# INTERNSHIP REPORT

## PROJECT TITLE

**Optical Character Recognition on Lecture Videos for enrichment of Automatic Speech Recognition**

Report by **NISHANT MISHRA**

Supervised by

- Dr.Christine SENAC
- Dr.Benjamin BIGOT

Team  SAMOVA

# Table of Contents

## Acknowledgement

With immense gratitude I would like to thank the whole team SAMOVA at IRIT for providing me the opportunity to carry out this internship.I would also like to thank my project supervisors here Dr.Christine Senac and Dr.Benjamin Bigot and my guide at BIT Mesra,Dr.Mahesh Chandra  for their constant support and guidance throughout the project without which this would not have been possible.I also acknowledge Dr.Benjamin for the carefully designed Roadmap which kept me motivated.

There is still a lot of work to be done and I have pointed the way forward through this report.

## Abstract

In this project,optical character recognition was performed on lecture videos belonging to a variety of topics.The precision and recall of the video OCR system was evaluated.The results obtained from OCR were then used to enrich the lexicon of Automatic Speech Recognition system and the improvement in the ASR on these videos due to the OCR was evaluated.

# INTRODUCTION

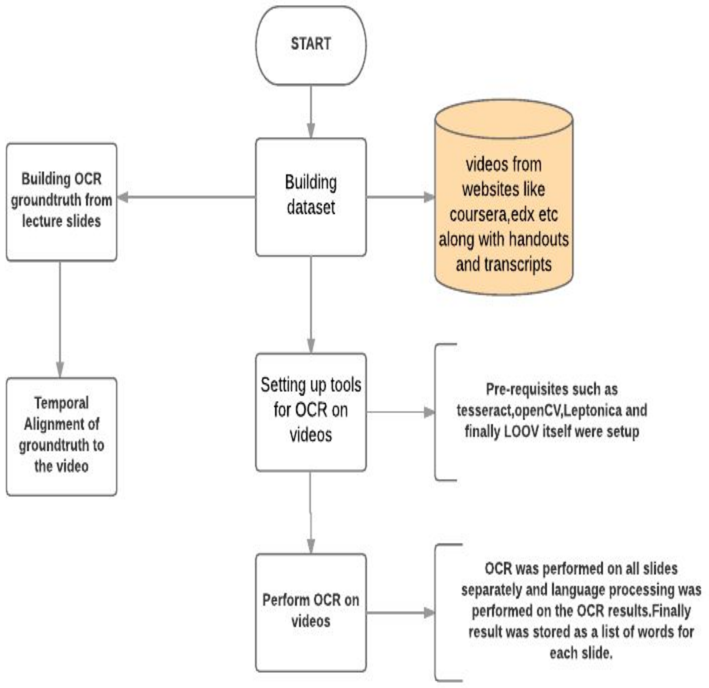
Nowadays,online courses and MOOCs are really popular. There is a huge content in terms of courses and lectures available on the internet related to various topics catering to a large community of people. Performing Automatic Speech recognition on these lectures for transcription and indexing is a bit difficult because different videos have a specific set of words depending on the domain of the video called jargon,which are not present in general lexicons we use to train speech recognition models.

