# Dokumentacja wstępna ASO

*Reidentyfikacja plam zabrudzeń*

Jan Kwiatkowski, Alicja Turowska, Konrad Gieleta
Kwiecień 2023

## 1. Cel i analiza tematu

Celem zadania jest stworzenie modelu służącego do reidentyfikacji plam zabrudzeń. Reidentyfikacja jest niezbędna w celu śledzenia poruszających się obiektów w sekwencji klatek. Różni się ona przede wszystkim od klasyfikacji tym, że podczas gdy celem klasyfikacji jest ustalenie identyfikatora/tożsamości obiektu, to w problemie reidentyfikacji sam identyfikator nie jest ważny, a jedynie znalezienie zdjęcia/klatki, gdzie jest ten sam obiekt (porównanie czy na dwóch zdjęciach/klatkach jest ten sam obiekt). Najczęściej za pomocą klasyfikacji jednego ze zdjęć/klatki ustalany jest identyfikator obiektu, który jest "propagowany" za pomocą reidentyfikacji na kolejne zdjęcia/klatki. Problem reidentyfikacji sprowadza się zatem do znalezienia dla zdjęcia obiektu "najbliższego" mu innego zdjęcia.

Sam problem reidentyfikacji plam zabrudzeń nie jest zbyt rozpowszechniony, tzn. nie znaleziono artykułów poruszających to zagadnienie. Problem reidentyfikacji jest najczęściej poruszany w kontekście reidentyfikacji osób. Z kolei zagadnienia związane z plamami zabrudzeń są najczęściej problemami klasyfikacji lub detekcji. Powoduje to brak możliwości porównania uzyskanych wyników i utrudnia oszacowanie przewidywanych rezulatów.

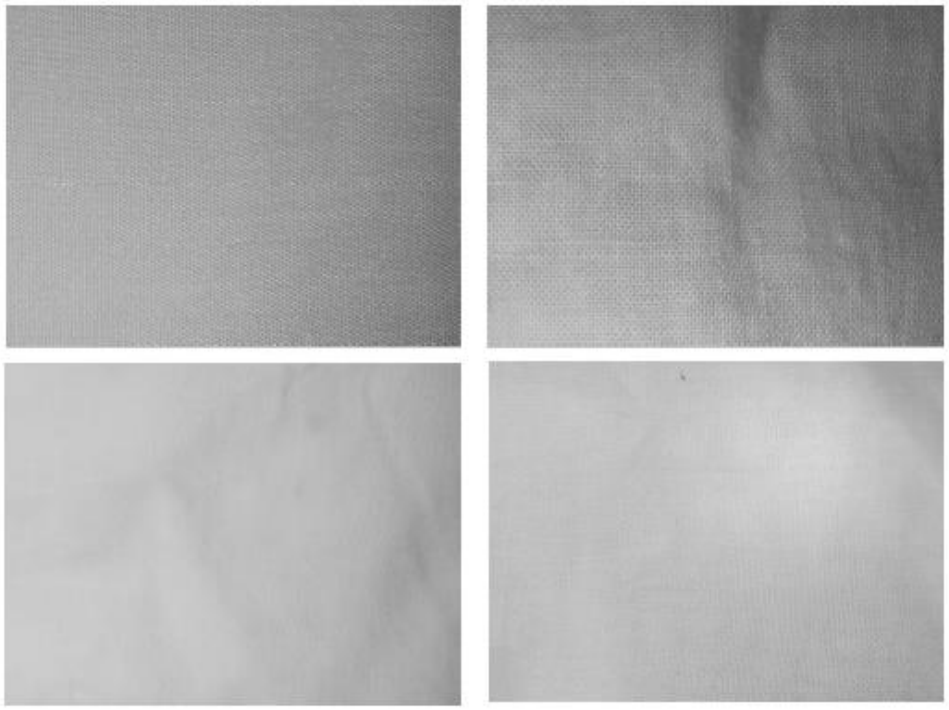
Jak wspomniano wyżej, mimo, że problem reidentyfikacji plam zabrudzeń nie jest zbyt popularny, sama reidentyfikacja jest często poruszana w kontekście reidentyfikacji osób, gdzie do najczęściej wykorzystywanych metod należą konwolucyjne sieci neuronowe (CNN). Zaletą tych sieci jest automatyczna ekstrakcja cech obrazu, co odróżnia je od innych algorytmów rozpoznawania obrazu, które wymagają wiedzy eksperckiej z zakresu inżynierii cech. Głębokie sieci neuronowe samodzielnie potrafią wykrywać istotne cechy z badanego obrazu, które mają kluczowe znaczenie dla poprawnej klasyfikacji czy reidentyfikacji. Dodatkowo sieci konwolucyjne są w stanie uczyć się na podstawie dużych zbiorów danych, dzięki czemu mogą rozpoznawać obiekty i cechy, których nie da się łatwo opisać w sposób analityczny. Dzięki temu, sieci te są w stanie wykonywać zadania, których nie da się łatwo lub w ogóle nie da się wykonać za pomocą prostych metod przetwarzania obrazu.

