UKRAINIAN CATHOLIC UNIVERSITY

BACHELOR THESIS

# Weakly-supervised individual human cell classification

*Author:*
Danylo KOLINKO

*Supervisor:*
Igor KRASHENYI

*A thesis submitted in fulfillment of the requirements*
*for the degree of Bachelor of Science*

*in the*

Department of Computer Sciences
Faculty of Applied Sciences


Lviv 2021

# Declaration of Authorship

I, Danylo KOLINKO, declare that this thesis titled, "Weakly-supervised individual human cell classification" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

*"Simple ingredients can be used to make elegant dishes with just a little extra attention to detail."*

Marcus Samuelsson

UKRAINIAN CATHOLIC UNIVERSITY

Faculty of Applied Sciences

Bachelor of Science

**Weakly-supervised individual human cell classification**

by Danylo KOLINKO

# *Abstract*

Proteins have an essential role in all cellular processes and greatly determine their functional properties. With the development of microscopy, we have the ability to study localization patterns of proteins within a cell group. Differences in the location among cell populations are called single-cell variability (SCV). SCV accounts for a difference in reactions between cells. Studying this connection may aid the development of treatment for cancer [23]. To this end, we develop a cell classifier trained to determine the correspondence of protein location to cell organelles in a single cell. A classifier is trained with image-level labels, so the training process is weakly supervised.
Code is publicly available here.

# *Acknowledgements*

I am grateful to my supervisor Igor Krashenyi for his time and thorough mentorship.

Thanks to Ostap Viniavskyi, Yuriy Yeliseev, and Andriy Prysiazhnyk, who were always open for questions, discussions, and help.

I want to thank Olena Domanska for her support.

I am obliged to my family for always encouraging me towards education.

# Contents

# List of Figures

# List of Tables

# List of Abbreviations

| | |
|---|---|
| **SCV** | **S**ingle **C**ell **V**ariability |
| **HPA** | **H**uman **P**rotein **A**tlas |
| **CNN** | **C**onvolutional **N**eural **N**etworks |
| **AP** | **A**verage **P**recision |
| **mAP** | **m**ean **A**verage **P**recision |
| **SoTA** | **S**tate **o**f **T**he **A**rt |
| **CAM** | **C**lass **A**ctivation **M**ap |
| **GAP** | **G**lobal **A**verage **P**ooling |
| **IoU** | **I**ntersection **o**ver **U**nion |
| **BCE** | **B**inary **C**ross **E**ntropy |
| **TTA** | **T**est **T**ime **A**ugmentation |

*Dedicated to my family.*

# Chapter 1

# Introduction

## 1.1 Context

Advancement in high-resolution microscopy allowed the production of large volumes of biological imaging data. Many biological structures can be analyzed now, starting from tissues and zooming up to cellular and subcellular structures. One of the phenomena studied is protein localization on a cell and subcell level. There has been shown a high degree of complexity in cells regarding the localization of proteins. Approximately half of the human proteins are stored in multiple cell compartments called organelles.

Another phenomenon connected to protein localization is single-cell variability (SCV): which expresses that genetically identical cells sometimes show different localization patterns of a distinct protein. More specifically, the difference in organelles containing the protein. The connection between this phenomenon and the functional properties of the cell is not well understood while understanding this relation might be beneficial for drug development.

## 1.2 Problem

A large amount of high-resolution microscopy data indeed provides lots of insight. However, it requires expert-level accuracy and long hours of human expert work to provide accurate annotation. The research group Human Protein Atlas (HPA) that aims to map human proteins in cells and tissues, developed a machine learning model that could classify images with mixed protein patterns with the F1-score of 0.47, while measured expert F1 was around 0.73 [19]. To this end, HPA and Kaggle (machine learning competition platform) held a competition to develop better image classification models. Developed models classified images for the localization of certain proteins in 28 different organelles. The best solution scored 0.593 (F1). However, performance drops with multi-label images; the score is significantly higher for single-cell images [19].

Another problem is the abovementioned single-cell variability. Image-level classification has little use in images where SCV is expressed since protein localization may vary significantly among cells. Therefore in such cases, cell-level classification is required.

It would take substantial effort to comprise a cell-level labeled dataset. However, we may use the existing image-level labeled dataset, localize each cell and perform weakly supervised training to obtain a classification for each cell, which is the scope of this paper.

## 1.3 Data

We use the subset of the HPA dataset that the research group noted as high in single-cell variability. Each sample in the data consists of four files. Each file represents a specific subcellular protein patterns represented by the sample obtained with different filter, specifically:

- Microtubule channel: red;
- Nuclei channel: blue;
- Endoplasmic Reticulum (ER) channel: yellow;
- A protein of interest: green;

Three first channels may be used to define the cell structure, while the fourth is used as a reference channel for classification. Proteins of interest are undisclosed. For each sample, annotation is provided, noting a subset of 19 essential organelles that confine protein of interest in some/all cells on the image. The dataset contains 23609 samples with 21806 training images with annotation and the rest withheld as a test subset. The test is comprised out of two parts:

1. the test set with higher SCV than training subset: 559 samples;
2. the test set with higher SCV than first test subset: 1244 samples;

In the training subset, there are 10508 images with a single label and the rest with multiple. Also, there are 432 counts of unique labels in this part of the dataset.

## 1.4 Our approach

In this thesis, we explore the possibility of building a deep learning-based instance segmentation model for cells, followed by the classification of instances based on weak annotations. In this thesis, we focus on a single model approach rather than an ensemble of several models. We aim to obtain cell-level classification by merging class activation maps and cell instance segmentation masks.

## 1.5 Goal

The goal of this thesis is to develop a tool for classifying protein localization patterns in human cells. Such a model is trained to be robust to single-cell variability. We aim to develop a deep learning-based solution train with weak supervision. We also evaluate proposed solutions and provide a comparison among themselves and other published solutions.

## Structure of the thesis

### Chapter 2. Biological Background

This chapter contains information on general cell structure, protein creation. Also, this section explains how microscopic data is obtained.

### Chapter 3. Literature review

This chapter provides an overview of advancements in the image segmentation domain of computer vision, and the image classification approaches for protein localization.

### Chapter 4. Background and methodology

Here we introduce the dataset and metrics used for this problem along with building blocks for our solution.

### Chapter 5. Experiments

In this chapter, we provide a detailed overview of our solution transformation. This chapter also provides a detailed discussion for motivation regarding each step.

### Chapter 6. Results

Outcomes of the most successful experiments are presented.

### Chapter 7. Overview of alternative solutions

Here we present and discuss alternative solutions for this problem with their results.

### Chapter 8. Conclusions and Future work

This chapter contains conclusions on the topic. We discuss insights from our solution along with the alternatives. Based on this discussion, future work is motivated.

### Chapter 9. Implementation details

Here we provided system and hardware settings for our experiments.

# Chapter 2

# Biological background

## 2.1 Cell structure

A cell is the smallest biological integral unit, structured and endowed with defined functions. Organelles are small subcellular structures that perform specific functions. Examples of such functions are energy production or protein synthesis. Organelles are made of macromolecules: chains of smaller organic molecules. Protein is an example of a macromolecule. The cell maintains its shape and structure with cytoskeleton: a network of protein fibers. A nucleus is a center of cell control that contains and maintains genetic information. A nucleus is enclosed in a cell with a material called cytoplasm.

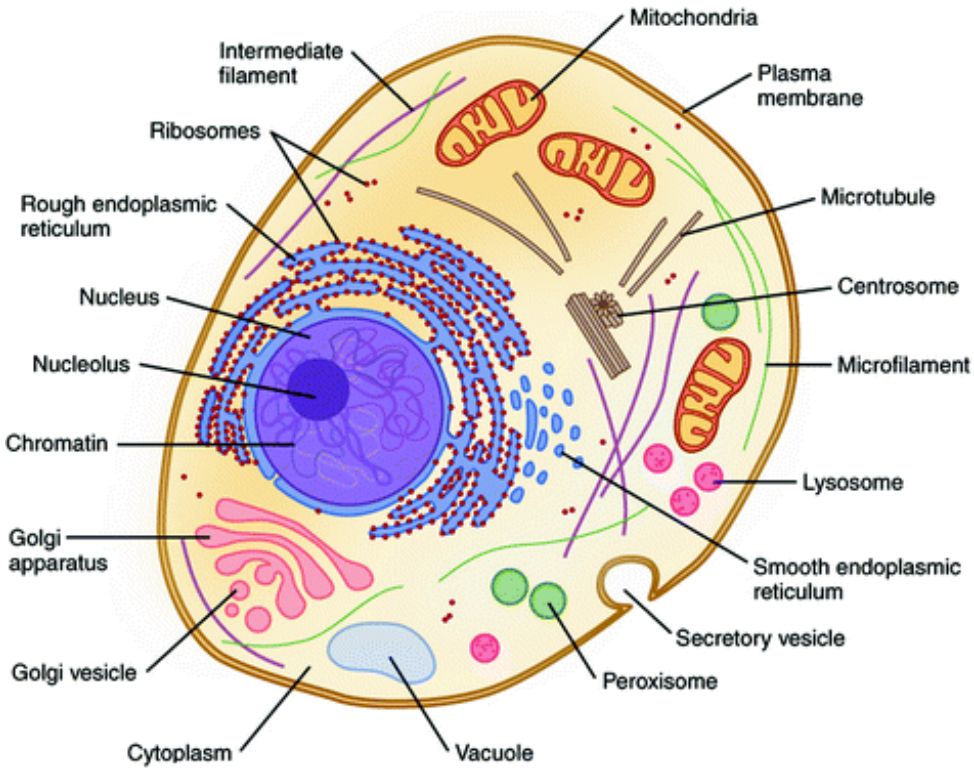



FIGURE 2.1: Cell structure with main organelles. Figure taken from [4].

**Protein Types and Functions**

| Type | Examples | Functions |
|---|---|---|
| Digestive Enzymes | Amylase, lipase, pepsin, trypsin | Help in food by catabolizing nutrients into monomeric units |
| Transport | Hemoglobin, albumin | Carry substances in the blood or lymph throughout the body |
| Structural | Actin, tubulin, keratin | Construct different structures, like the cytoskeleton |
| Hormones | Insulin, thyroxine | Coordinate different body systems' activity |
| Defense | Immunoglobulins | Protect the body from foreign pathogens |
| Contractile | Actin, myosin | Effect muscle contraction |
| Storage | Legume storage proteins, egg white (albumin) | Provide nourishment in early embryo development and the seedling |

FIGURE 2.2: Proteins. Figure taken from [6].

## 2.2 Protein in the cell

Protein is a type of biomolecule in a cell. Proteins often can act as a catalyst for cellular processes or regulators for different physiological functions. In the figure 2.2 represented few examples of proteins and functions they aid.

