**Overview Documentation(Task C)**

This is the overview documentation for TaskA and TaskB. IT contains steps, methodologies, tools and technologies used to perform taskA and taskB. IT also contains the points with which the result and performance of the tasks could be enhanced.

The code for task A and B can be found on: <https://github.com/ntesh21/f1-task-assignment.git>

**Initial observation of the Nepali Driving License**

The nepali driving licence is the document which contains various information regarding its holder along with the image of the person.

These are following data heads(information) that every driving licence holds:

* Licence Number
* Blood Group
* Name of the person
* Address
* License Office
* Date of Birth
* Father/Husband Name
* Citizenship Number
* Passport Number (optional)
* Phone Number
* Category
* Date of Issue
* Date of Expiry

Some of the interesting finding regarding data extraction from the initial observation of the document

Types:

By observing the images of driving license in the web, various types of driving licenses were found such as professional, non-professional and regular.

In this task the regular type of liscence is being used to extract the information.



Fig: Regular Type Nepali Driving License Format

**Task A**

**Problem Statement**

Extract all the relevant information from the Nepali Driving Licence sample in order and compare the extracted information confidence with the actual information in the given license.

Discuss and explore the ways to increase the confidence and the extraction process.

**Methodology:**

**Sample License:**

The sample license image was taken from the internet to use for extraction.



Fig: Image of sample lisence for this task

**Methodology Overview:**

For this task OCR(Optical Character Recognition) is applied to be able to read and extract the text from the image.

The general available OCR is Tesseract which can not only recognize the character in the image but also identify and extract the words. It uses the combination of CNN(Convolution Neural Network) and LSTM(Long Short Term Memory) . The python library of ‘pytesseract’ is used.

The task has three main steps:

**Preprocessing**

The sample image like a regular license is the colour image of the document. Applying the OCR directly in the image may not give the best result. It may not detect the text properly due to various noise and impurities in the raw image. Due to this reason the sample image is preprocessed by applying various image processing techniques.

Techniques applied for image preprocessing

* **Convert to Grayscale**

The sample image is in three channels i.e color, so it would be a lot easier to extract the text form the image if the pixel is segregated into less channels. If the image is converted into grayscale the individual text will be a lot visible from the background and the edges to extract the words will be also more precise. So the image is converted into grayscale.



Fig: Image after changing into grayscale

* **Adjust the brightness and contrast**

Contrast and brightness are increased in order to make the contents more visible and remove the background pixels.

Brightness is set to 10 out of 100 and Contrast is set 1.6 out of 3.



Fig: Image after converting to grayscale and increasing saturation

* **Apply Threshold**

Applying thresholding segregates the pixel value based on the threshold value set. If the pixel value is less than the threshold value then it is set 0. Adaptive gaussian thresholding is applied on the image which picks the threshold value according to the pixels in the image. Thresholding helps to enhance the edges of the text and text segments and also intensifies the text.

