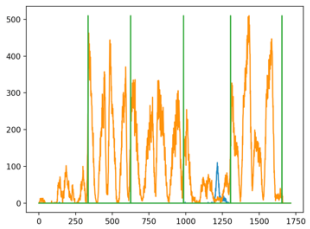
アプリの構造は 2.7 と同様で、PCA が AE に置き換わったものである。AE のネットワークモデルは以下のように実装した。ここで、N1 は低解像度化後の画素数 N2 は削減後次元数×クリップ長を動的に代入する。

ニューロン数	入力層	N1	N2
	中間層	N1/16	N2/4
	出力層	N1	N2
活性化関数	中間層	relu	relu
	出力層	linear	linear
損失関数		二乗誤差	二乗誤差
最適化アルゴリズム		adam	adam
バッチサイズ		256	256
エポック数		50	50

異常度を算出してみた所、次のようになった



一次異常検知



二次異常検知

３．結果

比較の結果、一次検知において、停止直前のノイズに強くなっている事が見受けられたが、総合的には今回のデータで性能の向上はあまり見られなかった。使用映像の情報と各所要時間を下に記す。

使用映像	解像度	低解像度化	時間	FPS	総フレーム
正常データ	1920×1080	160×90	2 m12s	10	1323
異常データ	1920×1080	160×90	2 m54s	10	1737

所要時間[s]	前処理1	前処理2	一次学習	二次学習	異常検知
PCA	31.0	26.0	4.8	4.6	4.5
AE	31.0	26.0	111.5	6.5	12.1

４．まとめ

今回の活動は、課題解決の為にアプリケーション案を考案し、実際に機能するかプロトタイプを通して確認した後、異なる機械学習手法を用いて改良を試みた。

結果として異常は検知できたものの、環境変化への対策は不安が残る。また、AE のモデルに関しても簡単なものを用いており、畳み込み層の追加や中間層を増やす等まだまだ改良の余地はあると思われる。これに関し、今後の学習でニューラルネットワークに関する理解を深め、最適なモデルを考えていきたい。

また、このアプリケーションを実用段階にするには、異常を知らせるアラーム機能、異常が検知されたクリップ映像を書き出す機能、異常領域の示唆、また、それが正常であったときの追加学習機能やアプリケーションの UI 実装、IoT 対応等の課題が考えられる。

加えて、今回はフーリエ解析等の知識不足によって断念したが、音声分析による異常検知等もぜひ実装したいと考えているので、今後の展望としてこれらの実装へ向けより一層学習を進めていきたいと考えている。

最後に、この活動を通し、python プログラミングや各機械学習手法について知見を深めることが出来た。特に python プログラミングに関し全くの無知であった活動当初と比べとても成長できたと感じているので、今後の学習や活動にもこれらの経験をぜひ活かしていきたいと思った。

なお、本活動は愛知総合工科高等学校にテスト用装置の協力の元、単独で行なった。

参考文献

[1]https://ja.wikipedia.org/wiki/主成分分析

[2]https://ja.wikipedia.org/wiki/K 近傍法

[3]https://jp.mathworks.com/discovery/autoencoder.html

[4]https://ibisforest.org/index.php?球面集中現象

[6]https://omathin.com/keras-nn-basic/

[7]https://www.hellocybernetics.tech/entry/2016/11/21/041607

[8]その他 python コーディングについての多数記事

[9]活動記録詳細

https://drive.google.com/drive/folders/16JPDiXf0S2EtVXGlJvYBTYnmu7B3Dafd?usp=share_link