### 2.2.1 Protein production

Cells produce proteins based on the DNA genes contained in the nucleus. Specific cell enzyme attaches to DNA to produce messenger RNA out of DNA instructions: transcription of the DNA. Then additional processing of mRNA follows that involves removing and adding sections of RNA. After this RNA is transported to cytoplasm messenger, RNA gets translated into a chain of amino acids that folds into complex 3D-shaped protein.

## 2.3 Protein detection

There are different ways to observe proteins in a cell. In this dataset, the methods of observing proteins were immunocytochemistry and microscopy.

### 2.3.1 Immunocytochemistry

This chapter explains the samples in the dataset were obtained. The first step of immunocytochemistry is to seed the cell on a glass slide and leave for a particular incubation time.

The second step of immunocytochemistry is immunostaining. The steps of this process are:

1. Fixation: preserves the structural state of the protein and location in the cell;
2. Permeabilization: puncturing the cell membranes with various chemicals enabling the antibodies to cross the cell membranes;
3. Antibody Incubation: the antibody binds to the desired protein;

Next step is immunofluorescence. This process visualizes the protein of interest with a fluorescent tag using antibodies specific to the protein of interest. Then images are produced with fluorescent microscopy. Four channels correspond to specific wavelengths with which waves excite proteins of interest.

## 2.4 Organelles and corresponding classes

### 2.4.1 Nucleus: blue channel

A nucleus is an organelle that contains a cell's genome and maintains the integrity of its genes. It is considered a control unit of the cell. A nucleus is usually found in the center of the cell. A nucleus can be found with the signal in the blue channel. Here we may see the overview of organelles belonging to a nucleus and therefore stained on the blue channel.

#### Class 0. Nucleoplasm

Nucleoplasm is a fluid inside the nucleus. Nucleoplasm staining is nucleus staining without nucleoli (Class 2) staining (if present).

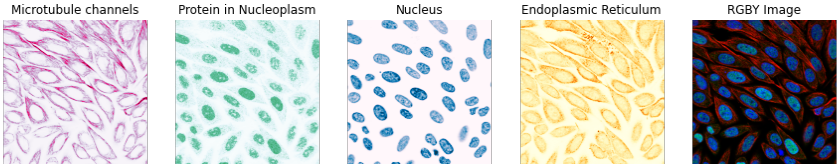



FIGURE 2.3: Protein of interest in Nucleoplasm.

#### Class 1. Nuclear membrane

A nucleus is bounded with a nuclear membrane; therefore, the membrane can be located in a thin circle around the nucleus. In some of the cells, the membrane folds may be located inside the nucleus.

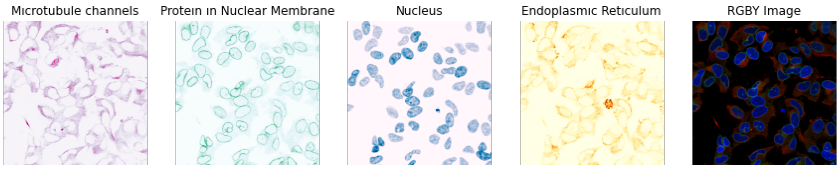



FIGURE 2.4: Protein of interest in Nuclear Membrane.

#### Class 2. Nucleoli

A nucleolus is a subcompartment of a nucleus where the synthesis of ribosomes happens. Nucleoli can be seen as circles in the nucleoplasm. There might be several nucleoli in a nucleus.

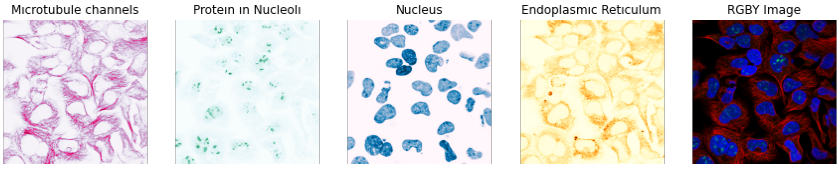



FIGURE 2.5: Protein of interest in Nucleoli.

### Class 3. Nucleoli fibrillar center

Nucleoli fibrillar center is a part of the nucleoli—proteins between the border of this and other nucleolus components responsible for transcribing DNA to pre-rRNA [6]. Usually appears as a dotted cluster in a nucleolus.

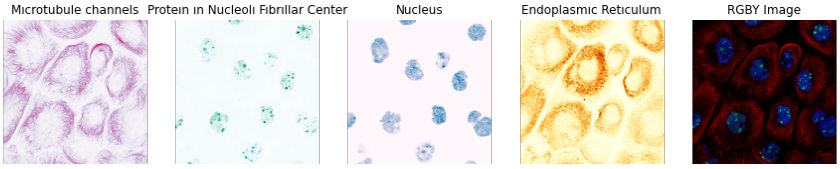



FIGURE 2.6: Protein of interest in Nucleoli fibrillar center.

### Class 4. Nuclear speckles

Nuclear speckles are clusters inside a nucleoplasm. As described above, pre-mRNA has to be processed before translation. One way pre-mRNA is processed by removing extra pieces of RNA that do not belong in the final mRNA molecule. Subcompartments of the nuclear speckles responsible for this and for reconnecting the coding parts together. This organelle appears as a spot or cluster in various shapes.-

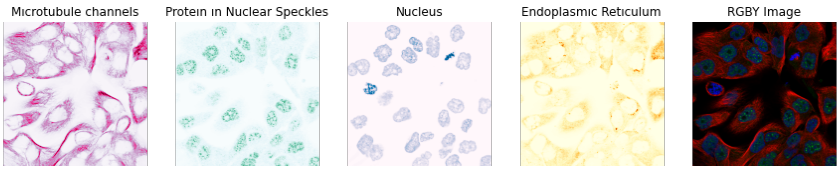



FIGURE 2.7: Protein of interest in Nuclear speckles.

### Class 5. Nuclear bodies

Nuclear bodies are positioned in the nucleoplasm. These bodies include various proteins and molecules of RNA. The type of reactions that occur here is the abovementioned RNA processing and DNA repair. Their shapes and number are irregular as well.

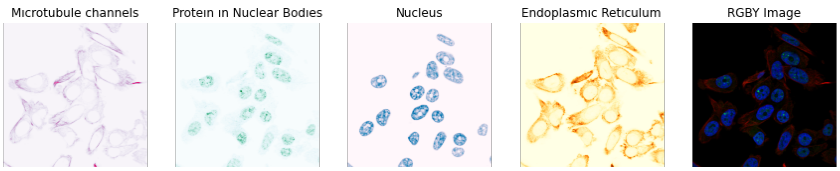



FIGURE 2.8: Protein of interest in Nuclear bodies.

### 2.4.2 Endoplasmic reticulum: yellow channel

This organelle is visible on the yellow channel.

#### Class 6. Endoplasmic reticulum

An endoplasmic reticulum is a network that spans the cytoplasm and nuclear membrane. The endoplasmic reticulum (ER) is recognized by network-like staining in the cytosol (Class 16), usually stained with decreased intensity from the nucleus to the edges of the cell.

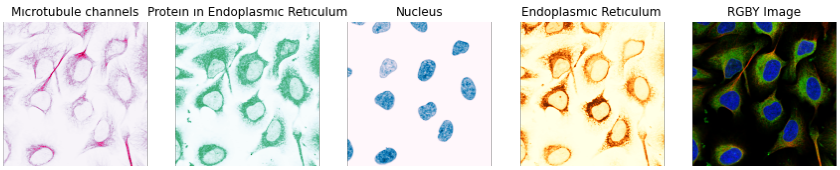



FIGURE 2.9: Protein of interest in Endoplasmic reticulum.

### 2.4.3 Microtubules: red channel

This organelle is visible on the red channel.

#### Class 10. Microtubules

A microtubule is the first part of the cytoskeleton mentioned above. Therefore microtubules take part in maintaining cell shape and are responsible for the cell's mechanical support. Microtubules can grow and shrink fast. Usually, they appear as thin strings throughout the cell.

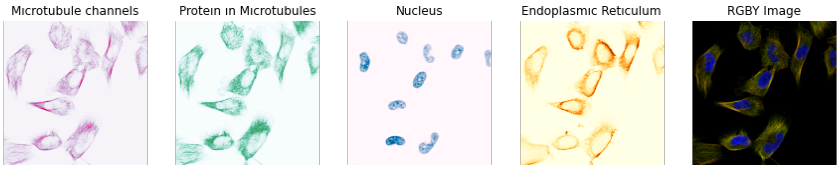



FIGURE 2.10: Protein of interest in Microtubules.

#### Class 11. Mitotic spindle

Mitotic spindles are particular microtubules. When the cell divides, a process called mitosis, spindles facilitate chromosome movement [6]. During mitosis, the shape of the organelle changes. We consider this organelle the most beautiful.

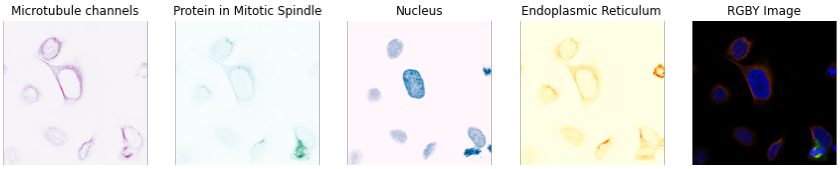



FIGURE 2.11: Protein of interest in Mitotic spindle.

### Class 12. Centrosome

A centrosome is the center of the microtubule organization that takes part in mitosis. The location of the centrosome is determined by the cell cycle. The centrosome is usually viewed as distinct staining at the origin of the microtubules, near the nuclear membrane.

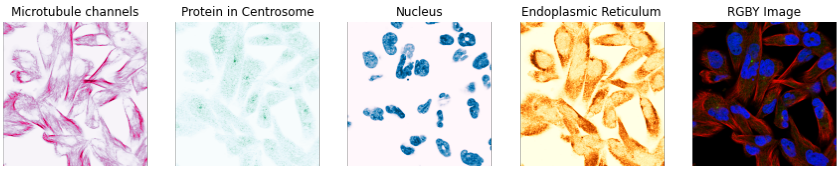



FIGURE 2.12: Protein of interest in Centrosome.

## 2.4.4 Cell level: multiple channels

Here are organelles that span across all of the channels.

### Class 7: Golgi apparatus

Golgi apparatus is a large organelle involved in tagging the transport structures of proteins to assist their transportation to the correct cell region [6]. Golgi apparatus located near the nucleus. It is folded into a shape similar to a ribbon.

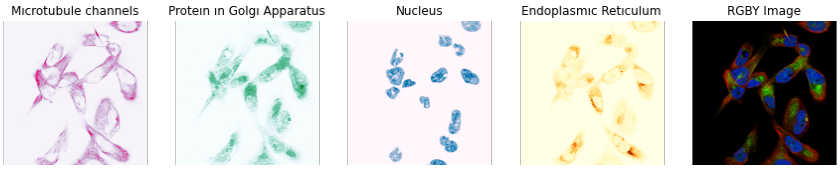



FIGURE 2.13: Protein of interest in Golgi apparatus.

