MASTER’S PROJECT REPORT

**COURSE :**

CS 700B 624 – MASTER’S PROJECT

**GUIDED BY :**

PROF. GRACE WANG

**TEAM MEMBERS FOR MOVIE SCRIPT ALIGNMENT NLP PROJECT:**

A.M. Muntasir Rahman

Jingyi Gu

Partha Mehta

**COURSE :**

CS 700B 624 – MASTER’S PROJECT

**GUIDED BY :**

PROF. GRACE WANG

**Master’s Project Report**

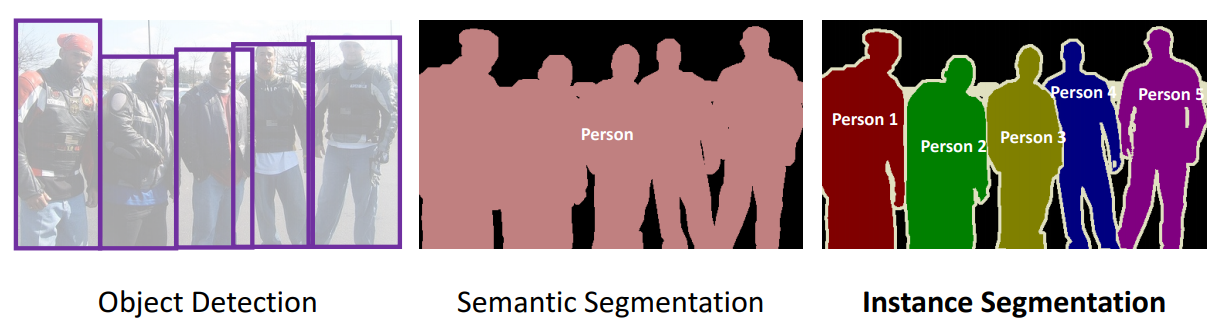
Projects which I worked on:

1. A deep-learning / computer vision based project and the work was to segment nuclei in an image and classify. **(Instance Segmentation & Semantic Segmentation)** The link for the challenge was this : <https://conic-challenge.grand-challenge.org/> . The name of the challenge was **Conic Challenge.**
2. My other project was on **Deriving High-Quality Book Comprehension Benchmarks via Book-Movie Script Alignments** which was based on **NLP**.

Instance Segmentation & Semantic Segmentation – What are these really?

* Instance segmentation treats multiple objects of the same class as distinct individual instances.
* Semantic segmentation associates every pixel of an image with a class label such as a person, flower, car and so on. It treats multiple objects of the same class as a single entity.

Instance Segmentation & Semantic Segmentation – An example



**Project 1: Conic Challenge based Nuclei Segmentation – What did the dataset look like?:**

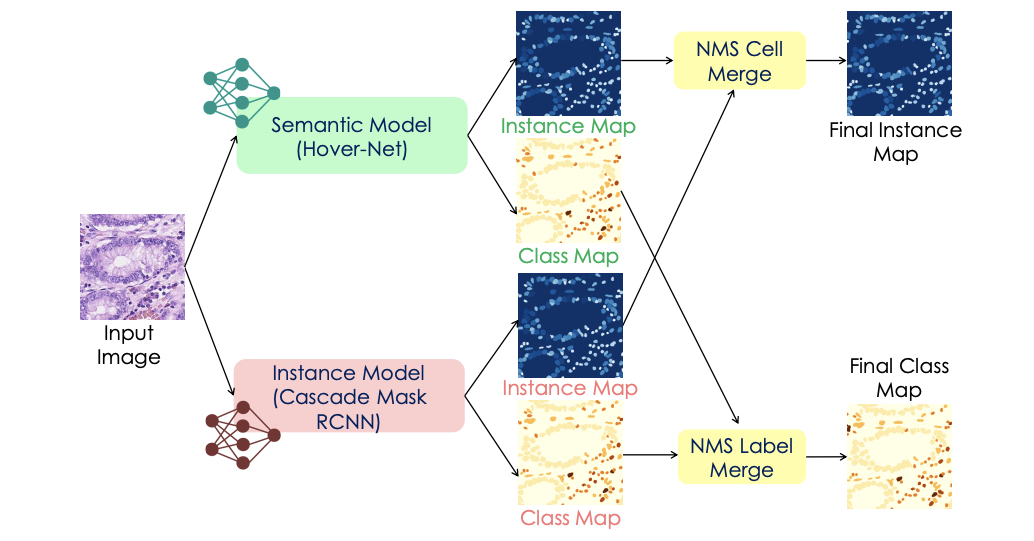
* The dataset for this challenge consisted of Haematoxylin and Eosin stained histology images. For each image, an instance segmentation and a classification mask is provided. Each nucleus was assigned to one of the following categories: **Epithelial, Lymphocyte, Plasma, Eosinophil, Neutrophil and Connective tissue.**
* The provided patch level dataset consists of 4981 non-overlapping images of size 256\*256 provided in the following formats: **RGB images, Segmentation & classification maps, Nuclei counts.**
* The RGB images and segmentation/classification maps are each stored as a single numpy array. RGB image array has dimensions **4981\*256\*256\*3** whereas the segmentation & classification map array has dimensions **4981\*256\*256\*2**.
* The **first** **channel** is **instance** **segmentation** **map** and the **second** **channel** is the **classification** **map** & for the nuclei counts, a single csv file is provided, where each row corresponds to a given patch and the columns determine the counts for each type of nucleus.

