
Image processing pipeline: from raw to archival and web

Cuneiform Digital Library Initiative: CDLI

BASIC INFORMATION

Name: Aman Singh

Email: amanggnlps@gmail.com

Github/Gitlab: @amansheaven

LinkedIn: www.linkedin.com/in/amansheaven/

Phone Number: +91-8743040743

Country / Region : Delhi, India

University: The NorthCap University

Field of Study: Computer Science Engineering (Batch of 2020)

EXPERIENCE

- **Enactus NCU:** Worked as VFX Artist, graphic artist assisting them with social media campaigns to project presentations. Worked extensively with RAW Images, Photoshop. Batch Processing of images.
- **Eckovation.com:** Worked as a Full Stack Developer intern working on their B2C product Eckovation.com, working on distributed systems. Worked extensively on Node.js, CircleCI, API testing.
- **Progilence Capability Development Ltd:** Worked as a technical research intern for a computer vision project, worked on Matlab, OpenCV, Ski-Image, and Python.

PROJECTS

- **WebOpenCV:** Implemented a simple image inversion program, that utilizes server side image processing with its server via sockets.

PERSONAL MOTIVATION

I have been working on OpenCV and image processing since last year. This project gives me the complete freedom to design and make the image pipeline with my learning and shortcomings. I think with my functional knowledge and previous development experience really encourages me to pick this project up. With the inputs of my mentors and my hard work, I would deliver high throughput and continue to support CDLI even after GSOC.

PROJECT DETAILS

Introduction

CDLI is an international project aimed towards indexing text and images of historical relics digitally. It currently catalogs more than 500,000 tablets distributed over museums and archives. This project is collaborated by leading experts from the field of Assyriology, curators of European and American museums. The electronic documentation contains text & image, combines with document transliterations, text glossaries and digitizes originals and photo archives of cuneiform. Currently, data format includes ATF and CDLI-CoNLLs, and lossless images.

Currently, CDLI is working on a framework that would let users easily write a summary of these tablets over a simplified interface. A large part of the framework efficiency depends on efficient image processing. The current flow of image processing is as follows :



While this method results in better outputs it comes with the following problems:

- This created a hectic environment on the person who is involved in manual editing and dependency which can not be averted.
- This also forces people to adopt the skill of graphic editing.
- Hinders empirical automation in the framework.

Technical Analysis

Image Processing

As per the information from the mentors, the images are majorly generated using a **flat-bed scanner** or **conventional photography** setup. The result is usually 6 images of the tablets corresponding to each side. On analyzing the problem closely it's actually an image segmentation problem. The images in the provided dataset are TIF(F), which is a lossless image format.