### Class 8: Intermediate filaments

Intermediate filaments are the second component of the cytoskeleton; therefore, they take part in maintaining cell shape and are responsible for cell's mechanical support, but also intercellular organization and cell signaling. By shape, they are similar to microtubules and located inside cytosol (Class 16).

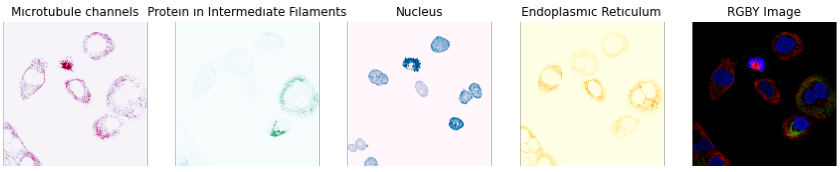



FIGURE 2.14: Protein of interest in Intermediate filaments.

### Class 9: Actin filaments

Actin filaments are another part of the cytoskeleton. Actin filaments can be found close to the edges of the cells and organized in long clumps or branched networks.

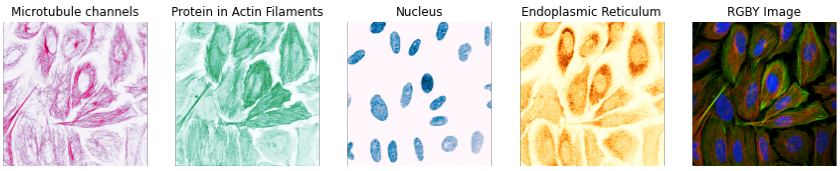



FIGURE 2.15: Protein of interest in Actin filaments.

### Class 13: Plasma membrane

The plasma membrane bounds and protects the cell; proteins take up to half of the total membrane mass. The plasma membrane can be found at an edge of the cell.

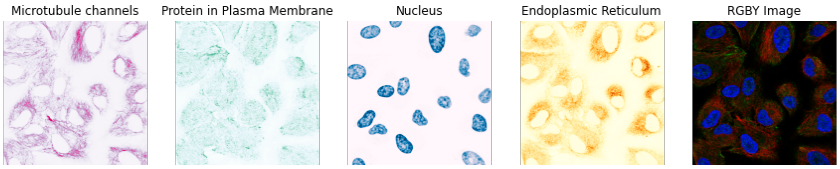



FIGURE 2.16: Protein of interest in Plasma membrane.

### Class 14: Mitochondria

Mitochondria are small rods in the cytosol (Class 16), spans mainly along microtubules. Mitochondria is responsible for energy production and tracking the cellular life cycle.

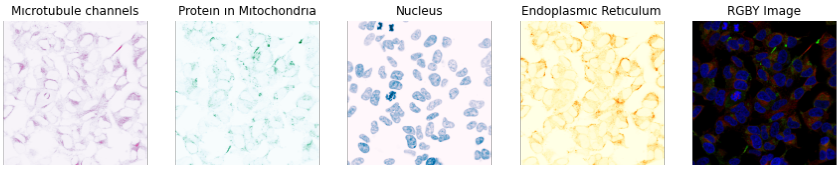



FIGURE 2.17: Protein of interest in Mitochondria.

### Class 15: Aggresome

Aggresome formed with the group of disrupted proteins. This organelle can be located in the cytosol near the nucleus and has dense staining.

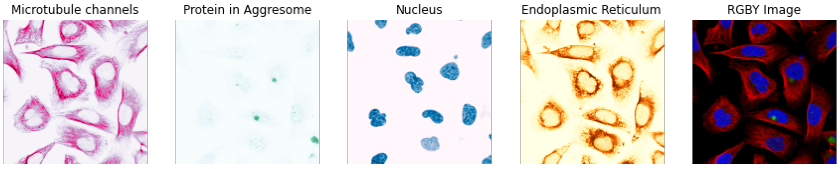



FIGURE 2.18: Protein of interest in Aggresome.

### Class 16: Cytosol

The cytosol takes place among the plasma membrane to the nuclear membrane. It is a part fluid and part molecule structure that surrounds other organelles. It can

appear smooth or granular, and the staining is often stronger close to the nucleus. The cytosol is a region where many cellular processes occur. Its staining is evenly spread throughout the cell.

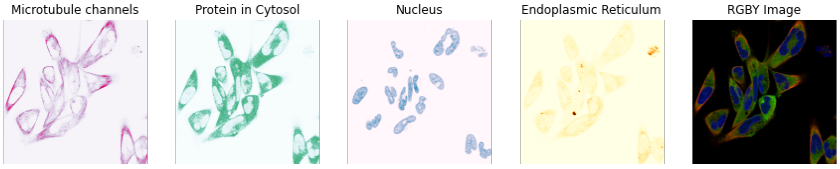



FIGURE 2.19: Protein of interest in Cytosol.

**Class 17: Vesicles and punctate cytosolic patterns**

This class includes small circular compartments in the cytosol. They rapidly change form and number in response to the environment. Usually, they have a circular appearance.

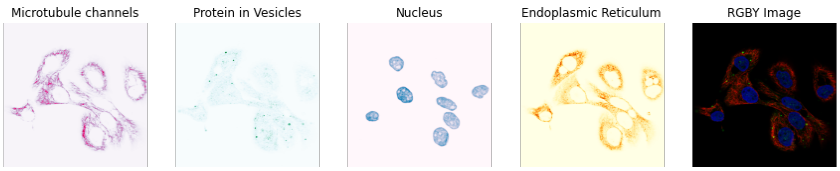



FIGURE 2.20: Protein of interest in Vesicles and punctate cytosolic patterns.

**Class 18: Negative**

This class includes negative stainings (no staining) and unspecific patterns; if most of the cell is out of the image, it is considered negative.

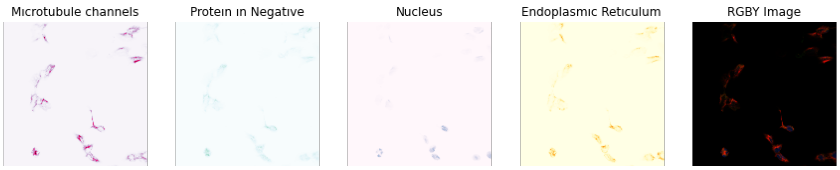



FIGURE 2.21: Protein of interest in Negative.

### 2.4.5 SCV and multi-localization example

Single or multiple organelles may contain a protein in a cell. Any combination of patterns can be possible for a single cell. There is no other pattern assigned if a Negative label is present. All cells will inherit the labels from image-level labels for some images, but where SCV occurs, they do not.
Here is an example of multi-localized protein. In this image, in all of the cells, protein resides in the nucleoplasm, nucleoli fibrillar centers, and mitochondria.

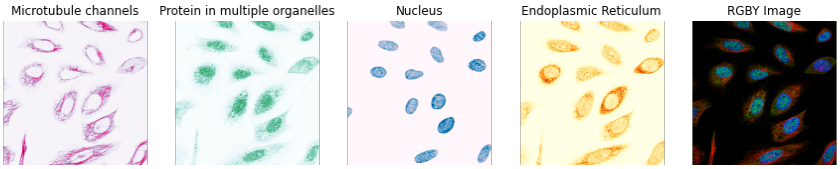



FIGURE 2.22: Protein of interest in Nucleoplasm, Nucleoli fibrillar centers, and Mitochondria in all of the cells.

Regarding SCV, here is a beautiful example of this phenomenon. Protein resides in different organelles in different cells. The classes present are Nucleoplasm, Nucleoli, Intermediate filaments, and Cytosol.

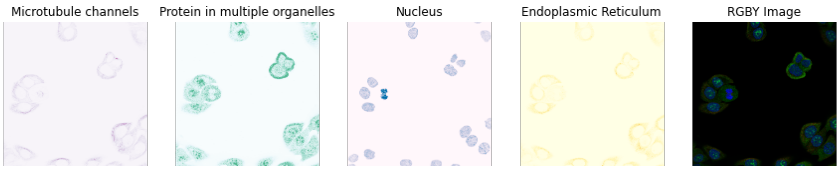



FIGURE 2.23: Single Cell Variability example.

# Chapter 3

# Literature review

As proposed by Human Protein Atlas research group, the problem can be considered instance segmentation with weakly supervised classification. Since this problem was posed recently, works related to this particular task are yet to be published. However, we will split our review into a summary of advancements in the field of image segmentation with its subfield of weakly supervised image segmentation and a summary of best solutions for protein pattern classification to gather valuable insights.

## 3.1 Semantic segmentation

Semantic segmentation is a task of pixel-wise classification. It has recently found numerous applications. In robotics and autonomous driving, it is applied to more finely define space for machine perception. In the photo and video editing, it is used to substitute costly manual work and avoid unwanted artifacts. In the medical domain, it is used to detect tumors, various tissues and formations.

The most notable solutions in semantic segmentation are based on Fully convolutional networks [24]. This approach produced segmentation masks by harnessing capabilities of the VGG network that was a SoTA architecture for image classification at the time [26]. VGG was used as a feature extractor by removing final fully-connected layers, and extracted features were upscaled to the image size followed by 1x1 convolution to predict the final segmentation mask.

Such a strong upsampling was suffering from various artifacts and insufficient resolution. SegNet architecture was developed by Badrinarayanan et al. [2] to improve resolution with upsampling decoder and skip connections for pooling indices. Noh et al. [18] addressed the same issue by introducing a deconvolutional decoder with a fundamental building block of transpose convolution to upsample feature maps more precisely. However, this method was suffering from "checkerboard artifacts": stridden patterns caused by an uneven amount of convolution outputs summed across feature maps.

To avoid artifacts and maintain higher resolution, Ronneberger et al. [20] introduced UNet using interpolation for smooth upsampling and skip connections for feature maps. Skip-connections are used to transfer spatial information across a network better. This network was introduced for cell segmentation, it is widely used in the biomedical domain now. PSPNet was proposed by Zhao et al. [35] that used feature pooling at different scales and convolving them, followed by feature upsampling to image size and concatenation across scales. Concatenated maps convolved together to aggregate information from all of the scales.

A similar idea is used by Chen et al. [5] in DeepLab V3 that also applied dilated convolutions to expand a receptive field, introducing more of a global context into architecture. Lin et al. [16] introduced Feature Pyramid Network to incorporate different scales to training by predicting interpolated masks at all spatial levels.

## 3.2 Instance segmentation

Here are three common approaches:

- FCN based[24]: train for semantic segmentation and then divide into instances;
- RCNN based[8]: start with generating instance proposals and classify them;
- Proposal free: train directly by predicting boxes for each spatial location;

For FCN based instance segmentation, TernausNetV2 was introduced by Iglovikov et al. [13]. TernausNetV2 is an extension of UNet with WideResNet [34] as an encoder; it incorporates boundary class into the training. With predicted boundary class, authors use watershed transformation to split masks into the instances. A possible variation is introduced with Deep Watershed Transform by Bai et al. [3] that with segmentation incorporates watershed energy prediction, that is then used to split masks up.