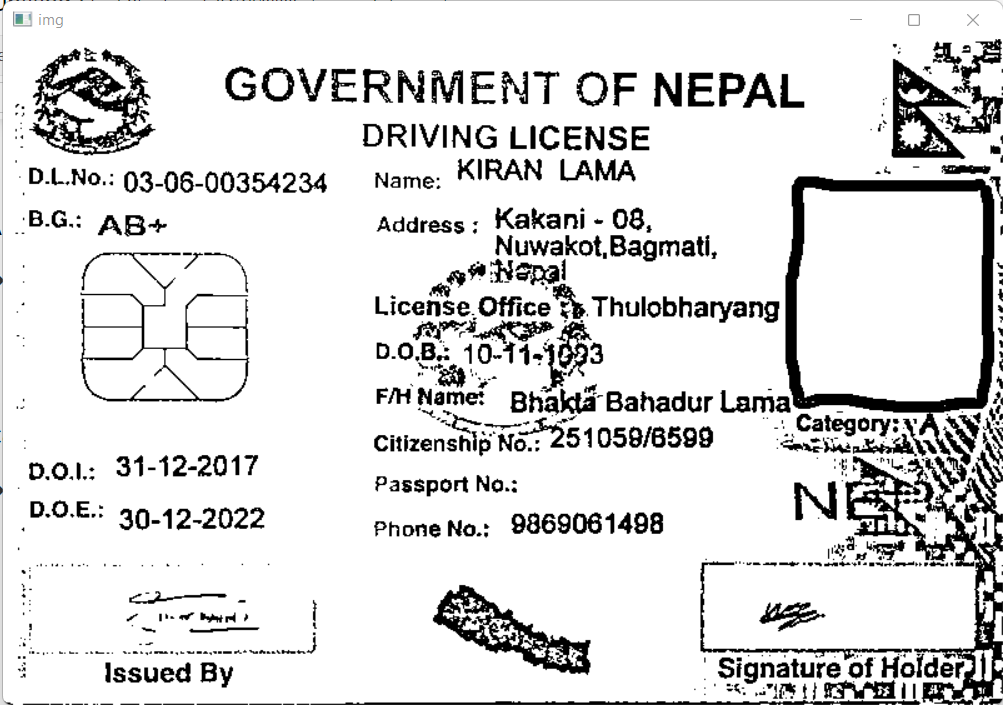


Fig: Image after converting into grayscale, adjusting saturation and applying the thresholding

* **Blurring/Remove Noises**

Blurring helps to smooth the image by removing the noises in the pixel. This helps to maintain the shape of the text intact without noise in the pixel.

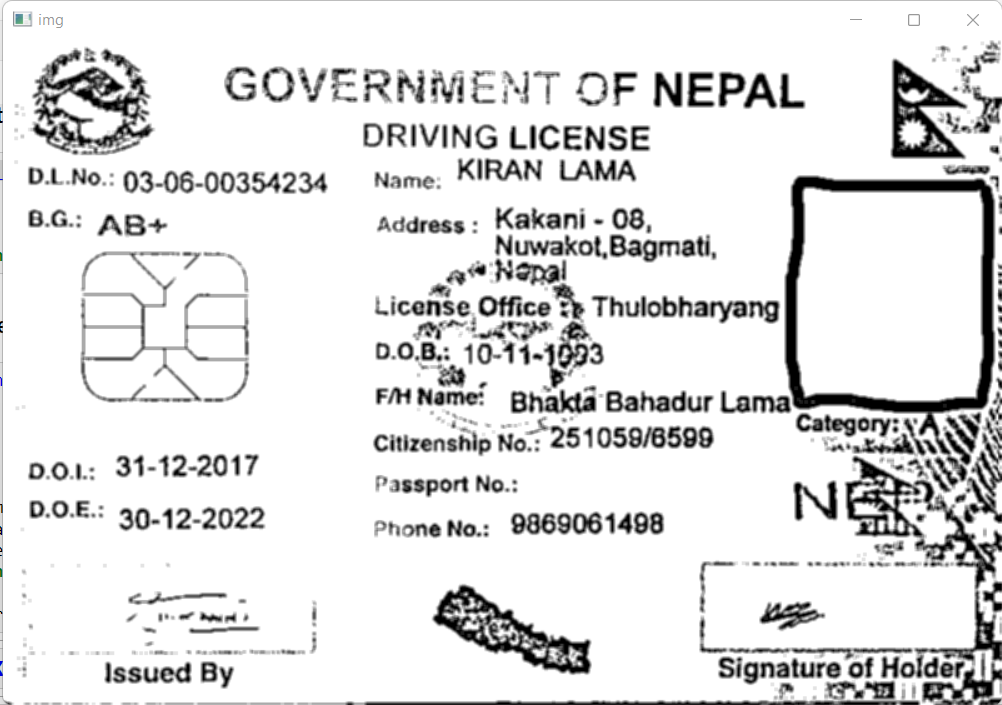


Fig: Image after final preprocessing(Grayscale, Saturation, Thresholding, Blur

**Text Extraction**

Pytesseract library is used to extract the text from the preprocessed image. Pytesseract library uses tesseract and extracts all the text that it detects in the image and constructs the words out of those.