**Solutions already Implemented by Authors through this challenge:**

* The body of literature on semantic segmentation has reported several potential methods, where the most widely used model is U-Net (<https://arxiv.org/pdf/1505.04597.pdf>).
* The U-Net model uses an encoder-decoder architecture to reconstruct segmentation masks from CNN features.
* Based on U-Net, Ψ-Net achieves better semantic segmentation accuracy by stacking densely convolutional LSTMs.
* Another well-known model is Hover-Net. It further leverages the instance information encoded within the vertical and horizontal distances of nuclear pixels to their centers of mass to perform instance segmentation.
* Unlike semantic models, instance techniques usually utilise a two-stage architecture to detect and segment each instance. A well-established technique is Faster RCNN.
* It is composed of a region proposal stage and a bounding box regression stage, which is effective for detection tasks.
* Mask-RCNN extends Faster R-CNN by adding a mask prediction branch, in parallel to the bounding box recognition branch, to perform instance detection and segmentation simultaneously.
* Cascade Mask R-CNN achieves better segmentation and classification performance than previous models.
* It works by adding a sequence of detectors trained with increasing IoU thresholds on Mask-RCNN.

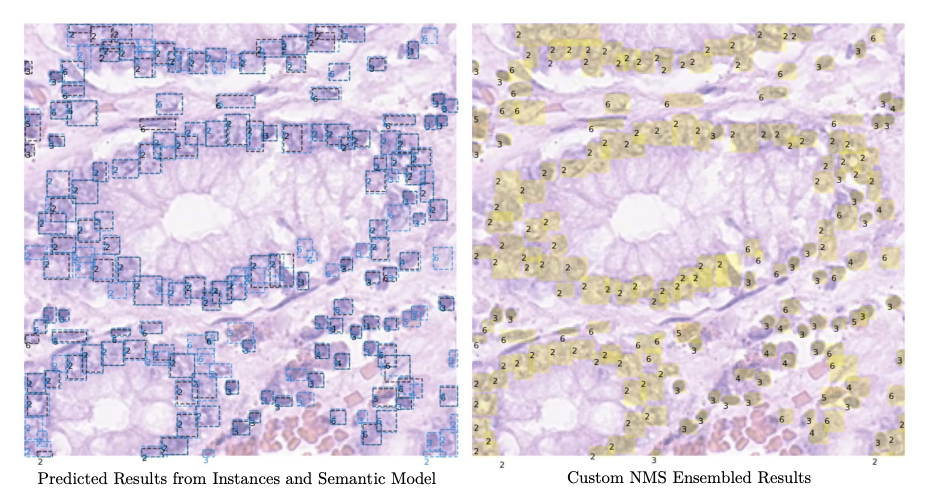
**Architecture for the 1st solution**

The architecture is on the next page.