Sieci CNN składają się z dwóch części. Pierwsza część, składająca się z warstw konwolucyjnych (wykorzystujących sploty) oraz warstw aktywacyjnych i redukujących rozmiar, umożliwia wspomnianą ekstrakcję cech. Druga część stanowi klasyczny klasyfikator neuronowy. W problemie reidentyfikacji zamiast prawdopodobieństw przynależności do danej klasy otrzymujemy na wyjściu wektor osadzenia. Modele wykorzystywane do reidentyfikacji często wyróżniają się zastosowaniem funkcji kosztu triplet loss, polegającej na dostarczeniu na raz wektorów osadzenia trzech zdjęć: zdjęcia wyjściowego, zdjęcia mu "bliskiego" oraz zdjęcia "dalekiego". Użycie triplet loss jest jedną z najbardziej popularnych sposobów rozwiązania

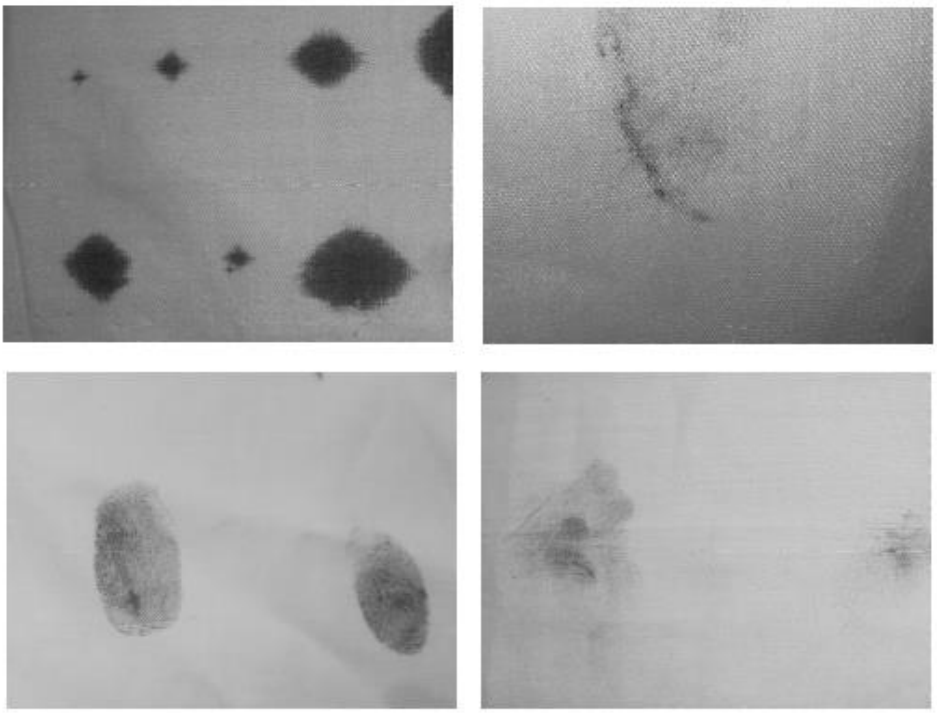
problemu reidentyfikacji, nie jest jednak jedyną możliwością. W znalezionym artykule [3] autorzy przedstawiają sieć syjamską łączącą w celu reidentyfikacji dwa modele: identyfikacyjny i weryfikacyjny. Sieć ta dla pary obrazów szkoleniowych przewiduje tożsamość (model identyfikacyjny), oraz to czy należą do tej samej tożsamości, czyli kategorii (model weryfikacyjny).