**Information Extraction from the text**

The text extracted from the preprocessed image with pytesseract is in raw from. It contains all the text in one place and by looking at the text it is very difficult to extract the correct information out of the texts for specific data heading. So the extracted text is processed using the regular expression to catch the pattern specific to certain data heading as well as matched on the basis of keywords.

Various data fields require various patterns specific to them in order to catch the data so multiple methods are used for individual data headings.

The methods are:

**Name Extraction**

For this task the name extraction is solely based on keyword and pattern. First the line containing ‘Name’ is searched from the extracted text. The name of the lisence holder is supposed to be in all Uppercase letters. So if the name keyword is found in any line then regular expression is used to match the pattern of the words in that line where the words are in all uppercase. Once the uppercase words are matched they are further divided into first name, middle name and last name based on their occurrences.

**License Number Extraction**

On observation of few license documents there are a specific couple of license number patterns only so for determining the license number regular expression is used to match those specific patterns in the text and that pattern is very unique from the rest of the text. So it is easy to extract the liscence number as long as the ocr determines the text from the document.

**Blood Group Extraction**

The pattern for the blood group is also very much unique as the blood group contains only few alphabets with + and - in them so by matching that pattern it is quite easily extracted from the extracted text.

**Phone Number Extraction**

Similar to license number the pattern of the phone number is also unique to the whole document text. It is either a 10 digit number for mobile number or 9 digit for telephone number including area code. So as long as OCR extracts the phone number we can identify from the text.

**Citizenship Number Extraction**

There are a couple of patterns for matching citizenship numbers. Since there are multiple patterns in different citizenship all possible types of patterns were observed and all those patterns were matched. The pattern which is most common is matched if it is not found then the sequentially all the patterns are searched and matched based on their rarity.

**Address Extraction**

The address here is also extracted based on the keyword. The line with the keyword address is looked at on the extracted text and if the keyword is found then the position of the keyword on the line is looked into and the words after that position are extracted.

**Date of Birth Extraction**

For the extraction of the date of birth, the first keyword ‘D.O.B’ is matched. If the keyword is found then the regular expression is used in that sentence to match the date format and find it. If the keyword is not found then the whole extracted text is subjected to the match the date pattern and out of the all matched date patterns each date is compared with the current date and if the date has a difference of more than 18 and less than 100 then that date is selected as the date of birth. This is only done on the assumption that the date of issue is not more than 18-20 years.

**Date of Issue Extraction**

Similar to date of birth extraction, date of issue is extracted by first finding the keyword D.O.E in the text. If not found then the regular expression is used to match the date pattern in the whole text and out of the all matched dates the date which is less than 20 years than the current date and not any more than current date is taken as the date of issue.

**Date of Expiry Extraction**

For the extraction of the date of expiry, first the keyword D.O.E is matched, if found then the date pattern is matched in that particular sentence and returned as the date of expiry. If the keyword is not found then the pattern to match all dates from the text is used and out of all those matched dates, the date which is more than the current date is used as the date of expiry. This is done on the assumption that anything with a date of expiry less than the current date is an expired/invalid lisense.

**Category Extraction**

The category extraction is done by matching the keyword containing first and then if found matching the pattern of category codes such as A, A,B K etc

**License office extraction**

This extraction is similar to the address extraction. As it is based on the keyword lisence office. This field is not found in all types of lisence documents.

**Note:**

**Passport number** and **F/H name** fields are not addressed in this task. As the passport number is very rare and not common but very easy to extract based on the pattern. The f/h name is quite challenging without Named Entity Recognition as it will affect the main holder’s name.

Result

By applying preprocessing, extracting and applying various methods mentioned above to extract the specific information from the text the result is obtained on the sample image.

