ISSN:0975-9646
Samir Khanal et al/ (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 12 (5) , 2021, 97-101

Nepali Printed Text To Speech with Accurate Transliterated Form

Samir Khanal[@1], Ranjan Paudel[#2], Rajendra Chataut[*3]

[@]*Samakhushi, Kathmandu, Nepal*
[#]*Chandragiri, Kathmandu, Nepal*
[*]*Baneshwor, Kathmandu, Nepal*

3rajendrachatauthq@gmail.com

***Abstract*— As the title of the paper, the research is conducted to read Nepali printed text from a digital image file. The input digital image is obtained by scanning through a scanner or taking a picture from a digital camera of printed Nepali text. It uses Optical Character Recognition (OCR) methodologies to recognize characters in the text image and text to speech conversion approaches to read the identified text aloud. During Optical Character Recognition the input image is passed through different image preprocessing techniques like conversion to grayscale, noise filtering, image smoothing, thresholding (binarization) and then segmentation. The obtained character segments are then fed to a trained Convolutional Neural Network (CNN) which recognizes the Nepali characters. After this, the recognized Nepali characters are converted to Graphemes of English characters using IPA transliteration standard plus our additional characters. The output of the research is the more accurate transliterated form of Nepali text.**

***Keywords*— OCR, TTS, Transliteration, Neural Network**

I. INTRODUCTION

The name 'Nepali Printed Text To Speech' simply means reading the machine generated or printed Nepali text present in a digital image. This research consists of two sections: One, the recognition of Nepali characters (OCR) from the image and the another is converting the recognized Nepali characters text into their respective transliterated form which later will be used by the TTS system to read the recognized text aloud.

OCR, the name itself describes the process of recognizing optically scanned characters. It is an application of machine learning which enables computers to learn without being explicitly programmed.

Optical Character Recognizer(OCR) is the conversion of images of typed, handwritten or printed text(of any language) into machine-encoded text, from photos of a handwritten document, a scanned document, a scene photo or from subtitle text on a picture. It is widely used to convert the entry from bank statements, invoices, business cards, passport documents, computerized receipts, mail, printouts of static-data, or any suitable text documentation. It's a standard method of digitizing printed or drawn texts to be electronically stored, edited, searched, sorted, displayed online, and also used in machine processes like text-to-speech, cognitive computing, machine translation and text mining. OCR is a field of research in computer vision and pattern recognition.

In general, OCR working process can be divided into three stages:

A. Document Pre-Processing

This stage involves digitization of printed text image, noise removal, orientation correction, document layout analysis and other several image operations in order to make the image ready for recognition.

B. Recognition Process

This involves identifying the textual information by analysing the image. The recognized text is then converted to a standard text format.

C. Verification Process

In this stage, the resultant text is looked up for possible errors and this is a multistage process.

The process of recognition consists of a series of different stages, with each stage passing its results on to the next in a pipeline fashion. The recognition process can be divided into three major steps: Pre-processing, Recognition and Post Processing.

After Recognition, comes the Text-To-Speech(TTS) synthesizer which is a computer-based system that should be able to read any text aloud, whether it was directly introduced in the computer by an operator(User) or submitted by an Optical Character Recognition (OCR) System.

II. METHODOLOGY

A. Image Acquisition

Input image for the system is better to have been taken from a scanner or a good quality digital camera with proper lighting. The image is automatically converted to digital form(eg. JPEG, PNG, etc.) by the acquiring devices.

The input image is then carried out through different processing techniques which are explained below.

B. Grayscale Conversion

Grayscale conversion can be done in many ways to convert colorful(RGB) images into monochrome(Gray) form. We have used luminosity method, a weighted mean

method suitable for human eye perception, to convert color image into gray and is carried out by formula:

$$GrayValue = 0.21 \times R + 0.72 \times G + 0.07 \times B \quad (1)$$

Where, *R, G* and *B* are Red, Green and Blue intensity values of a pixel respectively.