For RCNN branch solutions are based on RCNN object detector introduced by Girshick et al. [8]. RCNN extracts region proposals and then predicts whether an object is indeed present and, if present, what class does it belong to. Based on this detection method, a widely used MaskRCNN architecture was introduced [10], incorporating an additional head for pixel-wise mask prediction for each object.

For a proposal-free approach, a fast alternative was proposed by Lee et al. [15] to achieve real-time instance segmentation named CenterMask. This network bases on the one-stage (no proposals required) anchor-free detector FCOS developed by Tian et al. [29] which for each spatial location predicts centeredness for its assumed object, classification, and box location; these predictions are made across all of the spatial levels in an FPN-alike manner. CenterMask was built by extending FCOS with a mask head that performs pixel-wise predictions for detected objects with aggregation and alignment of predictions at multiple scales.

## 3.3 Weakly supervised semantic segmentation

Most of the SoTA solutions for weakly supervised semantic segmentation are based on the class activation maps introduced by Zhou et al. [36]. CAMs are usually produced by using CNN backbone for feature extraction and perform global average pooling to receive a feature vector. With a linear layer on top, classes for an image are predicted. Then using weights from the last linear layer, we can form an activation map for each class by the weighted sum of features from the backbone. With this approach to classification, feature maps are trained to excite at discriminative locations for each class since spatial information is retained in a CNN.

Since weights from the linear layer used in the weighting of each feature map for contribution to a class map are simply a gradient of this class for each feature map, Selvaraju et al. [22] proposed a generalized version called Grad-CAM that calculates weights for a weighted sum as a propagated gradient from classification by any task-specific network.

The issue arises when several objects or their features are most discriminative and located at a specific part of the class; thus, a network overfits CAMs to excite one object or these discriminative locations. To battle this issue, PuzzleCAM was introduced by Jo et al. [14]. PuzzleCAM performs two forward paths to predict class on image-level and patches formed by the same image split into several pieces. Except for the image-level CAM, another CAM is produced by reassembling CAMs from patches to the corresponding location. During training, reconstruction loss term is progressively added to achieve a smooth CAM map across each class.

## 3.4 Human Protein Atlas 2018 challenge

Analysis of the Human Protein Atlas Image Classification competition by Ouyang et al. [19] is another valuable piece of literature for this problem. HPA 2018 Classification Challenge aimed to solve the multi-label classification problem with highly imbalanced classes. The classes present were the same or subsets of the classes at hand. Understanding best fitting strategies may be beneficial for developing a weakly supervised classification solution for very similar data.

Most teams developed deep learning-based solutions using a variation of Resnet [9], Densenet [12], or Inception [27] architectures as a backbone. Teams commonly used binary cross-entropy to handle multi-label prediction. Since data was imbalanced, teams often used multi-label stratification to achieve good Train-Val split and focal loss to focus on the hard-to-classify/ insufficiently represented classes. Some teams enriched their data with other datasets released by the HPA group. The top four solutions received a significant boost from four different strategies:

1. image retrieval + arcface [7] loss;
2. advanced data processing;
3. automatic data augmentation search;
4. model ensembling;

# Chapter 4

# Background and Methodology

The following methodology was developed as a solution to a Human Protein Atlas - Single Cell Classification challenge hosted jointly by Kaggle and HPA group. This was a code competition meaning that solutions had to be submitted as runnable inference code for test data.

Test data was represented with two subsets. First subset metric scores were visible immediately after the inference was completed; hence subset was called the public test set. Second subset metric scores were computed after the end of the competition; hence subset was named the private test set. Private test set scores are used as final scores for the competition.

## 4.1 Dataset

The dataset released by HPA consists of 21806 high-resolution images. Resolutions represented in the dataset are 2048x2048, 1728x1728, 3072x3072. As mentioned above, each sample in the dataset consists of four files corresponding to different filters. We viewed the data as a four-channel image dataset, with red, green, blue, and yellow (RGBY) channels corresponding to its filters.

Exploring the dataset, we can see that most of the training set consists of multiple localization patterns.

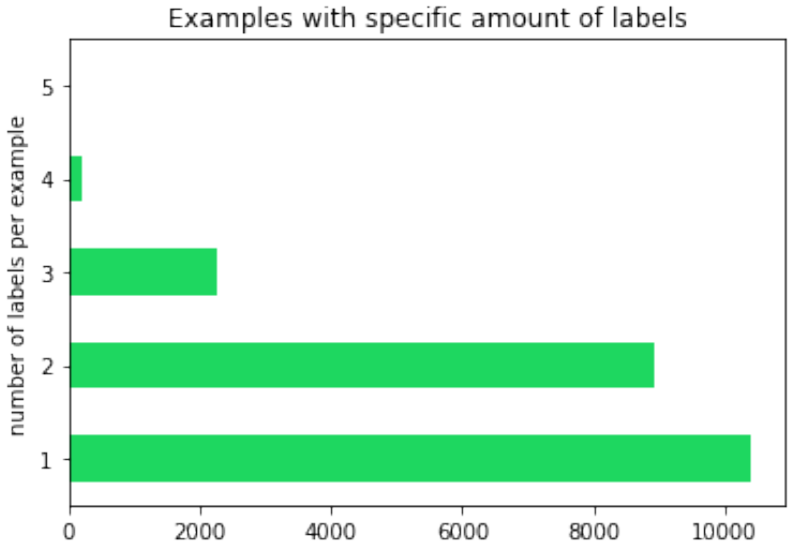



FIGURE 4.1: Number of labels in dataset.

Each sample has an annotation providing information on where the protein of interest (visible on the green channel) resides. Location is characterized by 18 cell organelles and a negative class. The dataset is explicitly crafted to contain multi-localization patterns and CSV. Here we see the distribution of labels with its unique and full count:

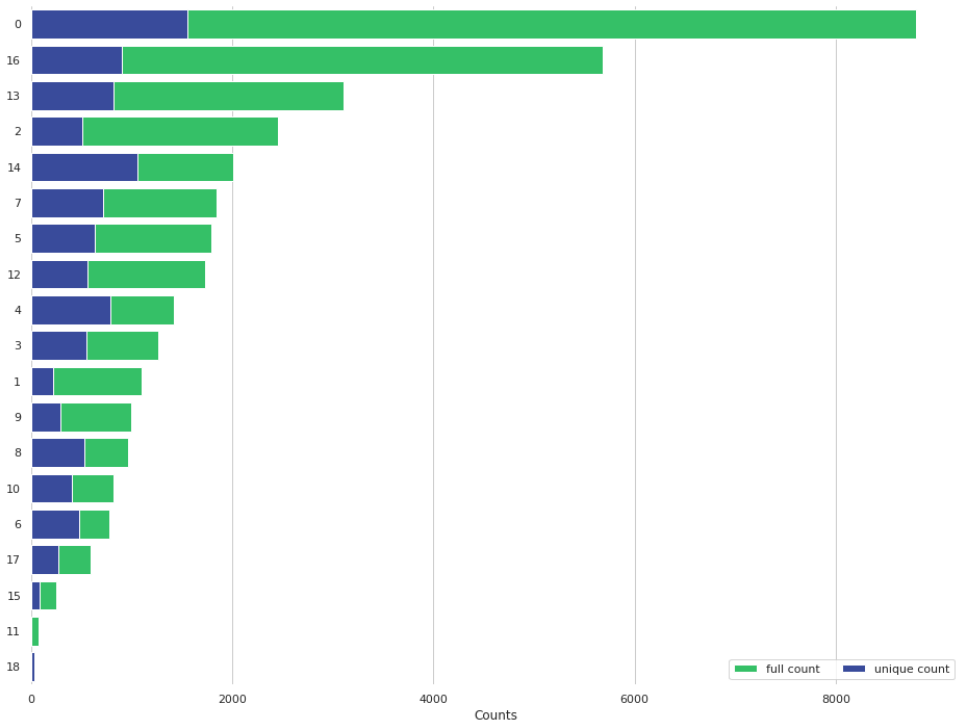



FIGURE 4.2: Distribution of labels.

Here are few samples from provided data:

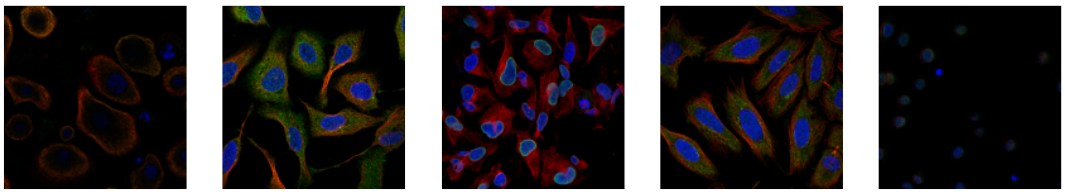


FIGURE 4.3: HPA data samples.

## 4.2 Train-Val split

It is crucially important to reflect all of the classes equally in the validation and training dataset with a multi-label and highly imbalanced setting. In other words, we aimed for a stratified split of our data. Sechidis et al. [21] proposed an algorithm for multi-label stratification that now has open-source implementation. This MultilabelStratifiedKFold algorithm was employed to split data into 80/20 ratio.

## 4.3 Metric

As a metric for the model performance, the HPA group chose to use mean Average Precision. A prediction for each image contained mask predictions with class probabilities. Hence, mAP was computed based on the matching of predicted instance masks with ground truth.

### 4.3.1 Average Precision

Average Precision metric is used to evaluate the quality of classification model predictions; it combines recall and precision for ranked predictions. AP can be calculated as the area under the precision-recall curve, or if simplified, can be either calculated for the p-r curve with some fixed recall levels or for the p-r curve interpolated to maximum successive precision level.

The precision-recall curve plots out how precision and recall are distributed against sorted predictions. In other words, how precision and recall change based on predictions confidence. At each level, precision and recall are calculated for sorted predictions being cut off at a particular confidence threshold.

$$\mathrm{AP} = \Sigma \left(r_{n+1} - r_n\right) p_{\text{interp}} \left(r_{n+1}\right)$$
$$p_{\text{interp}} \left(r_{n+1}\right) = \max_{\tilde{r} \geq r_{n+1}} p(\tilde{r})$$

### 4.3.2 Jaccard index

Jaccard index is an intersection-over-union coefficient for discrete predictions, it is commonly employed in evaluating image segmentation results since it provides appropriate relevance to small objects and false negatives compared to per-pixel losses.