****

**Solutions Implemented By Me in my work (1st Solution used three parts – Instance Segmentation Model, Semantic Segmentation & a customized Non-Maximum Suppression Algorithm):**

* My model was initially an ensemble model. I first trained a semantic segmentation model and then an instance segmentation model (separately)
* I used backbone as HoverNet (<https://arxiv.org/pdf/1812.06499.pdf>) and Cascade Mask-RCNN (<https://arxiv.org/pdf/1906.09756.pdf)and> used ResNeXt-152 as backbone for Cascade Mask-RCNN model. Then I ensembled the results with a customized Non-Maximum Suppression embedding algorithm [NMS\_Algorithm](https://towardsdatascience.com/non-maximum-suppression-nms-93ce178e177c)
* One observation from previous tests is that semantic segmentation part can achieve an accurate class prediction for the cells while instance segmentation provides a refined segmentation.
* From the observations during the implementation, I observed that semantic models are able to do highly accurate class predictions for cells; as these types of techniques provide a global understanding of the image content.
* For instance-segmentation, every single nucleus in an image would matter and so, I tried to optimize the element-wise cross entropy loss over all the data points and all categories.
* If the image is too small, then the segmentation is coarse.
* To meet this specification, I set the input as 512 × 512, which is currently the smallest image size for instance segmentation models.
* Moreover, I decreased the original anchor size setting from (32, 64, 128, 256, 512) to (8, 16, 32, 64, 128) due to the smaller size of the images and their targets (nucleus).
* With these changes, the model is more robust when detecting smaller objects like the nucleus. A sample output of the instance model is shown on the next slide. Most notable, the output is marked using the blue bounding boxes.

****

Above is the illustration of the ensemble process from the NMS. The left side figure displays the predicted bounding boxes from the semantic model (black bounding boxes) and the instance model (blue bounding boxes), respectively. Then, the ensemble results are shown using customised NMS algorithm. The final output is displayed, on the right side, showing the ensemble results visualised on the yellow bounding boxes.

**Solution Implemented (1st Solution) – About Semantic Segmentation Now:**

* Semantic segmentation is to provide a denser prediction than instance segmentation models, as the goal is to predict pixel-wise the label of objects related to a set of defined classes. To apply a semantic segmentation model on instance segmentation tasks, an additional post-processing step is needed.
* This post-processing process can split the semantic segmentation result into multiple single objects from the same class (i.e. into instance segmentation).
* To avoid re-implementing this function, HoverNet is directly selected as a semantic training model, that is, to optimise another loss (the cross-entropy loss as mentioned in case of Instance Segmentation Model). For clarity purposes, let’s denote the loss as Lsemantic.
* The focus is on the model design rather than the post-processing. To keep the same setting as the instance segmentation model, ResNeXt152 is selected as the backbone for our semantic segmentation model, and use the same input size as 512\*512 for consistency.
* The sample output for semantic model is shown on the previous slide, where output is highlighted using black bounding boxes.
* We follow the work where here and we seek to optimise the following the element-wise cross entropy loss over all voxels and all categories:

**Linstance or L-semantic = - (1/ (mxk) ) x ∑sum(m=1 to M) x ∑sum(k=1 to K) y-subscript-m^k \* log(p-subscript-m^k)**

where M denotes the total number of pixels in an input image, and K is the cell cat- egories in our application we have seven categories including the background (details on the categories are displayed in the experimental section). Moreover, ymk refers to the ground truth label at the m-th pixel in the k-th category; pkm is the softmax output value, which represents a predicted probability indicating the m-th voxel belongs to the k-th category.

**Solution Implemented (1st Solution) – About NMS Embedding Algorithm as a Model Ensemble Now:**

* After obtaining the results from the semantic and instance segmentation models, a custom NMS model has been adopted to ensemble the detected instances from the two models.
* Note that this step is not a learning process. NMS seeks to find overlapped instance results based on the position of the detected cells bounding boxes. By setting an Intersection over Union (IoU) threshold, it can be determined if two or more cells from the two models identify the same cell.
* The standard NMS only eliminates multiple excessive cells and keep only one cell. In this work, a customised NMS has been used, which not only detects multiple cells but also merges the segmentation mask and the classification labels.
* Firstly, the model detected the overlapped segmentation results for one cell / one position. Then take as a final segmentation output the merge of such overlapped regions of a cell.
* Then, we assign a classification label to the merged segmentation results by a weighted voting mechanism, which details are as follows. The semantic segmentation model is capable of a great understanding of the global information content.
* Hence, it can achieve a great classification performance in cells with a small number such as neutrophils, plasma, and eosinophils.
* For the label merging process, the weight of the neutrophil, epithelial, lymphocyte, plasma, eosinophil, and connective tissue cells is taken from the semantic segmentation model, and set to (2, 1, 1, 2, 2, 1).
* While the weight of neutrophil, epithelial, lymphocyte, plasma, eosinophil, and connective tissue cells is taken from the instance segmentation model and set to (1.5, 1.5, 1.5, 1.5, 1.5, 1.5).
* Then a weight of 1.5 is assigned to the weight of the instance model’s prediction to avoid the same voting number.
* For example, two cells overlapped in the NMS stage, where the semantic model predict it as neutrophil and the instance model as epithelial.
* Then the final label for this cell is assigned as a neutrophil.

