



Cyfrowa rewolucja w Life Science

Widzenie komputerowe i sztuczna inteligencja.

Dawid Rymarczyk, Data Scientist @ Ardigens

PyStok 20.03.2019



- Kilka słów o firmie i o mnie
- SI i Widzenie komputerowe w Naukach o Życiu
- Studia przypadku:
 - Automatyczna ocena stopnia owrzodzenia tkanki na zdjęciach histopatologicznych z przewodu pokarmowego
 - Detekcja i segmentacja jąder komórkowych na zdjęciach mikroskopowych
- Podsumowanie

Dlaczego powinniśmy się przejmować rakiem?

Z wszystkich żyjących ludzi...

**jeden na trzech będzie miał
raka**

**jeden na dwóch chorych na
raka umrze**

TECHNOLOGY PLATFORMS

Increasing response rates
in Immuno-Oncology
with Artificial Intelligence

Neoepitope Prediction Platform

Prioritization of immunogenic neoepitopes
for effective cancer vaccines design and
pHLA for TCR screening

Microbiome Analysis Platform

Leveraging the whole metagenome
information
for effective therapy design

Biomarker Discovery Platform

Maximizing the value of clinical trials data
for biomarker discovery



ardigen

Artificial
Intelligence
& Bioinformatics
for Precision
Medicine

70+
employees

CUSTOM SERVICES

Partner in Drug Discovery
in the era of
Artificial Intelligence

Methods and Tools

Bioinformatics, AI, Data Science,
Software Engineering

Data Expertise

Genomics, Transcriptomics, Proteomics,
Metabolomics, Metagenomics,
Epigenomics, Chemical Data

Domain Knowledge

Cancer Biology, Immunology, Microbiome

Competence Centers

ardigen



Marek Piątek, PhD
Bioinformatics
Director



Barbara Kalinowska,
PhD
Biophysics



Marzena Mura,
PhD
Bioinformatics



Michał Warchał,
PhD
AI Labs Director



Jan Kaczmarczyk,
PhD
Data Science



Bartosz Zieliński,
PhD
Computer Vision

Biology & Bioinformatics



Piotr Radkowski
Software Eng.
Director



Karol Horosin



Michał Kurdziel

Software Engineering

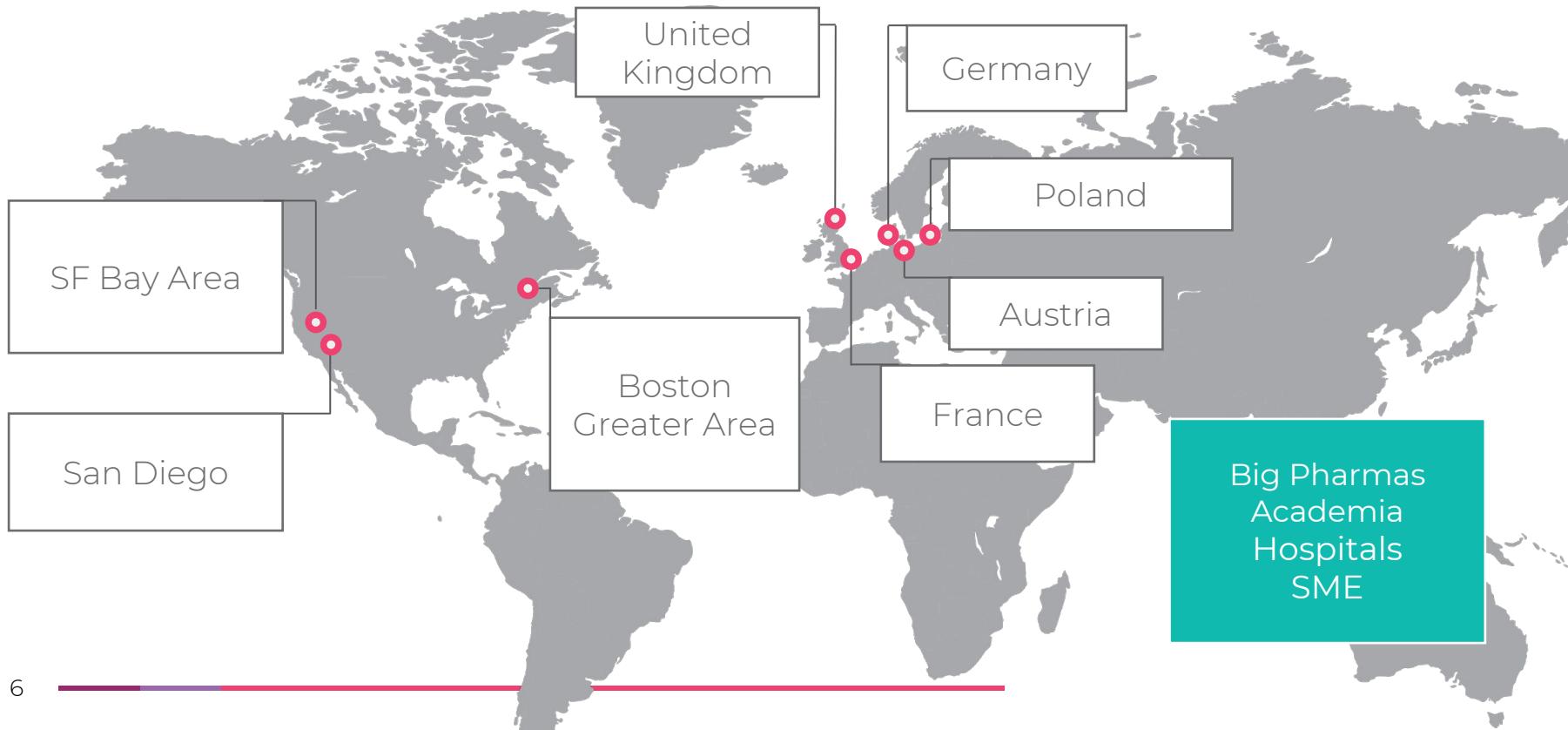
AI & Data Science

70+ People

Bioinformatics	23 (6 PhD)
AI & Data Science	11 (4 PhD)
Software Engineering	34

Fast growing customer base

ardigen



Ardigen

Make a difference. Coda against cancer.

ardigen

Through our know-how, experience and products we support Life Science & Healthcare industries towards Precision Medicine.

ardigen

**Make a difference,
Code Against Cancer**

ardigen.com



Dawid Rymarczyk

ardigen

Data Scientist. Computer Vision Specialist.

- Absolwent Teleinformatyki na IET AGH w 2018
- Brązowy medalista Kaggle
- Certyfikowany przez Stanford University w zakresie Statistical Learning
- Od 2017 @ **Ardigen** jako Data Scientist and Computer Vision Specialist
- Od 2018 współpracuje z GMUM UJ.

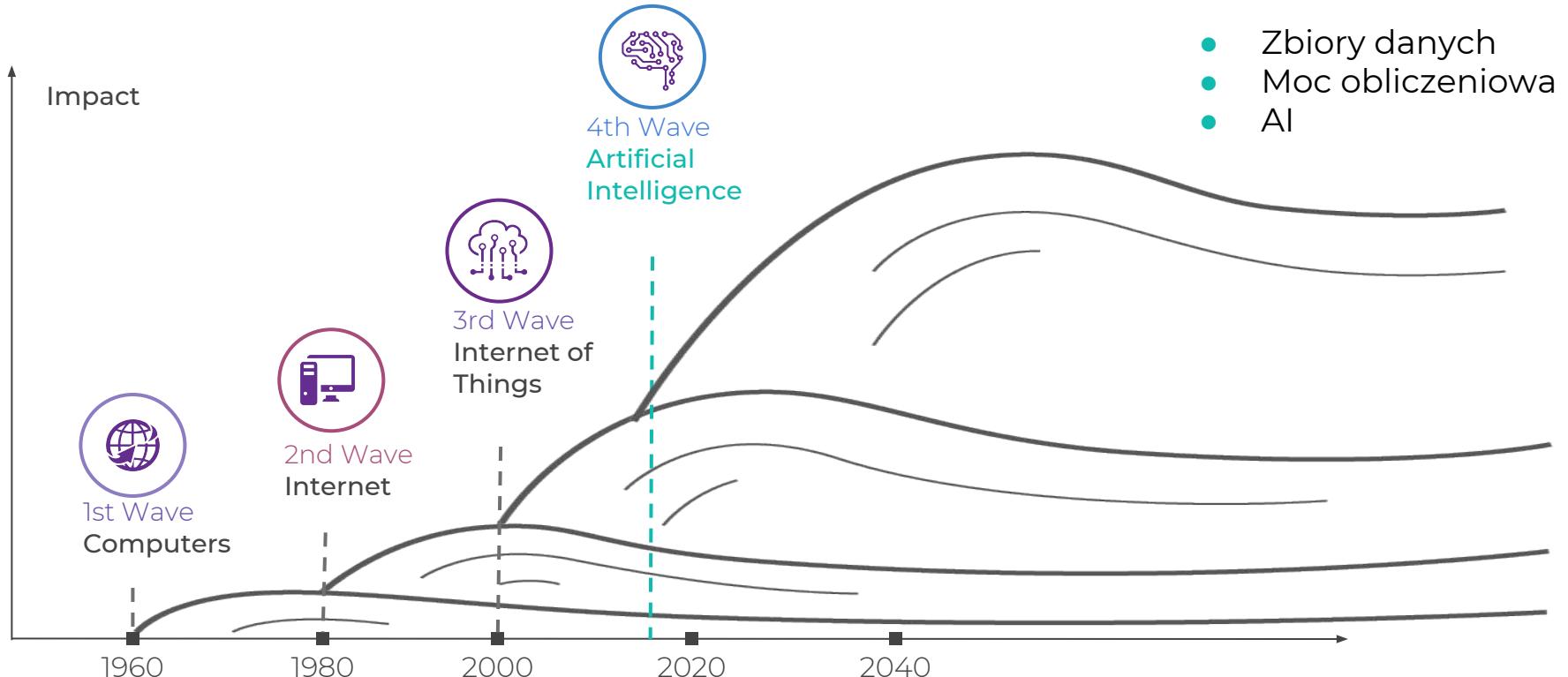


SI i Widzenie komputerowe w Naukach o Życiu



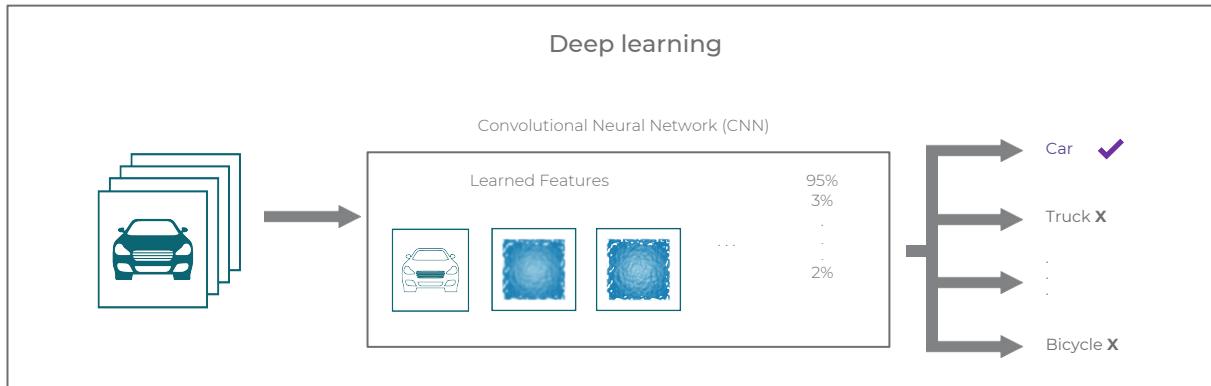
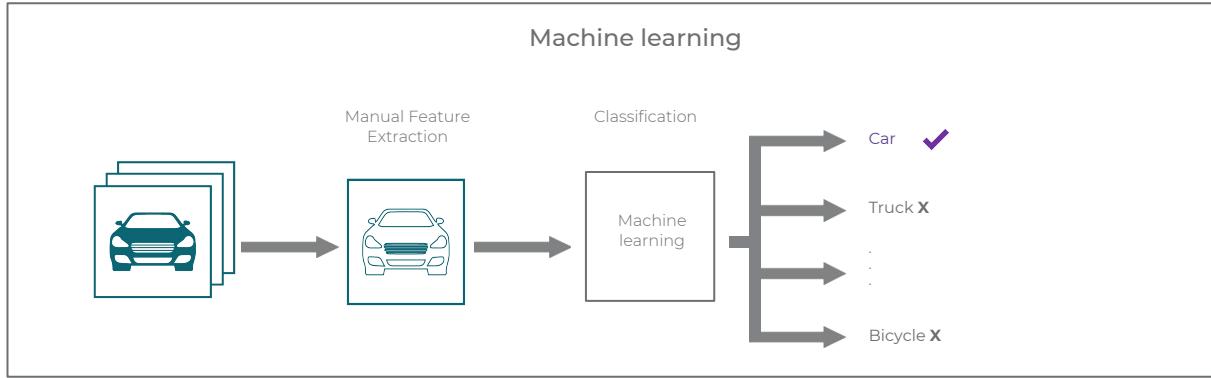
Kolejne cyfrowe rewolucje

ardigen



Uczenie maszynowe i głębokie

ardigen



Modele uczenie głębokiego uzyskują wyniki lepsze od ludzi w dobrze zdefiniowanych zadaniach