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

### 4.3.3 Mean Average Precision

Finally, mAP is calculated as the mean of average precision across classes or/and across different IoU detection levels. In this competition threshold for matching was at 0.6 IoU and mAP was calculated across classes.

$$mAP = \frac{1}{n} \sum_{k=1}^{k=n} AP_k$$

## 4.4 Segmentation model

HPA research group provided cell and nuclei segmentation models accompanying this dataset, which prove to be valuable in our research. The joint model is named HPACellSegmentation.

Segmentation models, though exact implementation details are not provided, use the TernausNetV2-alike approach. Both models are based on UNet architecture. The nuclei model on inference uses a one-channel image with a blue (nuclei) filter, while the cell model infers based on three-channel image RBY channels in our data. Each model has two target classes: nuclei/cell class and borders.

Later both can be post-processed separately with Otsu thresholding and watershed transform and matched to obtain unified labels for each cell. HPA hosts suggested that the tool provides masks with at least 0.6 IoU with ground truth for 90% of the test data.

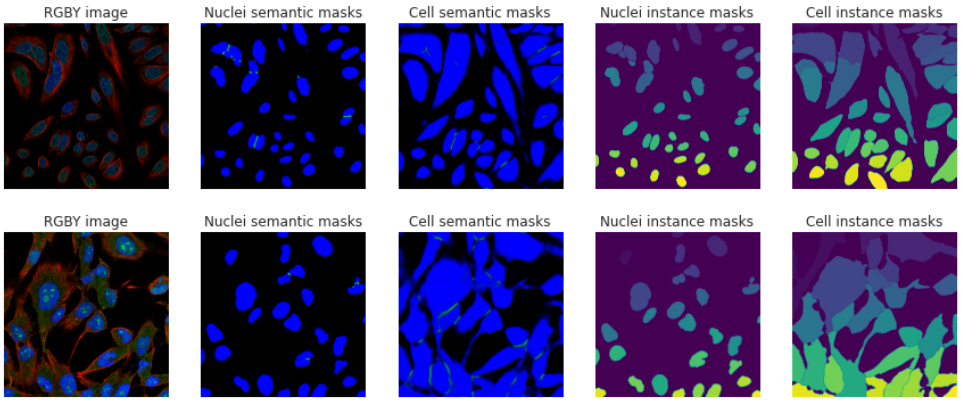



FIGURE 4.4: Output samples from HPACellSegmentations.

## 4.5 Preprocessing

As a preprocessing step, all of the images were cropped with a fraction of the width and height of the input image. Fraction is chosen randomly from [0.8, 1] interval. This crop was then resized into 512x512 resolution. Such training size was chosen following most of the teams from the HPA18 classification challenge. Another reason was to train efficiently under memory constraints.

All of the values were transformed to [0, 1] domain, dividing by maximum pixel value (255) and then normalized by dataset statistics to reflect sample differences better. Another normalization setting that was used in our experiments was instance normalization. Here instead of dataset parameters, we normalized each instance by its own statistics.

### 4.5.1 Augmentation pipeline

Augmentation is a technique to expand training data synthetically and prevent overfitting. The first step of the augmentation should be considered a random crop before image resizing. Excluding crop, augmentation was not set precisely for all of the experiments, but the following list of four groups includes augmentations for the best solution.

The first group contained random dropouts for masks. They are used to achieve smooth CAMs. Dropout was applied to masks obtained from the HPACellSegmentation. We were zeroing down all of the regions in the image corresponding to dropped-out masks. We aimed to reduce the focus of the produced activation maps on the least challenging cells for detection and span CAMs evenly.

The second group was augmenting data spatially , namely horizontal and vertical flips, transposing of the image, slight shifts, scaling, and rotations. Each transform was applied with a probability of 0.5. Also, one non-rigid augmentation was applied: an elastic transform described by Simard et al. [25]; this transformation was used with 0.2 probability.

The third and fourth groups follow each other containing the same transformations. These transformations alter color values. From each group, we sample one augmentation, thus with two stacked blocks, we end up with different combinations of color augmentations. Namely, this group includes per channel multiplicative noise, blur with a maximum size of the kernel 3x3, random brightness/contrast changes with the 0.25 change limit for both to original values.

Another augmentation we used in some of our experiments was a dropout of the channel.

## 4.6 Losses

A loss function is a function that estimates certain losses with respect to an objective; thus, minimization of these losses aims to improve model performance.

### 4.6.1 Cross entropy loss

A commonly used loss function for training classification and segmentation models is cross-entropy. Loss value is calculated by measuring the difference between true distribution $p(x)$ and estimated distribution $q(x)$ over discrete variable $x$ in the following way:

$$H(p,q) = -\sum_{\forall x} p(x)\log(q(x))$$

### 4.6.2 Binary cross-entropy loss

Binary adaptation of this loss is calculated for binary distribution of discrete variable x. BCE loss fits well for a multi-label problem by averaging binary losses for each class.

$$H_p(q) = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot \log\left(p\left(y_i\right)\right) + \left(1-y_i\right)\cdot\log\left(1-p\left(y_i\right)\right)$$

### 4.6.3 Focal loss

Focal loss is an adaptation to the standard cross-entropy criterion introduced by Lin et al. [17]. It adds a modulating factor to cross-entropy to prioritize training for poorly predicted samples.

$$\mathrm{FL}\left(p_{\mathrm{t}}\right) = -\left(1-p_{\mathrm{t}}\right)^{\gamma}\log\left(p_{\mathrm{t}}\right)$$

Where $\gamma \geq 0$ controls penalization level for easily predicted samples. The larger the gamma, the heavier is the penalization. Focal loss is commonly used for imbalanced datasets or refining predictions for challenging samples.

### 4.6.4 Lovasz Hinge loss

Lovasz loss is used for training segmentation models as an alternative for cross-entropy. It is crafted by Yu et al. [33] to optimize the Jaccard index with a Jaccard loss piecewise linear surrogate.

### 4.6.5 Regression losses

L1 and L2 losses are common choices for training regression models.

L1 loss or Least Absolute Deviation function

$$LAD = \Sigma_{i=1}^{n} |y_i - f(x_i)|$$

L2 loss or Least Squared Deviation function

$$LSD = \Sigma_{i=1}^{n} (y_i - f(x_i))^2$$

L2 loss is continuously differentiable across a rational domain, unlike L1 loss. L2 makes training more stable and allows for better gradient-based optimization.

## 4.7 Dilated convolution (2D)

Dilated convolution is a generalized version of a convolution introduced by Yu et al. [32]. It is applied to input with chosen dilation step $l$. It is a common technique to expand the receptive field of the network.

$$(F *_l k)(\mathbf{p}) = \sum_{\mathbf{s}+l\mathbf{t}=\mathbf{p}} F(\mathbf{s})k(\mathbf{t})$$

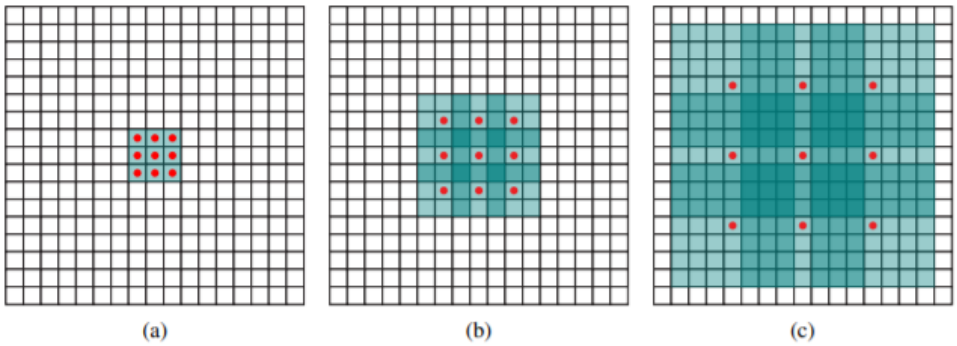



FIGURE 4.5: Dilated convolution with steps 1, 2 and 4. Table taken from [32].

## 4.8 Architecture families

### 4.8.1 ResNet

ResNet [9] stands for a residual network, the network build of residual blocks. Residual block is skip-connection enabled convolutional block referencing previous blocks through mentioned skip-connections. Skip-connections are built by appending as the final operation a sum of its input and output.
Such a strategy benefits training by preventing gradient vanishing in deep networks by providing additional paths for gradient flow through residual connections.

### 4.8.2 EfficientNet

EfficientNet family [28] with its eight versions was introduced to allow for better model scaling concerning performance. Authors employ neural architecture search to obtain this family, each with specifically calibrated depth, width, and resolution. For efficient scaling, authors suggest using compound factor $\phi$:
depth: $d = \alpha^\phi$
width: $w = \beta^\phi$
resolution: $r = \gamma^\phi$
s.t. $\alpha \cdot \beta^2 \cdot \gamma^2 \approx 2$
$\alpha \geq 1, \beta \geq 1, \gamma \geq 1$

Since FLOPS of a convolution is close to being proportional to $\alpha$, $\beta^2$, and $\gamma^2$, such condition is suggested to keep model FLOPS under $2 * *\phi$.

EfficientNet baseline architecture is build of a mix of inverted residual blocks and squeeze-and-excite blocks (described below). For this baseline, with a grid search, the authors found baseline parameters of $\alpha 1.2$, $\beta = 1.1$, $\gamma = 1.15$ satisfying conditions above. This baseline called EfficientNet-b0 offered an excellent trade-off between performance and computational complexity. Different compound factors $\phi$ were then used to receive versions b1-b7.

### 4.8.3 Inverted Residual Block

An inverted residual block is a residual block with a larger channel number inside the convolution block than at its input and output. This change to residual block was made for efficiency reasons since it drastically reduces the number of parameters compared to residual block.

Squeeze and Excite block SE block [11] is designed to achieve better representations using a channel-wise calibration of the output. The basic idea is the weighting output of the convolutional block. Weighting vector is produced by 'excitation' side-network from vector obtained from globally averaged input channels.

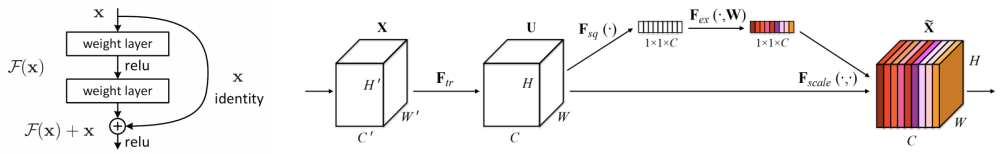



FIGURE 4.6: Residual and squeeze-and-excite blocks

## 4.9 CAM

Class activation maps were introduced for discriminative object localization. In this setup, a CNN-based feature extractor is applied to receive final feature maps. To these features, global average pooling is applied to receive vectors with channel number length. Class activation map for a particular class is computed through weighting features with weights from the final fully-connected layer. These weights identify in what ratio each map contributed to the score for a particular class.

The illustration from the paper provides a nice overview of this process.