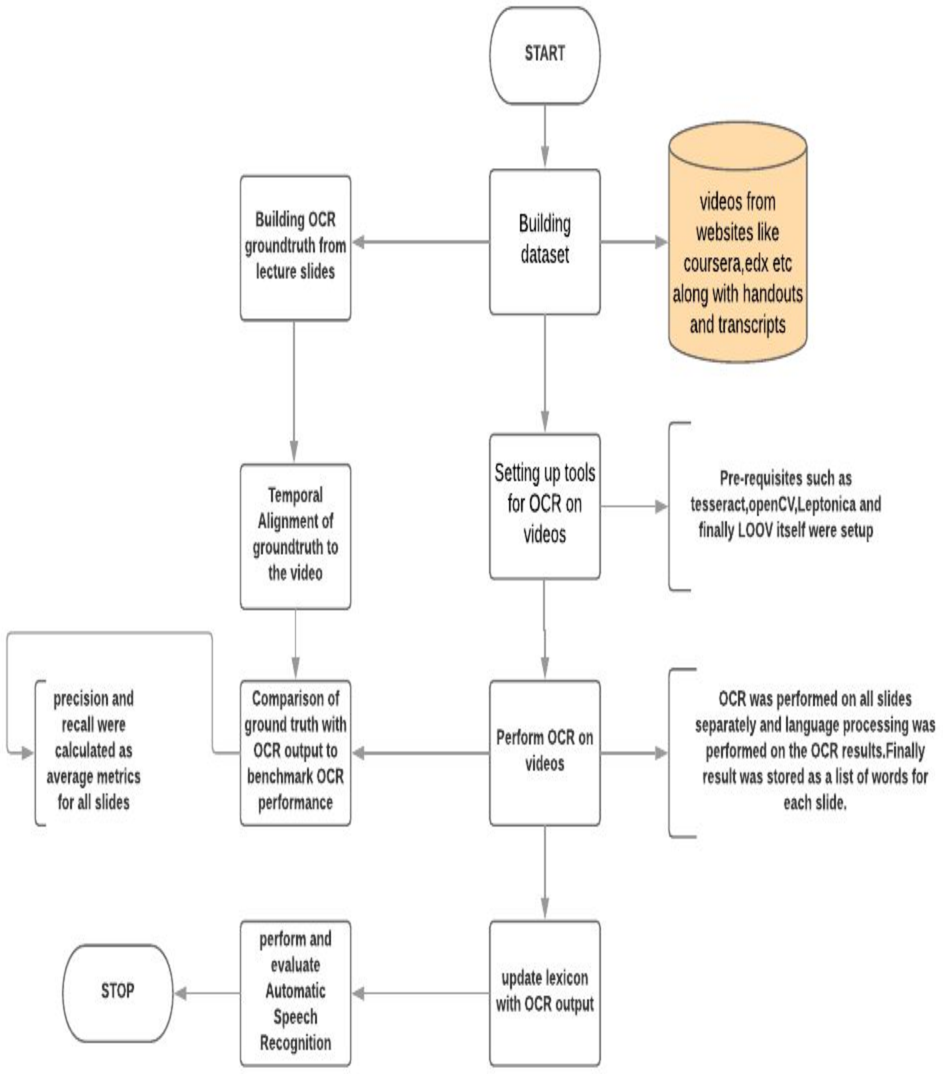
Therefore the lexicon needs to be updated with these domain-specific words/phrases for better ASR performance.Through this project,I attempt to use the results obtained from performing OCR on these videos to enrich the lexicon for speech recognition and observe the improvement,if any,in the ASR result.

## ROADMAP

Exploiting Overlaid text for ASR enrichment in Video lectures

Nishant Mishra | September 12, 2017


START
Building dataset
videos from websites like coursera,edx etc along with handouts and transcripts
Building OCR groundtruth from lecture slides
Temporal Alignment of groundtruth to the video
Setting up tools for OCR on videos
Pre-requisites such as tesseract,openCV,Leptonica and finally LOOV itself were setup
Perform OCR on videos
OCR was performed on all slides separately and language processing was performed on the OCR results.Finally result was stored as a list of words for each slide.


START
Building dataset
videos from websites like coursera,edx etc along with handouts and transcripts
Building OCR groundtruth from lecture slides
Setting up tools for OCR on videos
Pre-requisites such as tesseract,openCV,Leptonica and finally LOOV itself were setup
Temporal Alignment of groundtruth to the video
Perform OCR on videos
OCR was performed on all slides separately and language processing was performed on the OCR results.Finally result was stored as a list of words for each slide.
Comparison of ground truth with OCR output to benchmark OCR performance
precision and recall were calculated as average metrics for all slides
update lexicon with OCR output
perform and evaluate Automatic Speech Recognition
STOP

## DATASET

We made our own custom dataset for this project. We collected video lectures and corresponding subtitles and slides from different courses taken from multiple sources involving a wide variety of topics in order to provide a balanced conclusion.

## OPTICAL CHARACTER RECOGNITION

Optical Character Recognition refers to the detection localisation and subsequent extraction of all textual information present in any digital entity such as Images,videos by computer.OCR can be performed by traditional Image processing based methods that work based on features such as edges,pixel density,contrast and sharpness or by state of the art Neural network based models trained on a large dataset. Although the traditional methods are very engineered and not very generic they are still in use because they prove to be more accurate for specific use cases and the data available to train Machine Learning based models is still not extensive.All in all OCR remains one of the most complex computer vision problems.

In our case,it was even more daunting a task considering we needed to perform OCR on videos.

## LOOV:A step towards video OCR

Any OCR pipeline involves three steps:

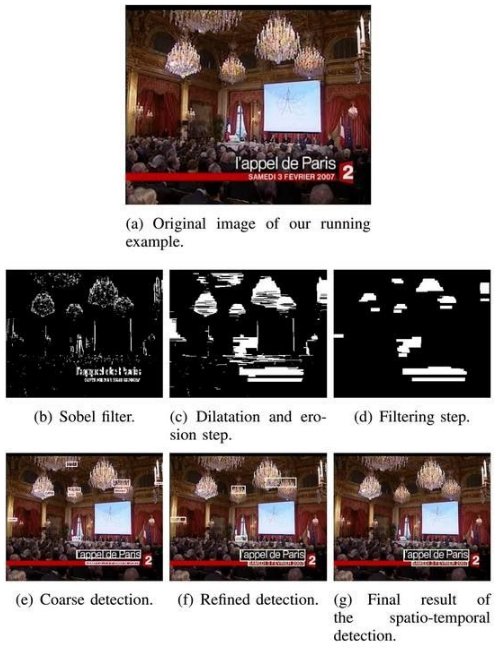
- TEXT DETECTION
- TEXT LOCALISATION
- TEXT RECOGNITION

Poignant et al in their paper([https://ieeexplore.ieee.org/document/6298510](https://ieeexplore.ieee.org/document/6298510)) titled **“From Text Detection in Videos to Person Identification”** proposed a unique solution for this problem of performing video OCR.Here they proposed a three step process:

- TEXT DETECTION

  They implemented text detection in a two step process

  1. COARSE DETECTION: Coarse detection involved detection of text in video using horizontal sobel filter on each frame of the video followed by binarization, an operation of horizontal dilation and erosion that connects characters of a same string.A filtering step cleans the image with vertical and horizontal erosion and dilation . In the resulting image, we select the areas that satisfy geometric constraints. We thus obtain a coarse detection of the texts boxes coordinates as shown in fig



(a) Original image of our running example.

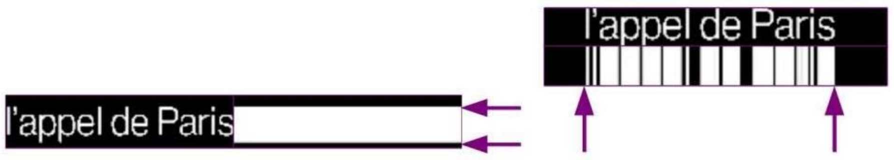
(b) Sobel filter. (c) Dilatation and erosion step. (d) Filtering step.

(e) Coarse detection. (f) Refined detection. (g) Final result of the spatio-temporal detection.

Figure 2. Different steps of spatial-temporal detection.

  2. FINE DETECTION: A second detection is performed on each individual text box. After binarization using the Sauvola algorithm, the number and the area variance of connected components allows us to detect if the text is written in

black on white or vice versa in the binarized image. This fine detection allows us to filter boxes that do not respect the text geometrical constraints.



(a) find Y coordinates (b) find X coordinates

- Text tracking on successive frames

  The next step takes advantage of the fact that a given text appears identically on many successive frames. The temporal information is used to filter out false positive boxes but also to recover boxes for which the detection locally failed.This accounts for the temporal information,and in turn,the effectiveness of this method on videos.

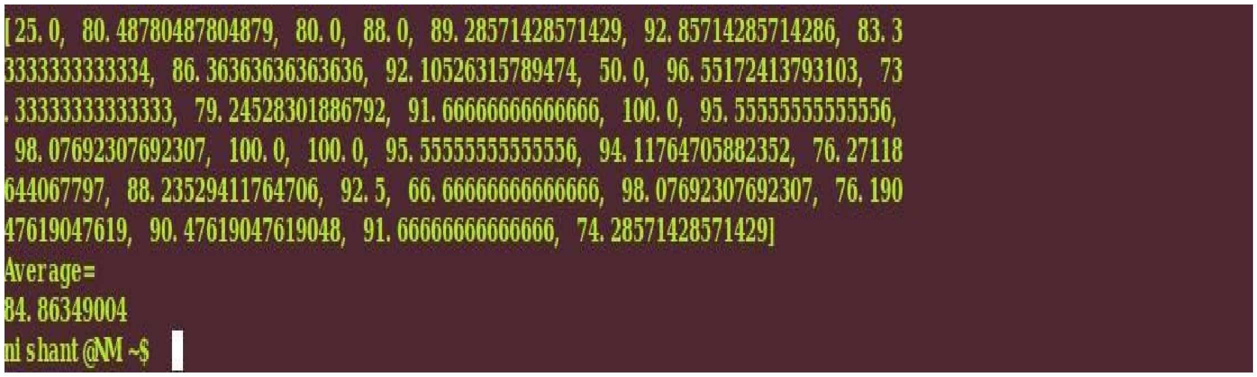
- Adapting images boxes for the OCR software and combination of multiple transcriptions

  After the text detection step, we need to adapt text box images to the OCR software. We artificially increase the resolution of these images with a bicubic interpolation. Next we apply a binarization on images using a threshold calculated with the Sauvola algorithm. To enhance the quality of the text transcription, we apply the OCR on several images, for a same box, temporally shifted. the first transcription corresponds to the global average image the others correspond to the transcriptions temporally shifted. The Viterbi algorithm is used to select the most probable path from among the different mesh networks(hypotheses).

  The recognition engine being used here is Tesseract 3.0 ,an open source OCR engine by microsoft that has been trained on a large amount of data

## STEP BY STEP

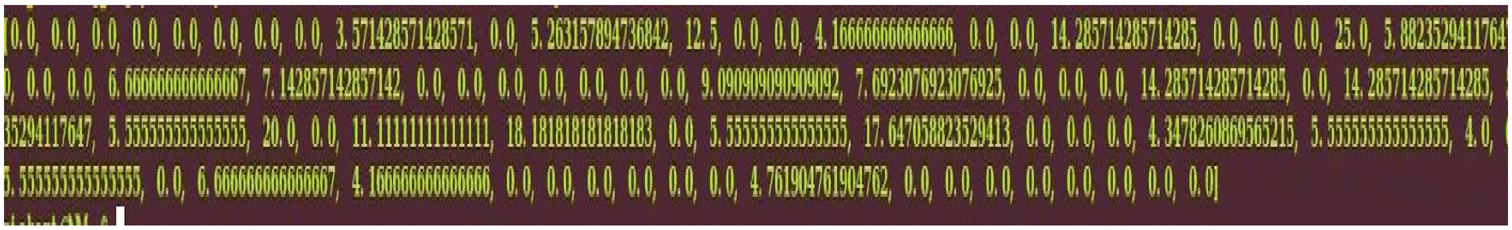
- We first collected the data in the form of videos,their corresponding slides and handouts and the subtitles.
- We then proceeded to process the raw data to create ground truth.Here we used the Apache Tika output to extract all text from the slides and arranged them in the form of list of lists with words for each slide as a list
- We made a semi manual program that we could use to mark the start and end of each slide in a video before performing OCR on the videos
- Then we started to work on extracting video text using LOOV,and store the output again in the form of lists of lists with words for each slide as a list.
- Then we benchmarked the LOOV performance by comparing its output to the groundtruth.We obtained remarkable results on our data with >84 percent OCR accuracy for all slides in the video combined.



```
[25.0,  80.48780487804879,  80.0,  88.0,  89.28571428571429,  92.85714285714286,  83.3
3333333333334,  86.36363636363636,  92.10526315789474,  50.0,  96.55172413793103,  73
.33333333333333,  79.24528301886792,  91.66666666666666,  100.0,  95.55555555555556,
 98.07692307692307,  100.0,  100.0,  95.55555555555556,  94.11764705882352,  76.27118
644067797,  88.23529411764706,  92.5,  66.66666666666666,  98.07692307692307,  76.190
47619047619,  90.47619047619048,  91.66666666666666,  74.28571428571429]
Average=
84.86349004
nishant@NM ~$
```

- Now that we had established and benchmarked LOOV's performance for vIdeo OCR,we proceeded to create speech ground truth ie the subtitles which based on timestamps were then aligned according to the frame lengths of various slides in the video

- Finally we compared the ocr processed text excluding garbage and repetitive words with the subtitles for all slides in order to come up with a metric of sorts that told how many additional words are there in the extracted OCR output that were not present in the subtitles,which in other words,are absent from the speech lexicon and upon population might improve the Automatic Speech Recognition performance by a certain degree.
- Here is a sample percent of additional words for each slide in a video,variable in nature but nearly 10% average



```
[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 3.571428571428571, 0.0, 5.263157894736842, 12.5, 0.0, 0.0, 4.166666666666666, 0.0, 0.0, 14.285714285714285, 0.0, 0.0, 0.0, 25.0, 5.882352941176
0, 0.0, 0.0, 6.666666666666667, 7.142857142857142, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 9.090909090909092, 7.6923076923076925, 0.0, 0.0, 0.0, 14.285714285714285, 0.0, 14.285714285714285,
35294117647, 5.555555555555555, 20.0, 0.0, 11.111111111111, 18.181818181818183, 0.0, 5.555555555555555, 17.647058823529413, 0.0, 0.0, 0.0, 4.3478260869565215, 5.555555555555555, 4.0,
5.555555555555555, 0.0, 6.666666666666667, 4.166666666666666, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 4.761904761904762, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
```

- Due to time crunch we could not concretely test our hypothesis on an actual ASR system.

## STATE OF THE ART AND WAY FORWARD

With newer OCR technology available now,more data,better networks and better algorithms for example,say,EAST or Connectionist Text Proposal Network for text detection,advancements of LSTMs(also used in tesseract 4.0) and other seq2seq network s we now have much more powerful,almost human level OCR capability at our hands,which has not necessarily been tapped for this multi-modal end,whereby it operates in conjunction with speech recognition for improvement of results.

Further with networks such as bilinear convolutional networks such as those used for Visual Question Answering we can develop feedback based networks that perform OCR and Speech recognition together,each improving the other.

Having said that,with better speech recognition implementations coming through that are learning to pronounce or recognise  words on their own based on spectral properties or based just simply on the block of speech,Wavenet being a case in point,we might no longer need to worry about domain specific words or any word,for that matter.

## UPDATE:PYLOOV

The LOOV developers also came up with a python version of the same called pyloov which is more convenient while using it for machine learning purpose.It had some issues with the implementation,which we managed to fix.Just like loov it uses tesseract at its heart for as text detection engine,for python,we use pytesseract which is basically a python wrapper for tesseract

## CONCLUSION

So,to conclude,over the course of the project we tried to come up with the best OCR solution for extraction text from videos.LOOV turned out to be the best suited for our use case.The accuracy we benchmarked on our data for LOOV was around 80%.

The ground truth generation remains a problem as it’s a difficult task to recognise when a slide changes,usually categorised as a computer vision problem called scene change detection and is still an open problem.As of now we made do with a semi-manual process whereby a person marked the beginning and end of each slide.Although ,this will no longer be a problem when we consider entire videos at a time.

Upon subsequent analysis,we found out that using OCR output to populate ASR response (subtitles here) does enrich it with improvements reaching as high as 20% in specific cases,with an overall average being 10% on our data

## REFERENCES AND BIBLIOGRAPHY

- R. Lienhart and W. Effelsberg "Automatic text segmentation and text recognition for video indexing " ACM/Springer Multimedia Systems pp. 69-81 1998.
- Poignant et al. - 2012 - From Text Detection in Videos to Person Identification,2012 IEEE International Conference on Multimedia and Expo
- Haojin Yang and C. Meinel, "Content Based Lecture Video Retrieval Using Speech and Video Text Information," in IEEE Transactions on Learning Technologies, vol. 7, no. 2, pp. 142-154, 2014.
- Shetty, Shashank, et al. "Ote-OCR based text recognition and extraction from video frames." Intelligent Systems and Control (ISCO), 2014 IEEE 8th International Conference on. IEEE, 2014.
- Yang, Haojin, et al. "Automatic lecture video indexing using video OCR technology." Multimedia (ISM), 2011 IEEE International Symposium on. IEEE, 2011.
- Smith, Ray. "An overview of the Tesseract OCR engine." Document Analysis and Recognition, 2007. ICDAR 2007. Ninth International Conference on. Vol. 2. IEEE, 2007

## CODES

All the programming for the project was done using python with an subprocess call being made to use LOOV(written in C).Here is a very brief walk through of the programs involved.

***wordprocessing.py***- makes a list of list of all words in each page of a slide/pdf document
**command line**
*python wordprocessing.py -t [address to the text document(output of apache-tika)] -n(optional.define in case you want numbers)*

**NOTE**:the text document should be the output of tika in structured-format for detection pf slide changes.

***ground_truth.py***-for building the groundtruth.It uses the wordprocessing script to build a list of words for each slide and opencv for alignment of slides with video.Outputs a json file containing list of lists,each list containing the starting and ending frame numbers,the start and end position in video in msec,for each slide in video along with all the words in the slide.
**command line**
*python ground_truth.py -v [specify the path to video] -o [specify the path to output and name of the output file] -t [address to the text document(output of apache-tika)] -n(optional.define in case you want numbers)*
upon running this command a window opens playing the video.Use **space** to pause the video.Once paused, you can assign a frame as starting by pressing the key **'s'** and as ending frame by pressing the key **'e'**.Press **space** again to play the video and repeat the steps for next slides.The video will automatically stop once we have obtained starting and ending frames for each slide,or if the video playback time is exceeded,in which case unmarked slides will have 0 as starting/ending frame numbers.In case you want to exit midway,just press the key **‘q’** while video is running.

**NOTE**: *requirement openCV
*the text document should be the output of tika in structured-format for detection of slide changes.

***loovprocessing.py*** script to run loov on the video for each slides and save the loov output for each slide in a directory.Then process the loov output to obtain a list of words and evaluate(find average and individual precision and recall) by comparing with the groundtruth.

**command line**
*python loovprocessing.py -v [specify the path to video] -g [path to the groundtruth,ie the json file we obtained from ground_truth.py] -l [specify path to build directory of LOOV ending with '/',e.g /home/here/is/LOOV/build/] -o [specify the output directory ending with '/',e.g /home/my/output/directory/]*

output of this program- *loov output of each slide is saved sequentially as .OCR files in the output directory
*list of all words in loov output for all slides is saved as list.json in the output directory
*list of individual precision for each slide(printed on command prompt)
*list of individual recall for each slide(printed on command prompt)
*average precision(printed on command prompt)
*average recall(printed on command prompt)
*total number of words in ocr output for the video(printed on command prompt)
*total number of words in the groundtruth for the video(printed on command prompt)

**NOTE**-in certain cases,there might be a single slide covering multiple videos,in that case use cat to combine the videos using the command
*ls 01.mp4 02.mp4 | perl -ne 'print "file $_"' | ffmpeg -f concat -i - -c copy* Movie_Joined.mp4
01.mp4 and 02.mp4 are the videos to be joined and Movie_Joined.mp4 is the name of the new combined video.
similarly there might be multiple slides for one video,we can use cat to combine all the slides for a particular video.

**Edit**
**convert.py** to convert a video from color to binary.A makeshift solution to address the LOOV problem where color text was not being recognised

**<u>commandline</u>**
python convert.py -v [*specify the address of the input video*] -o [*specify output video name with path*] -b [*window size while finding threshold in adaptive thresholding*]

Once you run this program the video plays in binary and once the video playback is complete,you will get an output video file which is the binary version.In case you are not happy with the binarization you can stop the program midway by pressing **space** and then make suitable changes to the parameters and rerun the script.

**THANK YOU**