Language Translations

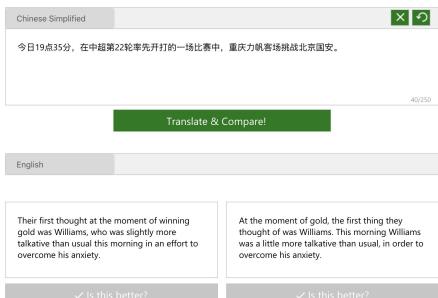
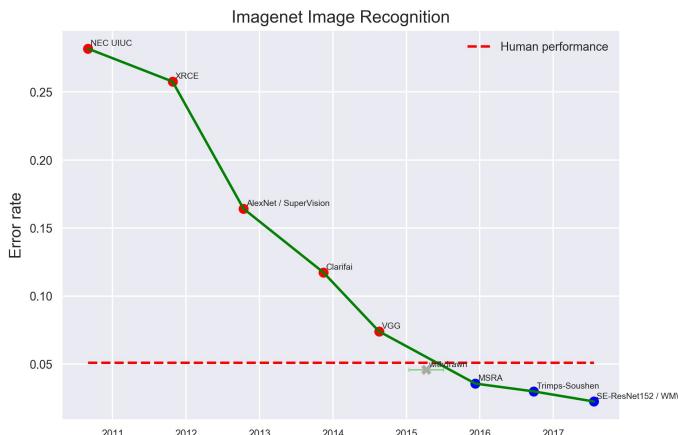
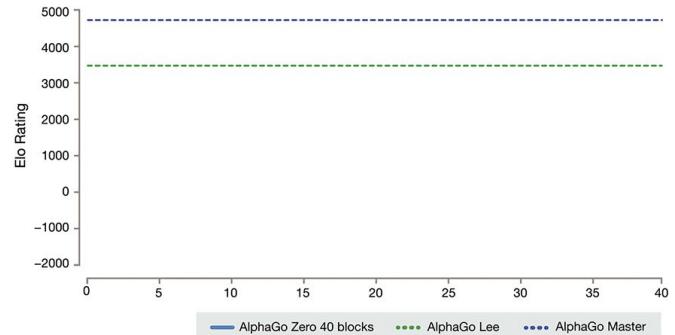


Image recognition



Games (Go, chess, Starcraft, ...)



SI a opieka zdrowotna

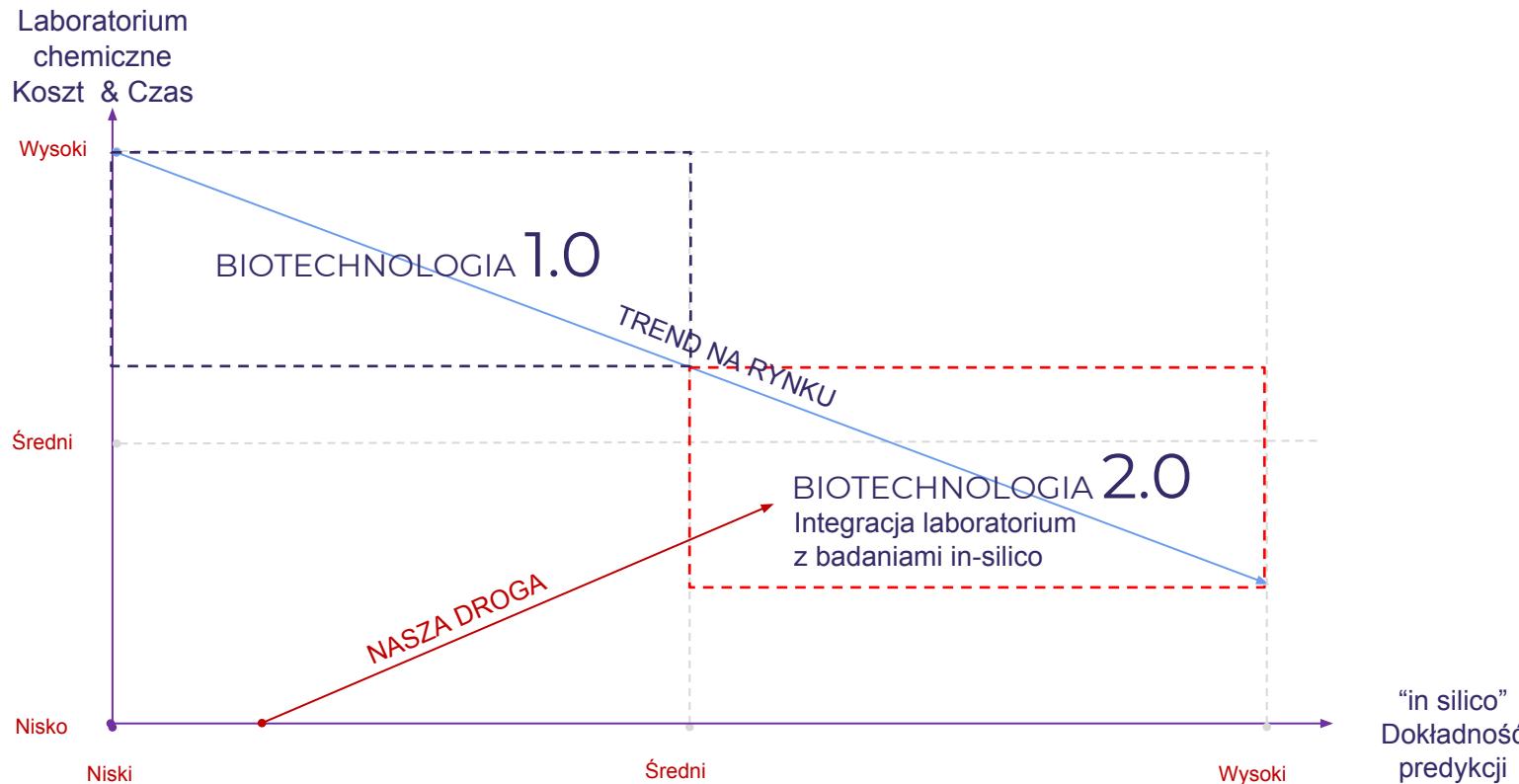
ardigen



- Badania
- Obrazowanie medyczne
- Diagnostyka
- Odkrywanie leków
- Operacje
- Zarządzanie danymi
- Wearables
- Asystenci wirtualni
- Odżywki
- ...

Od *in silico* do *in vitro* i *in vivo* ewaluacji leków

ardigen



Automatyczna ocena stopnia oworzodzenia tkanki na zdjęciach histopatologicznych z przewodu pokarmowego

- Wprowadzenie do problemu
- Preprocessing danych
- Uproszczenie problemu jako krok pośredni
- Dodatkowe filtrowanie danych
- Ostateczny model
- Python

Widzenie komputerowe studium przypadku

Automatyczna ocena stopnia zaawansowania IBD

ardigen

Tło biologiczne:

- Choroba zapalna jelit (IBD) jest terminem obejmującym przewlekłe zapalenie przewodu pokarmowego
 - Dwa schorzenia:
 - Ulcerative colitis (UC)
 - Owrzodzenie odbytnicy rozszerzające się do jelita krętego
 - Relatywnie jednorodna
 - Choroba Leśniowskiego-Crohna (CD)
 - Zapalenie obejmuje dowolną część jelit, jelito kręte i odbytnicę
 - Rozproszona i nieciągła
 - Nie występuje we wszystkich regionach jednocześnie
 - Jednym ze sposobów oceny zaawansowania zmian jest ocena histopatologiczna
 - Ocena aktualnie wymaga wyspecjalizowanego histopatologa i jest czasochłonna
-

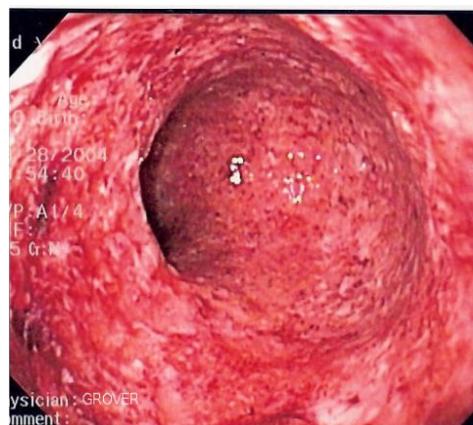
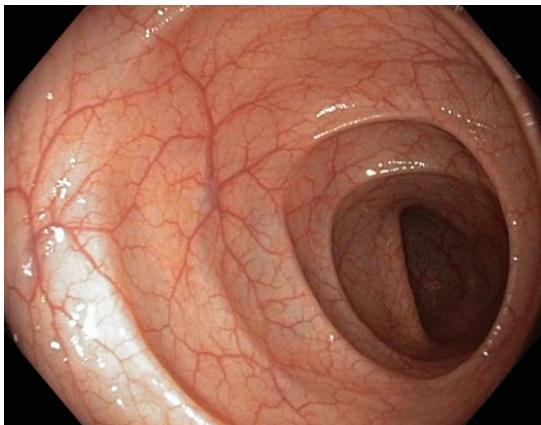
Widzenie komputerowe studium przypadku

Automatyczna ocena stopnia zaawansowania IBD

ardigen

Diagnoza IBD:

- Wykluczanie przyczyn zakażenia
- Badania kultur bakterii, C. diff, Crypto/Giardia
- Test krwi: są nieczułe i niedokładne
- Test stolca (calprotectin): czuły ale nie dokładny
- **Złoty standard:** endoskopia/kolonoskopia i biopsja



Widzenie komputerowe studium przypadku

ardigen

Dane

- Zbiór treningowy: 1669 zdjęć:
 - 792 zdjęć jelita krętego
 - 877 zdjęć odbytnicy
 - W rzeczywistości zbiór danych jest mniejszy ze względu na problemy z jakością
- Zbiór testowy: 388 zdjęć:
 - 173 zdjęć jelita krętego
 - 215 zdjęć odbytnicy
 - W rzeczywistości: 368 zdjęć

Widzenie komputerowe studium przypadku

ardigen

Dane

- Dystrybucja ocen:
 - Kategoria 1

	0	1	2	
Jelito krete		385	197	55
Odbytnica	558	115	42	

- Kategoria 2

	0	1	2	
Jelito krete		260	273	104
Odbytnica	461	158	96	

- Kategoria 3

	0	1	2	NA	
Jelito krete		422	165	49	1
Odbytnica	582	103	30	0	

Widzenie komputerowe studium przypadku

ardigen

Automatyczna ocena stopnia zaawansowania IBD

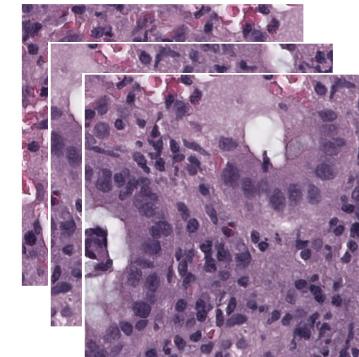
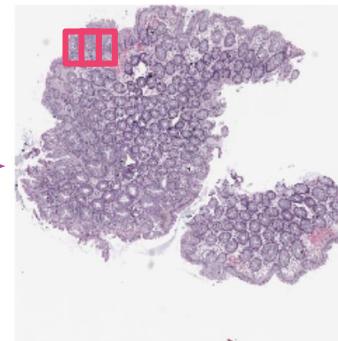
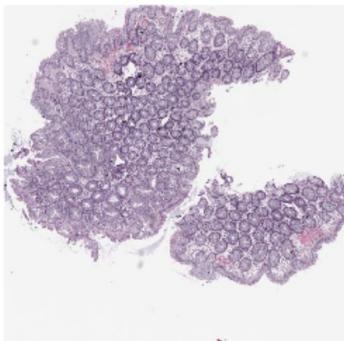
Preprocessing

