Capsulorrhexis Trajectories for Automated Surgical Training Feedback

Project Category: Life-Sciences and Computer Vision
Project Mentor: Yining Chen

Name: David Kuo
SUNet ID: dekuo
Department of Biomedical Data Science
Stanford University
dekuo@stanford.edu

Name: Riya Sinha
SUNet ID: riyasinh
Department of Biomedical Data Science
Stanford University
riyasinh@stanford.edu

Name: Ben Viggiano
SUNet ID: viggiano
Department of Biomedical Data Science
Stanford University
viggiano@stanford.edu

1 Key Information to include

- External collaborators: Ben Ehlert (CS231N student), Susan Qi (MS Clinical Informatics, Ophthalmology resident), Emmett Goodman (postdoctoral fellow, MARVL Lab)
- This is a shared project between CS229 and CS231N for the Spring Quarter of 2022. David Kuo and Ben Viggiano are in both classes.

2 Introduction

Given the inherent variability in the number and type of surgical cases that surgical trainees have the opportunity to perform, even within the same institution, there is great demand among residents and fellows for any resource that can help maximize the educational value of each case. In recent years, internet platforms like YouTube have enabled surgeons to share cases and discuss tips and tricks, and allowed trainees to receive valuable feedback on their cases from more experienced surgeons; however, these initiatives are limited by the availability of experienced surgeons willing to review trainee-submitted cases outside of work hours.

To address this limitation, we propose a machine learning platform that provides automated feedback from trainee-submitted surgical videos, specifically cataract surgery. Because cataract surgery is performed within a space measured in microns to millimeters, every movement must be purposeful and precise. We believe that being able to track instrument keypoints and anatomical landmarks over the course of cataract surgery and compare instrument trajectories between experts and trainees will yield valuable feedback for the training cataract surgeon.

For this project, we focus on the capsulorrhexis step of cataract surgery, which is uniquely stresses the foundational skills of 'floating' and 'pivoting' within the 2mm main incision necessary for maneuvering safely and effectively within the eye, and is one of two critical steps in which complications most commonly occur.

In the CS231N portion of this project, we trained a Keypoint R-CNN (ResNeXt101-FPN) to detect the tips of the utrada forceps instrument used for the capsulorrhexis in each frame of a surgical video.

For the CS229 portion of this project, we generate frame-by-frame predictions of utrada forceps tips using our CS231N keypoint detection model as well as frame-by-frame predictions of anatomic landmarks of interest (namely the pupil center, pupil boundary, and incision location) using a pre-trained semantic segmentation model from our dataset of surgical videos from several resident (trainee) and attending (expert) surgeons. We then generate features from this trajectory data to perform binary classification of capsulorrhexis videos by skill level: resident (trainee) or attending (expert).

Our inputs for this CS229 project are CSV files containing frame-by-frame predicted instrument tip and anatomic landmark locations and our output is a scalar value indicating the predicted classification: resident or attending.

3 Related Work

Our project is heavily inspired by work done in the the Stanford MARVL Lab and previous work done in automating the assessment of surgical skill from surgical videos.

Jin et al. fine-tuned a ImageNet-pretrained Faster R-CNN[1] model to perform bounding box detection of surgical tools for laparoscopic cholecystectomy, and used the predicted spatial annotations to create timelines of tool usage, heat maps of bounding box locations, and tool trajectory maps for the critical clipping phase of laparoscopic cholecystectomy and estimate the Global Operative Assessment of Laparoscopic Skills (GOALS) rating for each video [2].

More recently, Goodman et al. curated the Annotated Videos of Open Surgery, a dataset which included 343 YouTube videos of 23 different open surgical procedures annotated with hand and instrument bounding boxes, 21 hand joint keypoints, and per-second resolution temporal labels of cutting, tying, suturing, or background. Using this dataset, they developed a deep multi-task neural network to classify actions and track tools, hands, and hand pose. They generated hand and hand pose trajectories for 23 open surgeries with 14 surgeons (including medical students, residents, fellows, and attendings) at Beth Israel Deaconess, and calculated metrics such as distance traveled, velocity, acceleration, hand steadiness, hand pose variation that were found to provide useful feedback for skill improvement [3].