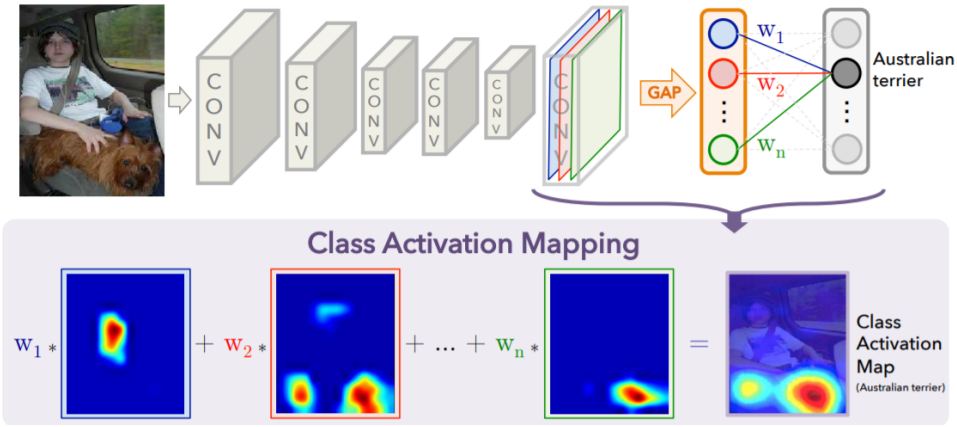



FIGURE 4.7: Class activation maps. Visualization taken from [36].

## 4.10 PuzzleCAM

PuzzleCAM pipeline was designed to improve CAM quality with minimal computational cost. The authors decided to achieve this by performing two forward paths through the feature extractor: one path with an image and one path with its patches. After patches are processed, their CAMs are placed according to the patch's original arrangement. Reconstruction loss is computed between image CAMs and patch CAMs to achieve better activation at different spatial levels. Reconstruction term is added to the final loss, progressively increasing its weight during training.

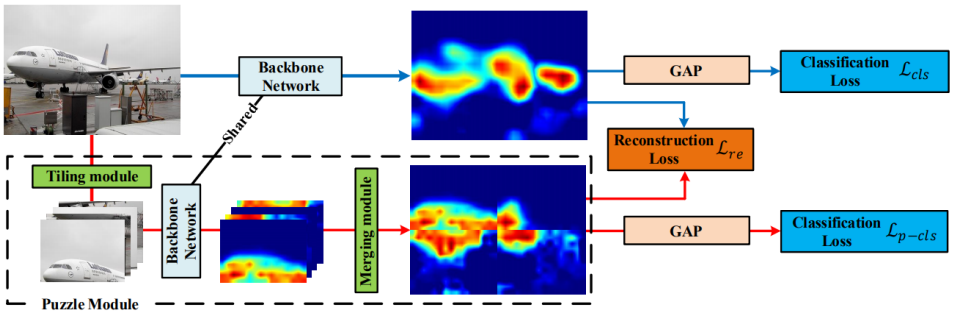



FIGURE 4.8: PuzzleCAM pipeline. Visualization taken from [14].

## 4.11 Additional data

HPA group also provided access to a larger cell dataset. This dataset has fewer multi-label instances than competition data. However, we used it at later stages of experimentation to improve the score on the worst predicted classes. At the mentioned stage of experiments, hard-to-learn classes were considered: Endoplasmic Reticulum, Actin Filaments, Mitotic Spindle, Cytosol, Vesicles, and Negative. We expanded training data with subsets from provided additional data. This subset contained 2429 samples. From additional public data, we pooled all of the samples concerning Mitotic Spindle and Negative. Also, we added all of the instances with a label count larger than three with at least two hard-to-learn classes.

# Chapter 5

# Experiments

This section will reflect an evolution of our solution to this problem: including assumptions that succeeded in practice and the ones that did not. Experiments are described in groups with provided settings and results only for the best experiment in a group. Besides that, a discussion for experiments that did not yield improvement is provided.

## Baseline: ResNet cell-wise

As a baseline solution, we trained a cell classifier. For training, we subsetted images with a single label. This way, we know that cells on an image inherit image-level labels. With the HPACellSegmentation-provided masks, we sliced images into patches of bounding boxes with tightly contained cells. Each patch inherited an image-level label for training.
**Size**: each cell patch was rescaled to 224x224 size and normalized as described above;
**Normalization**: normalized by dataset statistics;
**Augmentation**: horizontal flips;
**Backbone architecture**: ResNet-50;
**Loss functions**: BCE loss;
**Optimizer**: Adam with learning rate of 1e-4;

## PuzzleCAM w/ ResNet

Since we could train only on single-label, our model could not generalize well on images with multiple compartment localization patterns.
To address this problem, we used the PuzzleCAM classification pipeline to train directly for multi-label instances.
**Size**: each image rescaled to 512x512;
**Normalization**: normalized by dataset statistics;
**Augmentation**: horizontal flips;
**Backbone architecture**: ResNet-50;
**Loss functions**: BCE loss as classification loss, L1 as reconstruction loss for CAMs;
**Optimizer**: Adam with learning rate of 1e-4;

On the inference, HPACellSegmentator was used to obtain instance masks. For each instance, class scores were computed by averaging each CAM inside the mask region. To the produced vector, we applied sigmoid to transform logits to [0, 1] domain. Essentially, we used the average pooling operation as it was used during the training, but on the cell level.

## PuzzleCAM w/ EfficientNet

To increase the speed of the training process, we decided to use the architecture from the more memory-efficient family of EfficientNets despite its unimaginative name.

The positive activation term was added progressively to the training with the same schedule as the reconstruction loss term. Using this loss term, we were aiming to penalize each image's activation of positive classes outside the cell masks to achieve better focus. However, experimentation with positive activation term did not yield better results.
**Size**: each image rescaled to 512x512;
**Normalization**: normalized by dataset statistics;
**Augmentation**: horizontal flips;
**Backbone architecture**: EfficientNet-b4;
**Loss functions**: BCE loss as classification loss, L1 as reconstruction loss for CAMs;
**Optimizer**: Adam with learning rate of 1e-4;

## PuzzleCAM w/ L2, Focal, and spatial augmentations

BCE loss was substituted by Focal loss to achieve better prediction on tricky cases and significantly underrepresented classes. We also refined activation masks with changing L1 by L2 loss, assuming it would penalize heavier more significant deviations while providing a bit more freedom for the training process at small margins. To augmentation pipeline, new spatial augmentations were introduced. All of the changes subsequently caused a performance boost.

**Size**: each image rescaled to 512x512;
**Normalization**: normalized by dataset statistics;
**Augmentation**: spatial augmentations;
**Backbone architecture**: EfficientNet-b4;
**Loss functions**: Focal loss w/ gamma=1, L2 as reconstruction loss;
**Optimizer**: Adam with learning rate of 1e-4;

Since HPACellSegmentator produced nuclear segmentation, we decided to test if score pooling for classes contained in the nucleus might be improved through pooling from the nuclear region instead of the total cell area. To this end, we measured average precision for cell-level predictions on the single-label subset. Average precision was measured for all of the classes pooling scores from cell or nuclei areas. Nuclear AP was worse across all of the classes; thus, we decided to pool from the cell region further.

## PuzzleCAM w/ truncated feature extractor and color augmentations

At this stage, normalization by dataset statistics was tested against instance normalization. Instance normalization did not provide any substantial boost to performance.

Since we used a full EfficientNet-b4 feature extractor, we ended up with features of scale 16x16. We subsequently changed the depth of our extractor from 5 to 4 and later to 3 blocks, enlarging feature maps to 64x64 and boosting the AP score. Also, color augmentation groups were introduced to the pipeline (mentioned above).
**Size**: each image rescaled to 512x512;
**Normalization**: normalized by dataset statistics;
**Augmentation**: augmentations from second, third, and fourth augmentation group except for elastic transform;
**Backbone architecture**: EfficientNet-b4 truncated;
**Loss functions**: Focal loss w/ gamma=1, L2 as reconstruction loss;
**Optimizer**: Adam with learning rate of 1e-4;

## PuzzleCAM w/ further augmentation

We also performed experiments with channel dropout. There are two reasons behind it. The first reason is the usual attempt to achieve better training since dropout techniques are designed to regularize the training process. Another is that in the HPA2018, some teams did not include Y-channel as an input to their models because it did not seem to bring any performance boost. Since it looks like a blue channel can be identified by a red and yellow channel, it seemed worth training a more robust model dropping one of the RBY channels. However, these experiments did not lead to a performance improvement.

To produce finer CAMs at this stage, we changed the last convolution with stride 2x2. Instead, we used dilated convolution with the dilation 2x2. This way, we produce CAMs of size 128x128; pretrained weights can still be used.
Though adding elastic transform proves to be beneficial by both cell and image-level average precision.
**Size**: each image rescaled to 512x512;
**Normalization**: normalized by dataset statistics;
**Augmentation**: full augmentation pipeline;
**Backbone architecture**: EfficientNet-b4 truncated-dilated;
**Loss functions**: Focal loss w/ gamma=1, L2 as reconstruction loss;
**Optimizer**: Adam with learning rate of 1e-4;

## UNet trained with HPACellSegmentator

Boosting up the classification score, we decided that it was worth giving a shot for a training segmentation pipeline to improve test mAP. Thus we could receive three potential benefits. With this approach, we could potentially train to generalize better on our data, reduce inference time with one forward path, and we could use segmentation as an auxiliary loss to improve CAMs.

Predictions were made for four classes: cells, cell borders, nuclei, nuclei borders. These were obtained from HPACellSegmentator and thresholded by 0.5 confidence to receive binary maps for each class.

UNet was trained with Lovasz-hinge for each class to maximize the Jaccard index. Based on the inference to split and match cells and nucleus, we used post-processing tools provided by open-source HPACellSegmentator code based on the Otsu thresholding and watershed transform.

We also run an inference with CAMs obtained from this network but masks from the HPACellSegmentation tool (see Results, Encoder from UNet with HPACellSegmentation). Experiments were done based on a pre-trained encoder from previous experiment.

**Size**: each image rescaled to 512x512;
**Normalization**: normalized by dataset statistics;
**Augmentation**: full augmentation pipeline;
**Backbone architecture**: EfficientNet-b4 truncated-dilated;
**Loss functions**: Focal loss w/ gamma=1, L2 as reconstruction loss, lovasz for segmentation masks;
**Optimizer**: Adam with learning rate of 1e-6 for encoder and learning rate of 1e-3 for decoder;

In the Results section, we demonstrate that this experiment received better results with CAMs from encoder and masks from HPACellSegmentator on the first test set, while on the second test set, the entire developed pipeline was predicting better. However, since the private test scores were received later, we concluded that this approach indeed helped to achieve better CAMs but did not prevail on the HPACellSegmentator for the cell segmentation. Thus we decided to base the final solution masks on the HPACellSegmentator while refining CAMs with similar techniques.