Biopsja z oceną

Generowanie
maski

Mechanizm
okna
przesuwnego

Próbki



Ocena zmian dla
biopsji taka sama dla
każdej próbki (w
pewnym stopniu
fałszywe)

Widzenie komputerowe studium przypadku

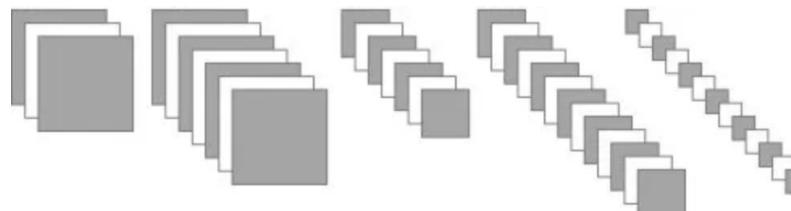
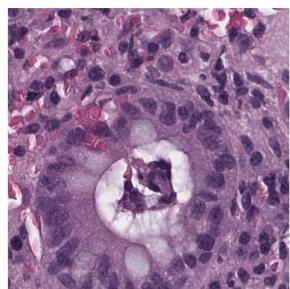
Automatyczna ocena stopnia zaawansowania IBD

ardigen

Wykorzystanie modeli

512 x 512 x 3

CNN models



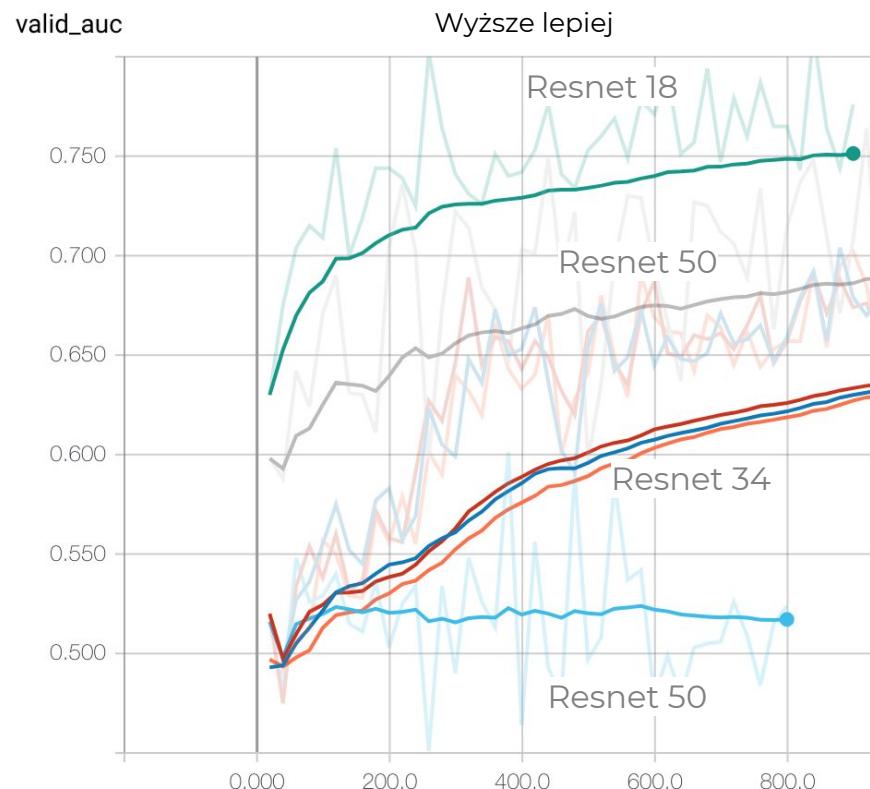
Wyjście: 1
(chory/zdrowy)

- Transfer wiedzy
- Nauka od podstaw

Widzenie komputerowe studium przypadku

Automatyczna ocena stopnia zaawansowania IBD

ardigen

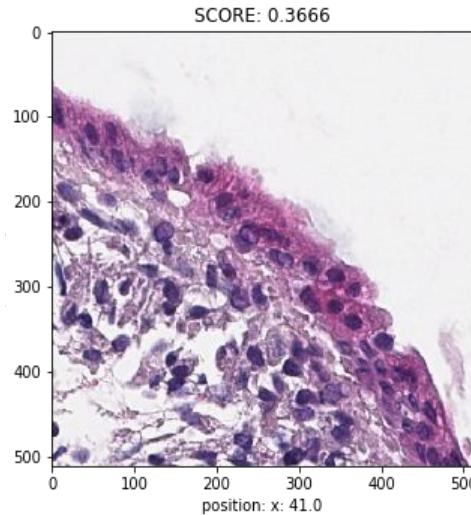
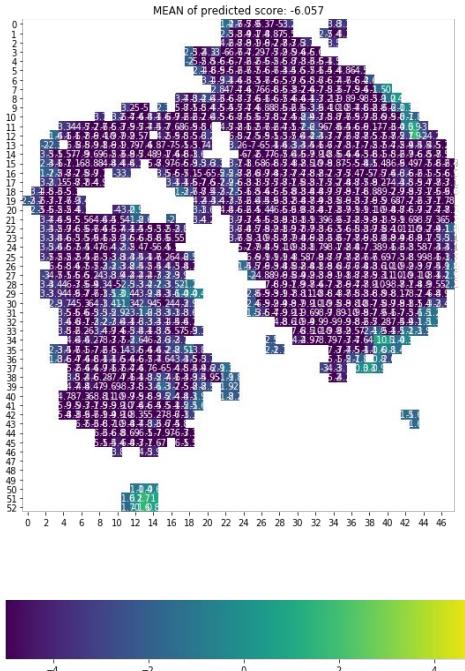


Widzenie komputerowe studium przypadku

Automatyczna ocena stopnia zaawansowania IBD

ardigen

Duże obrazy, a małe obszary owrzodzenia



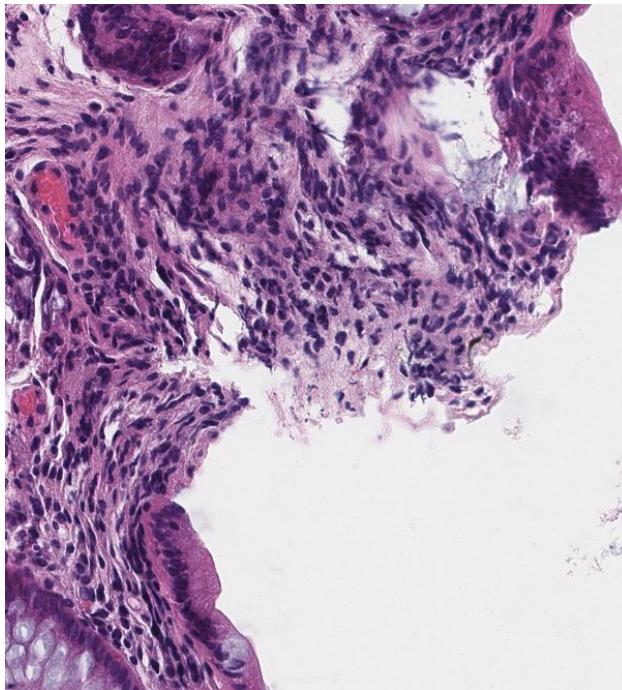
Uszkodzony nabłonek

Widzenie komputerowe studium przypadku

Automatyczna ocena stopnia zaawansowania IBD

ardigen

Dystraktory

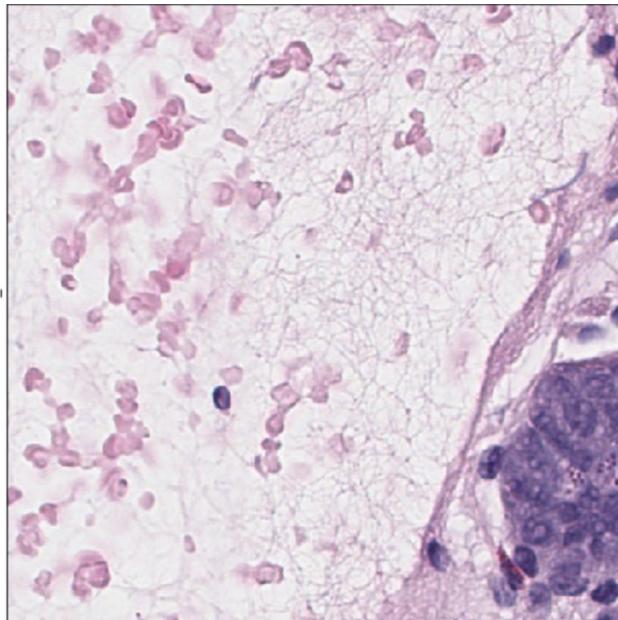


Widzenie komputerowe studium przypadku

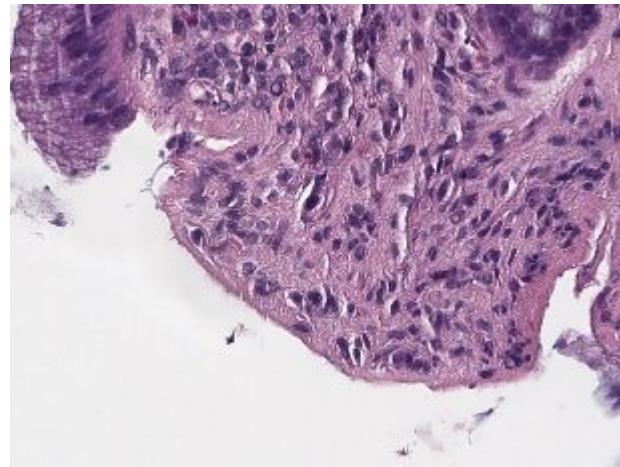
Automatyczna ocena stopnia zaawansowania IBD

ardigen

Komórki krwi



Błędy przy pobieraniu próbek

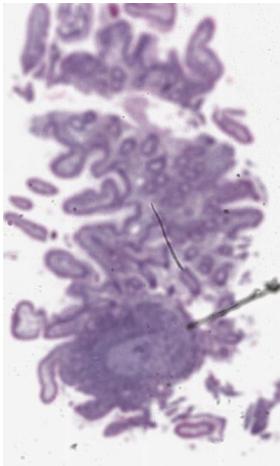


Widzenie komputerowe studium przypadku

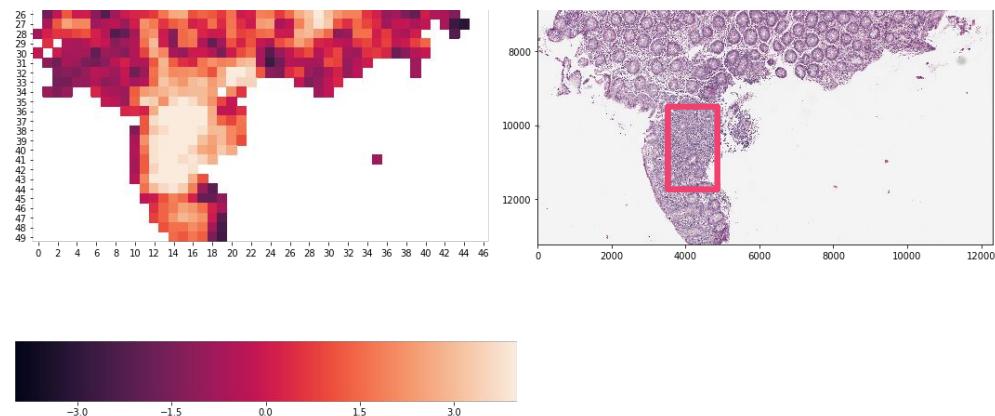
Automatyczna ocena stopnia zaawansowania IBD

ardigen

Jakość danych - zawsze kłopot ;)



Naciek zapalny



Widzenie komputerowe studium przypadku

Automatyczna ocena stopnia zaawansowania IBD

ardigen

1. Usunięcie rozmytych próbek:
 - a. Odchylenie standardowe wyniku operatora Laplaciana
2. Usunięcie białych fragmentów:
 - a. Analiza percentylu kanału S w reprezentacji HSV

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Usunięcie 10% próbek ze zbioru treningowego