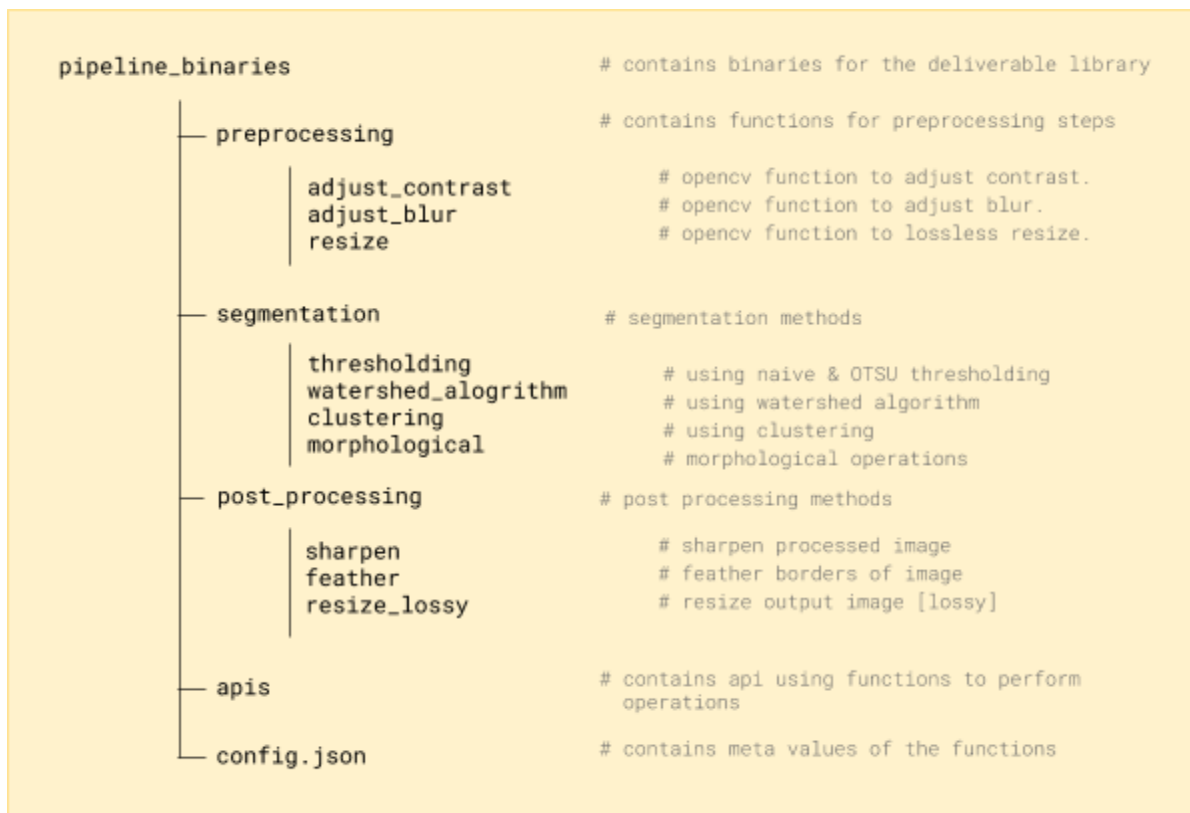
Web Interface

The second part of the problem is web-part. This involves building an application which can alone generate the Artifact Individual View of the tablets. This part should be designed that way that there is an abstraction between the applications and the core library which processes the image because this part also needs to be merged in the framework.

Deliverables

Image Processing Library

This library contains four modules namely **preprocessing**, **segmentation**, **post-processing**, and **compression**. These modules would be written in **python**, for easy migration between the framework. This library would contain the scripts needed to make manipulations on the images.