**Pre-processing and Data Augmentation:**

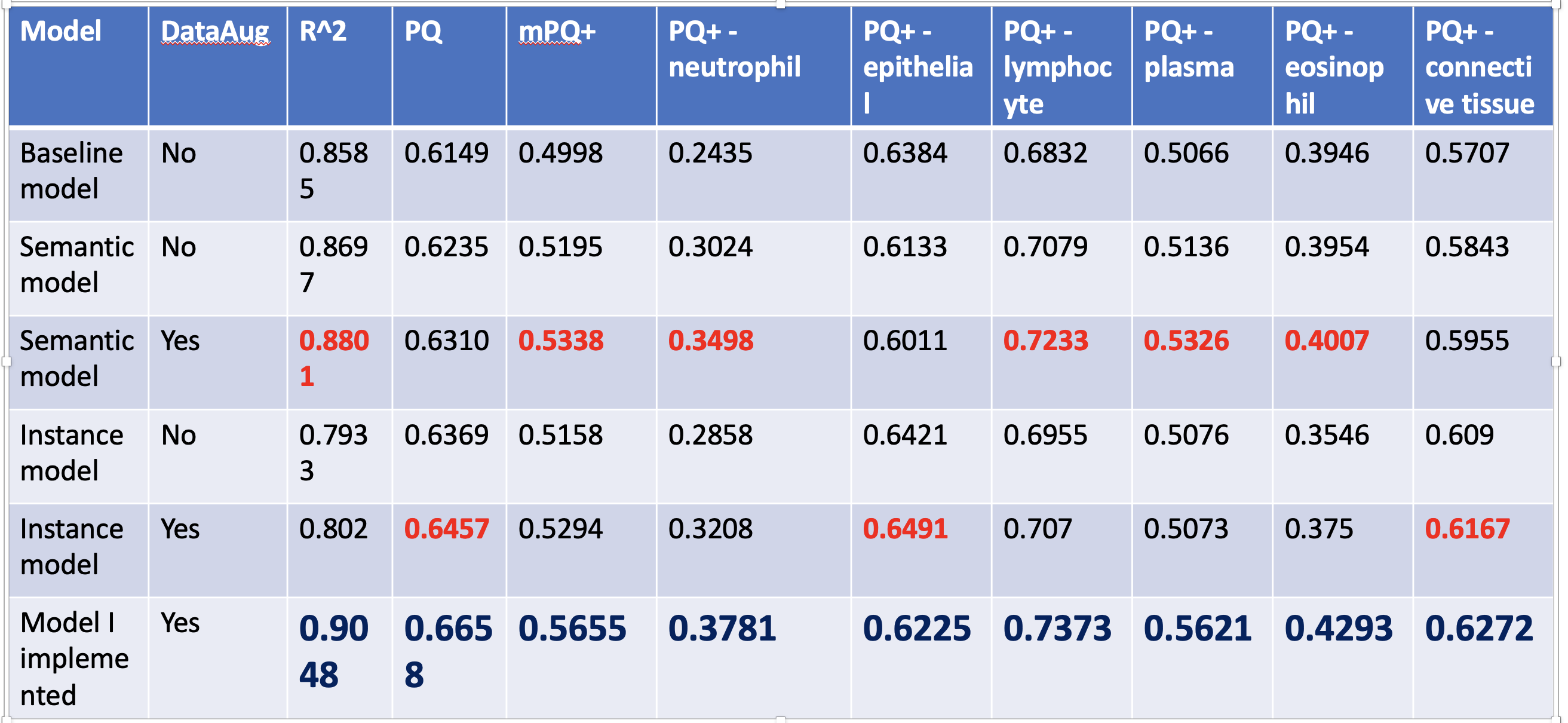
For a robust training process, a 4 step data augmentation process is used to enrich the training data.