## 1.1. Publiczne zbiory

Wyszukano 2 interesujące publiczne zbiory związane z plamami zabrudzeń. Jeden z nich - TILDA nie dotyczy jednak samych plam, co wszelakich defektów tkanin. Drugi zbiór - Fabric Stain Dataset, powstał z myślą o problemie klasyfikacji. Możliwe jest jednak wykorzystanie go również w celach reidentyfikacji. Zawiera on 466 zdjęć podzielonych na 2 kategorie: 68 zdjęć tkanin bez defektów (14.59% zbioru) oraz 398 zdjęć (85.41% zbioru) tkanin z plamami atramentu, brudu, oraz oleju (bez podziału na podkategorie). Zdjęcia są w dwóch rozdzielczościach: 1488x1984 (305 zdjęć) oraz 1984x1488 (161 zdjęć). Zbiór nie został podzielony na dane treningowe, walidacyjne i testowe. Poniżej przedstawiono przykładowe zdjęcia tkanin bez defektów (rysunek 1) oraz tkanin z plamami (rysunek 2).



Rysunek 1 - przykładowe zdjęcia bez defektów



Rysunek 2 - przykładowe zdjęcia tkanin z plamami

Analizę zbioru Fabric Stain Dataset, którego podsumowanie przedstawiono powyżej, umieszczono w notatniku “Analiza_zbioru.ipynb”. Przedstawia on informacje o liczności poszczególnych klas oraz rozmiaru zdjęć. Dodatkowo przedstawia wszystkie zdjęcia (z podziałem na klasy) oraz historgramy kilku zdjęć z poszczególnych kategorii.

## 2. Założenia projektu

Podczas projektu zostanie wykorzystany publiczny zbiór Fabric Stain Dataset omówiony w rozdziale 1.1. oraz dodatkowo stworzony na potrzeby tego projektu zbiór plam zabrudzeń. W ramach zadania zostaną wytrenowane 3 modele sieci CNN o podanych architekturach:

- Resnet-50 z wykorzystaniem funkcji kosztu triplet loss,
- VGG16
- EfficientNet B7

Modele te będą wytrenowane dla problemu reidentyfikacji, rozumianego jako wyznaczenie “najbliższego” zdjęcia dla podanego. Projekt zostanie zrealizowany w języku python z wykorzystaniem biblioteki PyTorch oraz PyTorch Lightning.

## 3. Etapy

- Stworzenie własnego zbioru,
- Analiza i przygotowanie zbiorów (podział na zbiór treningowy, testowy i walidacyjny, zmiana rozmiarów zdjęć do jednej rozdzielczości, przygotowanie trójek - ze wzg. na funkcję kosztu triplet loss),
- Implementacja i wytrenowanie modeli,
- Dobór hiperparametrów,
- Ostateczne wytrenowanie i przetestowanie modeli,

## 4. Modele

### *Resnet-50*

Sieć resnet-50 jest siecią z rodziny sieci resnet (Residual Neural Network). Jest to rodzina głębokich sieci konwolucyjnych rozwiązująca problem zanikania gradientów. Sieci te składają się z bloków rezydualnych, zawierających 2 lub więcej warstwy konwolucyjne oraz funkcję aktywacji, po której do danych wyjściowych bloku dodawane są oryginalne dane wejściowe. Sieć renset-50 w każdym z 16 bloków rezydualnych zawiera po 3 warstwy konwolucyjne, gdzie pierwsza i ostatnia w bloku wykorzystuje maskę 1x1 co tworzy charakterystyczne "wąskie gardło". Dodatkowo przed blokami sieć zawiera jedną warstwę konwolucyjną oraz warstwę MaxPool, co łącznie daje 50 warstw. Rysunek 3 przedstawia architekturę sieci resnet, w tym Resnet-50.