For cataract surgery, much of previous work has focused on the temporal classification of the different steps of cataract surgery: Yeh et al. trained a VGG16-LSTM to classify between 13 steps of cataract surgery[4] and Nespolo et al. developed real-time feedback system for cataract surgery using Faster R-CNN and optical flow which offers audiovisual warnings in the event of potentially risky instrument movements or decentration of the eye from primary position[5].

Finally, the CATARACTS Semantic Segmentation Grand Challenge 2020 and CADIS dataset lead to semantic segmentation models of surgical instruments and anatomical landmarks for cataract surgery[6]. We modified the current state-of-the-art, a ResNet encoder-OCRNet decoder model, to generate segmentation masks on our surgical videos[7].

4 Dataset and Features

Two ophthalmologists curated a dataset of roughly 200 surgical videos performed by resident and attending surgeons from different institutions as well as from YouTube, recorded from the surgical microscope and capturing the surgeon's point of view while operating. The surgical videos were manually trimmed and converted to series of JPEG images. Next, 1000 video frames were randomly selected from 40 videos and manually labeled using CVAT.[8] These images and their annotations in COCO JSON format [9] were used to train our CS231N keypoint detection model to generate trajectories of the utrada forcep tips. We also used a CaDiS-pretrained semantic segmentation modelMICCAI2021 Cataract Semantic Segmentation Model[7] from the Robotics and Vision in Medicine Lab at King's College of London to identify the positions of relevant anatomical landmarks.

We then used these two models to generate the inputs to our CS229 final project: CSV files containing frame-by-frame locations in X,Y coordinates of the utrada forceps tips, pupil center, pupil boundaries, and incision location for each individual 'pull' for 20 capsulorrhexis videos. Each 'pull' refers to a surgical movement in which the utrada forceps are used to pull on the anterior lens capsule flap and incrementally tear a circle in the anterior lens capsule. Our full analysis pipeline is shown in Figure 1.

4.1 Temporal Data

From the coordinate data, we generated temporally dependent features for each pull, such as average velocity, average acceleration, a custom feature we termed 'sliding window angle counts' (a measure of the smoothness or jaggedness of a trajectory), and the standard deviation of pupil center location. Average velocity was calculated as the mean L2 difference between consecutive video frames, and the average acceleration was calculated as the mean difference of consecutive velocities. For each consecutive triplet of frames, the sliding-window angle was calculated using the following procedure:

Sliding Angle Calculation

1. Collect the x,y coordinates of the forceps in the 3 consecutive frames $(x_a, y_a), (x_b, y_b), (x_c, y_c)$
2. Calculate the vectors between the center point to the other points
$$v_{ab} = \begin{bmatrix} x_a - x_b \\ y_a - y_b \end{bmatrix} \qquad v_{cb} = \begin{bmatrix} x_c - x_b \\ y_c - y_b \end{bmatrix}$$
3. Utilize the dot product to determine the angles between the consecutive points:
$$\theta_{abc} := cos^{-1}\left(\frac{<v_{ab}, v_{cb}>}{||v_{ab}||*||v_{cb}||}\right)$$

The angles are then separated into bins based on their values to create a feature of counts corresponding to the number of angles that were placed into each bin. As we will discuss later , through hyperparameter tuning our model, we determined that the optimal number of angle bins to utilize was 21.

4.2 Image data

Our second approach sought to look at the full trajectory of the forceps as an image relative to the pupil only, thus disregarding any movement of the eye or camera itself. The advantage of this method was that by using the pupil as a reference object, differences in resolution of videos and zoom of the camera would be normalized for.

In order to accomplish this, we used the generated keypoints for the pupil's horizontal and vertical extents in each frame to project the pupil onto a 100x100 matrix. The forceps keypoint position was then transformed and scaled to determine the index in the matrix it corresponds to (Figure. 2a). The value placed in the matrix was a "1" to binarily indicate that a location had been visited during a rhexis trajectory. Applying this procedure for all frames in a video provided visually distinctive trajectories between videos (Figure. 2b).