Widzenie komputerowe studium przypadku

Automatyczna ocena stopnia zaawansowania IBD

CNN

Osobny model
dla różnych
części ciała

2 stopień CNN

Opis

1. Podział biopsji na próbki
2. Przypisanie próbkom wartości odpowiadających biopsji
3. Nauka modelu
4. Agregacja informacji ze wszystkich próbek z biopsji
5. Znalezienie progu decyzyjnego

ardigen

Dokładność
(~50%)

80%

85%
79% jelita
krętego
91% odbytnica

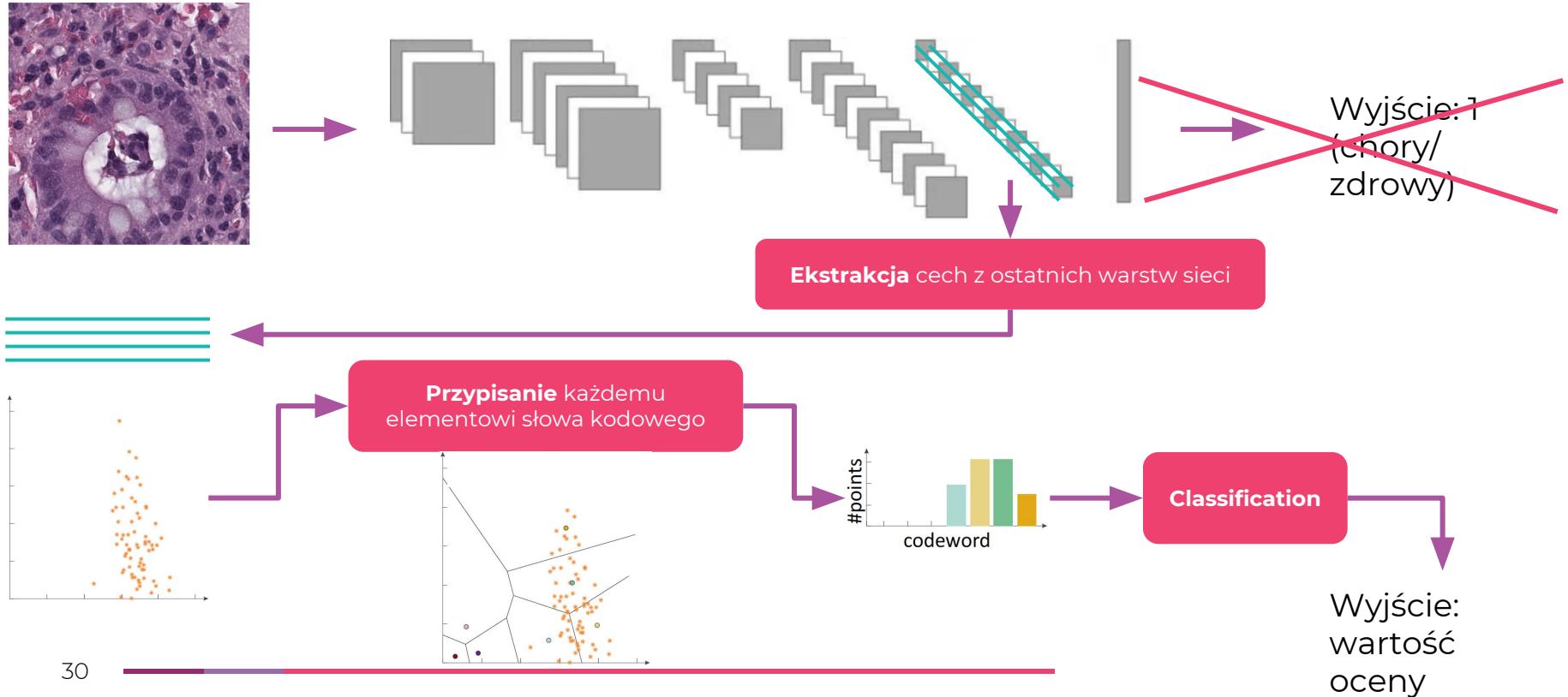
6. Budowa osobnych modeli dla odbytnicy i jelita krętego
7. Próbkowanie nowego zbioru danych:
8. Chore próbki: 5% próbek z największą wartością z modelu jako próbki chore
9. Zdrowe próbki: próbki z najniższą i najwyższą wartością z modelu
10. Powtórny trening

89%
85% jelita
krętego
94% odbytnica

Widzenie komputerowe studium przypadku

Automatyczna ocena stopnia zaawansowania IBD

ardigen



Widzenie komputerowe studium przypadku

Automatyczna ocena stopnia zaawansowania IBD

ardigen

Odbytnica

	0	1	2
0	111	16	6
1	8	15	1
2	1	0	5
	0	1	2
0	93	9	7
1	12	18	4
2	2	4	14

Kategoria 1

Kategoria 2

Kategoria 3

Jelito kręte

	0	1	2
0	64	15	12
1	16	27	8
2	0	4	8
	0	1	2
0	38	14	11
1	17	36	10
2	2	7	19

Kategoria 1

Kategoria 2

Kategoria 3

BOW

Widzenie komputerowe studium przypadku

Automatyczna ocena stopnia zaawansowania IBD

ardigen

Zastosowanie Python'a i jego bibliotek:

- Sieć neuronowa: *PyTorch*
 - Gotowe implementacje modeli jak i ich wytrenowane wersje
 - Prosty interfejs do ładowania i przetwarzania danych (*dataset*, *DataLoader*)
- Analiza wyników: *TensorBoard* w opakowaniu *TenforBoardX*
- Model BoVW: *PyVIFeat* oraz *scikit-learn*:
 - *PyVIFeat*:
 - Jedyna biblioteka w Pythonie pozwalająca na korzystanie z *FisherVector* i *GMM* dla obrazów
 - Ciężko instalowalna, dobrze wykorzystać Anacondę
 - *Scikit-learn*:
 - Definiowanie *Pipeline*'u
 - Wykorzystanie klasyfikatora *SVM*
 - Walidacja krzyżowa

Widzenie komputerowe studium przypadku PyTorch

ardigen

```
27     extractor = models.alexnet(pretrained=True).features.eval().to(device)
28     features = extractor(images)

77     loader = data.DataLoader(
78         dataset,
79         batch_size=config.batch_size,
80         shuffle=True,
81         num_workers=0,
82         pin_memory=True,
83         worker_init_fn=lambda x: np.random.seed(torch.initial_seed()))
```

Widzenie komputerowe studium przypadku PyVIFeat

ardigen

```
26     means, covars, priors, ll, posteriors = gmm(
27         X,
28         n_clusters=self.gmm_clusters_number,
29         init_mode=self.init_mode,
30     )
31     means = means.transpose()
32     covars = covars.transpose()
33     self.gmm_ = (means, covars, priors)

41     means, covars, priors = self.gmm_
42     x = x.transpose()
43     return fisher(x, means, covars, priors, improved=True)
```

Widzenie komputerowe studium przypadku

Automatyczna ocena stopnia zaawansowania IBD

ardigen

```
fv_pipeline = Pipeline(  
    steps=[  
        ('fisher vector', FisherVectorTransformer()),  
        ('svc', svm.SVC(probability=True)),  
    ]  
)
```

```
53 | pipeline = bow_pipeline if args.bow else fv_pipeline  
54 | param_grid = config.bow_param_grid if args.bow else config.fv_param_grid  
55 | pipeline = model_selection.GridSearchCV(pipeline, param_grid, n_jobs=24)
```

Detekcja jąder komórkowych na zdjęciach mikroskopowych

1. Detekcja i segmentacja jąder komórkowych - problem - Kaggle Data Science Bowl 2018
2. Instance and Semantic segmentation
3. U-Net i Mask R-CNN
4. Usprawnienia
5. Zwycięski projekt
6. Python

Detekcja jąder komórkowych

ardigen

Czas trwania



Detekcja jąder komórkowych

Motywacja

ardigen

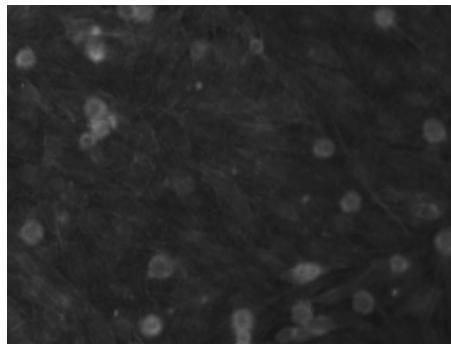
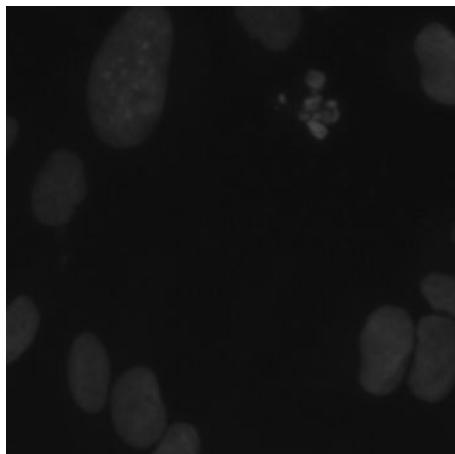
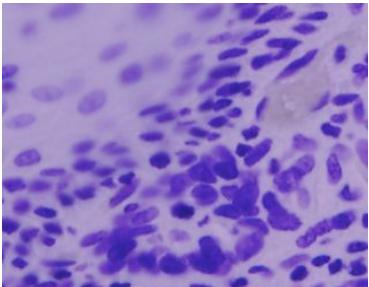
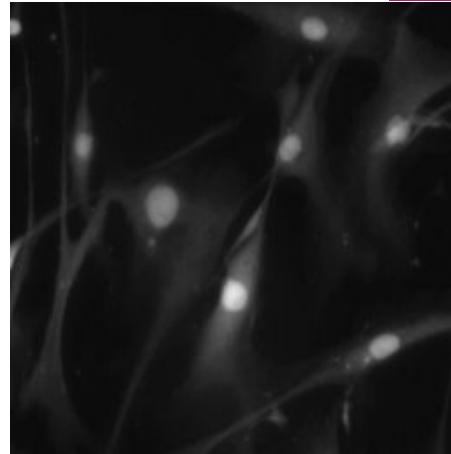
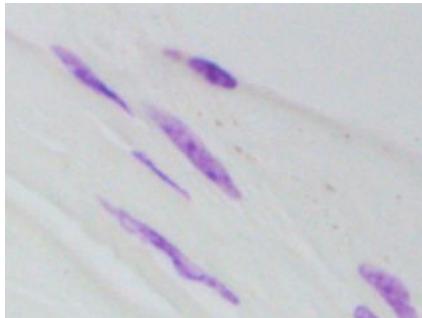
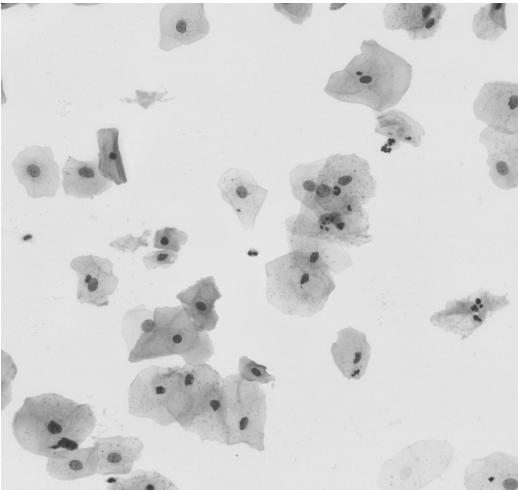