| layer name | output size | 18-layer | 34-layer | 50-layer | 101-layer | 152-layer |
|---|---|---|---|---|---|---|
| conv1 | 112×112 | 7×7, 64, stride 2 | | | | |
| conv2_x | 56×56 | 3×3 max pool, stride 2 | | | | |
| | | $\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$ | $\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$ | $\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$ | $\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$ | $\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$ |
| conv3_x | 28×28 | $\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$ | $\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$ | $\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$ | $\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$ | $\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$ |
| conv4_x | 14×14 | $\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$ | $\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$ | $\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$ | $\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$ | $\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$ |
| conv5_x | 7×7 | $\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$ | $\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$ | $\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$ | $\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$ | $\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$ |
| | 1×1 | average pool, 1000-d fc, softmax | | | | |
| FLOPs | | $1.8\times10^9$ | $3.6\times10^9$ | $3.8\times10^9$ | $7.6\times10^9$ | $11.3\times10^9$ |

Rysunek 3 – architektura sieci rodziny resnet

Sieć ta jest popularną siecią wykorzystywaną do problemu reidentyfikacji. W artykule [7] sieć resnet-50 posłużyła za podstawę modelu służącego do reidentyfikacji samochodów, zaś w artykule [8] opisano Batch DropBlock - dwugałęziowa sieć, gdzie jedną z gałęzi jest konwencjonalny resnet-50.

### *VGG16*

Sieć VGG16 należy do rodziny sieci VGGNet (architektura zdefiniowana przez Visual Geometry Group) i jest zazwyczaj stosowana do klasyfikacji obrazów. Wybrana architektura

zakłada wykorzystanie sieci złożonej z: 13 warstw konwolucyjnych (+maxpooling po każdym bloku) oraz 3 warstw wpełni połączonych. Dzięki temu jest w stanie efektywnie ekstrahować z obrazu cechy o różnym stopniu złożoności. Podstawowym założeniem tej architektury jest wykorzystanie we wszystkich warstwach niewielkich filtrów o rozmiarze 3x3. Po każdym bloku ilość tych filtrów jest zwiększana dwukrotnie, a wielkość map zmniejszana o połowę. Zastosowanie dużej liczby niewielkich filtrów przy danym polu recepcyjnym zwiększa głębokość sieci w porównaniu z pojedyńczym filtrem o większym rozmiarze (zwiększona liczba filtrów zwiększa liczbę nieliniowości). Po każdej warstwie agregującej maleje rozmiar przestrzenny (szerokość × wysokość) następnej warstwy, ale podwaja się jej głębokość (liczba filtrów).
Rysunek 4 przedstawia architekturę sieci VGGNet (w tym VGG16).

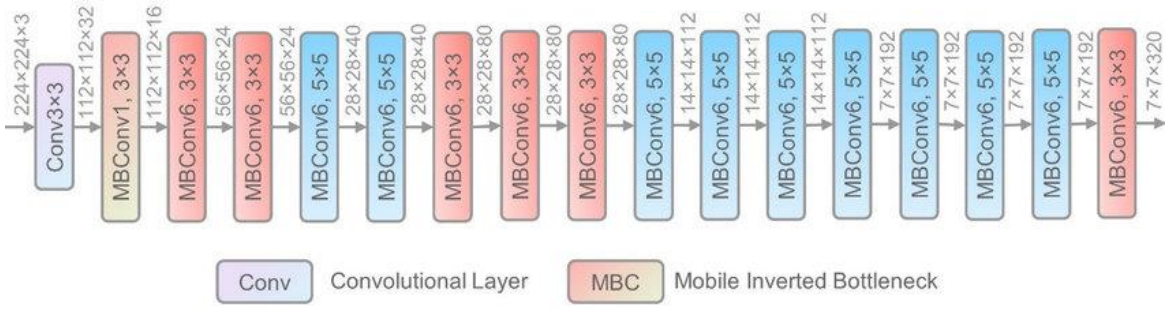
| ConvNet Configuration | | | | | |
|---|---|---|---|---|---|
| A | A-LRN | B | C | D | E |
| 11 weight layers | 11 weight layers | 13 weight layers | 16 weight layers | 16 weight layers | 19 weight layers |
| input (224 × 224 RGB image) | | | | | |
| conv3-64 | conv3-64<br>**LRN** | conv3-64<br>**conv3-64** | conv3-64<br>conv3-64 | conv3-64<br>conv3-64 | conv3-64<br>conv3-64 |
| maxpool | | | | | |
| conv3-128 | conv3-128 | conv3-128<br>**conv3-128** | conv3-128<br>conv3-128 | conv3-128<br>conv3-128 | conv3-128<br>conv3-128 |
| maxpool | | | | | |
| conv3-256<br>conv3-256 | conv3-256<br>conv3-256 | conv3-256<br>conv3-256 | conv3-256<br>conv3-256<br>**conv1-256** | conv3-256<br>conv3-256<br>**conv3-256** | conv3-256<br>conv3-256<br>conv3-256<br>**conv3-256** |
| maxpool | | | | | |
| conv3-512<br>conv3-512 | conv3-512<br>conv3-512 | conv3-512<br>conv3-512 | conv3-512<br>conv3-512<br>**conv1-512** | conv3-512<br>conv3-512<br>**conv3-512** | conv3-512<br>conv3-512<br>conv3-512<br>**conv3-512** |
| maxpool | | | | | |
| conv3-512<br>conv3-512 | conv3-512<br>conv3-512 | conv3-512<br>conv3-512 | conv3-512<br>conv3-512<br>**conv1-512** | conv3-512<br>conv3-512<br>**conv3-512** | conv3-512<br>conv3-512<br>conv3-512<br>**conv3-512** |
| maxpool | | | | | |
| FC-4096 | | | | | |
| FC-4096 | | | | | |
| FC-1000 | | | | | |
| soft-max | | | | | |