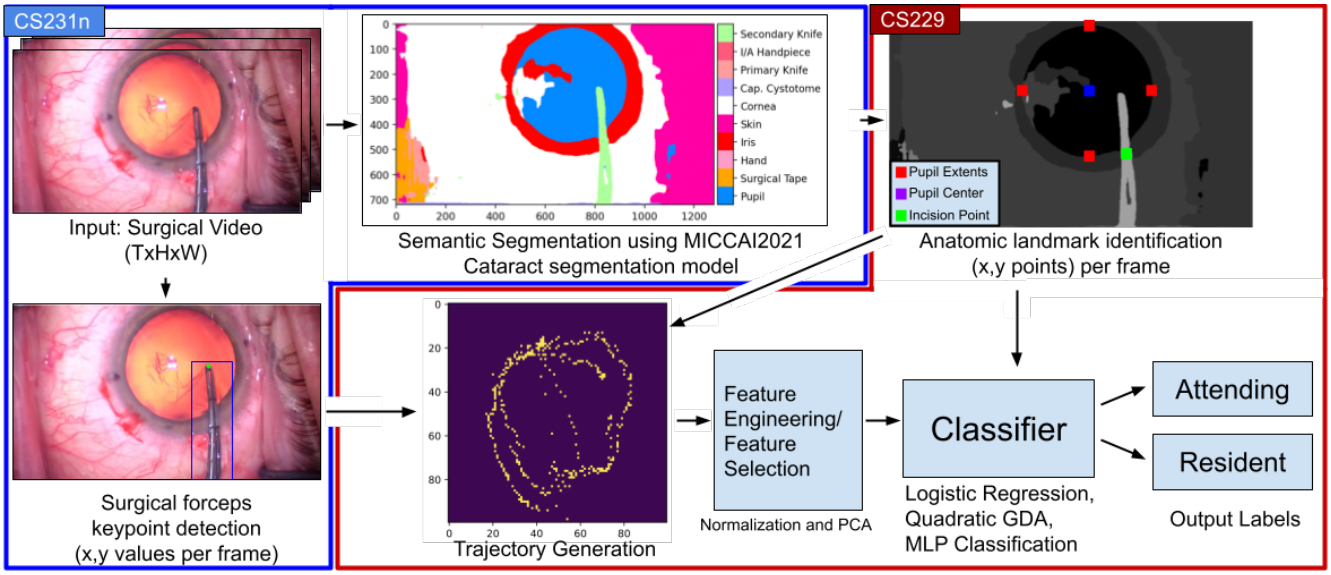
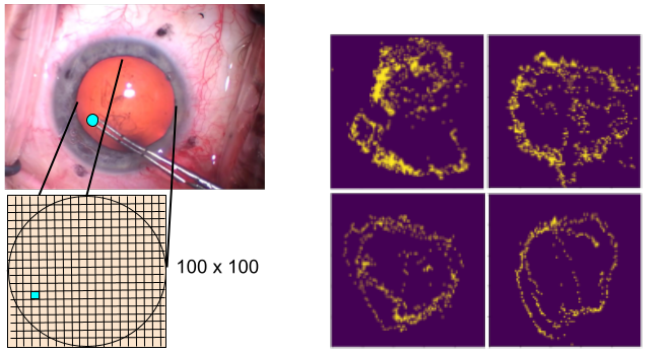


Figure 1: Illustration of our full methodology including both CS231N (blue) and CS229 (red) sub-projects.



(a) Projection of single frame onto 100x100 matrix

(b) Pupil-standardized trajectories from four videos

Figure 2: The pupil standardization process

5 Methods

5.1 PCA

To classify the trajectory images, a pipeline was built to flatten the images into a 1x10000 vector, normalize each pixel's mean and standard deviation, and apply a kernelized PCA prior to fitting to a classifier. The goal was to elucidate a set of "eigen"-trajectories that all other trajectories could be deconstructed as a linear combination of, thus determining a reduced feature space on which to do classification (Figure. 3). The kernel maps the existing features to another feature space where it may be easier for PCA to find high-variance and non-linear features, but the choice of kernel does influence the types of non-linearities which can be detected. Standard PCA was applied to the temporal data.