The result is as following:

**Exact result on the document:**

{

'citizenship\_no': '251059/6599',

'lisence\_no': '03-06-00354234',

'phone\_no': '9869061498',

'blood\_group': 'AB+',

'first\_name': 'KIRAN',

'middle\_name': '',

'last\_name': 'LAMA',

'lisence\_office': 'Thulobharyang',

'address': 'Kakani - 08, Nuwakot, Bagmati,Nepal',

'category':'A',

'date\_of\_birth': 10-11-1993,

'date\_of\_issue': '31-12-2017',

'date\_of\_expiry': '30-12-2022'

}

**Result of Data Extraction**

{

'citizenship\_no': '251059/6599',

'lisence\_no': '03-06-00354234',

'phone\_no': '9869061498',

'blood\_group': 'AB+',

'first\_name': 'KIRAN',

'middle\_name': '',

'last\_name': 'LAMA',

'lisence\_office': 'te Thulobharyang',

'address': 'Kakani 08, Nuwakot, Bagmati',

'category': None,

'date\_of\_birth': None,

'date\_of\_issue': '2017-12-31',

'date\_of\_expiry': '39.42-2022'

}

**Evaluation**

The evaluation for the result is done by calculating the text similarity between exact data in the document for each field and the result for each field.

**Evaluation Report:**

{

'citizenship\_no': 100.0,

'lisence\_no': 100.0,

'phone\_no': 100.0,

'blood\_group': 100.0,

'first\_name': 100.0,

'middle\_name': 100.0,

'last\_name': 100.0,

'lisence\_office': 74.9599814786962,

'address': 91.20217842351352,

'category': 0.0,

'date\_of\_birth': 0.0,

'date\_of\_issue': 97.64283968039754,

'date\_of\_expiry': 92.06335584414785

}

The evaluation report shows that the result is pretty good for the data fields which have very specific patterns such as citizenship number, lisence number, phone number, blood group.

In terms of name also the result has reflected satisfactory results. But with the data fields with more complexity to match such as category, address, specific date the result fails to reflect accurate results. In some fields the confidence is 0 because of the performance of the text extraction process(OCR) as well as having very complex patterns.

**Note:**

The result is on the reflection of all the population of the lisence documents as the method to extract information is developed by only observing a few samples. So for all the general documents the result may vary drastically.

**Possible Improvements**

This task is just to show the overview of the basic processes involving the information extraction from the document with just sample data. For general application huge related data is needed to train and work specifically on the generalised ML models.

The information extraction can be improved from every aspect.

Improvements in various steps:

**Data Loading**

The data loading process can be more generalised. As in real life application there may be various types and formats of the document. The system should be able to take any valid type of document and process it. Like if the document is in pdf form then the system should be able to convert it into the image object by using pdf reader to read as an image object and convert it into the numpy array.

So the system can be improved by making the input uniform across all the platforms.

**Data Preprocessing**

Data preprocessing is the most vital step that will immensely affect the result of the task. So there are a lot of preprocessing steps that can be applied in order to improve the result.

**Uniform Alignment**

Starting with checking the angle of the input image. The alignment of the input image may not be uniform in the real world. So images may be rotated and others may be tilted. If this is the case then it will hamper the result very adversely. So checking the angle and alignment and modifying the image accordingly to get the uniform image will improve the result all around.

**Various Preprocessing Techniques**

General preprocessing techniques are used for this task. These steps have improved the performance very much. Furthermore we can apply additional advanced preprocessing techniques after accessing the whole large dataset patterns and work with other additional preprocessing steps which will improve the result. Some of the preprocessing techniques may be the pixel averaging, neutralising or enhancing the dominant pixels etc.

**Image Segmentation**

The result can also be improved by the segmentation of the Lisence document. For this we can train the model that will segmentize the document in different parts according to the information available. For example, we can segment the Lisence document in 5 different parts. One segment of the document will contain L.No and Blood Group, Second segment will contain date of issue and expiry, third will be the segment that has category, fourth can have the first three fields of the middle section and final can have the remaining section of middle part.