* Firstly, the image is enlarged to twice its original size 512 × 512. This step is necessary for our instance model training as described before.
* Secondly, a random scale has been performed to resize the images to 0.8 - 1.2 of the size drawn from previous step.
* Then randomly cropped, using a fixed size of 512×512, regions with zero padding.
* Thirdly, random flip has been used to augment the data from different views.
* As a fourth step, Gaussian blur, median blur, and additive Gaussian noise have been selected randomly.
* imgaug package has been used to enhance the pixels sensitivity of the input.

**Implementation Details:**

* The code is built based on Hover-Net and detectron2 repositories.
* For the loss function, binary cross-entropy loss for mask prediction has been used, and cross-entropy loss for label prediction in both repositories.
* For the instance model, SGD optimiser is used during training with the initial learning rate setting as 0.0001.
* While, for the semantic model, Adam optimiser is used with the same learning rate 0.0001.
* All models are run on an NVIDIA A100 GPU with 80G RAM, which takes around 12 hours to train the instance model for 60 epochs, and 96 hours to train the semantic models for 100 epochs.
* The data splitting process consisted of 3900 images as training and 1081 images as test set.
* To evaluate the model, firstly, multi-class panoptic quality (PQ) and multi-class PQ+ metrics have been usedto evaluate the segmentation and classification tasks. While, for the counting (cellular composition) the multi-class coefficient of determination R2.

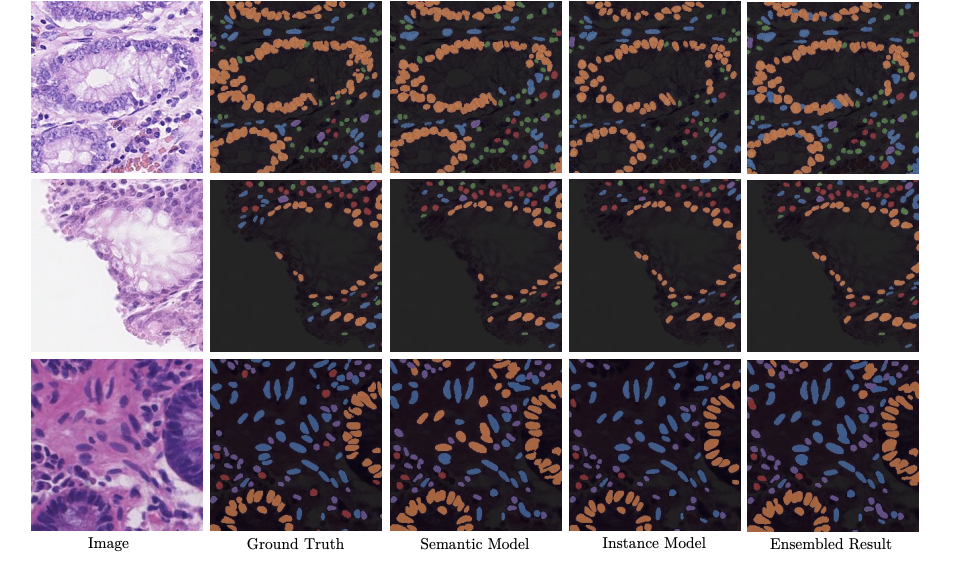
**Performance Metrics for 1st Solution Implemented:**



**Observations:**

* The results are reported in the previous slide, where my framework outperformed all compared models.
* In a closer inspection, I observed that my modified Semantic Model performs better than the baseline model. The performance improvement comes from the data augmentation and better backbone model. Moreover, one interesting finding is that the Instance Model performs better than the Semantic Model in terms of PQ evaluation metrics, while the Semantic Model outperforms the instances model in terms of mPQ+ and R2 evaluation metrics.
* This is because, when performing segmentation, the Instance Model focuses only on the nucleus region. Hence, it has a better segmentation accuracy (PQ).
* While the Semantic Model has a great semantic understanding of the content of the image, i.e. how many cells are in the images. Hence, the classification accuracy (R2 ) is better. Also, even with a low PQ, the Semantic Model can also outperform the Instance Model regarding the final mPQ+.
* I plan on writing a paper soon and make the code available soon on Github once I format the code properly.