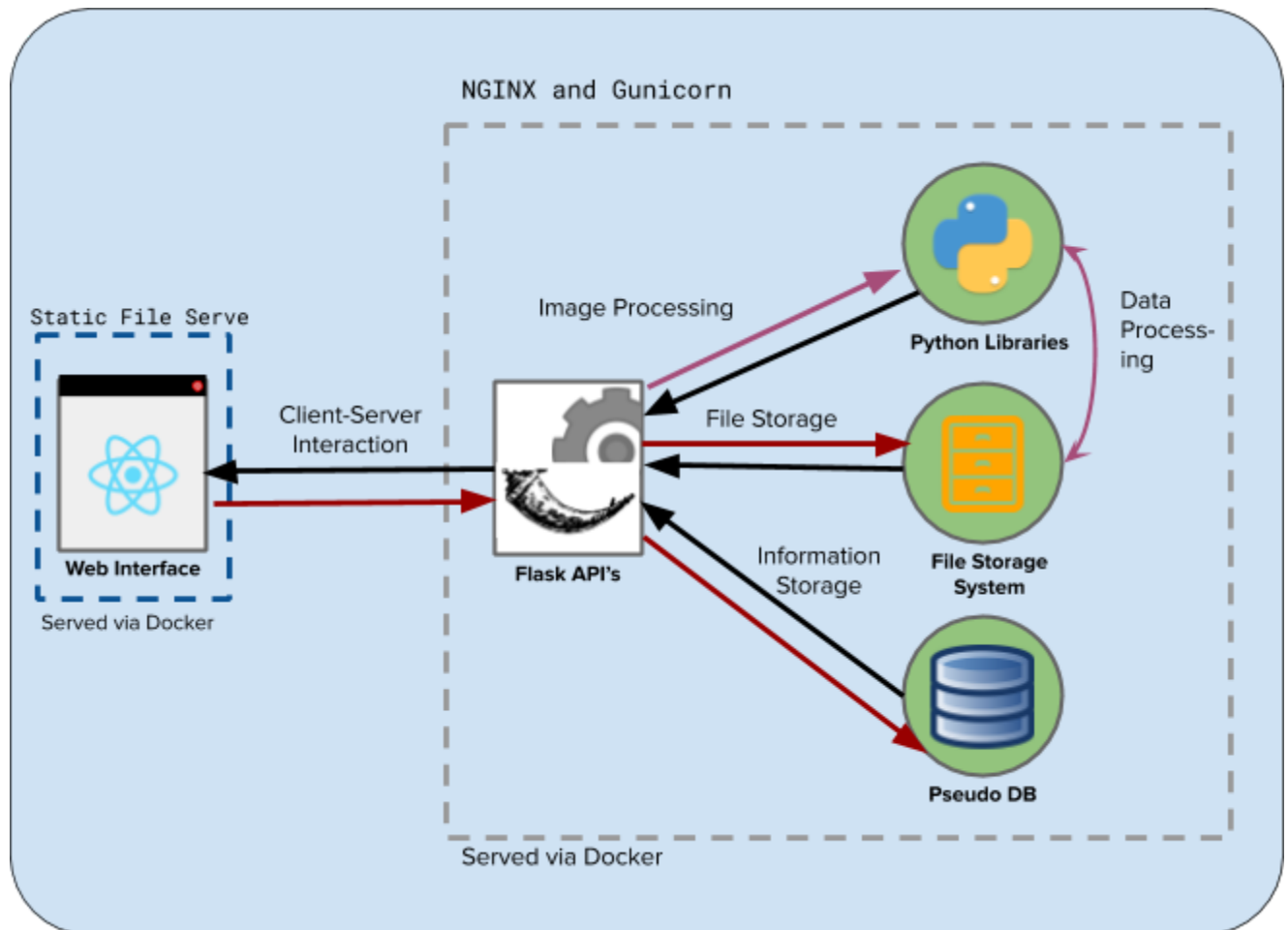


Web Application

The web application part of the project would deal with the interaction between the final user and the binaries, this web application is meant to be used by users with limited or no functional knowledge on processing and web infrastructure. The whole application would be wrapped and shipped via docker to make DevOps and further development easy.

These stakeholders have three functions to perform on the application namely:

- Process batch images of the tablets.
- Batch Renaming of the images set
- Making archival & web format outputs of the processed images.
- Generating the output that can be used further by the web admins ie: JSON, CSV, TSV of the metadata of the images.
- Edit and format the generated archival.



DETAILED EXPLANATION

Image Segmentation

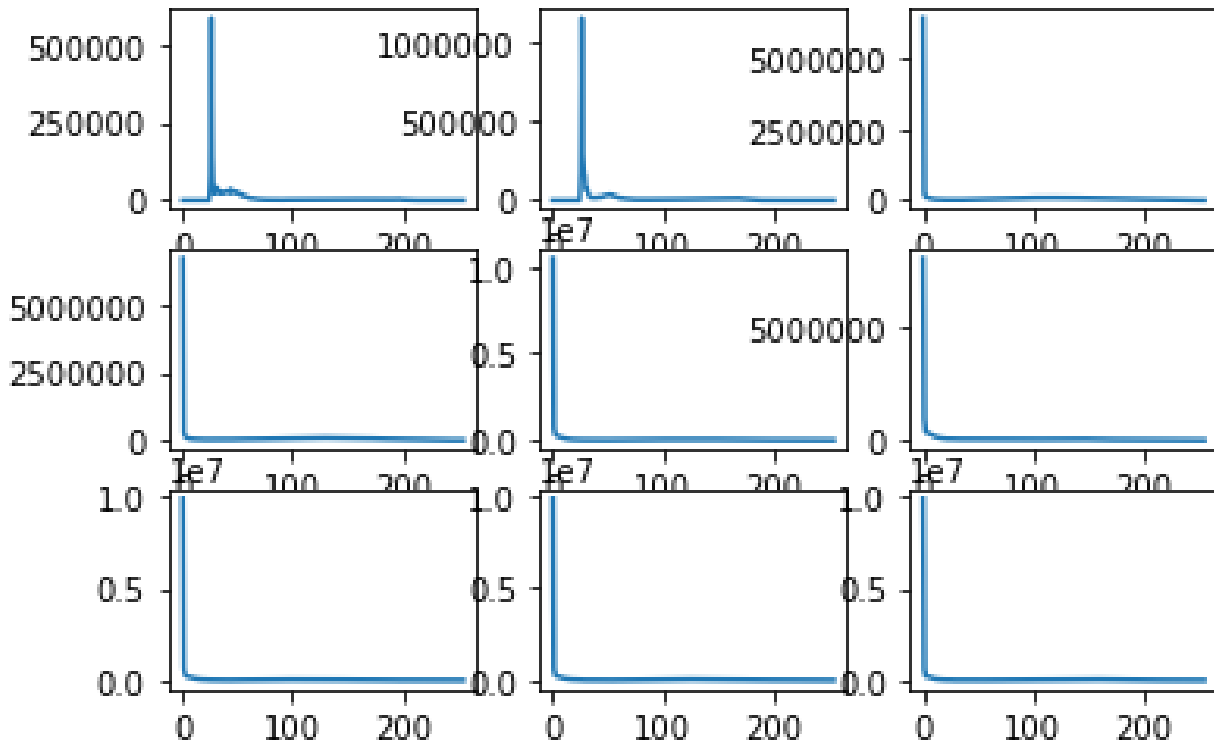
In the field of computer vision image segmentation is one of the prominent subprocesses, it is exhaustively used in deep learning and analysis. There are a lot of methods of image segmentation that can be used according to the corpus and data to be analyzed. There are a variety of techniques that are discussed in the research papers linked down in resources.