Source: kaggle.com

Detekcja jąder komórkowych

Zbiór treningowy: 670 zdjęć

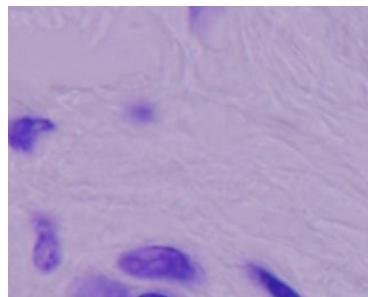
ardigen



Detekcja jąder komórkowych

ardigen

Dane

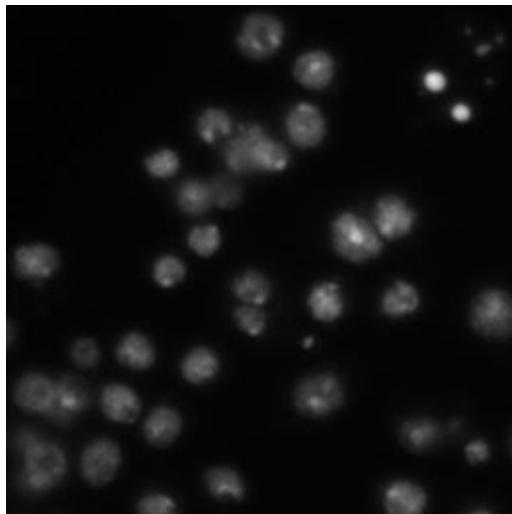


Detekcja jąder komórkowych

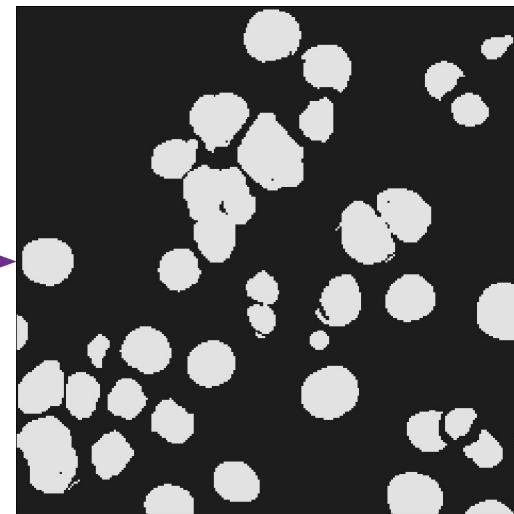
ardigen

Semantic segmentation

Image



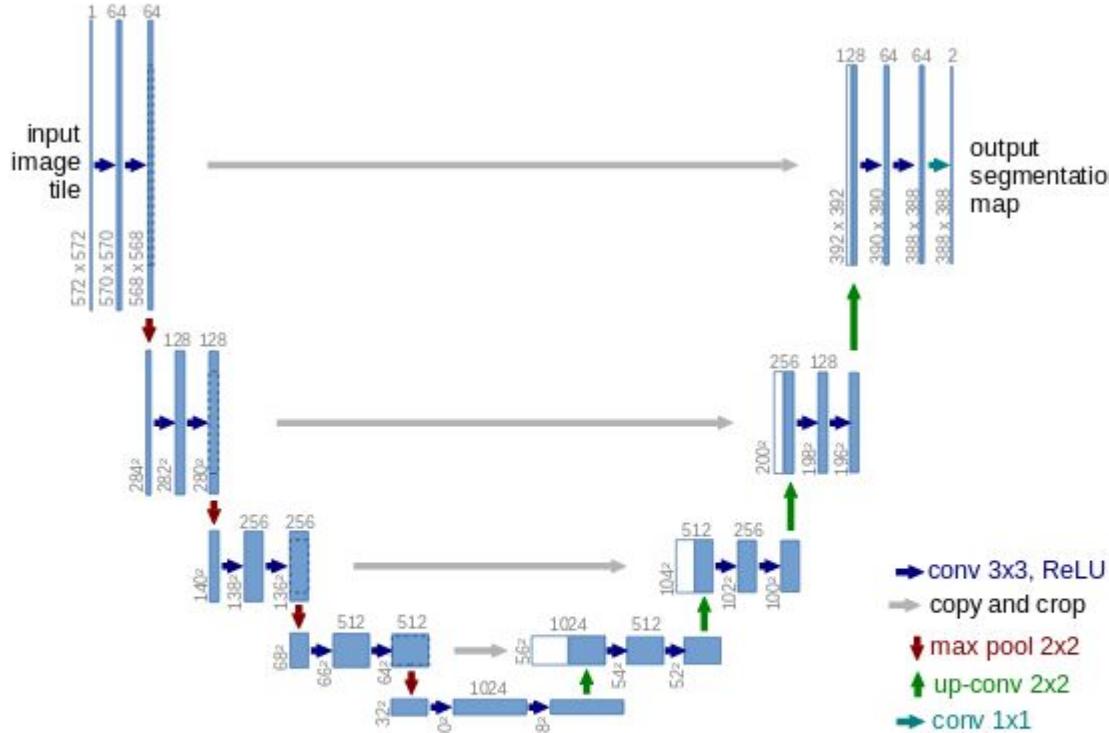
Mask



Detekcja jąder komórkowych

U-Net

ardigen

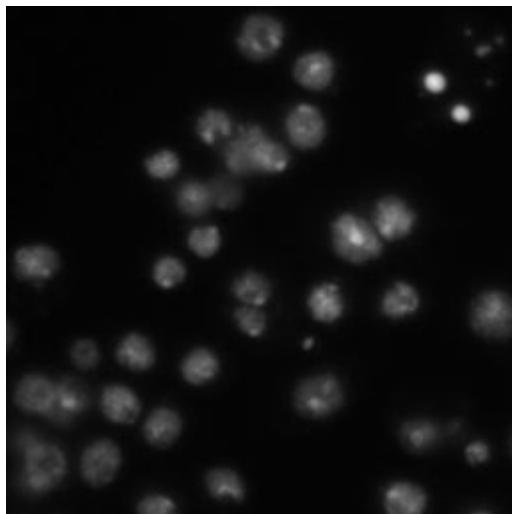


Detekcja jąder komórkowych

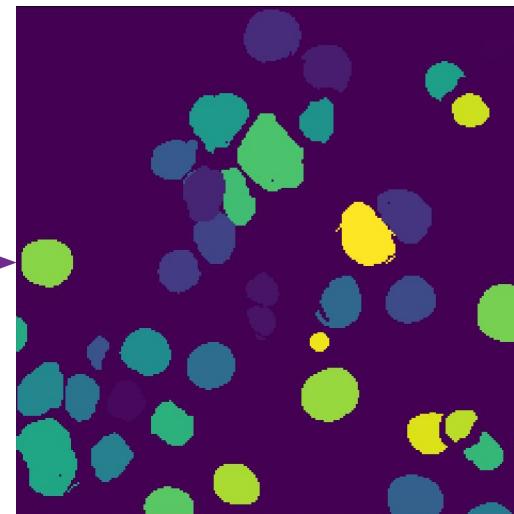
ardigen

Instance segmentation

Image



Mask



Detekcja jąder komórkowych

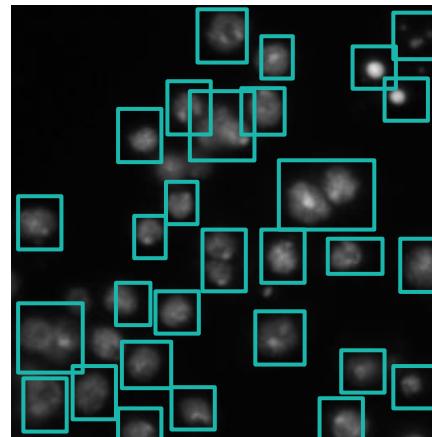
R-CNN

ardigen

Key steps of the model:

- Region proposals (Selective Search)
- Feature extraction (AlexNet)
- Classifiers (SVM)

Image with
Region
Proposals



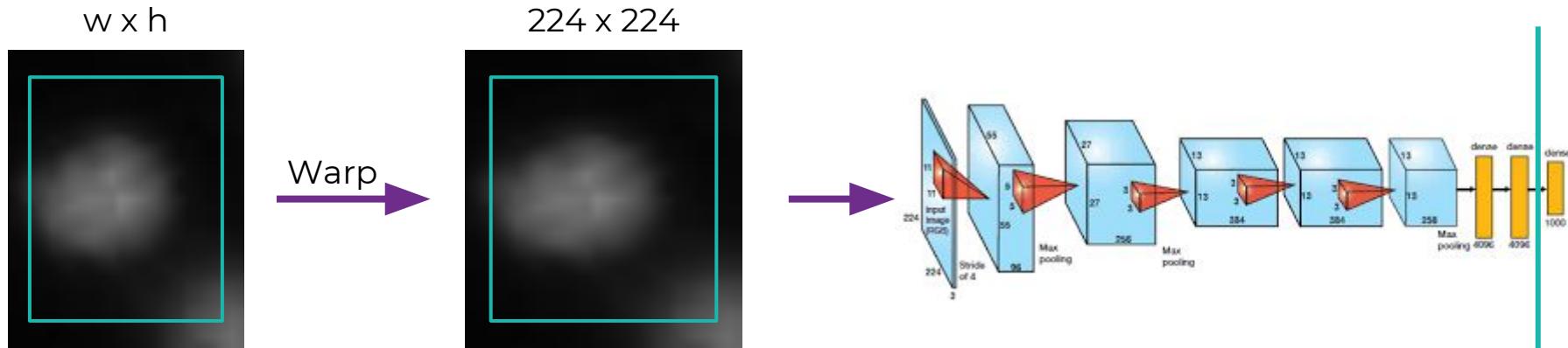
Detekcja jąder komórkowych

R-CNN

ardigen

Kroki modelu:

- Propozycja regionów
- Ekstrakcja cech
- Klasyfikacja



Source: CV-Tricks.com

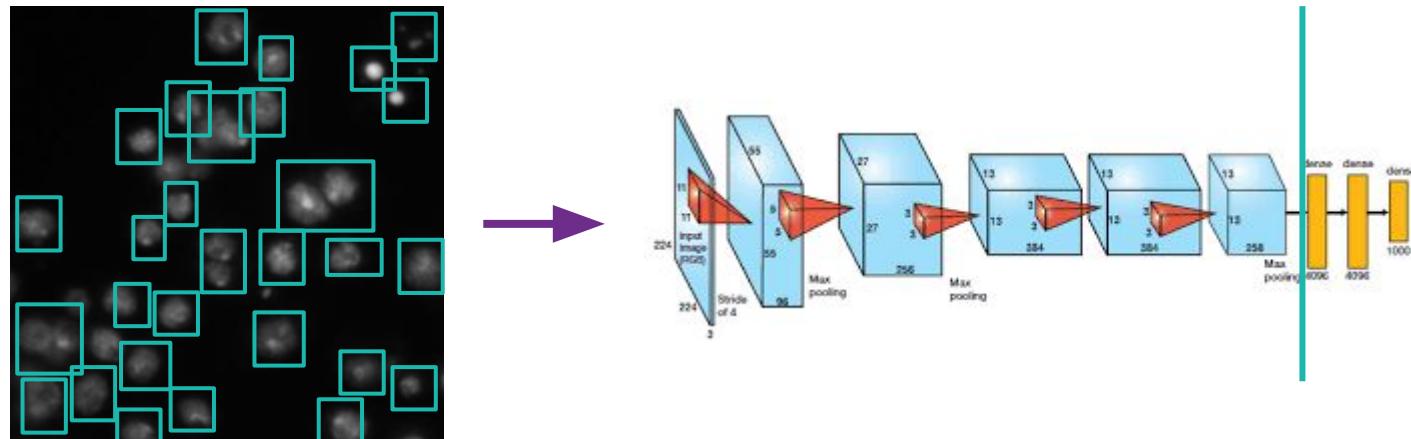
Detekcja jąder komórkowych

ardigen

Fast R-CNN

Usprawnienie:

- Przetwarzanie całego obrazu przez sieć
- Trening typu end-to-end



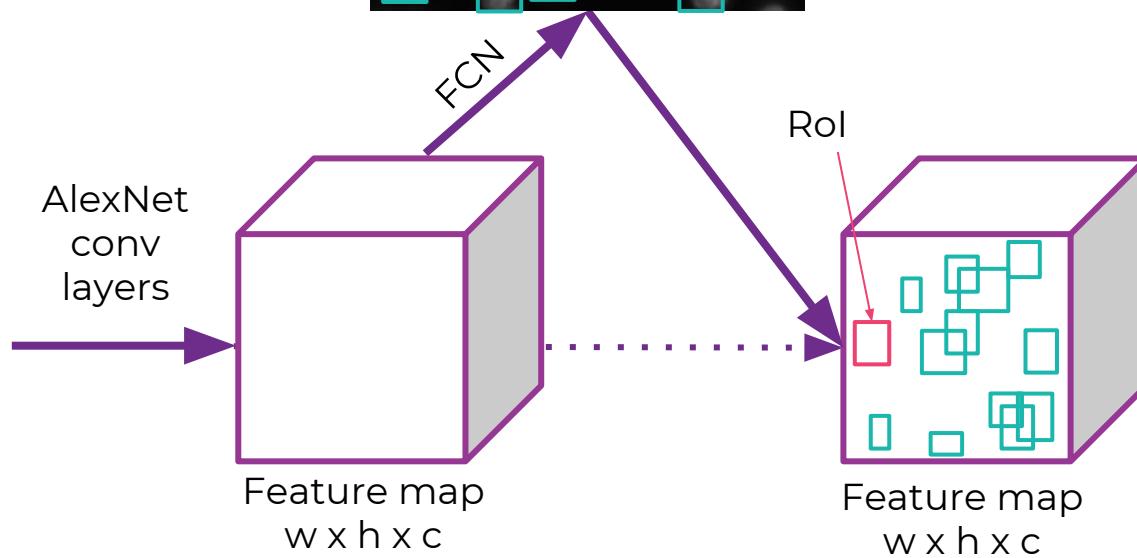
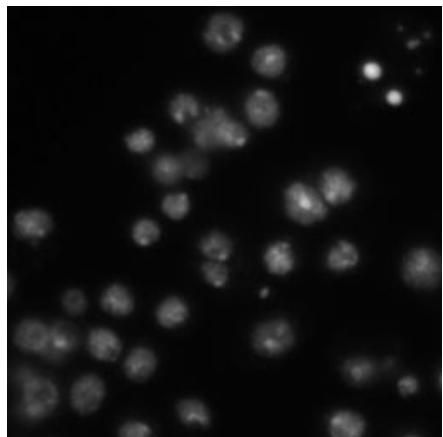
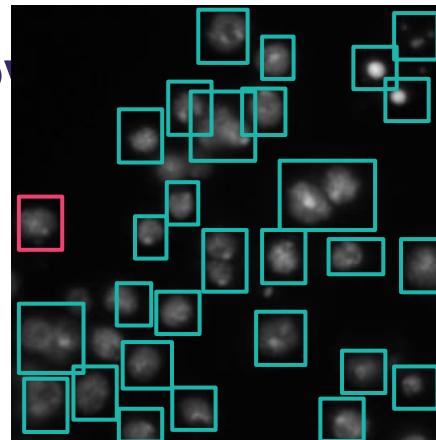
Detekcja jąder komórkowych

Faster R-CNN

ardigen

Ulepszenia:

- Region Proposal Network
- Perspektywy



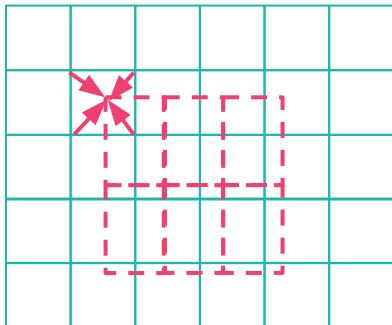
Detekcja jąder komórkowych

Mask R-CNN

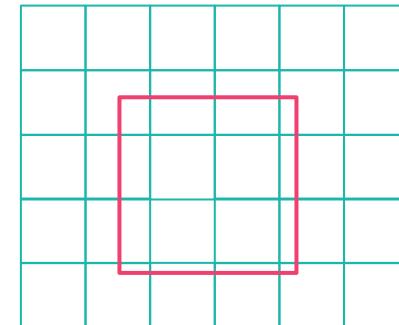
ardigen

Usprawnienia:

- Wielonauczanie
- RoI Align



Feature map



RoI

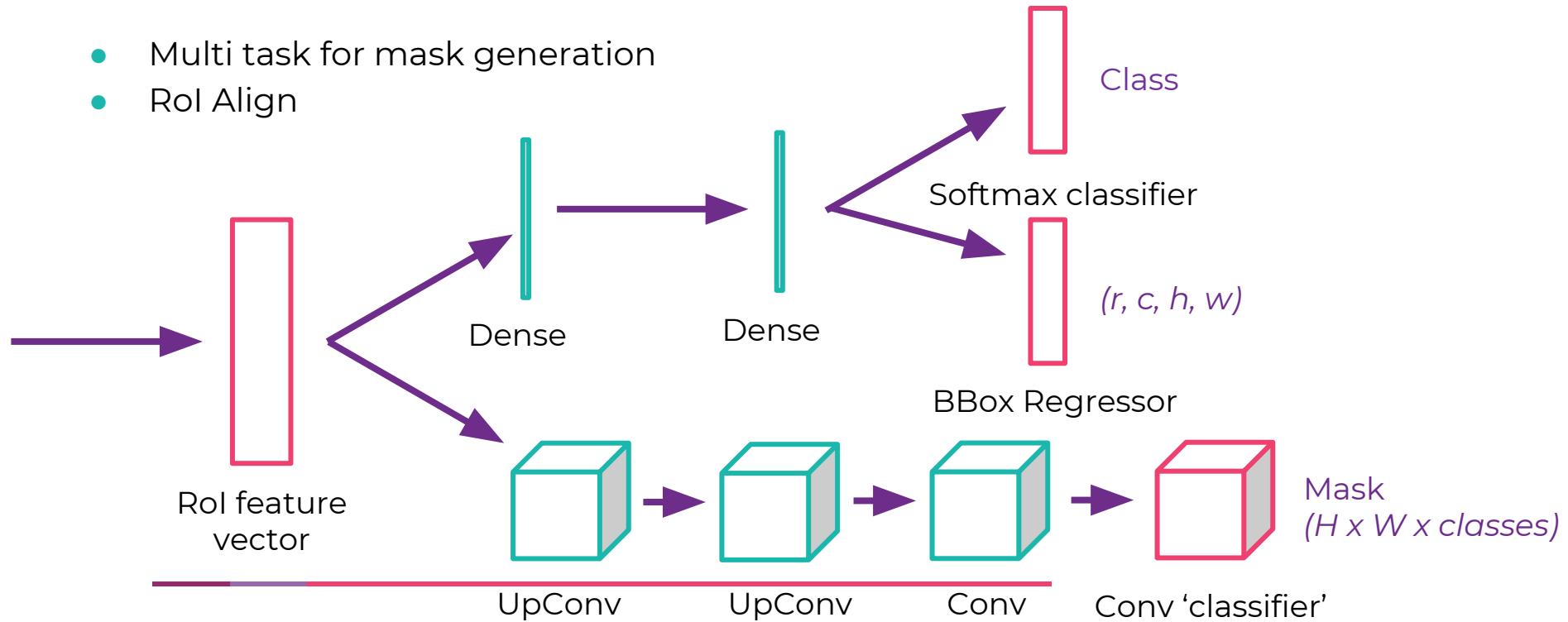
Detekcja jąder komórkowych

ardigen

Mask R-CNN

Enhancement:

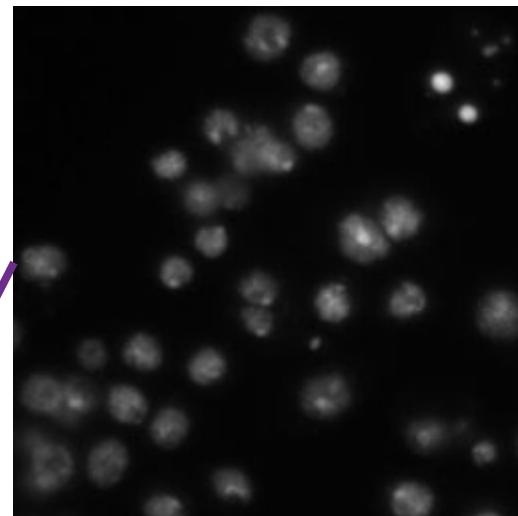
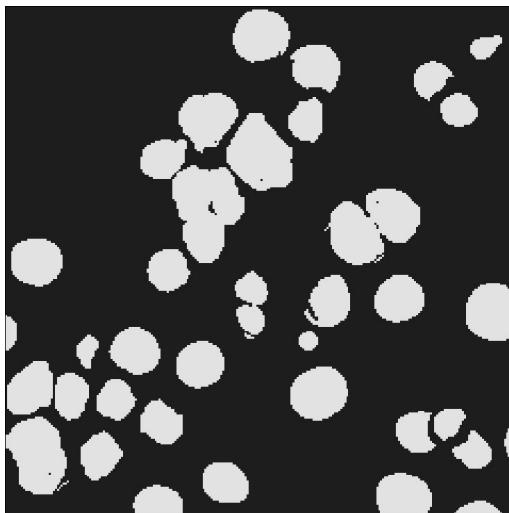
- Multi task for mask generation
- RoI Align



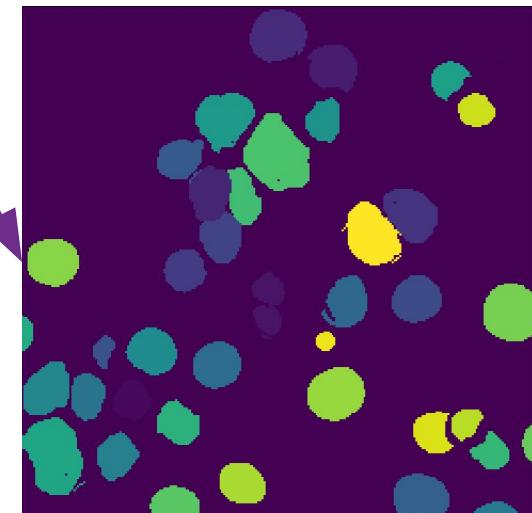
Detekcja jąder komórkowych

ardigen

Semantic



Instance

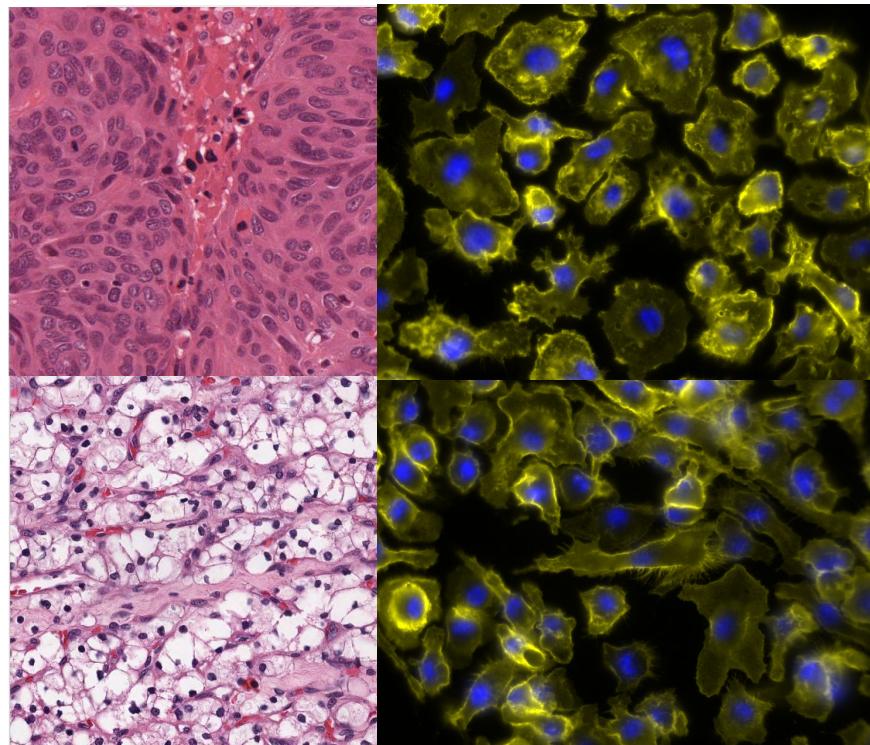
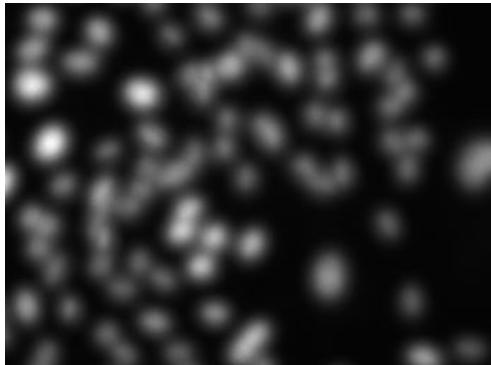