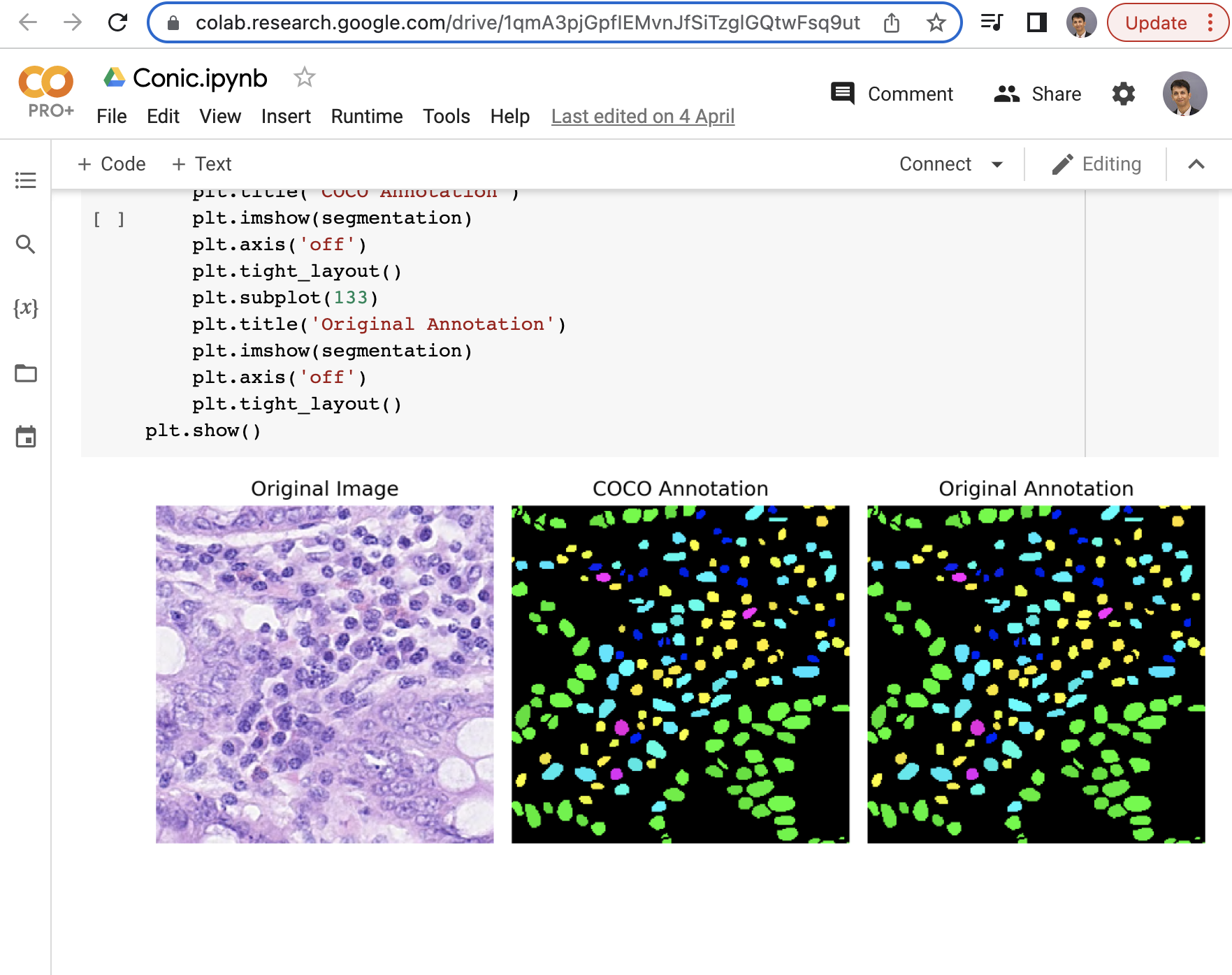
**Output Images for 1st Solution Implemented**