The following table depicts methods that can be used while dealing with an image segmentation problem :

Segmentation Technique	Description	Advantages	Disadvantages
Thresholding Method	Based on the histogram peaks to find particular threshold values.	Simplicity, no need for previous information.	Depends on the peak, Spatial details are not considered.
Edge Based Method	Based on discontinuity detection.	Works better images having good contrast.	Detects too many edges.
Region-Based Method	Based on partitioning the image into homogenous regions.	More immune to noise,	High time and space complexity.
Clustering Method	Based on the division into homogenous clusters.	Fuzzy based clustering.	Membership defining function is not easy.
Watershed Method	Work on topological interpretation.	Stable results detected continuously	High time complexity.
PDE Based Method	Based on the working of differential equations.	Fastest method, Best for time-critical application.	High time complexity
ANN Based Method	Based on the simulation of the learning process for decision making	Easy to use	Training required.

The following information is cited from the research paper {{2}},

From the set of images, we can assess a few typical properties of the images, the very first is the histogram of all the images follow a high peak of values apart from being scattered, As the conventional photography and image capture techniques these images with a dark background.



The following image shows the histogram of all the images provided in the dataset all of these images have a particular peak at some value declaring that there is a contagious homogenous color in a particular element.

Conclusions:

1. The images have a common pattern in which all the images are clicked over a particular background which contrasts the object.
2. For such kind of problems the following methods can be used:
 - Thresholding Image
 - Watershed Algorithm
 - PDE Methods
 - Clustering Methods
3. The two parameters that would depict the accuracy of the system is **speed** and **output**. The system would process heavy TIF(F) images and should be fast enough to carry out these in a batch with a good output.

Web Application

The web application part of the project is focussed on better UI and lightweight nature, to serve an application uses a lot of resources at the backend. After thorough research, the following stack does seem to help in our functionalities. The stack selection is as follows:

1. Image Processing Backend:

The application extensively uses and repeatedly needs to use these binaries, it would be fair to say that the performance of the application depends on these binaries.

Most of the computer vision libraries come with bindings for two popular programming language :

a. Python:

Python is one of the fastest-growing languages and is one of the preferred languages of machine learning and artificial intelligence.

Apart from being a scripting language, it is strongly typed and easy to grasp. Python has been used in development with the help of packages like Django or flask.

b. C++/CPP:

CPP is a rather older language than python, it is used heavily in embedded systems and systems which need raw performance. CPP is a compiled language hence is strongly typed, and hence faster as compared to python.

Popular computer vision libraries adopted CPP since the origination clause of the throughput it delivers, but the lack of suitable packages to extend this functionality is one factor that makes it less adaptable.

2. Framework Integration:

The web framework is responsible for the UI components and supports complex server-side integrations. The current framework is based on PHP, so we need to create an easy UI for users to process the images while having the integration intact.

