Reidentyfikacja plam zabrudzeń

Jan Kwiatkowski, Alicja Turowska, Konrad Gieleta
Czerwiec 2023

1. Wstęp

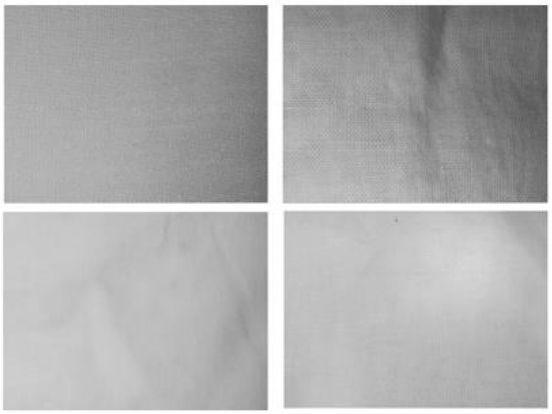
Przedstawiamy badania nad problemem reidentyfikacji plam zabrudzeń przy wykorzystaniu trzech modeli sieci konwolucyjnych: Resnet-50, VGG16 oraz EfficientNet. W analizowanych przypadkach badanie reidentyfikacji plam polega na znalezieniu plamy podobnej do wskazanej. W ogólności reidentyfikacja jest niezbędna w celu śledzenia poruszających się obiektów w sekwencji klatek. W przeciwieństwie do tradycyjnej klasyfikacji, w reidentyfikacji celem jest znalezienie obiektu na kolejnych zdjęciach, a nie ustalenie jego identyfikatora. W tym tekście przedstawiamy analizę tematu, opiszemy modele sieci oraz eksperymenty przeprowadzone na publicznym zbiorze danych plam zabrudzeń.

Reidentyfikacja plam zabrudzeń nie jest powszechnym zagadnieniem badawczym, co oznacza, że ma możliwości porównania uzyskanych wyników z innymi. Zazwyczaj problem reidentyfikacji jest omawiany w kontekście reidentyfikacji osób, podczas gdy zagadnienia dotyczące plam zabrudzeń są głównie związane z klasyfikacją lub detekcją.

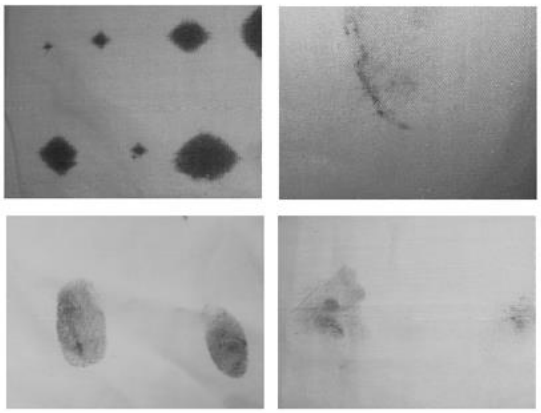
Zaletą wybranych sieci konwolucyjnych jest automatyczna ekstrakcja cech obrazu, co odróżnia je od innych algorytmów rozpoznawania obrazu, które wymagają wiedzy eksperckiej z zakresu inżynierii cech. Ma to kluczowe znaczenie dla poprawnej reidentyfikacji.

2. Zbiory danych

Zbiór - Fabric Stain Dataset, (powstał z myślą o problemie klasyfikacji). Możliwe jest 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). 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.



Rysunek 1 - przykładowe zdjęcia bez defektów



Rysunek 2 - przykładowe zdjęcia tkanin z plamami

Zbiór – własny, złożony jest ze zdjęć znalezionych w internecie. Zawiera 101 zdjęć z różnymi plamami oraz 33 zdjęć bez plam.

3. Wybrane architektury

Sieci CNN są złożone z dwóch głównych części. Pierwsza część składa się z warstw konwolucyjnych, które wykorzystują sploty, oraz warstw aktywacyjnych i redukujących rozmiar, co umożliwia ekstrakcję cech. Druga część sieci to klasyczny klasyfikator neuronowy.

W przypadku reidentyfikacji, zamiast otrzymywać prawdopodobieństwa przynależności do określonej klasy, otrzymujemy na wyjściu wektor osadzenia. Modele wykorzystywane do reidentyfikacji często stosują funkcję kosztu triplet loss. Polega to na dostarczeniu na raz wektorów osadzenia dla trzech zdjęć: zdjęcia wyjściowego, zdjęcia "bliskiego" (o podobnej tożsamości) i zdjęcia "dalekiego" (o innej tożsamości).

Chociaż użycie triplet loss jest jednym z najbardziej popularnych sposobów rozwiązania problemu reidentyfikacji, istnieją również inne podejścia. W artykule [3], którego odnaleziono, autorzy przedstawiają sieć syjamską, która łączy dwa modele - model identyfikacyjny i model weryfikacyjny - w celu reidentyfikacji. Ta sieć przewiduje tożsamość pary obrazów szkoleniowych (za pomocą modelu identyfikacyjnego) oraz określa, czy należą one do tej samej kategorii (za pomocą modelu weryfikacyjnego).