In this way segment this the document in different part and treating each segment as different image and extracting the text from each segment will make the distinction of the data head easier and there will be less chance of the error on information

**Text Extraction**

To make improvements in the text extraction process. The performance of the OCR should be improved. In this task tesseract is used which is a pretrained model and common one. The performance of this model is not quite on par with the highest accuracy. So to improve text extraction we can either use other pretrained OCR models which have far better performance in terms of accuracy or we can train our own ocr model with domain specific augmented data set which will perform good on particular documents rather than general document.

On research one of the better OCR currently is Google OCR by Google. This is supposed to be far better in performance than Tesseract. This can be explored and compared with current results.

**Data Extraction**

If the preprocessing steps and text extraction steps are improved then the data extraction will automatically improve.

One of the challenges in the Lisence document is extracting the fields such as date of birth, F/H Name, Address because beneath these data there is the watermark of Nepal Government that will create the noise and affect in the text extraction process. So this can be tackled by layering the image and removing the bottom layer containing the watermark.

Beside those data extraction can also be improved by applying the NLP techniques such as Named Entity Recognition(NER) model which can recognise the important information such as Names, Addresses, Dates etc.

**Task B**

**Problem Statement**

Extract the image from the document and Compare with the real live image.

Show the comparison between those.

**Methodology:**

**Sample License:**

The sample document(citizenship) image was used for image extraction.



Fig: Sample document for Image Extraction

**Methodology Overview:**

This task is divided into three stages:

**Face Detection**

For face detection Haar - a feature based cascade classification method is used. It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images.

Specifically, open cv’s Haar-Cascade Detection is used. It is the pretrained model which helps to detect faces in the image.

Using Haar-Cascade the face in the sample document is detected and a bounding box is created around the face.



Fig: face Detected Using Haar-Cascade and bounding box around the detected face

**Face Extraction**

Now the detected face is extracted with the help of the bounding box coordinates created by face detection mode. So with the help of bounding box coordinates the image inside the box is extracted as a face.



Fig: Face Extracted from the document

**Extracted image comparison**

To compare the face from the live image and the extracted face from the document. Face-Recognition library is used. It is the open source library built using Dlib state of art face recognition model.

To compare the faces we use face-recognition library which takes the live image and detects the face in the image and encodes the detected live face and it then encodes the face image extracted from the document.

Now it uses the face\_distance method to calculate the distance between two image encodings and returns the distance value.

It also uses the compare\_faces method to evaluate whether the two input faces are from the same face. This outputs True is the faces are same and False is the faces are different

**Result**

**Live Image to compare with:**

****

**Face Extracted from the document**

****

**Final Evaluation Result:**

* Are both images of the same person?
* True
* The extracted image has a distance of **0.44** from the live image.
* With a normal cutoff of 0.6, would the test image match the known image?
* True
* - With a very strict cutoff of 0.5, would the test image match the known image?

- True

**Note:**

In this task it is assumed that there is only one face in the document so the multiple faces is not handled.

**Possible Improvements**

Certain Improvements could be made in order to increase the similarity of the images.

**Preprocessing:**

In this task the preprocessing is not heavily used but we can use various preprocessing methods that could be applied in both the images to make both images even as possible in terms of colour, saturation, exposure.

Size of the both images can be made even so that the comparison would be uniform.

If the both images are subjected to similar general preprocessing techniques then if the faces are same then there will be the chance that the encodings are also nearly same thus increasing similarity.

**Face Detection Models**

More accurate and advanced face detection models can also be used or can be self trained

**Comparison:**

More rigid and precise comparison techniques can be used which compare the faces with more precision under challenging conditions as well.

Image comparison methods such as Template matching, Feature Matching, Comparing Histograms, Structural Similarity Index(SSIM), Dense Vector Representations could be used.

**Tools and Library Used:**

**Programming Language**

* Python

**Pretrained Models:**

* Haar Cascade
* Tesseract
* FaceNet

**Open Source Libraries:**

* OpenCV
* Pytesseract
* Numpy
* Matplotlib

**IDE/Platform**

* Jupyter Notebook

**References**

* <https://docs.python.org/3/>
* <https://docs.opencv.org/4.x/>
* <https://face-recognition.readthedocs.io/en/latest/readme.html#features>
* <http://dlib.net/>