TIMELINE:

Pre GSOC: March 7 - May 4

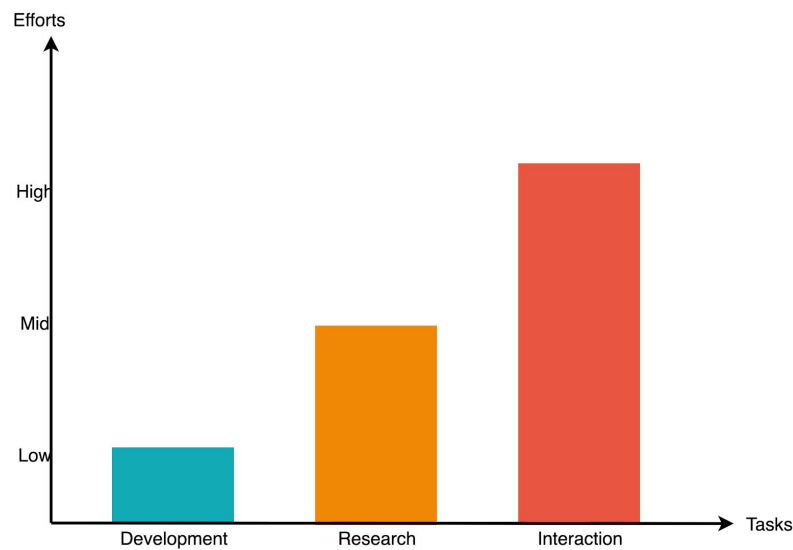
Implement algorithms dealing with the segmentation, using watershed, thresholding, and clustering techniques. Have python implementation for preprocessing and postprocessing methods with appropriate configs and tested parameters for preprocessing images.

Community Bonding Period: May 4 - June 1

I plan to use the community bonding period to research the way assyriologists, the way they document the tablets, what do the different parameters mean in the archival formats and most importantly how are these images captured and try to work my current algorithms on a small diverse dataset. I would utilize this period to work tentatively on my algorithms perfecting the outcomes.

The primary goal in this period would be:

- Familiarizing with the team.
- In-depth understanding of Archival Format.
- Understanding Photography methods.
- Compiling a diverse test dataset.



Phase 1: June 1 - June 29

Week 1 :

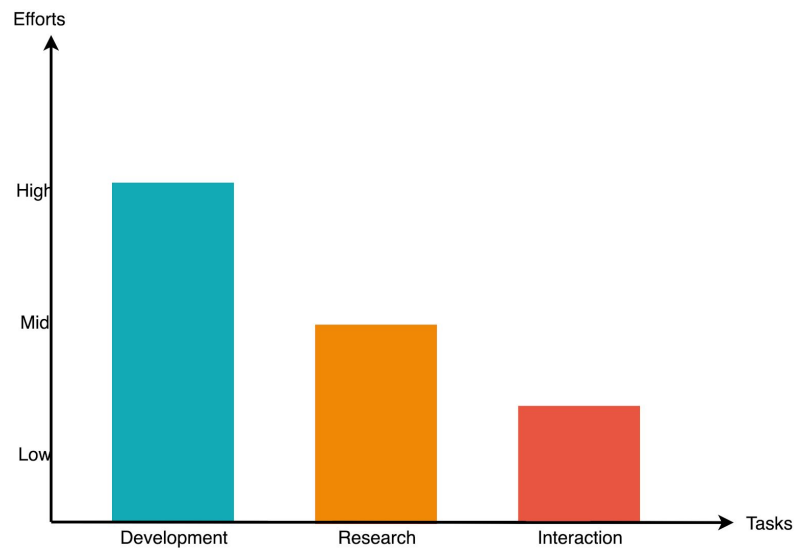
- Building and perfecting algorithms, pre-processing modules and post-processing methods.
- Implementing python scripts for modules and methods.
- Writing documentation for the implemented methods.

Week 2-3 :

- Basic Flask deployment of modules.
- Shipping Flask with Docker.
- Initializing API calls.