The kernels we tested were polynomial, linear, sigmoid, and radial-basis function (RBF). The polynomial kernel is defined as

$$K(x, y) = (a * x^T y + c)^d$$

and compares different polynomial combinations of vectors together. The linear kernel is a special case of the polynomial kernel where $d = 1$. The sigmoid kernel is

$$K(x, y) = tanh(a * x^T y + C)$$

which can mimic features from a multi-layer perceptron. Finally, the RBF kernel is defined as

$$K(x, y) = \exp\left(-\frac{\gamma * \|x - y\|^2}{2\sigma^2}\right)$$

and compares the similarity of two vectors' distance from a point. This is especially promising for our problem, since the trajectories tend to be quite circular.

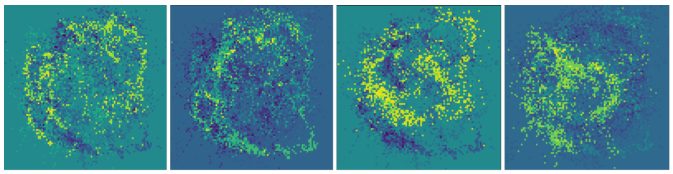


Figure 3: PCA of pupil standardized trajectory images with four components yields four "eigen"-trajectories

5.2 Classifiers

Logistic Regression In binary classification, logistic regression models the probability of one outcome of a binary event as the logistic function of a linear combination of one or more features x.

$$P(y = 1|x; \theta) = h_\theta(x) = g(\theta^T x) = \frac{1}{1 + e^{-\theta^T x}}$$

During training, the parameters θ are iteratively optimized via stochastic gradient ascent (shown below) or other methods to maximize the likelihood of the data.

$$\theta_j := \theta_j + \alpha(y^{(i)} - h_\theta(x^{(i)}))x_j^{(i)}$$

Quadratic Gaussian Discriminant Analysis Gaussian Discriminant Analysis is a generative learning algorithm: it models $p(y)$ and $p(x|y)$ and uses Bayes Rule to derive the posterior distribution on y given x.

$$p(y|x) = \frac{p(x|y)p(y)}{p(x)}$$

Gaussian Discriminant Analysis assumes that $p(x|y)$ has a multivariate normal distribution

$$p(x; \mu, \Sigma) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} exp(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu))$$

and models binary classification as

$$y \approx Bernoulli(\phi)$$

$$x|y=0 \approx (\mu_0, \Sigma_0)$$

$$x|y=1 \approx (\mu_1, \Sigma_1)$$

which has closed form solutions for the maximum likelihood estimate of the parameters $\phi, \mu_0, \mu_1, \Sigma_1$, and Σ_2.

Quadratic Gaussian Discriminant Analysis assumes different covariance matrices for different classes ($\Sigma_0 \neq \Sigma_1$) while Linear Gaussian Discriminant Analysis assumes that the multivariate normal distribution for each class shares the same covariance matrix ($\Sigma_0 = \Sigma_1 = \Sigma$).

Multilayer Perceptron Multilayer Perceptrons (MLP) are fully-connected feedforward neural networks that insert nonlinear activation functions between fully connected linear layers to add expressivity to the model, and are trained using backpropagation. Multilayer Perceptrons are universal approximators that can model any function given enough nodes and the right set of weights. Our implementation contains three hidden layers of 1000 neurons with ReLU activations, and is trained using the Adam optimizer and cross-entropy loss.

ReLU:	Cross entropy loss:
ReLU(x) = max(0, x)	$L_i = -\log(\frac{\exp f_{y_i}}{\sum_j \exp f_{y_j}})$

Adam optimizer:

while not converged **do**
- $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$ (gradient)
- $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1-\beta_1) \cdot g_t$ (biased 1st moment)
- $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1-\beta_2) \cdot g_t^2$ (biased 2nd moment)
- $\hat{m}_t \leftarrow m_t/(1-\beta_1^t)$ (biased-corrected 1st moment)
- $\hat{v}_t \leftarrow v_t/(1-\beta_2^t)$ (biased-corrected 2nd moment)
- $\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \hat{m}_t/(\sqrt{\hat{v}_t} + \epsilon)$ (update parameters)

end while

Histogram-based Gradient Boosting (HBGB) Gradient boosted trees build an ensemble of decision trees by sequentially constructing them to optimize for a loss function. At stage i of the training process for model F, it will add a new estimator h_i to account for the residual of F with the label y.

$$h_i(x) = F_i(x) - y$$

HBGB provides an improvement on this by binning features into discrete value sets, which enables more efficient construction of the tree structure. The approximation of the features may also have a positive effect on performance.

6 Experiments/Results/Discussion

6.1 Training, Validation and Test Datasets

From our auto-generated trajectories, we had a total of 79 samples to train on. We split 80% for training classifiers with 5-fold cross validation, and used the remaining 20% as a validation set that was used to evaluate each classifier trained. We used a 9 sample dataset of manually labeled trajectories as a golden test set to only be run on our classifier with the best performance.

6.2 Hyperparameter Tuning

In order to determine the best model and corresponding hyperparameters, we performed a grid search on a variety of possible parameters.

Angle Bins As a part of generating manual features for the data, we counted the number of angles in a dataset, binned across the range of 0°-180°4.1. We treated the number of bins as a hyperparameter of the temporal data, since too many bins may be too granular, but too few bins may mask important distinctions in trajectory variation. Across all classifiers, 21 bins was found to be the best performing parameter.

Kernelized PCA We tested the performance of different numbers of components, the type of kernel, and some kernel-specific hyperparameters. For polynomial kernels, we varied the degree of the polynomial. For the RBF kernel, we varied the gamma parameter. Over both image and numeric data, the linear kernel performed best. It was surprising that the RBF kernel was not able to outperform for the image classification, since the RBF is known to fit nonlinear data well. The optimal number of components varied between 2 and 10 based on the classifier.

Logistic Regression We varied the penalty for logistic regression between L1, L2 and Elasticnet. L1 regularization performed best. This may make sense since L1 promotes sparse parameter sets.

Quadratic Gaussian Discriminant Analysis We varied the regularization parameter between 1e-6 and 0.5. 0.4 or 0.5 gave best results.

Multilayer Perceptron (MLP) The learning rate of the MLP was tuned between 1e-9 and 0.1. We found 0.001 to be the best learning rate.

Histogram-based Gradient Boosting (HBGB) We tuned the loss function between log loss and cross-entropy, finding cross-entropy worked better. The best learning rate was found to be 0.01 with an L2 regularization of 0.3. The best minimum number of samples per leaf was 9, which made sense to be lower than the default 20, since we had fewer samples and would thus otherwise have shallow trees.

6.3 Results

After training and tuning our classifiers, we ended up with the following results in Table 1 on the validation dataset. The ROC curves for all models is also shown in Figure 4. The best model was the Histogram Based Gradient Boosting, with a validation accuracy of 0.89. Running this model on the test set yielded 100% accuracy.

Table 1: Best Validation Set Accuracy across all models on the image and temporal data

Model	Image Data	Temporal Data
Logistic Regression	0.5	0.78
Quadratic GDA	0.75	0.72
MLP	0.75	0.72
HBGB	0.50	0.89

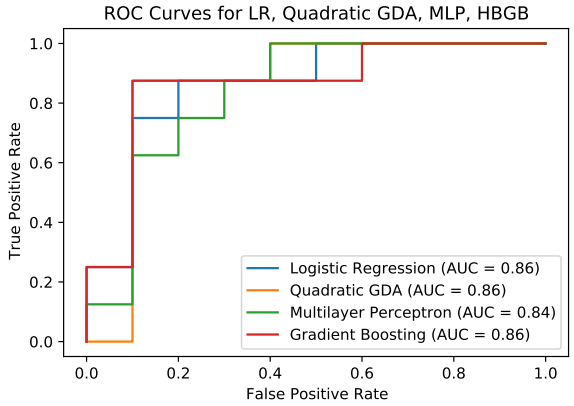


Figure 4: ROC curves for Logistic Regression, Quadratic Gaussian Discriminative Analysis, Multilayer Perceptron, and Histogram-based Gradient Boosting

7 Conclusion/Future Work

After experimentation with a variety of classification models and hyperparameters, we were able to achieve a test-set classification accuracy of 100.00% on our test set (n=9) with gradient boosting with L2 regularization of 0.3, learning rate of 0.01, binary cross-entropy loss, and a minimum of 9 samples per leaf.