Rysunek 4 – architektura sieci VGGNet [4].

### *EfficientNet B7 [5]*

EfficientNet B7 należy do rodziny sieci EfficientNet, która stanowi grupę skalowalnych, wydajnych konwolucyjnych sieci neuronowych opartych na technice Compound Scaling, polegającej na równoczesnym skalowaniu głębokości, szerokości i rozdzielczości sieci. EfficientNet-B7, jako jeden z najbardziej zaawansowanych modeli w tej serii, składa się z wielu bloków MBConv, zawierających konwolucje 1x1 do rozszerzenia i kompresji kanałów, konwolucje z przestrzeni grupowej (DWConv), a także mechanizm uwagi Squeeze and Excitation (SE). W niektórych blokach MBConv wykorzystuje się także połączenia typu skip,

które łączą wyjście z wcześniejszą warstwą. Sieć EfficientNet-B7 stosuje także warstwy normalizacji wsadowej (Batch Normalization) oraz funkcję aktywacji Swish zamiast standardowej ReLU. Na początku sieci znajduje się warstwa konwolucyjna 3x3 z 64 filtrami, a na końcu sieci używa się Global Average Pooling (GAP) do agregacji informacji na przestrzeń pojedynczego wektora, a następnie warstwa gęsta (Dense) przekształca wektor cech w wartości prawdopodobieństw dla poszczególnych klas. Rysunek 5 przedstawia architekturę sieci z rodziny EfficientNet.




Rysunek 5 – architektura sieci EfficientNet [6]

# Bibliografia:


[1] https://www.getklap.com/blog/what-is-reidentification,

[2] https://datagen.tech/guides/computer-vision/resnet-50/,

[3] ZHENG, Zhedong; ZHENG, Liang; YANG, Yi. A discriminatively learned cnn embedding for person reidentification. *ACM transactions on multimedia computing, communications, and applications (TOMM)*, 2017, 14.1: 1-20.

[4] https://cv-tricks.com/cnn/understand-resnet-alexnet-vgg-inception/

[5] Tan, M., & Le, Q. V. (2019). EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. Proceedings of the 36th International Conference on Machine Learning (ICML 2019), 97, 6105–6114.

[6] https://www.researchgate.net/figure/The-network-architecture-of-EfficientNet-It-can-output-a-feature-map-with-deep-semantic_fig3_349299852/

[7] Peng Wang, Bingliang Jiao, Lu Yang, Yifei Yang, Shizhou Zhang, Wei Wei, Yanning Zhang, *Vehicle Re-Identification in Aerial Imagery: Dataset and Approach*; Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 460-469

[8] Zuozhuo Dai, Mingqiang Chen, Xiaodong Gu, Siyu Zhu, Ping Tan; *Batch DropBlock Network for Person Re-Identification and Beyond*; Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 3691-3701