Wymienione poniżej modele zostały wytrenowane dla problemu reidentyfikacji, rozumianego jako wyznaczenie “najbliższego” zdjęcia dla podanego.

- **ResNet-50** [7] jest częścią rodziny sieci ResNet (Residual Neural Network). Jest to głęboka sieć konwolucyjna, która rozwiązuje problem zanikania gradientów. Składa się z bloków rezydualnych, które zawierają 2 lub więcej warstw konwolucyjnych oraz funkcję aktywacji. Dane wyjściowe bloku są dodawane do oryginalnych danych wejściowych. Sieć ResNet-50 składa się z 16 bloków rezydualnych, z których każdy zawiera 3 warstwy konwolucyjne. Dodatkowo, przed blokami, sieć zawiera jedną warstwę konwolucyjną i warstwę MaxPool, co daje łącznie 50 warstw. ResNet-50 jest popularną architekturą wykorzystywaną w problemie reidentyfikacji, a także w innych dziedzinach.

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		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

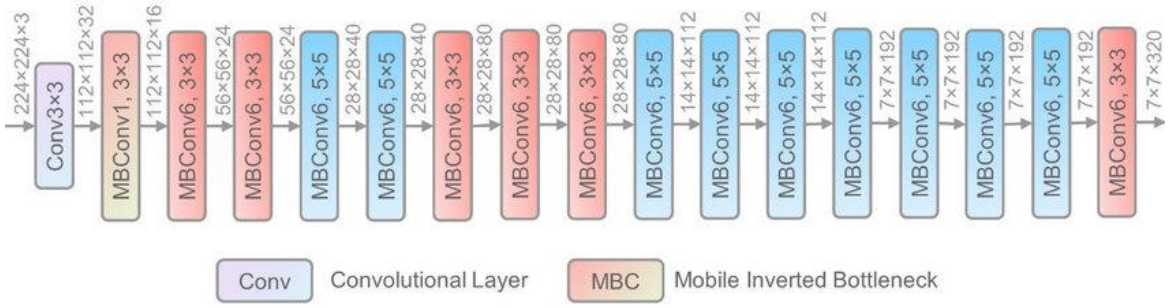
Rysunek 3 – architektura sieci rodziny resnet

- **VGG16** należy do rodziny sieci VGGNet (Visual Geometry Group) i jest powszechnie stosowana do klasyfikacji obrazów. Składa się z 13 warstw konwolucyjnych, po których następuje operacja max pooling, oraz 3 warstw w pełni połączonych. Wykorzystanie sieci z wieloma niewielkimi filtrami o rozmiarze 3x3 pozwala na efektywną ekstrakcję cech o różnym stopniu złożoności. Liczba filtrów jest zwiększana dwukrotnie po każdym bloku, a wielkość map jest zmniejszana o połowę. VGG16 charakteryzuje się dużą głębokością i liczbą filtrów, co przekłada się na jej zdolność do wykrywania bardziej skomplikowanych cech w obrazach.

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 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Rysunek 4 – architektura sieci VGGNet [4].

- **EfficientNet B7** należy do rodziny sieci EfficientNet, która opiera się na technice Compound Scaling, czyli równoczesnym skalowaniu głębokości, szerokości i rozdzielczości sieci. EfficientNet B7 składa się z bloków MBConv, które zawierają konwolucje 1x1 do rozszerzenia i kompresji kanałów, konwolucje z przestrzeni grupowej (DWConv) oraz mechanizm uwagi Squeeze and Excitation (SE). Wykorzystuje również połączenia typu skip w niektórych blokach MBConv, które łączą wyjście z wcześniejszą warstwą. Sieć wykorzystuje warstwy Batch Normalization i funkcję aktywacji Swish. Na końcu sieci używany jest Global Average Pooling (GAP), a następnie warstwa gęsta (Dense) przekształca wektor cech w prawdopodobieństwa dla poszczególnych klas. EfficientNet B7 jest zaawansowanym modelem o wysokiej wydajności, który osiąga dobre rezultaty w różnych zadaniach związanych z przetwarzaniem obrazów.



Rysunek 5 – architektura sieci EfficientNet [6]

4. Realizacja

Wstępnie podzielono zbiory wejściowe na treningowy, walidacyjny oraz testowy. Przed przystąpieniem do trenowania modeli zbiory te został poddane transformacji (między innymi przekształceniu na oczekiwany format/rozmiar) tak aby były gotowe do wykorzystania w procesie uczenia.

Obrazy są przechowywane w tablicy NumPy a po przetworzeniu zwracane w postacji tensora wraz z odpowiadającą mu etykietą.

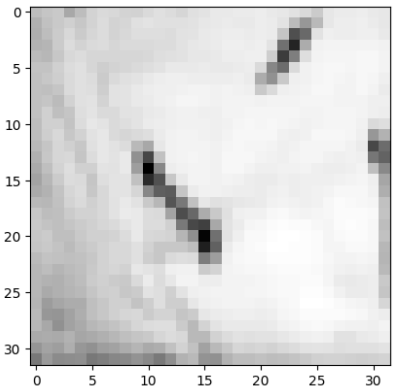
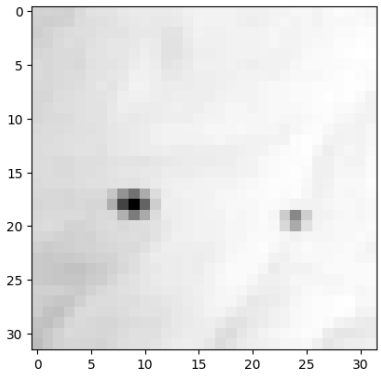
Proces uczenia polega na iteracyjnym wywoływaniu metody training_step dla każdej porcji danych treningowych oraz metody validation_step dla danych walidacyjnych. Straty są obliczane i aktualizowane w procesie uczenia, a postęp jest monitorowany i zapisywany za pomocą narzędzi dostarczonych przez PyTorch Lightning.

Sprawdzanie poprawności dla danego modelu realizowane jest przy pomocy funkcji compute_embeddings. Działa w trybie ewaluacji, iteruje po danych wejściowych z dataset'u i dla każdego przykładu oblicza osadzenie za pomocą modelu. Osadzenia są gromadzone i zapisywane w tensorze embeddings. Z kolei funkcja find_nearest_neighbours znajduje najbliższych sąsiadów dla danego osadzenia.

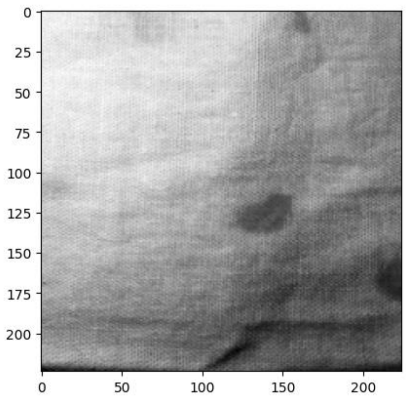
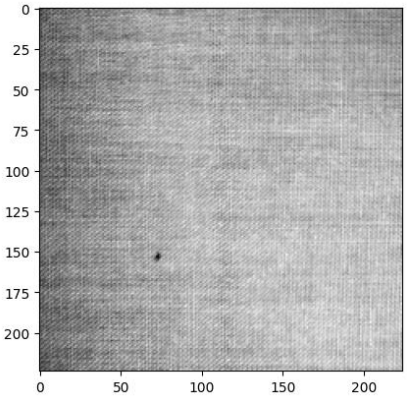
5. Wyniki i wnioski

W większości przypadków uzyskano niezadowalające wyniki czego powodem jest najprawdopodbniej mały zbiór danych. Przykłady (dla obrazu po lewej model zakwalifikował najbliższy mu obraz jako ten po prawej)

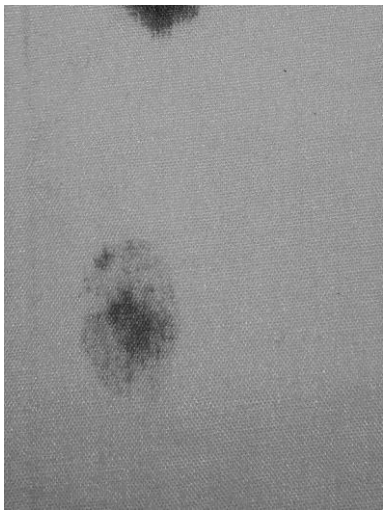
VGG16:

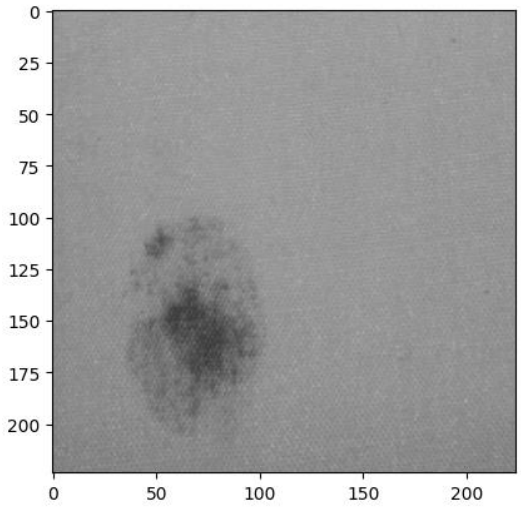



ResNet50:

EfficientNet B7:





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