## PuzzleCAM w/ Image-level and TTA, lovasz-loss

Since class scores were pooled out of the total cell area and at least one class should be assigned to each cell, we attempted to span true label class masks over the total cell area by adding additional loss term. With channel-wise max pooling across CAMs, we calculate a Lovasz-hinge loss term.

The Lovasz-hinge term was added progressively to the training with the same schedule same as the reconstruction loss term. In this fashion, we directly optimize for intersection-over-union among CAMs (in general) and cell area from which scores were pooled.

Another idea to improve classification was to use image class predictions from the PuzzleCAM classification head with a weighted sum of cell-level logits and image-level logits. Although the score was improved, peaking at a 0.7/0.3 ratio, we excluded this technique since it was not robust against SCV.

However, to make inference more robust, we used a test-time augmentations pipeline. Namely, we inferred the original image and three flipped copies of this image. Then all of the CAMs were flipped back to initial orientation and averaged across samples.

Another boost was received from using additional training data applied as described in the corresponding section above.
**Size**: each image rescaled to 512x512;
**Normalization**: normalized by dataset statistics;
**Augmentation**: full augmentation pipeline;
**Backbone architecture**: EfficientNet-b4 truncated-dilated;
**Loss functions**: Focal loss w/ gamma=1, L2 as reconstruction loss, lovasz for CAM pooling;
**Optimizer**: Adam with learning rate of 1e-5;

## Self-supervision PuzzleCAM

Another idea to produce a better representation of protein patterns was borrowed from self-supervised learning. PuzzleCAM pipeline was slightly changed; we loaded two representations of input that were augmented in the same way with mask dropout and spatial block but differed in their color augmentations. Then classification loss was computed for both as usual, while reconstruction loss was computed with patches of the opposite sample. This way, we directly optimize CAMs producing them as invariant to color deviations.

**Size**: each image rescaled to 512x512;
**Normalization**: normalized by dataset statistics;
**Augmentation**: full augmentation pipeline for self-supervised learning;
**Backbone architecture**: EfficientNet-b4 truncated-dilated;
**Loss functions**: Focal loss w/ gamma=1, L2 as reconstruction loss, lovasz for CAM pooling;
**Optimizer**: Adam with learning rate of 1e-5;

# Chapter 6

# Results

We used mean average precision for image classification to validate our experiments. In this context averaging across all of the class APs.
We also logged each class AP. Few other metrics were logged for better interpretation:

1. single-label subset mAP on a cell level. Cell level score was computed with the help of HPACellSegmentation mask in the same way it was done during the inference. It was logged to check quality/good spanning of class activation maps; We also logged each class AP;

2. AP on images for a multi-label subset: logged to track whether the model generalizes well on the multi-label instances, ofcourse;

3. Briefly, we tracked nucleus-level mean AP and AP for all of the classes to decide on pooling nucleus contained classes from a nucleus region;

4. For UNet segmentation, the Jaccard index with HPACellSegmentation masks was logged. However, it was logged solely to check on gradual improvements during training. Since HPACellSegmentation masks are not ground truth masks, validation scores are more or less useless;

5. Also, we provide mAP scores on two test sets for best experiments in each group, except test scores for second test experiments, where results, unfortunately, were not logged properly.

While inspecting results, keep in mind that the private test set has a more significant SCV presence. This phenomenon is usually hard to predict.

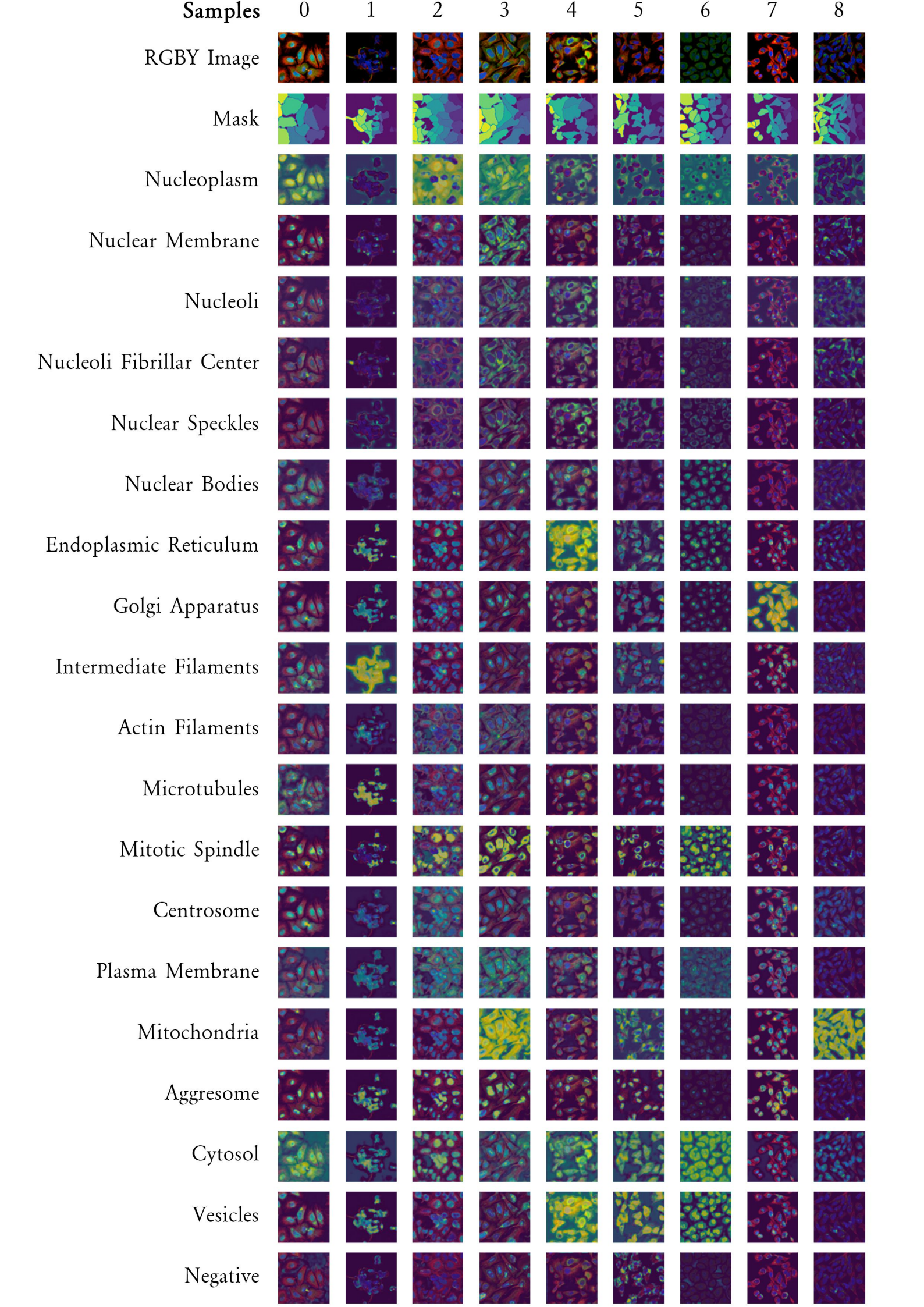
| | **AP** | **single-label cell AP** | **multi-label AP** |
|---|---|---|---|
| Baseline: ResNet cell-wise | — | 0.585 | – |
| PuzzleCAM w/ ResNet | 0.651 | 0.395 | 0.547 |
| PuzzleCAM w/ EfficientNet | 0.723 | 0.515 | 0.62 |
| PuzzleCAM w/ L2, Focal, spatial augmentations | 0.751 | 0.56 | 0.663 |
| PuzzleCAM w/ truncated feature extractor | 0.76 | 0.668 | 0.7 |
| PuzzleCAM w/ further augmentation | 0.781 | 0.726 | 0.752 |
| UNet | 0.797 | 0.734 | 0.787 |
| Encoder from UNet with HPACellSegmentator | 0.815 | 0.761 | 0.79 |
| PuzzleCAM w/ TTA, lovasz-loss. | 0.824 | 0.778 | 0.81 |
| Self-supervised PuzzleCAM | 0.8 | 0.756 | 0.794 |

TABLE 6.1: Results on validation set

| | **AP** | **public test mAP** | **private test mAP** |
|---|---|---|---|
| Baseline: ResNet cell-wise | — | 0.304 | 0.152 |
| PuzzleCAM w/ ResNet | 0.651 | 0.327 | — |
| PuzzleCAM w/ EfficientNet | 0.723 | 0.346 | — |
| PuzzleCAM w/ L2, Focal, spatial augmentations | 0.751 | 0.362 | 0.261 |
| PuzzleCAM w/ truncated feature extractor | 0.76 | 0.387 | 0.33 |
| PuzzleCAM w/ further augmentation | 0.781 | 0.4 | 0.365 |
| UNet | 0.797 | 0.431 | 0.411 |
| Encoder from UNet with HPACellSegmentator | 0.815 | 0.44 | 0.401 |
| PuzzleCAM w/ TTA, lovasz-loss. | 0.824 | 0.451 | 0.415 |
| Self-supervised PuzzleCAM | 0.8 | 0.443 | 0.419 |

TABLE 6.2: Results on test set compared to validation AP

Below you may see CAMs produced by the last model. Two first rows contain input image and predicted masks with HPACellSegmentation. Next rows contain activation masks overlayed with input image. Activated regions of specific cell represent spatial locations 'in favor' of specific class (organelle) containing a protein of interest.


Samples
0
1
2
3
4
5
6
7
8
RGBY Image
Mask
Nucleoplasm
Nuclear Membrane
Nucleoli
Nucleoli Fibrillar Center
Nuclear Speckles
Nuclear Bodies
Endoplasmic Reticulum
Golgi Apparatus
Intermediate Filaments
Actin Filaments
Microtubules
Mitotic Spindle
Centrosome
Plasma Membrane
Mitochondria
Aggresome
Cytosol
Vesicles
Negative

# Chapter 7

# Overview of alternative solutions

Here we provide a summary of two best scoring teams at the problem at hand. We provide an overview of building blocks used for their solutions and results reported on private and public test sets.

## 7.1 Best solution

The team suggested that the problem of a CAM-based solution is that network localizes most discriminative features, which leads to a low recall. The team then based its solution on PuzzleCAM to battle this spatial overfitting and added metric learning loss to the training mix. Metric classes were staining antibodies that were mined from publically available data. The team inferred each cell separately, thus reducing noise from neighboring cells.
Losses:

1. for classification: Focal, symmetric Lovasz and HardLog loss;
2. for reconstruction: L2;
3. for metric learning: ArcFace;

The solution was enhanced by ensembling with the second model: Swin Transformer. The transformer was trained on individual cells with different certainty labels that were assigned by rule-based approach. Spatial augmentation pipeline was used for train time and test time. Also, an additional model was applied that verified the 'borderness' of the cell as a postprocessing step since border cells should be indeed considered negative if 0.5 of the cell area is out of the image.

Public test mAP: 0.590; Private test mAP: 0.566

## 7.2 Second-best solution:

This team developed the ensemble of four models:

1. The double-headed model for cell tiles. The model predicted both cell and image levels. It was trained with BCE. Hard augmentation was applied to sampled cells. Backbone: Efficientnet-b3;
2. The image-level model with masked cell inference. Inference was carried out the same way as the first team did. The model was trained with primarily spatial augmentations and BCE loss;

3. Cell-level model with pretrained weights from the first model. Since training cell-level classification with image-level labels can introduce much noise, the team finetuned this model for two epochs. The model was trained with additional data but without a yellow channel.;

4. Metric learning model trained with Arcface loss on antibodies. The same approach as the first team applied;

Public test mAP: 0.602; Private test mAP: 0.553;

# Chapter 8

# Conclusion and Future work

## 8.1 Conclusions

In this work, we dealt with the problem of weakly supervised single-cell classification. We investigated state-of-the-art approaches for weakly supervised semantic segmentation and cell classification. Fusing discovered approaches lead as to three promising solutions:

1. PuzzleCAM tuned with Lovasz loss that scored well on the public test set;
2. UNet that improved on seemingly more complex data from the private test set;
3. PuzzleCAM trained with self-supervision that achieved the best results on the private test set;

Regarding UNet: since results from the private test set were obtained after all of the experiments were finalized, we did not carry out enough tuning into this solution. To this end, the additional dataset released by the HPA team could be very useful for training a good segmentation model.

Since we uncovered that our UNet was predicting reasonably well on the private test set, it should be beneficial to investigate better segmentation approaches for SCV-rich data.

With this work, we scored 55th out of 757 teams in this weakly supervised classification challenge on both test sets.

## 8.2 Future work

Based on top solutions and our results, we determined promising areas for future work as:

1. Better inference techniques: TTA, masked cell inference;
2. Mining of rare cases and additional classes (e. g. antibodies);
3. Ensemble of various models;
4. Metric learning auxiliary losses;
5. Segmentation for SCV-rich data;

# Chapter 9

# Implementation details

To achieve faster training, we used torch.cuda.amp package for automatic mixed-precision setup. We could automatically switch to float16 precision where higher precision is not required for convergence; thus, enabling us to reduce memory and computation costs.
For classification pipeline we used work of Wightman [30];
For segmentation pipeline we used work of Yakubovskiy [31];
For augmentation we used work of A. Buslaev et al. [1];

All experiments were trained on the system with 20-core intel core i9 10900k, 24268 Mb of Ram, and Nvidia RTX 3090. All models were trained with CUDA 11.1, CUDnn 7.6.5, and PyTorch 1.8.1.

# Bibliography


[1] E. Khvedchenya V. I. Iglovikov A. Buslaev A. Parinov and A. A. Kalinin. "Albumentations: fast and flexible image augmentations". In: *ArXiv e-prints* (2018). eprint: 1809.06839.

[2] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation". In: *CoRR* abs/1511.00561 (2015). arXiv: 1511.00561. URL: http://arxiv.org/abs/1511.00561.

[3] Min Bai and Raquel Urtasun. "Deep Watershed Transform for Instance Segmentation". In: *CoRR* abs/1611.08303 (2016). arXiv: 1611.08303. URL: http://arxiv.org/abs/1611.08303.

[4] Michael Chappell and Stephen Payne. *Physiology for Engineers: Applying Engineering Methods to Physiological Systems*. 1st ed. 2016. Biosystems Biorobotics. Springer International Publishing: Imprint: Springer, 2016. ISBN: 9783319261973.

[5] Liang-Chieh Chen et al. "Rethinking Atrous Convolution for Semantic Image Segmentation". In: *CoRR* abs/1706.05587 (2017). arXiv: 1706.05587. URL: http://arxiv.org/abs/1706.05587.

[6] Mary Ann Clark et al. *Biology*. 2020. ISBN: 9781947172517.

[7] Jiankang Deng, Jia Guo, and Stefanos Zafeiriou. "ArcFace: Additive Angular Margin Loss for Deep Face Recognition". In: *CoRR* abs/1801.07698 (2018). arXiv: 1801.07698. URL: http://arxiv.org/abs/1801.07698.

[8] Ross B. Girshick et al. "Rich feature hierarchies for accurate object detection and semantic segmentation". In: *CoRR* abs/1311.2524 (2013). arXiv: 1311.2524. URL: http://arxiv.org/abs/1311.2524.

[9] Kaiming He et al. "Deep Residual Learning for Image Recognition". In: *CoRR* abs/1512.03385 (2015). arXiv: 1512.03385. URL: http://arxiv.org/abs/1512.03385.

[10] Kaiming He et al. "Mask R-CNN". In: *CoRR* abs/1703.06870 (2017). arXiv: 1703.06870. URL: http://arxiv.org/abs/1703.06870.

[11] Jie Hu, Li Shen, and Gang Sun. "Squeeze-and-Excitation Networks". In: *CoRR* abs/1709.01507 (2017). arXiv: 1709.01507. URL: http://arxiv.org/abs/1709.01507.

[12] Gao Huang, Zhuang Liu, and Kilian Q. Weinberger. "Densely Connected Convolutional Networks". In: *CoRR* abs/1608.06993 (2016). arXiv: 1608.06993. URL: http://arxiv.org/abs/1608.06993.

[13] Vladimir I. Iglovikov et al. "TernausNetV2: Fully Convolutional Network for Instance Segmentation". In: *CoRR* abs/1806.00844 (2018). arXiv: 1806.00844. URL: http://arxiv.org/abs/1806.00844.

[14] Sanghyun Jo and In-Jae Yu. "Puzzle-CAM: Improved localization via matching partial and full features". In: *CoRR* abs/2101.11253 (2021). arXiv: 2101.11253. URL: https://arxiv.org/abs/2101.11253.

[15] Youngwan Lee and Jongyoul Park. "CenterMask : Real-Time Anchor-Free Instance Segmentation". In: *CoRR* abs/1911.06667 (2019). arXiv: 1911.06667. URL: http://arxiv.org/abs/1911.06667.

[16] Tsung-Yi Lin et al. "Feature Pyramid Networks for Object Detection". In: *CoRR* abs/1612.03144 (2016). arXiv: 1612.03144. URL: http://arxiv.org/abs/1612.03144.

[17] Tsung-Yi Lin et al. "Focal Loss for Dense Object Detection". In: *CoRR* abs/1708.02002 (2017). arXiv: 1708.02002. URL: http://arxiv.org/abs/1708.02002.

[18] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. "Learning Deconvolution Network for Semantic Segmentation". In: *CoRR* abs/1505.04366 (2015). arXiv: 1505.04366. URL: http://arxiv.org/abs/1505.04366.

[19] Wei Ouyang et al. "Analysis of the human protein atlas image classification competition". In: *Nature methods* 16.12 (2019), pp. 1254–1261.

[20] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: *CoRR* abs/1505.04597 (2015). arXiv: 1505.04597. URL: http://arxiv.org/abs/1505.04597.

[21] Konstantinos Sechidis, Grigorios Tsoumakas, and Ioannis P. Vlahavas. "On the Stratification of Multi-label Data". In: *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2011, Athens, Greece, September 5-9, 2011, Proceedings, Part III*. Ed. by Dimitrios Gunopulos et al. Vol. 6913. Lecture Notes in Computer Science. Springer, 2011, pp. 145–158. DOI: 10.1007/978-3-642-23808-6_10. URL: https://doi.org/10.1007/978-3-642-23808-6_10.

[22] Ramprasaath R. Selvaraju et al. "Grad-CAM: Why did you say that? Visual Explanations from Deep Networks via Gradient-based Localization". In: *CoRR* abs/1610.02391 (2016). arXiv: 1610.02391. URL: http://arxiv.org/abs/1610.02391.

[23] Sydney M Shaffer et al. "Rare cell variability and drug-induced reprogramming as a mode of cancer drug resistance". In: *Nature* 546.7658 (2017), pp. 431–435.

[24] Evan Shelhamer, Jonathan Long, and Trevor Darrell. "Fully Convolutional Networks for Semantic Segmentation". In: *CoRR* abs/1605.06211 (2016). arXiv: 1605.06211. URL: http://arxiv.org/abs/1605.06211.

[25] Patrice Y. Simard, David Steinkraus, and John C. Platt. "Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis". In: *7th International Conference on Document Analysis and Recognition (ICDAR 2003), 2-Volume Set, 3-6 August 2003, Edinburgh, Scotland, UK*. IEEE Computer Society, 2003, pp. 958–962. DOI: 10.1109/ICDAR.2003.1227801. URL: https://doi.org/10.1109/ICDAR.2003.1227801.

[26] Karen Simonyan and Andrew Zisserman. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. 2015. arXiv: 1409.1556 [cs.CV].

[27] Christian Szegedy et al. "Going Deeper with Convolutions". In: *CoRR* abs/1409.4842 (2014). arXiv: 1409.4842. URL: http://arxiv.org/abs/1409.4842.

[28] Mingxing Tan and Quoc V. Le. "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks". In: *CoRR* abs/1905.11946 (2019). arXiv: 1905.11946. URL: http://arxiv.org/abs/1905.11946.

[29] Zhi Tian et al. "FCOS: Fully Convolutional One-Stage Object Detection". In: *CoRR* abs/1904.01355 (2019). arXiv: 1904.01355. URL: http://arxiv.org/abs/1904.01355.

[30] Ross Wightman. *PyTorch Image Models*. https://github.com/rwightman/pytorch-image-models. 2019. DOI: 10.5281/zenodo.4414861.

[31] Pavel Yakubovskiy. *Segmentation Models Pytorch*. https://github.com/qubvel/segmentation_models.pytorch. 2020.

[32] Fisher Yu and Vladlen Koltun. "Multi-Scale Context Aggregation by Dilated Convolutions". In: *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. 2016. URL: http://arxiv.org/abs/1511.07122.

[33] Jiaqian Yu and Matthew B. Blaschko. "The Lovász Hinge: A Novel Convex Surrogate for Submodular Losses". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 42.3 (2020), pp. 735–748. DOI: 10.1109/TPAMI.2018.2883039. URL: https://doi.org/10.1109/TPAMI.2018.2883039.

[34] Sergey Zagoruyko and Nikos Komodakis. "Wide Residual Networks". In: *CoRR* abs/1605.07146 (2016). arXiv: 1605.07146. URL: http://arxiv.org/abs/1605.07146.

[35] Hengshuang Zhao et al. *Pyramid Scene Parsing Network*. 2017. arXiv: 1612.01105 [cs.CV].

[36] Bolei Zhou et al. "Learning Deep Features for Discriminative Localization". In: *CoRR* abs/1512.04150 (2015). arXiv: 1512.04150. URL: http://arxiv.org/abs/1512.04150.