****

**Solutions Implemented By Me in my work (2nd Solution) based on Pure Detectron2:**

* My second model was just based on Mask-RCNN framework using Facebook’s Pre-trained Detectron2 model. Here I just performed Segmentation and Classification and no counting due to time constraints. Paper on Detectron2 ([Detectron2Paper](https://research.fb.com/wp-content/uploads/2019/12/4.-detectron2.pdf)) and Github repo for Detectron2 ([Detectron2Github](https://github.com/facebookresearch/detectron2))
* The result was I got IOU score (Intersection Over Union – A Performance Metric to be checked while Segmenting Images) as 0.72. More metrics such as dice loss, panoptic quality can be used as well. I used IOU.
* What is IOU? - Intersection over Union (IoU) is used when calculating mAP. It is a number from 0 to 1 that specifies the amount of overlap between the predicted and ground truth bounding box.
* What is mAP? - To evaluate object detection models like R-CNN and YOLO, the mean average precision (mAP) is used.
* The mAP compares the ground-truth bounding box to the detected box and returns a score. The higher the score, the more accurate the model is in its detections.
* For this, the notebook Conic.ipynb has been attached.

**Output Images for 2nd Solution Implemented**

****

**Project 2: Deriving High-Quality Book Comprehension Benchmarks via Book-Movie Script Alignments:**

**What is Scene Segmentation?**

* Acquiring high-quality annotated benchmarks for book story understanding is challenging.
* These usually require the annotators to read the whole books before annotations.
* On the other hand, movie scripts are semi-structured documents.
* There are many kinds of important story structure information that can be automatically extracted from scripts.
* By collecting movie scripts and books adapted from the movies (such as Star Wars trilogy and their book versions), we can use sentence embedding and paraphrasing techniques to align the movies with books adapted from movies.
* Then we propose to project the structures from movie scripts to annotate books, so as to automatically create book understanding benchmarks with high quality.
* Main task was to perform Scene Segmentation.

**What is Scene Segmentation in NLP?**

* Scene Segmentation - one fundamental question is, what is the basic content unit of a story.
* Previous NLP works like NarrativeQA and BookSum use paragraphs or fixed-length chunks.
* However, actually in movie scripts, a natural unit , i.e., scenes, has been defined by the authors.
* Splitting the books into scenes is a unique idea and also an important problem for follow-up annotation tasks.
* Settings – when are a specific subplot happens.
* Characters – identifying the characters involved in a scene and speakers of dialogue utterances, if there are no explicit identifiers; this also extends to NER/coreference over characters.

**My Task in the NLP Project**

* My task was to split one particular Movie Book consisting of different chapters with each of it having some content from a word file.
* I had to split these chapters into individual chapters into their individual word files consisting of all the content as given originally.
* Finally, the json content consisting of type of script (dialogue or scene) was given to Muntasir and Jingyi.
* They performed Scene Segmentation and aligning movie scenes, that is, to which chapters the scenes belong using SentenceBert.

**References:**

**Key References:**

[**https://arxiv.org/pdf/1506.06724.pd**](https://arxiv.org/pdf/1506.06724.pdf)**f ,** [**https://github.com/booknlp/booknlp**](https://github.com/booknlp/booknlp)

**References to tools:**

* SentenceBERT: <https://sbert.net/> . And how it works to construct paragraph-level alignment in BookSum: <https://arxiv.org/pdf/2105.08209.pdf>
* Script Parser: We have an accurate script parser (<https://openreview.net/pdf?id=HK-_DteWlGq>). We have already processed the movie scripts with the parser. But we encourage you to check the details, because some of the ideas can be applied for bootstrapping the sentence alignment models, if out-of-box SentenceBERT does not provide very accurate results.

**Paper Reference**

* Kočiský, T., Schwarz, J., Blunsom, P., Dyer, C., Hermann, K. M., Melis, G., & Grefenstette, E. (2018). The narrativeqa reading comprehension challenge. *Transactions of the Association for Computational Linguistics*, *6*, 317-328.
* Kryściński, W., Rajani, N., Agarwal, D., Xiong, C., & Radev, D. (2021). Booksum: A collection of datasets for long-form narrative summarization. *arXiv preprint arXiv:2105.08209*.
* <https://arxiv.org/abs/1812.06499> - HoverNet Paper
* <https://github.com/vqdang/hover_net> - HoverNet Github Code
* <https://www.catalyzex.com/paper/arxiv:2203.00262> - Separable-HoverNet and Instance-YOLO for Colon Nuclei Identification and Counting
* <https://github.com/simongraham/hovernet_inference>
* <https://github.com/rshwndsz/hover-net>
* <https://github.com/uit-hdl/hovernet-pipeline>
* <https://github.com/TissueImageAnalytics/CoNIC>