C. Gaussian Blurring

Gaussian blurring is the process or technique of smoothing small dot-like noises and distorted edges present in an image. It is based on the normal distribution of intensity of a pixel in the image 2D-plane.

The function of the 2D-Gaussian distribution for the calculation of weights of pixels in a pixel-window is given by,

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{\left(\frac{-x^2+y^2}{2\sigma^2}\right)} \quad (2)$$

Where, *x* and *y* as *(x, y)* represent the pixel vector in a window and σ is the standard deviation of the normal distribution which is determined by using value of radius *r* using the relation,

$$r = \sigma\sqrt{2log(255)} - 1 \quad (3)$$

We have implemented this technique for the window of size *3×3* i.e. radius *r = 2*, which results the standard deviation to be $\sigma = 1.3674$.

D. Contrast Stretching

Contrast stretching is an image enhancement technique that changes the contrast in an image by stretching the range of intensity values it contains to span a desired range of values.

For a mono-channel (gray) image, this technique uses a range [I_{min}, I_{max}] of intensity-levels. Where, I_{min} will be the new minimum intensity level and I_{max} will be the new maximum intensity level after stretching. General formula used for contrast stretching is,

$$s = (r - r_{min}) \left(\frac{I_{max} - I_{min}}{r_{max} - r_{min}}\right) + I_{min} \quad (4)$$

Where, *r* = pixel intensity value; r_{min} = minimum intensity with least number of pixels; r_{max} = maximum intensity with highest number of pixels; *s* = new pixel intensity value.

For this project, we have used range of [*0, 255*] i.e. $I_{min} = 0$ and $I_{max} = 255$. So, the formula becomes,

$$s = 255 \left(\frac{r - r_{min}}{r_{max} - r_{min}}\right) \quad (5)$$

After calculating the value of *s* for each value of *r*, the pixels with intensity value *r* is assigned with corresponding value of *s*.

E. Binarization

An image is called binary if the intensity value of each pixel is either 255 (white color) or 0 (black color) i.e. logically binary values either 1 (high) or 0 (low) respectively. Hence, the process of conversion is called binarization.

We have used a simple thresholding method where the threshold value for all pixels of the image is the same, as the input image is expected to have a light background and dark foreground.

F. Nepali Character Segmentation

The objective of the segmentation module is to segment characters to extract them from the text present in the image document. Statistical approach of pixels for segmentation of characters has been used in this system. Segmentation of Nepali characters, from the family of Devanagari, mainly consists of three major steps: Line Detection, Word Segmentation, Character & Modifiers segmentation.

The approach of segmentation required for the Nepali and other Devanagari scripts is different from that of Roman script. Major characteristics that actually make the difference are:

- A horizontal line in each word that connects characters to form the word, called the header line. e.g. snd·
- Top & bottom modifiers of pronunciation. e.g. top-modifiers: / l / and /] / in / ldn]g /, and bottom-modifiers: / [/ and / ' / in / d[b'n /.
- Half characters. e.g. / G / is the half character of / g /.
- Connected characters. e.g / D / + / r / = / Dr / in / ufDrf /.

So, major steps taken to segment the Nepali characters are:

- Step 1: Detect consecutive rows of pixels, to represent a character line, that contain a number of black pixels greater than a threshold value to indicate presence of character(s), using horizontal projection of black pixels.
- Step 2: Segment words present in the detected line of characters using vertical projection of black pixels, where column(s) with number of black pixels less than a threshold value represent the space between two words.
- Step 3: From each word, remove the header line, pixel-row(s) with maximum number of black pixels, identified by horizontal projection. The section above the header line is of top-modifiers.
- Step 4: Use vertical projection to segment the outcome of Step 3. Each segment can contain: a single character, a half character, a character with top or bottom modifier, or a connected character.
- Step 5: Top modifier is the segment above the removed header line. So, segment the top modifier using the position of the header line.
- Step 6: Use horizontal projection to segment out the bottom modifier from the remaining segment of Step

5. Now the remaining part only is of the core character.

- Step 7: The core character may be a connected character. So, if the width of the character is less than a threshold value then consider it as a non-connected character and proceed, else consider it as a connected character and use the vertical projection to separate individual characters from it [6].