To our knowledge, this is the first study to utilize instrument trajectories from keypoint detection and tracked locations from semantic segmentation to generate automated surgical feedback for cataract surgery. Still, there is much room for further improvement: Our annotated dataset is still quite small and sparsely annotated: our keypoint detection model currently only predicts keypoint and bounding box annotations of the utrada forceps and our semantic segmentation model uses several heuristics to estimate pupil center, pupil boundaries, and incision location from segmentation masks. To provide the most useful feedback to training cataract surgeons, we would want to add annotations capturing the location and orientation of the capsulorrhexis flap as well as the orientation of the utrada forceps itself.

We also hope to generate a richer set of metrics that better captures the details of important skills like 'pivoting' and 'floating' in the main incision, maintaining flap control by keeping the flap folded over and 'walking the dog', and releasing the flap in such a way that it is easy to regrasp.

Ultimately, our goal is to generate truly meaningful and actionable clinical feedback for surgical trainees, going beyond merely grading a surgical video to providing detailed explanations and points of comparison and ultimately giving trainees clear actions that can be taken to improve their technique for the next case.

For example, we want to go beyond simply telling a resident that they performed the capsulorrhexis poorly in a given case, to highlighting that they were releasing the capsulorrhexis flap in locations that are difficult to regrasp from [10], showing them that an attending will pre-emptively release and regrasp before these locations if they feel that they might lose control of the flap and be forced to regrasp at one of these danger zones, and thus providing them with clear actions for improvement (pre-emptively regrasp early at a more favorable location if you think you may be forced to regrasp in a danger zone).

We hope to eventually extend our work to all steps of cataract surgery, and anticipate that such a platform can help ophthalmology residents and fellows maximize their growth as surgeons from every case.

A Project Hyperlinks

- Rhexis Trajectory Project GitHub
  https://github.com/davidekuo/rhexis-trajectory
- Google drive link to our Dataset
  https://tinyurl.com/53zh3ezf

Contributions

- David Kuo: I curated and annotated most of the data and helped with feature engineering, model implementation, and model training.
- Riya Sinha: I wrote the transformation to create the trajectory images, and designed the pipelines for streamlined model building and testing.
- Ben Viggiano: I wrote a lot of the under the hood functions in our codebase; including the functions responsible for extracting features from the semantic segmentation model.

References

[1] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6):1137–1149, 2017.

[2] Amy Jin, Serena Yeung, Jeffrey Jopling, Jonathan Krause, Dan Azagury, Arnold Milstein, and Li Fei-Fei. Tool detection and operative skill assessment in surgical videos using region-based convolutional neural networks. *CoRR*, abs/1802.08774, 2018.

[3] Emmett D. Goodman, Krishna K. Patel, Yilun Zhang, William Locke, Chris J. Kennedy, Rohan Mehrotra, Stephen Ren, Melody Y. Guan, Maren Downing, Hao Wei Chen, Jevin Z. Clark, Gabriel A. Brat, and Serena Yeung. A real-time spatiotemporal AI model analyzes skill in open surgical videos. *CoRR*, abs/2112.07219, 2021.

[4] Hsu-Hang Yeh, Anjal M. Jain, Olivia Fox, and Sophia Y. Wang. PhacoTrainer: A Multicenter Study of Deep Learning for Activity Recognition in Cataract Surgical Videos. *Translational Vision Science and Technology*, 10(13):23–23, 11 2021.

[5] Rogerio Garcia Nespolo, Darvin Yi, Emily Cole, Nita Valikodath, Cristian Luciano, and Yannek I. Leiderman. Evaluation of artificial intelligence–based intraoperative guidance tools for phacoemulsification cataract surgery. *JAMA Ophthalmology*, 140(2):170, 2022.

[6] Maria Grammatikopoulou, Evangello Flouty, Abdolrahim Kadkhodamohammadi, Gwenol'e Quellec, Andre Chow, Jean Nehme, Imanol Luengo, and Danail Stoyanov. Cadis: Cataract dataset for image segmentation, 2019.

[7] Theodoros Pissas, Claudio S. Ravasio, Lyndon Da Cruz, and Christos Bergeles. Effective semantic segmentation in cataract surgery: What matters most? *CoRR*, abs/2108.06119, 2021.

[8] Intel Corporation. Computer vision annotation tool. `https://github.com/openvinotoolkit/cvat`, 2018.

[9] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context, 2014.

[10] Uday Devgan MD. Cataract coach 1186: Capsulorrhexis danger zones.