Detekcja jąder komórkowych

ardigen

Dodatkowe dane

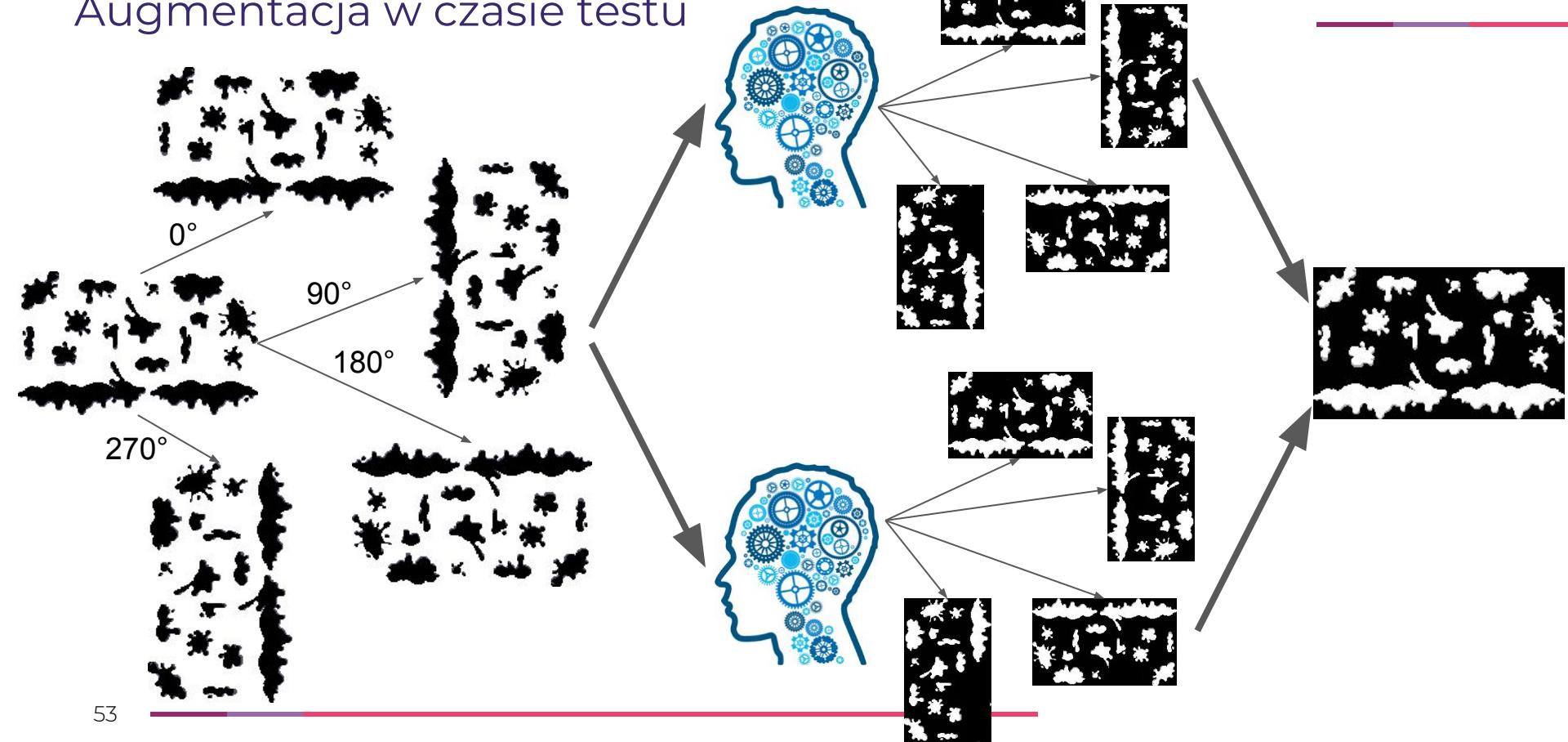
- Hematoxylin and eosin (H&E) stain
[<https://nucleisegmentationbenchmark.weebly.com/>]
- Broad Bioimage Benchmark Collection
[https://data.broadinstitute.org/bbbc/image_sets.html]
 - Human U2OS cells (out of focus)
 - Bone-marrow derived macrophages from C57BL/6 mice



Detekcja jąder komórkowych

Augmentacja w czasie testu

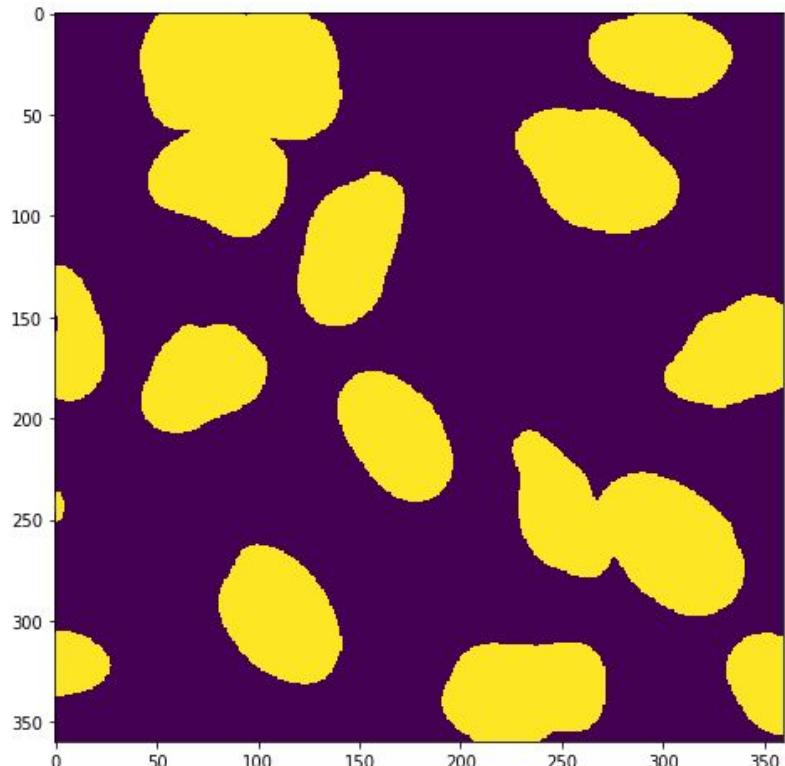
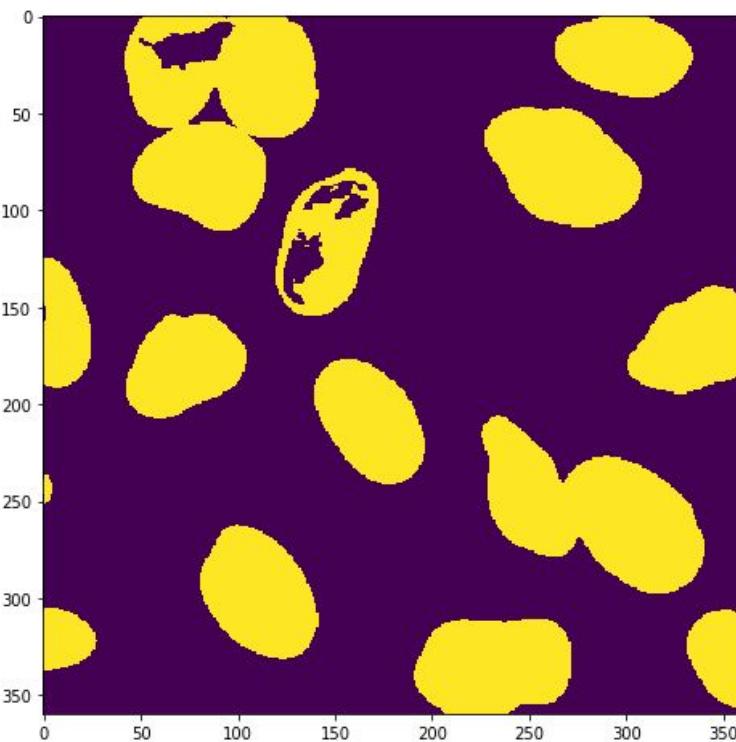
ardigen



Detekcja jąder komórkowych

ardigen

Post processing

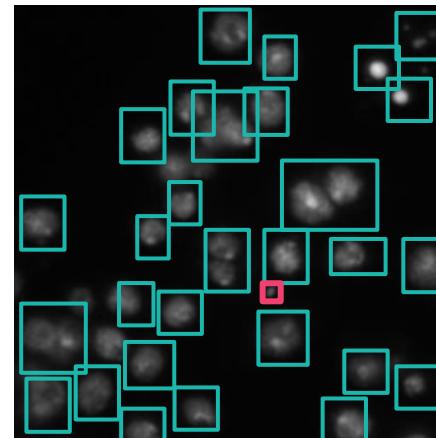


Detekcja jąder komórkowych

ardigen

Inne usprawnienia

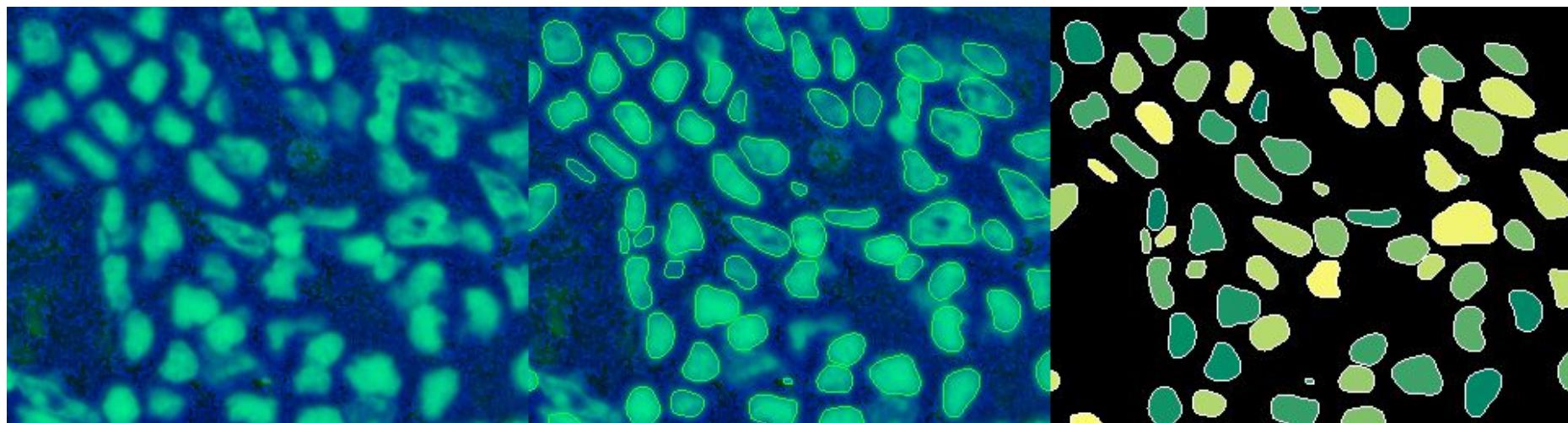
1. Dylacyjne konwolucje
2. Focal Loss
3. Różne wielkości mask
4. Zbalansowanie danych
5. Auxiliary tasks
6. Augmentacja danych w treningu