Week 4 :

- Implementing live hooks for editors.
- Documenting API calls and initiating frontend repository.
- Security measures implementation for images to be uploaded.
- Writing test cases for backend



Phase 1 Evaluation: June 29 - July 3

Phase 2: July 3- July 27

Week 1 :

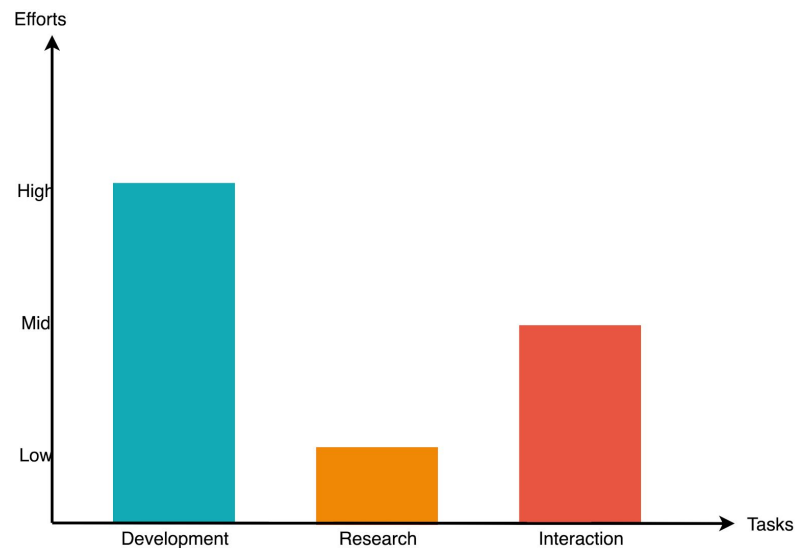
- Start integration work with the framework
- Define styles and components.

Week 2-3 :

- Completing Frontend and connection with backend
- Checking deliverables, working on saturating the deliverables
- Docker Wrapping of frontend

Week 4 :

- Working on dev-ops for the particular project.
- Deploying packages and docker files online and testing.
- A sample server deployment via Heroku + Netlify.



Phase 2 Evaluation: July 27 - July 31

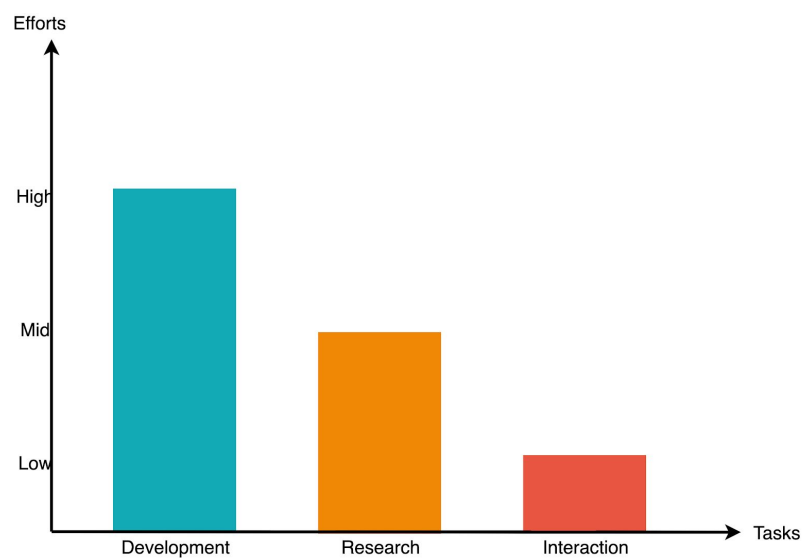
Phase 3: July 31 - August 24

Week 1-3 :

- Working on integration of the application with the framework.
- Connecting the library with the codebase.
- Writing test cases for the connection and testing efficiency.

Week 4 :

- Working on stretch goals.



Phase 3 Evaluation: August 24 - August 31

Program Completion: August 31 - September 7

Stretch Goals:









CDLI Catalyst

- Standalone binaries for processing images on local machines.
- No internet connectivity required.
- Batch processing multiple images, with user-specific filters.

Persistent Storage Cluster

-
- The current systems use a native file storage system to process and handle images.
 - This is a fast method but not for persistent storage.
 - Ceph is an open-source Amazon S3 like file storage solution which can be deployed via docker to control files.
 - This would let users easily manage files and maybe help them serving these files into their own applications

CONTRIBUTIONS

	cdli-gh / pyoracc	 #45	Merged
	cdli / framework	 ! 73	Merged
	cdli / framework	 ! 89	Merged
	cdli-gh / atf2conll-convertor	 #18	Open

Some Screenshots of Current Progress in Image Segmentation:

