OLAM: A Devanagari OCR employing Neural Networks

Group Members:

- Ankit Yadav(11740140)
- Ayush Sharma(11740240)
- Bhuvnesh Arora(11740270)
- Shivani Tripathi(11740970)
- Yatendra Singh(11741190)

Code Links:

- Object Detection Co-Lab - Models Based on Config
- CRNN Training Notebook Co-lab
- Data Generation Code & Analysis - GitHub

Data Links:

- Text & Line Segmentation Model Data
- Line to Word Segmentation Data
- Word Sequence Detection Data

Model Links:

- Object Segmentation Models
- Word Sequence Detection Model

Objective and Potential Applications:

Olam is a Neural Network based OCR for languages in the Devanagari Script family currently supporting Hindi. The proposed model can be optimized and implemented to tackle challenges across several Human-to-Machine as well as Machine-to-Machine interaction domains. Consumer Businesses can use this to adaptively widen their reach and expand their market to remote areas by understanding data in Hindi (Devanagari script better). Recruiters and Consultants can use this to provide solutions to clients beyond their current operational languages and hire regional talent surmounting the Language barrier. Academicians can extract and decipher long texts and Primary Sources in Hindi/Hindustani, a language that finds prevalence in quite a few Administrative records across regions and eras. Beyond that, Devanagari has been of critical aid in deciphering numerous historical edicts and texts of rich Cultural Heritage. An OCR equipped with and optimized for predicting phrases in columns can be of great use in decoding them. Today, Devnagri enjoys a massive usage by more than 608 million people across 120 different languages, from Hindi and Sanskrit to Nepali and Marathi, thus a project on this can positively impact many lives on a huge proportion across regions and cultures. Socially, much Legal information/documentation and Medical prescriptions as standard follow English or Hindi as their media for exchange which may not be very accessible to local populations in many regions of India. A Hindi based OCR can act as a bridge to streamline the translation of vital information into native languages/scripts quickly. Which can be a great social equalizer to the regional populace in terms of access to mainstream information and vital prescriptions. Consumer tools built around this can highly aid migrants and travelers traveling in Hindi speaking areas, helping them in capturing and translating signboards/Direction/Station Boards. This can also be of great value to translators or people looking to pick up the language who encounter it in Non-Digital form, say while reading a Newspaper. Such Handy tools can also bring cultures closer through streamlining everyday interactions like reading a Food Menu or the Local journal in different tongues, highly adding to the first-hand experience of traveling.

Approach:

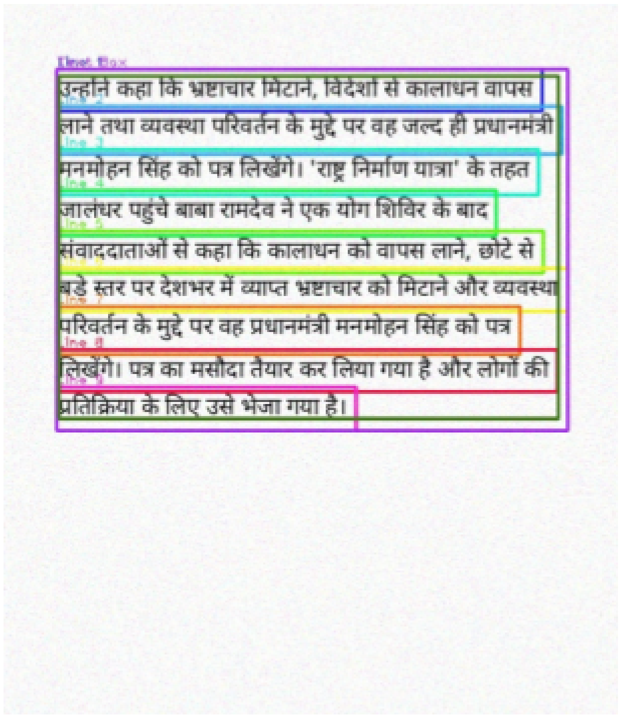
Nature of Dataset: The dataset for the model will be a huge collection of images with labeled bounding boxes around text, lines, words, and characters. We are going to create our dataset using the following approach:

We make a Python program to write text in Hindi on an image using libraries like Pillow and Opencv. Doing so we can easily compare the two successive images as we have White (later on introducing different sorts of backgrounds to make the model even robust) background and textual content is written in Black to find the pixel of the most recent word/ character. Once we have the pixel values of the new region we can pad the extremes of this region to form bounding boxes around the word. We can augment the data by changing the background color, using different text sizes. Further adding random noise or blurriness to the images can also be considered as an expansion of the dataset. Addition of this noise makes the generated images closer to photos of documents in grayscale.

Dataset Examples:

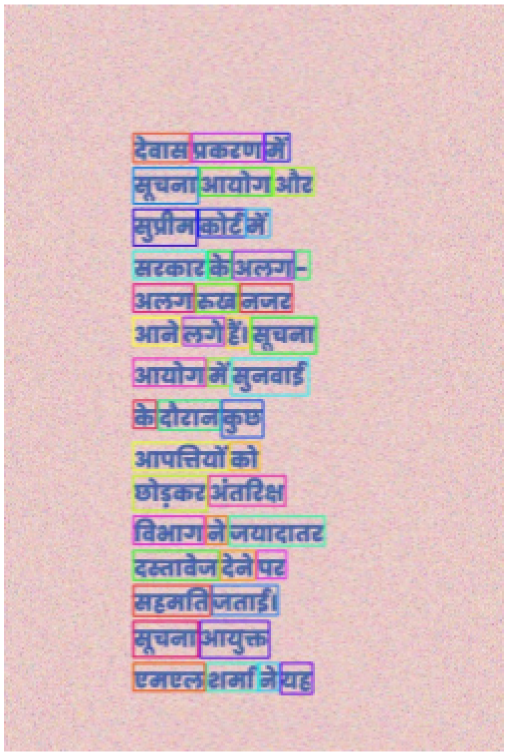
नेपाली सेना के प्रवक्ता [illegible] ने बुधवार को बताया कि काठमांडू से 300
किलोमीटर पश्चिम में स्थित [illegible] जिले के [illegible] गांव के पास जंगलों में इस
विमान का मलबा मिला। [illegible] ने कहा कि मारे गए लोगों में दो [illegible]
[illegible]

Text Area Boundiing Box with S&P noise

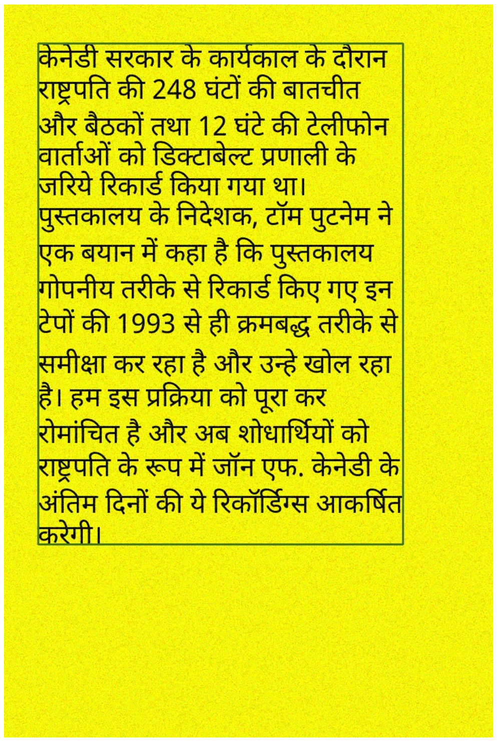


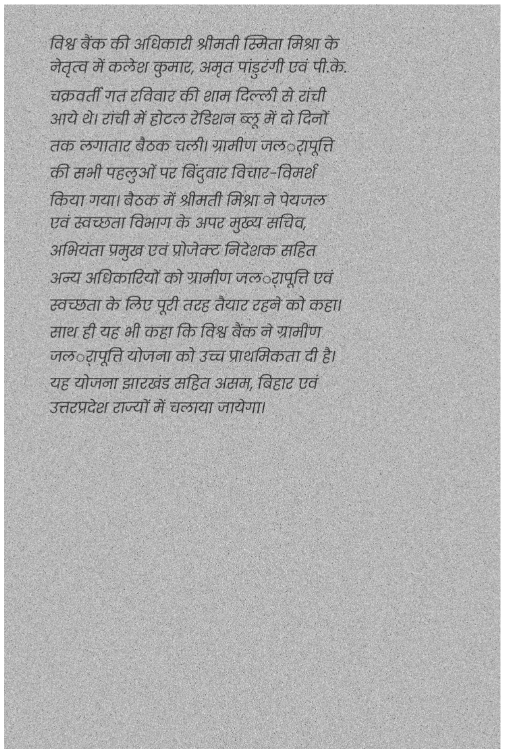
उन्होंने कहा कि भ्रष्टाचार मिटाने, विदेशों से कालाधन वापस
लाने तथा व्यवस्था परिवर्तन के मुद्दे पर वह जल्द ही प्रधानमंत्री
मनमोहन सिंह को पत्र लिखेंगे। 'राष्ट्र निर्माण यात्रा' के तहत
जालंधर पहुंचे बाबा रामदेव ने एक योग शिविर के बाद
संवाददाताओं से कहा कि कालाधन को वापस लाने, छोटे से
बड़े स्तर पर देशभर में व्याप्त भ्रष्टाचार को मिटाने और व्यवस्था
परिवर्तन के मुद्दे पर वह प्रधानमंत्री मनमोहन सिंह को पत्र
लिखेंगे। पत्र का मसौदा तैयार कर लिया गया है और लोगों की
प्रतिक्रिया के लिए उसे भेजा गया है।

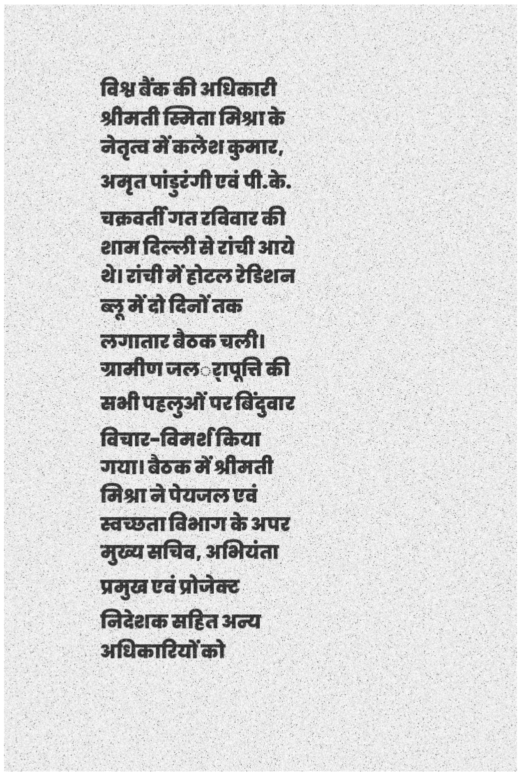
Line Bounding Boxes with Gaussian Noise


देवास प्रकरण में
सूचना आयोग और
सुप्रीम कोर्ट में
सरकार के अलग-
अलग रुख नजर
आने लगे हैं। सूचना
आयोग में सुनवाई
के दौरान कुछ
आपत्तियों को
छोड़कर अंतरिक्ष
विभाग ने ज्यादातर
दस्तावेज देने पर
सहमति जताई।
सूचना आयुक्त
एमएल शर्मा ने यह

Word Bounding Boxes


केनेडी सरकार के कार्यकाल के दौरान
राष्ट्रपति की 248 घंटों की बातचीत
और बैठकों तथा 12 घंटे की टेलीफोन
वार्ताओं को डिक्टाबेल्ट प्रणाली के
जरिये रिकार्ड किया गया था।
पुस्तकालय के निदेशक, टॉम पुटनेम ने
एक बयान में कहा है कि पुस्तकालय
गोपनीय तरीके से रिकार्ड किए गए इन
टेपों की 1993 से ही क्रमबद्ध तरीके से
समीक्षा कर रहा है और उन्हे खोल रहा
है। हम इस प्रक्रिया को पूरा कर
रोमांचित है और अब शोधार्थियों को
राष्ट्रपति के रूप में जॉन एफ. केनेडी के
अंतिम दिनों की ये रिकॉर्डिंग्स आकर्षित
करेगी।

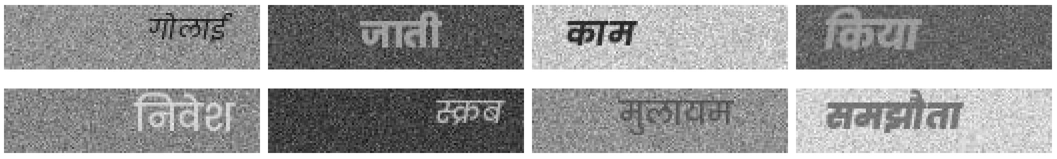

विश्व बैंक की अधिकारी श्रीमती स्मिता मिश्रा के
नेतृत्व में कलेश कुमार, अमृत पांडुरंगी एवं पी.के.
चक्रवर्ती गत रविवार की शाम दिल्ली से रांची
आये थे। रांची में होटल रेडिशन ब्लू में दो दिनों
तक लगातार बैठक चली। ग्रामीण जल◌ऱापूत्ति
की सभी पहलुओं पर बिंदुवार विचार-विमर्श
किया गया। बैठक में श्रीमती मिश्रा ने पेयजल
एवं स्वच्छता विभाग के अपर मुख्य सचिव,
अभियंता प्रमुख एवं प्रोजेक्ट निदेशक सहित
अन्य अधिकारियों को ग्रामीण जल◌ऱापूत्ति एवं
स्वच्छता के लिए पूरी तरह तैयार रहने को कहा।
साथ ही यह भी कहा कि विश्व बैंक ने ग्रामीण
जल◌ऱापूत्ति योजना को उच्च प्राथमिकता दी है।
यह योजना झारखंड सहित असम, बिहार एवं
उत्तरप्रदेश राज्यों में चलाया जायेगा।


विश्व बैंक की अधिकारी
श्रीमती स्मिता मिश्रा के
नेतृत्व में कलेश कुमार,
अमृत पांडुरंगी एवं पी.के.
चक्रवर्ती गत रविवार की
शाम दिल्ली से रांची आये
थे। रांची में होटल रेडिशन
ब्लू में दो दिनों तक
लगातार बैठक चली।
ग्रामीण जल◌ऱापूत्ति की
सभी पहलुओं पर बिंदुवार
विचार-विमर्श किया
गया। बैठक में श्रीमती
मिश्रा ने पेयजल एवं
स्वच्छता विभाग के अपर
मुख्य सचिव, अभियंता
प्रमुख एवं प्रोजेक्ट
निदेशक सहित अन्य
अधिकारियों को

- Added randomization for little noise, text width.
- Added randomization for text position.
- Randomization for different fonts.

Word Level Data Generation

We have generated images of words to be fed into the CRNN model in resolution 128 x 32, (in grayscale). Each image was labeled as a list of classes of characters in a word. We have used 154 classes including Hindi characters, English characters, English and Hindi numerals. English uppercase and lowercase were assigned the same class. Some examples are shown below.



Dataset

Hindi Text data is taken from Hindi data from [HC corpora: Collection of corpora for various language]

Code For Data Preparation:

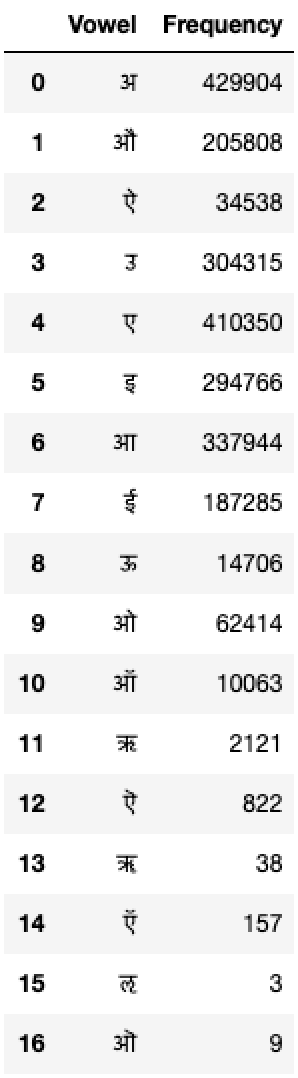
https://github.com/ayush1120/OLAM/tree/master

Data Analysis and Modeling:

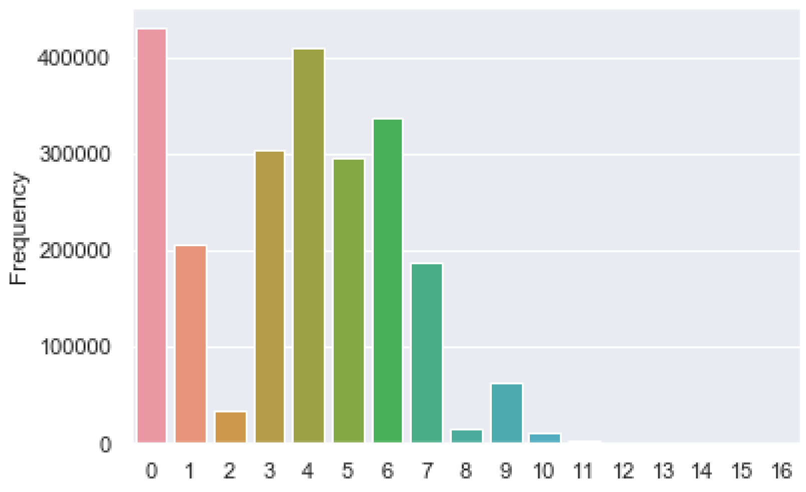
- Analysis of the frequency of various character labels to analyze the uniformity of data as text images will be generated using existing datasets.
- Analysis of chronology of words in the dataset.
- We tested the possibility of using models like YOLO, AlexNet, and others for transfer learning and using the pre-trained models for our use. We narrowed down on YOLO as we found it to be faster and capable of handling a large number of classes if required.

Vowel freq

- The frequency of all the vowels is measured from the sample dataset which shows only certain vowels are used more often.

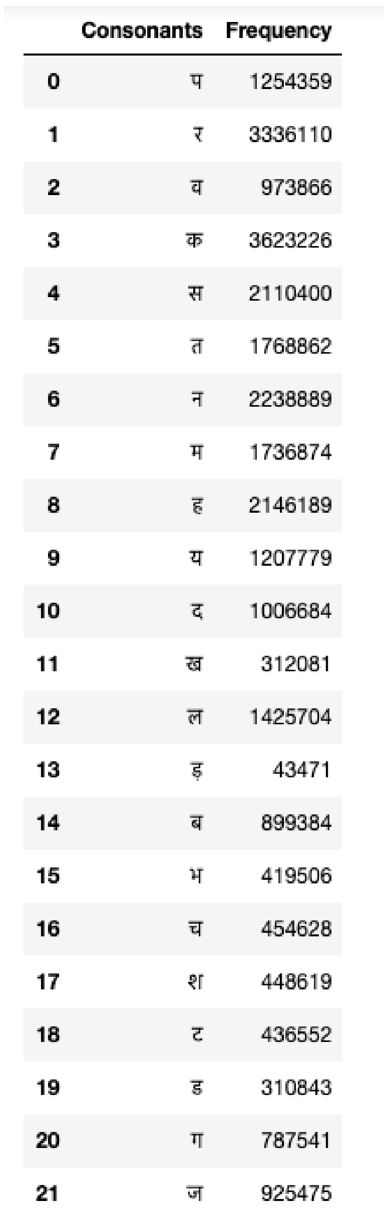
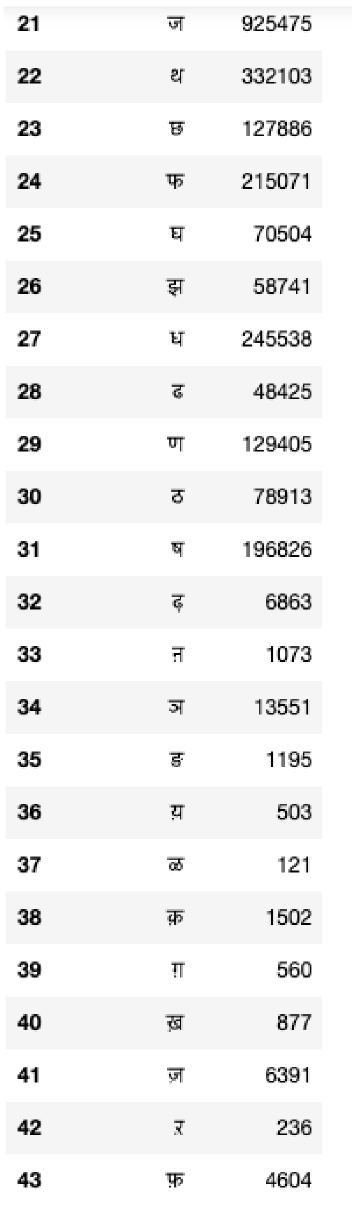


	Vowel	Frequency
0	अ	429904
1	औ	205808
2	ऐ	34538
3	उ	304315
4	ए	410350
5	इ	294766
6	आ	337944
7	ई	187285
8	ऊ	14706
9	ओ	62414
10	ऑ	10063
11	ऋ	2121
12	ऐ	822
13	ऋ	38
14	एं	157
15	ऌ	3
16	ओ	9

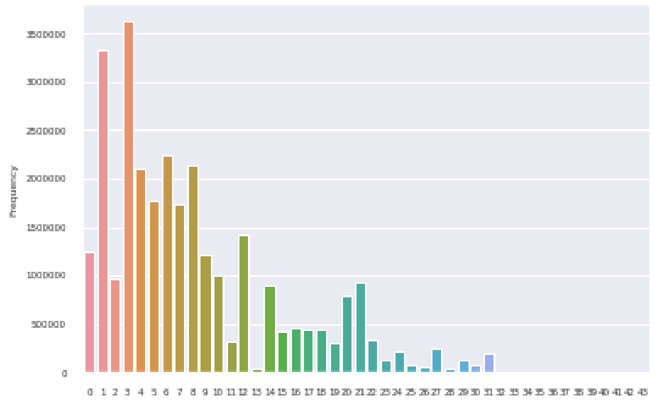


Consonant freq

- The frequency of each consonant is measured from the sample dataset. It can be seen that some consonants are used rarely.

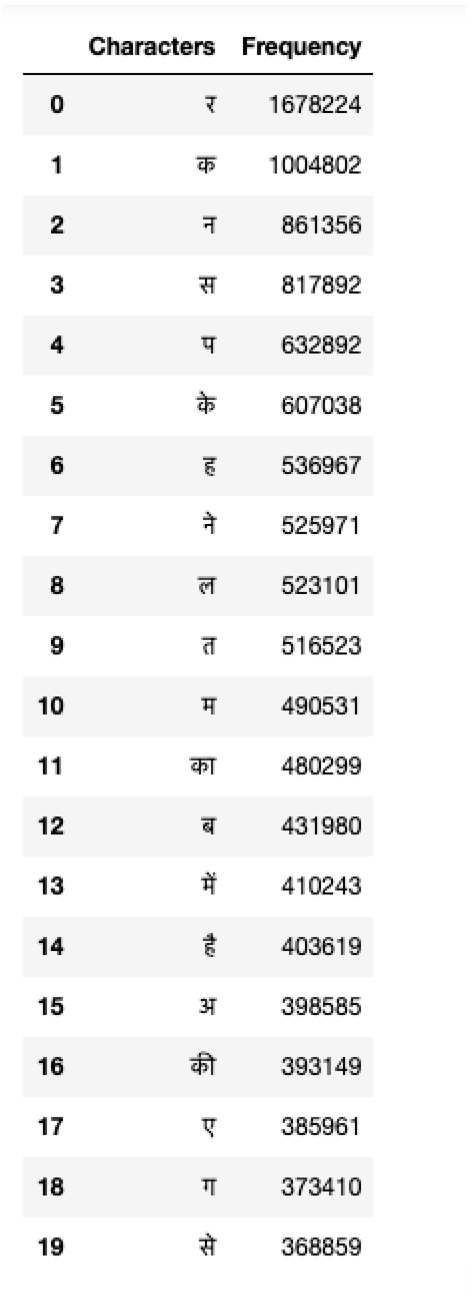



	Consonants	Frequency
0	प	1254359
1	र	3336110
2	व	973866
3	क	3623226
4	स	2110400
5	त	1768862
6	न	2238889
7	म	1736874
8	ह	2146189
9	य	1207779
10	द	1006684
11	ख	312081
12	ल	1425704
13	ड़	43471
14	ब	899384
15	भ	419506
16	च	454628
17	श	448619
18	ट	436552
19	ड	310843
20	ग	787541
21	ज	925475
22	थ	332103
23	छ	127886
24	फ	215071
25	घ	70504
26	झ	58741
27	ध	245538
28	ढ	48425
29	ण	129405
30	ठ	78913
31	ष	196826
32	ढ़	6863
33	त्र	1073
34	ञ	13551
35	ङ	1195
36	य़	503
37	ळ	121
38	क़	1502
39	ग़	560
40	ख़	877
41	ज़	6391
42	ऱ	236
43	फ़	4604

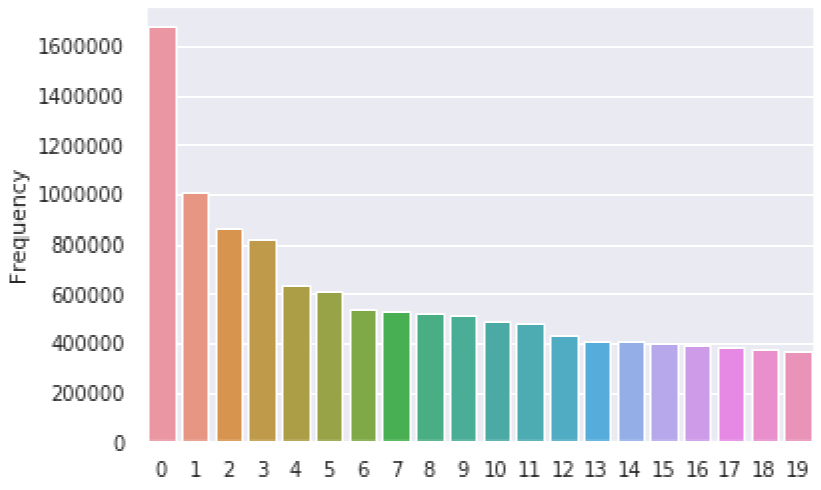


Detect Char

- We created a function to detect the characters of a word which is used to detect the characters by the model as the model will make a bounding box around these characters in images.
- Found the frequency of certain characters

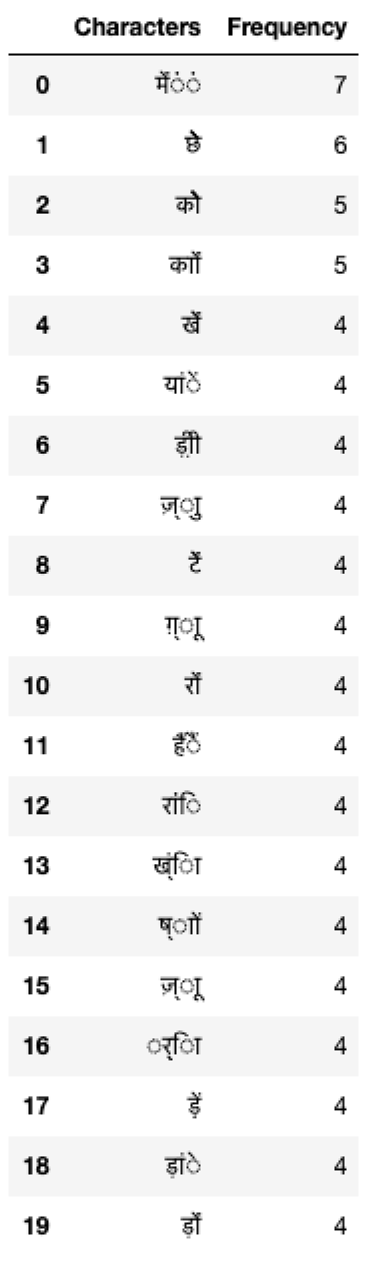


	Characters	Frequency
0	र	1678224
1	क	1004802
2	न	861356
3	स	817892
4	प	632892
5	के	607038
6	ह	536967
7	ने	525971
8	ल	523101
9	त	516523
10	म	490531
11	का	480299
12	ब	431980
13	में	410243
14	है	403619
15	अ	398585
16	की	393149
17	ए	385961
18	ग	373410
19	से	368859

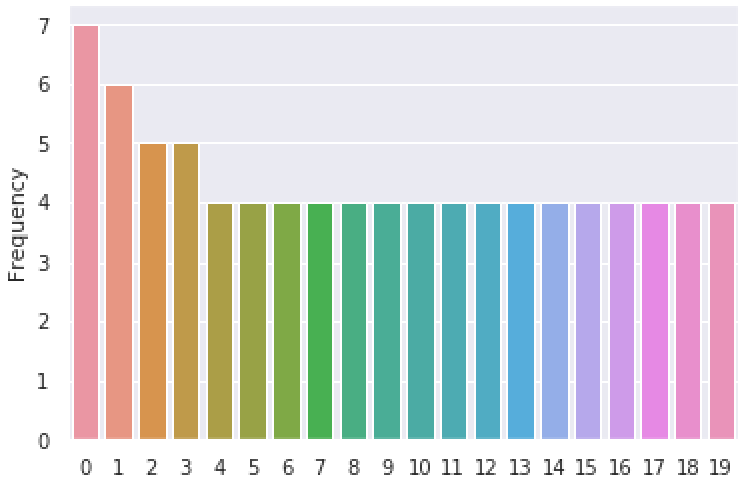


<u>Detect Consonant and Dependent Vowel with the frequency of Dependent Vowel</u>

- Detected characters that have consonants followed by Dependent Vowel and counted the number of Dependent Vowels.

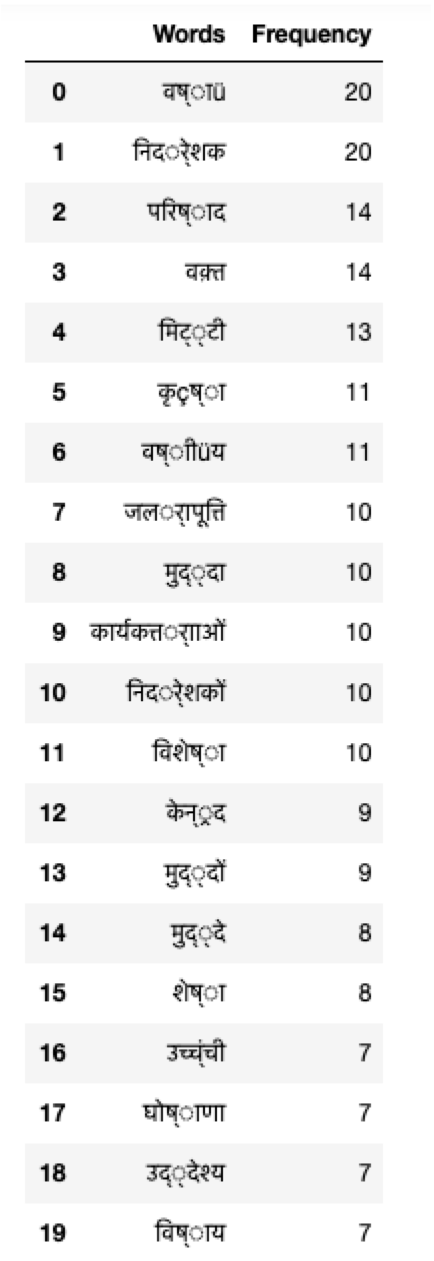


	Characters	Frequency
0	मैं◌ं◌ं	7
1	छे	6
2	कौ	5
3	कौं	5
4	खैं	4
5	यां◌ॅ	4
6	ड़ी	4
7	ज़्◌ु	4
8	टैं	4
9	ग़्◌ू	4
10	रौं	4
11	हैं◌ॅ	4
12	रां◌ि	4
13	ख़ं◌ि	4
14	ष्◌ौं	4
15	ज़्◌ू	4
16	◌ऱ्◌ि	4
17	ड़ें	4
18	ड़ां◌े	4
19	ड़ौं	4

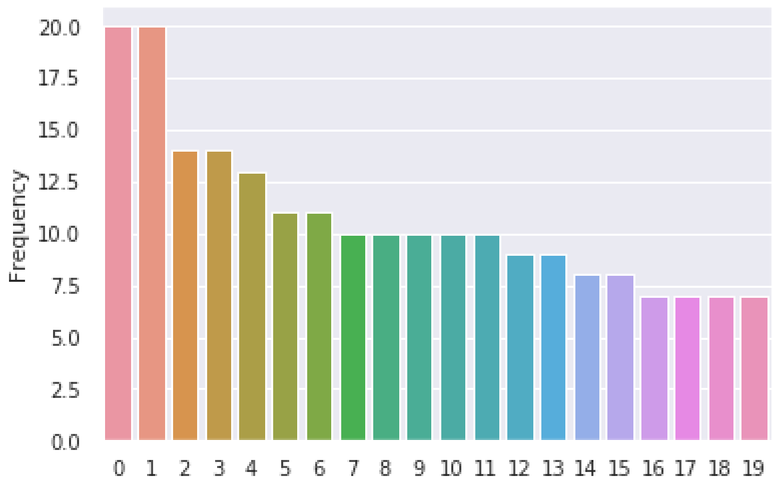


A half consonant followed by a Dependent Vowel

- The frequency of words with half constants followed by a Dependent vowel is calculated which can be used for data cleaning.

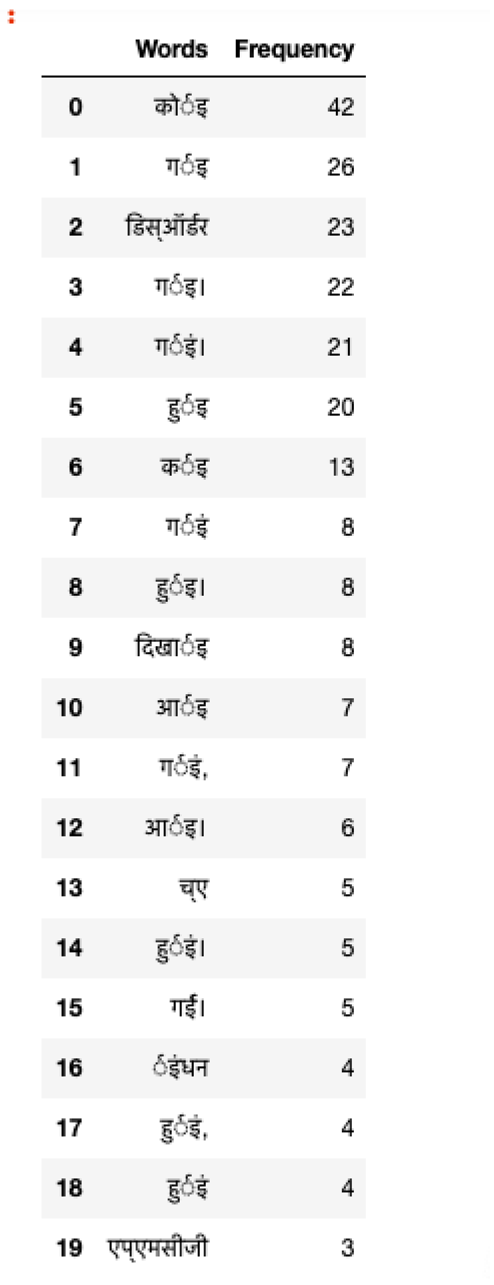


	Words	Frequency
0	वष्◌ाü	20
1	निद◌्रेशक	20
2	परिष्◌ाद	14
3	वक़्त	14
4	मिट्◌्टी	13
5	कृçष्◌ा	11
6	वष्◌ाीüय	11
7	जल◌्रापूत्ति	10
8	मुद्◌्दा	10
9	कार्यकत्त◌्राओं	10
10	निद◌्रेशकों	10
11	विशेष्◌ा	10
12	केन्◌्द	9
13	मुद्◌्दों	9
14	मुद्◌्दे	8
15	शेष्◌ा	8
16	उच्च्ंची	7
17	घोष्◌ाणा	7
18	उद्◌्देश्य	7
19	विष्◌ाय	7

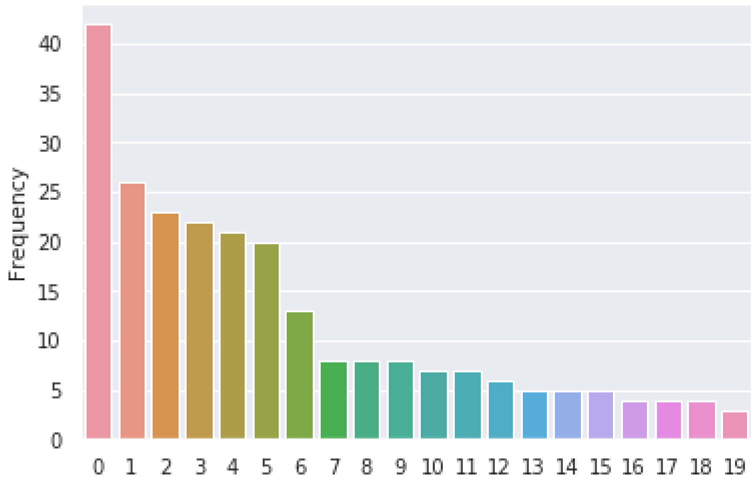


Half consonants followed by a vowel

- The frequency of words with half consonants followed by a Vowel is calculated which can be used for data cleaning.



	Words	Frequency
0	को◌́इ	42
1	ग◌́इ	26
2	डिस्ऑर्डर	23
3	ग◌́इ।	22
4	ग◌́इं।	21
5	हु◌́इ	20
6	क◌́इ	13
7	ग◌́इं	8
8	हु◌́इ।	8
9	दिखा◌́इ	8
10	आ◌́इ	7
11	ग◌́इं,	7
12	आ◌́इ।	6
13	च्ए	5
14	हु◌́इं।	5
15	गई।	5
16	◌́इंधन	4
17	हु◌́इं,	4
18	हु◌́इं	4
19	एप्एमसीजी	3



Inference

Due to a lack of Hindi grammatical rules, the function detected some words which were incorrect upon character segmentation. Thus analyzing such words helped us preprocess the data used to generate the Image Data set for training the model.

1. There were around 800 words that start with a dependent vowel.
    a. 'ॅषिकेश, 'ूबाजार, 'ंिचंताजनक, 'ेलेकिन, 'ँँँअविनाश
2. There were some words with half vowels which
    a. 'जकिउ्द्दीन', 'कोइ्रü', 'गइ्र', 'कोइ्र', 'हुइ्र', 'जाए्गा।', 'हुइ्र', 'सुरमइ्र्', 'जुलाॅइ्र', 'कोइ्र', 'उ्देश्य', 'राष्अ्रपति', 'आइ्र', 'स्काउ्ट्स', 'चढाॅइ्', '“देखो-श्-अ्!”', 'कोई्र', 'हेडलाइ्ट्स'
3. There were some words that have half consonants followed by a dependent vowel.
    a. जल्रापूत्ति, वष्ीüय, विशेष्ा, घोष्ाणा, चट््टानें’

Challenges and Redirections:

- The Segmentation Model Architecture will have to be reiterated time and again for obtaining a more accurate model.

- Pillow currently doesn’t have sufficient support for prominent Devnagari fonts and thus we had to make do with the not so popular alternative fonts that it does. (Ref: https://github.com/python-pillow/Pillow/issues/3191)

- Different fonts have varying rules for merging advanced (half) consonants. Hence for character segmentation, there can emerge an ambiguity in how we choose the correct bounding box.

- It was a bit challenging to coordinate tasks and timings being away from campus.

<u>**Modeling:**</u>

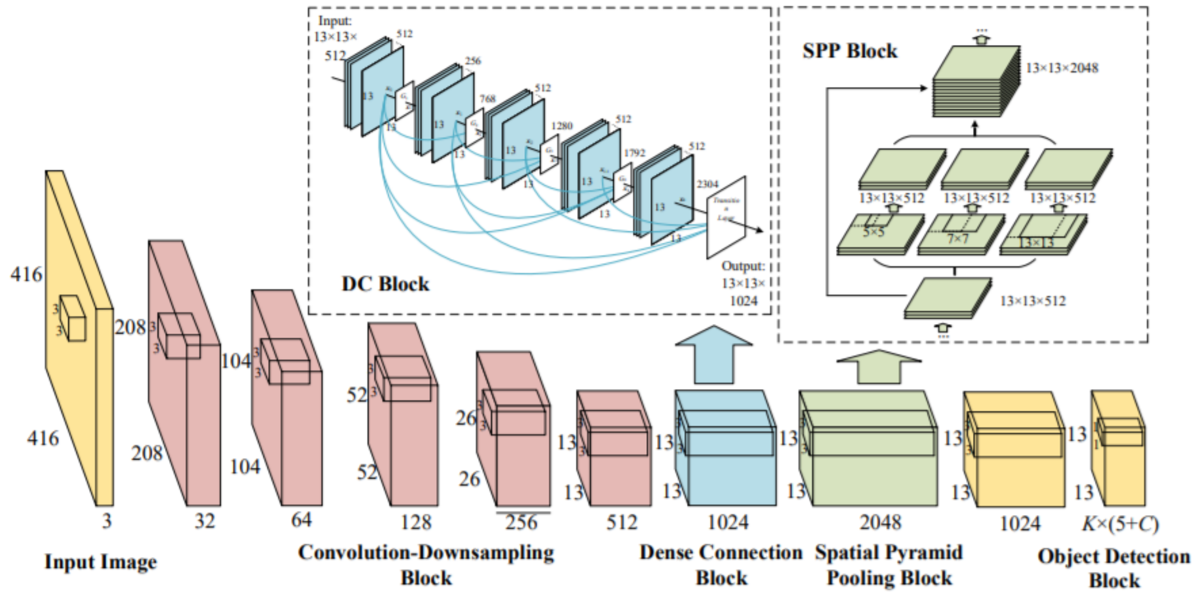
Currently, our model pipeline consists of 3 different learnable models that work together to attempt to reach our final goal of detecting the written letters from an optical image. We choose YOLO as a base model for object detection because the YOLO model is fast enough to be used in a scenario requiring real-time speeds in processing the image keeping the same accuracy as previous state-of-the-art models. Along with this, YOLO also learns a more general representation of the underlying structure The three models have their details as described below:

Text/Line detection model:

This is the first model in our pipeline which takes as input a full image with words written in Hindi and detects the lines or the text boxes based on which type of data the model is trained for. We use the current model for directly detecting the lines as that was the best we could do given the constraints. This is an object detection model which is based on the YOLO architecture. To be more precise, we use the YOLO-SPP model for detecting the lines in the image.

SPP stands for Spatial Pyramid Pooling which is used to increase the mAP(mean Average Precision) of the model which is used to detect the lines. The YOLO-SPP model takes the best features in the max-pooling layers along with the downsampling from the convolutional layers which is what a traditional full-size YOLO model does.

We have trained the YOLO-SPP model to around 3.0 average IOU values as provided by the darknet implementation of YOLO.



Word Detection Model:

The next model in our pipeline is again an Object detection model based on YOLO which takes the input of the lines we have segmented from the previous model. This model is not based on the full-sized YOLO but the smaller Tiny-YOLO architecture. This design decision was based on the fact that any image may contain a multitude of lines which will again run through the word detection model each time. Thus to make sure that the complete pipeline is not slowed down due to running on each image separately. The Tiny-YOLO model is a good fit for our needs and also provides good enough accuracy.

We have again trained the Tiny-YOLOmodel to around 3.0 average IOU values as provided by the darknet implementation of YOLO, similar to the YOLO-SPP model. There is room for improvement but the time constraints were severe and we were unable to train the model further.

CRNN word to sequence model:

The final model in our pipeline is a CRNN model written in Keras and TensorFlow which uses bi-directional LSTMs built on top of a conventional convolutional network to identify the correct sequence of characters in the given words which have been identified by the previous models. The model we have constructed is based on the research paper **An End-to-End Trainable Neural Network for Image-based Sequence Recognition and Its Application to Scene Text Recognition**.

This model has been trained using the connectionist Temporal Classification(CTC) loss function. The model was trained for 10 epochs but reached a point of diminishing returns sooner than expected and the validation loss started increasing at the 8th epoch. We still continued training and the validation loss did not decrease giving us the impression that the model was being overfitted at that point. We have chosen the model with the lowest validation loss as the best model for our final pipeline.

All three models described above can be found in the drive folder. The object detection models contain their respective special builds of darknet along with all the extra needed files to mitigate any problems which may arise due to minor changes in the code. The code is also compiled to be run on the CPU instead of GPU as most "consumer" GPUs would not be able to handle the model size of YOLO-SPP and would return a "CUDA: Out of memory" error, and it would also render the compiled binary being unusable on machines without NVIDIA GPUs supporting CUDA. Nevertheless, the complete training was carried out on GPUs provided by collab and the model was saved.

Although the accuracy achieved so far illustrates considerable progress from where we were a few weeks back, there is still room for improvement. We have been tackling constraints in terms of computing resources available on Google Colab as well as Time on our hands given it's currently a hectic phase for our batch. Nevertheless, we as a group are continuously working on optimizing OLAM parallel to all other undertakings and expect to achieve considerably improved accuracy through more rigorous training of the model as well as improving the architecture of our models. We’ll be able to better direct more of our efforts on this after our exams.

Work Distribution:

Task / Member	**Ayush Sharma**	**Yatendra Singh**	**Bhuvnesh Arora**	**Shivani Tripathi**	**Ankit Yadav**
Data Collection	Text Dataset gathering, Text wrapping for images, Bounding Box generation	Fonts gathering, Text Dataset Merging	Adding noise and different backgrounds to generated Images, Image Size fixing, and randomize the font size accordingly.	Merging existing datasets to augment our datasets	Serving text data from different to Image Generation Code
Data Analysis & Generation	Bounding Box Generation and text to lines to the word. Parsing data for modeling.	Analysis and Preprocessing ● Analysis of chronology of words in the dataset. ● Analysis of the frequency of various words and characters and several combinations. ● Preprocessing of the dataset by removing useless words.			Parsing data for modeling.
Document Reporting	Project Reporting				
Modelling	● Darknet Source Code Tweaks ● Model Training ● CRNN		● Reading Papers ● Constructing Models based on Research Content		