Detekcja jąder komórkowych

Wyniki

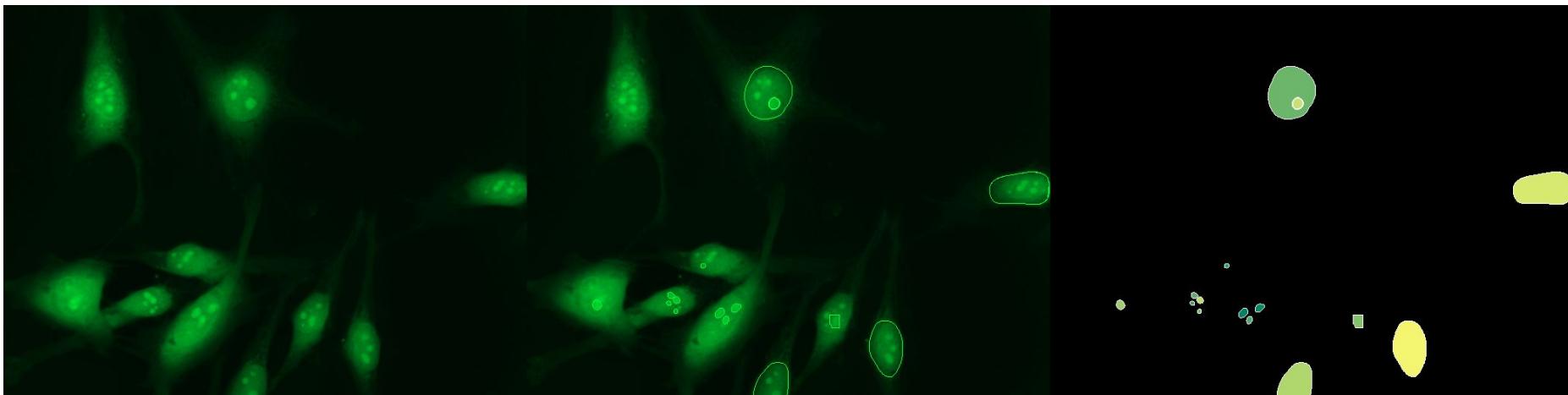
ardigen



Detekcja jąder komórkowych

Wyniki

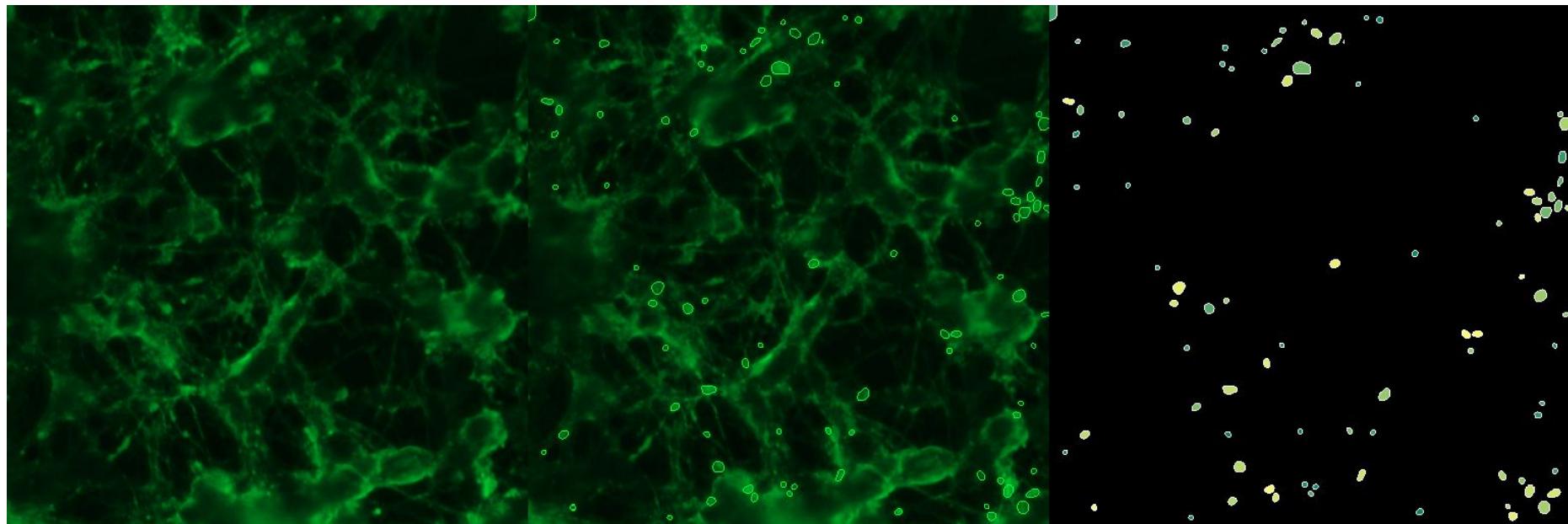
ardigen



Detekcja jąder komórkowych

ardigen

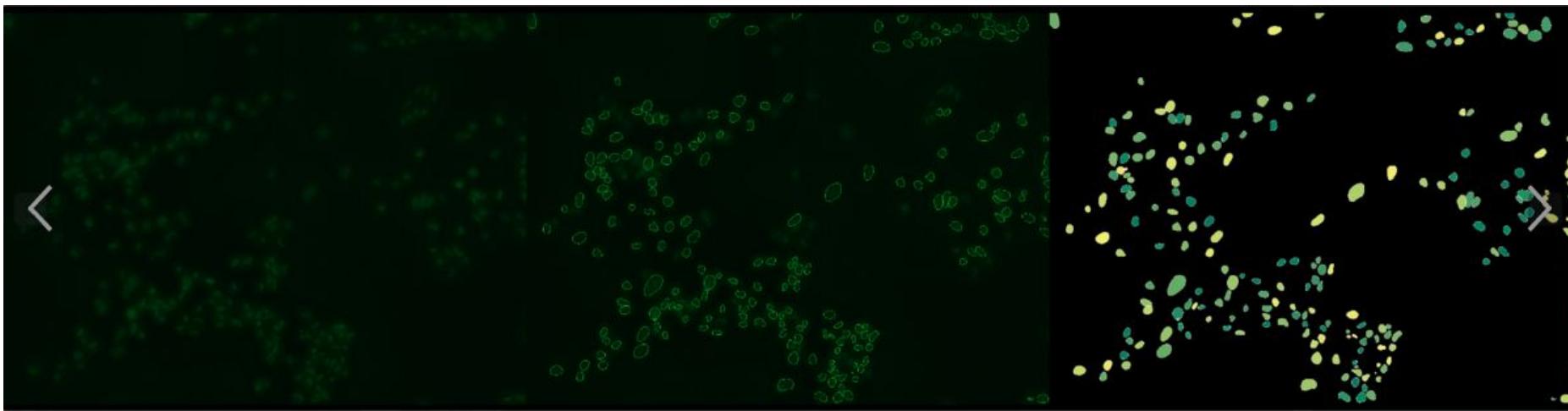
Wyniki



Detekcja jąder komórkowych

Wyniki

ardigen



Detekcja jąder komórkowych

Wyniki

ardigen

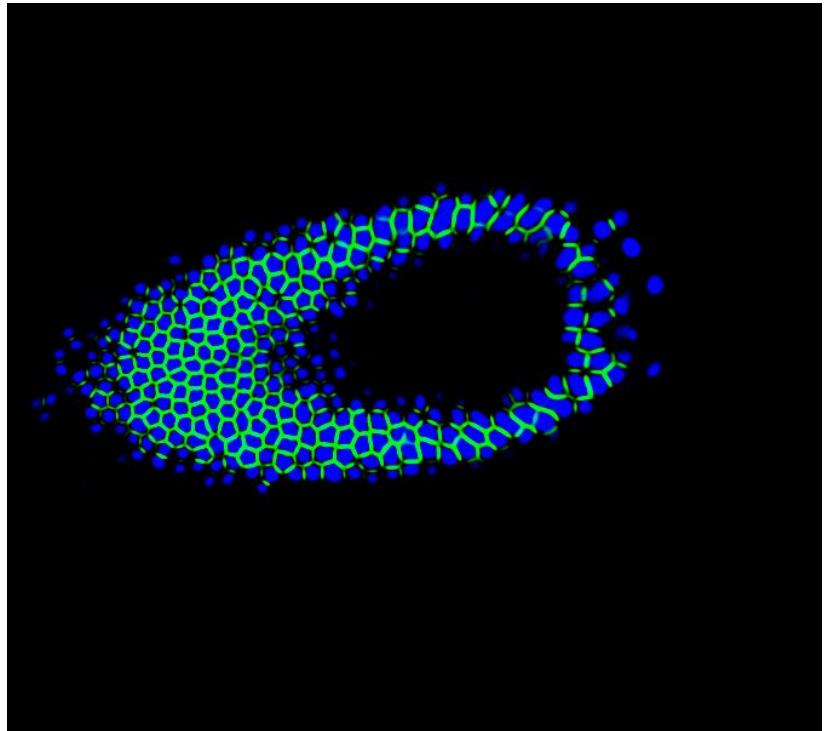


1st place U-Net on steroids

ardigen

Auxiliary tasks and post processing, deep model and augmentations:

- Clahe, Sharpen, Emboss
- Gaussian Noise
- Color to Gray
- Inverting
- Remapping grayscale to random color images
- Blur, Median Blur, Motion Blur
- contrast and brightness
- random scale, rotates and flips
- Heavy geometric transformations
- Random HSV
- Channel shuffle
- Nucleus copying on images.



Source kaggle.com

Detekcja jąder komórkowych

Python

ardigen

1. Wykorzystanie gotowego kodu w PyTorch do Mask R-CNN i rozbudowywanie go.
 - a. Moduł *transforms*
 - b. Prosty sposób na budowę funkcji kosztu złożonej z wielu funkcji
 - c. “Pythonic”

Python transform

ardigen

```
45     if self.augmentation:
46         self.transforms = [
47             transforms.ColorJitter(brightness=0.1, contrast=0.1, saturation=0.1),
48             transforms.RandomHorizontalFlip(p=0.5),
49             transforms.RandomVerticalFlip(p=0.5),
50             transforms.Resize((512, 512)),
51         ]
52
53
54
55
56
57
58
59
60
61
62
63     self.train_ds = self.dataset(
64         path_to_files=self.train_metadata['path'].tolist(),
65         labels=self.train_metadata['labels'].tolist(),
66         num_patches_per_block=self.train_num_patches_per_block,
67         transforms=self.transforms,
68         target=self.target,
69         labels_name=None,
70     )
```

Python

Loss function

ardigen

```
756 def loss(self, inputs, truth_boxes, truth_labels, truth_instances):
757     cfg = self.cfg
758
759     self.rpn_cls_loss, self.rpn_reg_loss = \
760         rpn_loss( self.rpn_logits_flat, self.rpn_deltas_flat, self.rpn_labels, self.rpn_label_weights, self.rpn_targets, self.rpn_target_weight
761
762     self.rcnn_cls_loss = torch.Tensor((cfg.rcnn_head_number)).zero_().cuda()
763     self.rcnn_reg_loss = torch.Tensor((cfg.rcnn_head_number)).zero_().cuda()
764     for i in range(cfg.rcnn_head_number):
765         rcnn_cls_loss, rcnn_reg_loss = \
766             rcnn_loss(self.rcnn_logits[i], self.rcnn_deltas[i], self.rcnn_labels[i], self.rcnn_targets[i], float(self.rcnn_deltas[i].std()), c
767             self.rcnn_cls_loss[i] = rcnn_cls_loss
768             self.rcnn_reg_loss[i] = rcnn_reg_loss
769
770     ## self.mask_cls_loss = Variable(torch.cuda.FloatTensor(1).zero_()).sum()
771     self.mask_cls_loss = \
772         mask_loss( self.mask_logits, self.mask_labels, self.mask_instances, cfg.mask_focal_loss )
773
774     self.total_loss = self.rpn_cls_loss + self.rpn_reg_loss \
775         + torch.mean(self.rcnn_cls_loss) + torch.mean(self.rcnn_reg_loss) \
776         + self.mask_cls_loss
777
778     return self.total_loss
```

Python

Loss function

```
756 def loss(self, inputs, truth_boxes, truth_labels, truth_instances):
757     cfg = self.cfg
758
759     self.rpn_cls_loss, self.rpn_reg_loss = \
760         rpn_loss( self.rpn_logits_flat, self.rpn_deltas_flat, self.rpn_labels, self.rpn_label_weights, self.rpn_targets, self.rpn_target_weight
761
762     self.rcnn_cls_loss = torch.Tensor((cfg.rcnn_head_number)).zero_().cuda()
763     self.rcnn_reg_loss = torch.Tensor((cfg.rcnn_head_number)).zero_().cuda()
764     for i in range(cfg.rcnn_head_number):
765         rcnn_cls_loss, rcnn_reg_loss = \
766             rcnn_loss(self.rcnn_logits[i], self.rcnn_deltas[i], self.rcnn_labels[i], self.rcnn_targets[i], float(self.rcnn_deltas[i].std()), c
767             self.rcnn_cls_loss[i] = rcnn_cls_loss
768             self.rcnn_reg_loss[i] = rcnn_reg_loss
769
770     ## self.mask_cls_loss = Variable(torch.cuda.FloatTensor(1).zero_()).sum()
771     self.mask_cls_loss = \
772         mask_loss( self.mask_logits, self.mask_labels, self.mask_instances, cfg.mask_focal_loss )
773
774     self.total_loss = self.rpn_cls_loss + self.rpn_reg_loss \
775         + torch.mean(self.rcnn_cls_loss) + torch.mean(self.rcnn_reg_loss) \
776         + self.mask_cls_loss
777
778     return self.total_loss
```

Python

Pyth~~on~~ic

ardigen

```
self.feature_net = FeatureNet(cfg, 3, feature_channels, non_local_block=self.non_local_block)
self.rpn_head    = RpnMultiHead(cfg, feature_channels)
self.rcnn_crop   = nn.ModuleList()
self.rcnn_head   = nn.ModuleList()
for i in range(cfg.rcnn_head_number):
    self.rcnn_crop.append(CropRoi(cfg, cfg.rcnn_crop_size))
    self.rcnn_head.append(RcnnHead(cfg, crop_channels))
self.mask_crop   = CropRoi  (cfg, cfg.mask_crop_size)
self.mask_head   = MaskHead (cfg, crop_channels)
```



ardigen

Artificial Intelligence & Bioinformatics
for Precision Medicine

ardigen.com

Dawid Rymarczyk

e: dawid.rymarczyk@ardigen.com