G. Character Recognition

Character recognition consists of two phases: training and recognition phase. For training and recognition of characters, we have used CNN.

We have implemented a Convolution Neural Network using tensorflow for the recognition of Nepali Characters. We have selected tensorflow because it automatically initializes and updates the weights and biases required for CNN implementation. For the basis , We have used two data-set, one train data-set for training and another test data-set for testing. These datasets have been created by ourselves. After the training of the model, we have saved the best model with less error rate and used it for recognition.

We have used three same Sequential CNN models for three different classes of characters; top modifiers, core characters and bottom modifiers. Adam Optimizer with learning rate 0.01 is used for all of them.

Some details of those models are shown below:

TABLE I
CNN MODEL DETAILS

Input Layer	Input Shape
Conv2D layer(2-dimensional convolutional layer)	32 filters with window size (3,3) Activation function: relu
MaxPooling2D(2-Dimensional pooling layer)	poolSize = (3, 3)
Conv2D layer(2-dimensional convolutional layer)	64 filters with window size (3,3) Activation function: relu
MaxPooling2D(2-Dimensional pooling layer)	poolSize = (2, 2)
Dropout Layer	Dropout value(percentage) = 0.1
Flatten Layer	It flattens the 2-dimensional data to linear 1-D data
Simple layer with 300 neurons	Activation function: relu
Simple layer with 200 neurons	Activation function: relu
Simple output layer with 4 neurons	Activation function: softmax

Accuracy of those models in step of training are given below:

- Model for top modifiers has an accuracy of 96.52%.
- Model for core characters has an accuracy of 97.12%.
- Model for bottom modifiers has an accuracy of 77.78%.

H. Transliteration

The conversion of a word written in one script to another script without losing its phonological characteristics is Transliteration. For the transliteration of Nepali language to English language, we have mapped Nepali syllables **To** English syllables as per ITRANS (Indian Languages TRANSliteration).

It is to note that the first vowel / c / in Nepali is mapped to the English letter ‘a’ (short vowel) while the second vowel / cf / is mapped to ‘a̅’ (long vowel as per IPA) in English. The alphabet ‘a’ in English is a short vowel equivalent to / c / which is also a short vowel in Nepali while / cf / in Nepali is a long vowel and mapped to capital ‘A’ in our phonetic scheme. Unicode and ISCII (Indian Script Code for Information Interchange) character encoding standards for Indic scripts are based on the full form of consonants.

We have used IPA(International Phonetic Alphabet) standards and ITRANS(Indian Languages TRANSliteration) for Transliteration. We have added some extra characters and changed some mappings(rules) for the better speech production using the gTTS system.We directly mapped the recognized character into their corresponding transliterated character forms. Transliteration used in our projects are shown in tables below:

TABLE II
FOR TOP MODIFIERS

Character	Label	Transliterated Form
]	0	ae
c	1	av
L	2	ii
l	3	i

TABLE III
FOR BOTTOM MODIFIERS

Character	Label	Transliterated Form
'	0	u
''	1	uu

TABLE IV
FOR CORE CHARACTER

Character	Label	Transliterated Form	Character	Label	Transliterated Form
c	1	a	O	2	i
p	3	u	pm	4	uu
C	5	RRI	P	6	e
s	7	ka	v	8	kha
u	9	ga	3	10	gha
ª	11	nga	r	12	cha
5	13	chha	h	14	ja
झ	15	jha	`	16	~na
6	17	Ta	7	18	Tha
8	19	Da	9	20	Dha
0f	21	nda	t	22	ta

Character	Label	Transliterated Form	Character	Label	Transliterated Form
y	23	tha	b	24	da
w	25	dha	g	26	na
k	27	pa	km	28	pha
a	29	ba	e	30	bha
d	31	ma	o	32	ya
/	33	ra	n	34	la
j	35	wa	z	36	sha
if	37	Sha	;	38	sa
x	39	ha	If	40	kSa
q	41	tra	1	42	Gya
S	43	k	V	44	kh
U	45	g	£	46	gh
R	47	ch	H	48	j
~	49	~n	0	50	nd
T	51	t	Y	52	th
W	53	dh	G	54	n
K	55	p	A	56	b
E	57	bh	D	58	m
N	59	l	J	60	w
Z	61	sh	i	62	Sh
:	63	s	I	64	ks
B	65	ddya	>	66	shra
Q	67	tta	qm	68	kra
¢	69	ddha	2	0	dda

After the transliteration to obtain the phonetic arrangement of vowels following rules are applied:

1. "aa" → "A"
2. "aii" → "aii"
3. "aiia" → "ii"
4. "aae" → "e"
5. "aaea" → "o"
6. "aav" → "ai"
7. "aava" → "au"
8. "ai" → "i"
9. "au" → "u"
10. "auu" → "uu"
11. "ii" → "I"

After the above rules, we implemented custom transliterated character representation forms to obtain the more accurate phonetic arrangement of vowels.

They are:

1. " na" → "Ya" → `
2. " n" → "n"
3. " Na" → "nGa" → ª
4. " N" → "nG"
5. "bha" → "Bha" → e
6. "kSa/kSha/xa" → "KSHa" → If
7. "Na" → "dna"
8. "Na " → "dn"
9. ".N" → "."

III.RESULTS

यहाँसम्म सबैको बुभफई र भोगाई समान छ

Fig. 1 Sample Test Image for recognition.

नङलाई स्वस्थ राख्न क्यालिसियम तथा जिंकको आवश्यकता पर्छ

Fig. 2 Sample Test Image for recognition.

राष्ट्रिय निकुञ्ज तथा वन्यजन्तु संरक्षण विभागले सन् दुई हजार अठारमा गरेको एक अध्ययन अनुसार

Fig. 3 Sample Test Image for recognition.

TABLE V
TRANSLITERATION FOR SHOWN IMAGES 1, 2 AND 3

Image	ITRANS	Optimized Word
Test Image[1]	yahA.Nsamma sabaiko bubhaphaI ra bhogAI samAna Cha	yahA.Nsamma > yahA.samma
Test Image[2]	na NalAI svastha rAkhna kyAlsiyama tathA jiMkako AvashyakatA parCha	na NalAI > nanGalAI
Test Image[3]	rAShTriya niku nja tathA vanyajantu saMrakShaNa vibhAgale san duI hajAra ThAramA gareko eka adhyayana anusAra	niku nja > nikunja And vibhAgale > viBhAgale

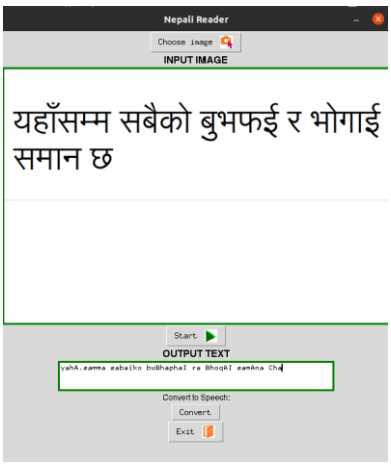

Fig. 4 Sample Output After Test

We have tested many Nepali words including `, C, e and If. Some of the words tested are given below.

TABLE VI
SOME TESTED NEPALI WORDS

Words	**ITRANS**	**ITRANS Pronunciation**	**After Custom Rules**	**New Pronunciation**
k~r	pa~ncha	[pa tild cha]	pancha	[pancha]
s'~h	ku~nja	[ku tild nia]	kunja	[kunja]
c~rn	a~nchala	[a tild chala]	anchala	[anchala]
vf\6f^{a}	khoTA~Na	[khota tild Na]	khoTAnGa	[khoTAnGa]
tfdf^{a}	tAmA~Na	[tama tild na]	tAmAnGa	[tAmAnGa]
ln^{a}\u	li~Nga	[li tild na]	linGga	[linGa]
c^{a}\u	a~Nga	[a tild na]	anGga	[anGa]
9^{a}\u	Dha~Nga	[Dha tild na]	DhanGga	[DhanGa]
cbe't	adabhuta	[adabhuta]	adaBhuta	[adaBhuta]
ceb|	abhadra	[abhadra]	aBhadra	[aBhadra]
cefuL	abhAgI	[abhAgi]	aBhAgI	[aBhAgi]
cfef/L	AbhArI	[AbhAri]	ABhArI	[ABHAri]
b]ze/	deshabhara	[deshabhara]	deshaBhara	[deshaBhara]
c+tl/If ofqL	aMtarikShayAtrI	[aMtarikShayAtrI]	aMtariKSHayAtrI	[aMtariKSHayAtrI]
k|ToIf bzL{	pratyakShadarshI	[pratyakShadarshI]	pratyaKSHadarshI	[pratyaKSHadarshI]
cf/If0f	ArakShaNa	[ArakShaNa]	AraKSHaNa	[AraKSHaNa]
bLIfLt	dIkShAMta	[dIkShAMta]	dIKSHAMta	[dIKSHAMta]
kIfkftL	pakShapAtI	[pakShapAtI	paKSHapAtI	[paKSHapAtI
ckx/0f	apaharaNa	[apaharaNa]	apaharadn	[apaharadn]
ck{0f	arpaNa	[arpaNa]	arpadn	[arpadn]
s?0ff	karUNA	[karUNA]	karUdnA	[karUdnA]
s[i0f	kRRiShNa	[kRiShNa]	kRRiShdn	[kRiShdn]
zf]if0f	shoShaNa	[shoShaNa]	shoShadn	[shoShadn]
C0f	RRiNa	[RRiNa]	RRidn	[RRidn]

IV. CONCLUSIONS

The method presented in this paper can process the Nepali Printed Text image, segment, recognize, transliterate the text in the image and convert it into more accurate speech using gTTS. We have also introduced some post-processing rules after the transliteration for more accurate pronunciation from transliterated form to speech.

There is still some further research to do. For example, it can't work with non-printed or handwritten Nepali text.This problem will be solved in the future.

REFERENCES

[1] Casey, R.G. and Lecolinet, E., A survey of methods and strategies in character segmentation, IEEE Transactions on Pattern Analysis and Machine Intelligence, 18, 690–706 (1996)

[2] BAG, S., HARIT, G., A survey on optical character recognition for Bangla and Devanagari scripts., Sadhana, 38,133–168(2013)

[3] Pant, Nirajan and Bal, Bal Krishna, Nepali Optical Character Recognition - A Hybrid Approach, (2016), 10.13140/RG.2.2.33676.72327

[4] Chen, Yutian and Assael, Yannis and Shillingford, Brendan and Budden, David and Reed, Scott and Zen, Heiga and Wang, Quan and Cobo, Luis C. and Trask, Andrew and Laurie, Ben and Gulcehre, Caglar and Oord, Aäron vanden and Vinyals, Oriol and Freitas, Nandode, Sample Efficient Adaptive Text-to-Speech,(2018)

[5] Gonzalez, Rafael C. and Woods, Richard E., Digital Image Processing, Pearson Education Pte. Ltd., 482 F.I.E. Patparganj, Delhi 110 092, India (2003)

[6] Veena Bansal, R.M.K. Sinha, Segmentation of touching and fused Devanagari characters,Pattern Recognition, Volume 35, Issue 4, 875-893 (2002),ISSN 0031-3203