HANDWRITTEN TEXT RECOGNIZER USING DEEP NET

A PROJECT REPORT

Submitted by

Varun Varatharajan 19BCE0538
Dussa Lalith Kumar 19BCE0532
Kush Dang 19BCE0573

Course Code: CSE 3013

Course Title: Artificial Intelligence

Under the guidance of

Dr. S. Anto

Associate Professor, SCOPE,

VIT , Vellore.

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

December, 2021

CONTENTS **PAGE NO.**

INTRODUCTION

- There is a dire need for an algorithm that can read and convert handwritten text to a computerized text format.
- The applications of such a software are endless, being able to convert physical chunks of scripts into a digital form and so much more.
- We intend to a make an algorithm that can effectively make meaningful sense of out different handwriting styles and accurately convert handwritten text to a digitally understandable format
- The challenging part is that there are infinite ways in which a user can write a text.
- Our system instead of using traditional approach of segmenting individual characters for recognition is using the combination of Neural Networks , classification and decoding algorithms for converting the the handwriting to digital text which are usable within computer and text processing applications

ABSTRACT

Given the universality of handwritten documents in human transactions, Optical Character Recognition (OCR) of documents have invaluable practical worth. Optical character recognition is a science that enables to translate various types of documents or images into analysable, editable and searchable data. The recognition of handwritten text is challenging as there are virtually infinite ways a human can write the same message.Our method provides an alternate approach to the popular strategy of using recurrent neural networks followed by connectionist temporal classification for character cleanup. Although connectionist temporal classification avoids the need for properly segmented labeled data, it can be difficult to tune.Our method aims to recognize common words as well as infinite symbol blocks such as surnames, phone numbers, and acronyms. The pairing of word blocklength prediction along with a family of even convolution filters enable accurate symbol alignment. We have noticed the usage of even convolution filters in signal processing hence we decided to adapt it in your convolutional neural network. When it comes to the performance metrics under consideration, obtaining 99% page level accuracy is not always good, because the 1% of the page with errors could contain characters required by the business.

PROBLEM DEFINITION

- We intend to create a handwritten text recognizer tool, that can efficiently convert images of hand written english text into a computerized format, with a high accuracy. Most algorithms are only successful with character recognition.While decoding a hand written word, most algorithms choose the best fit and not the most optimal word that makes sense.We aim to increase optimization.
- Handwriting Recognition, also known as Handwritten Text Recognition (HTR), is still regarded as a difficult problem statement. The wide range of handwriting styles among people, as well as the poor quality of handwritten text compared to printed text, present significant challenges in converting it to machine readable text.
- We need to structure an architecture and sequence of necessary neural networks to achieve the desired conversion with a respectable accuracy.

RESEARCH PAPERS

Base paper 1: Intelligent character recognition using fully convolutional neural networks

Authors: Raymond Ptuchaa, Felipe Petroski Sucha, Suhas Pillai

Methodology:

This paper describes a fully convolutional network architecture for generating arbitrary length symbol streams from handwritten text. A preprocessing step converts input blocks to a canonical representation, eliminating the need for expensive recurrent symbol alignment correction. When a lexicon is available, we add a probabilistic character error rate to correct errant word blocks. They claim their multi-state convolutional method is the first to achieve cutting-edge performance on both lexicon-based and arbitrary symbol-based handwriting recognition benchmarks.

Limitations and Future works:

Includes experimentation with larger contextual filters, the addition of sentence-level features such as sent2vec, the use of hierarchical processing to gain knowledge at the symbol, word, sentence, and paragraph levels, the use of fully convolutional recurrent networks, and the demonstration of these methods to more languages, resource constrained languages, and other recurrent applications are all planned for the future.

Base paper 2: Handwritten Optical Character Recognition (OCR): A Comprehensive Systematic Literature Review (SLR)

Authors: JAMSHED MEMON, MAIRA SAMI, RIZWAN AHMED KHAN

Methodology:

The goal of this review paper is to outline previous research on character recognition in handwritten documents and to suggest future research areas. We gathered, in this Systematic Literature Review (SLR), Research publications on handwritten OCR were compiled and assessed (and closely related topics) which were released during the years of 2000 and 2019. We used widely used electronic databases as a guide. following a predetermined review procedure Keywords and forward reference searching were used to find articles. Backward reference searching is used to find all articles that are relevant to the topic. following a thorough examination after the research selection procedure For this SLR, 176 articles were chosen. This article acts as a review. The goal of this paper is to discuss current state-of-the-art results and approaches in OCR, as well as research directions. pointing out research gaps.

Conclusion, Limitations and Future work:

For the past eight (8) decades, optical character recognition has been in use. Initially, however, products that recognise optical characters were primarily developed by large technology companies.

The majority of published research studies propose a solution for a single language or a subset of a language.

Publicly available datasets also include stimuli that are well aligned with one another but do not include examples that correspond well with real-life scenarios.

Researchers are also increasingly employing Convolutional Neural Networks (CNN) for the recognition of handwritten and machine-printed characters.

Characters written by different people create a lot of intra-class variability, making it difficult for classifiers to perform well. Although, with the increased use of complex deep learning architectures, obtained classification accuracy has improved, computational complexity (particularly during the classifier's training phase) has increased. This puts a stumbling block in the development of a real-time, robust system for htr.

Base paper 3: Offline Handwritten Numeral Recognition using Combination of Different Feature Extraction Techniques

Authors: Munish Kumar, M. K. Jindal, R. K. Sharma

Methodology:

They created a skeleton of the numeral in order to extract meaningful feature information about it. A combination of four types of features, namely centroid features, diagonal features, zoning features, and peak extent based features, was used for feature extraction. For classification, the SVM classifier was considered. 6000 samples of isolated handwritten numerals were considered for experimental results.

Conclusion, Future work and Limitations:

In this work, SVM with RBF kernel classifier was used to achieve maximum recognition accuracy of 85 percent or higher for handwritten numeral recognition.

This system can recognise handwritten characters in a variety of scripts, including Bengali, Devanagari, and Tamil. It has also been discovered that training a classifier with the greatest number of features acquired is not always the best option. As a result, the feature selection technique of PCA is used to reduce the dimensionality of the feature vector.

The methodology can be extended to alphabets and recognizing individual alphabets using similar feature extraction techniques.

OBJECTIVES OF WORK

Our system instead of using traditional approach of segmenting individual characters for recognition is using the combination of Neural Networks , classification and decoding algorithms for converting the the handwriting to digital text which are usable within computer and text processing applications

We intend to use a combination of convolutional neural networks, recurrent neural networks using a connectionist temporal classification with an optimal decoding algorithm.

- Being able to recognize sentences as a whole.
- Being able to accurately generate the word that makes sense and not the word that gives the best fit.

SOFTWARE REQUIREMENTS

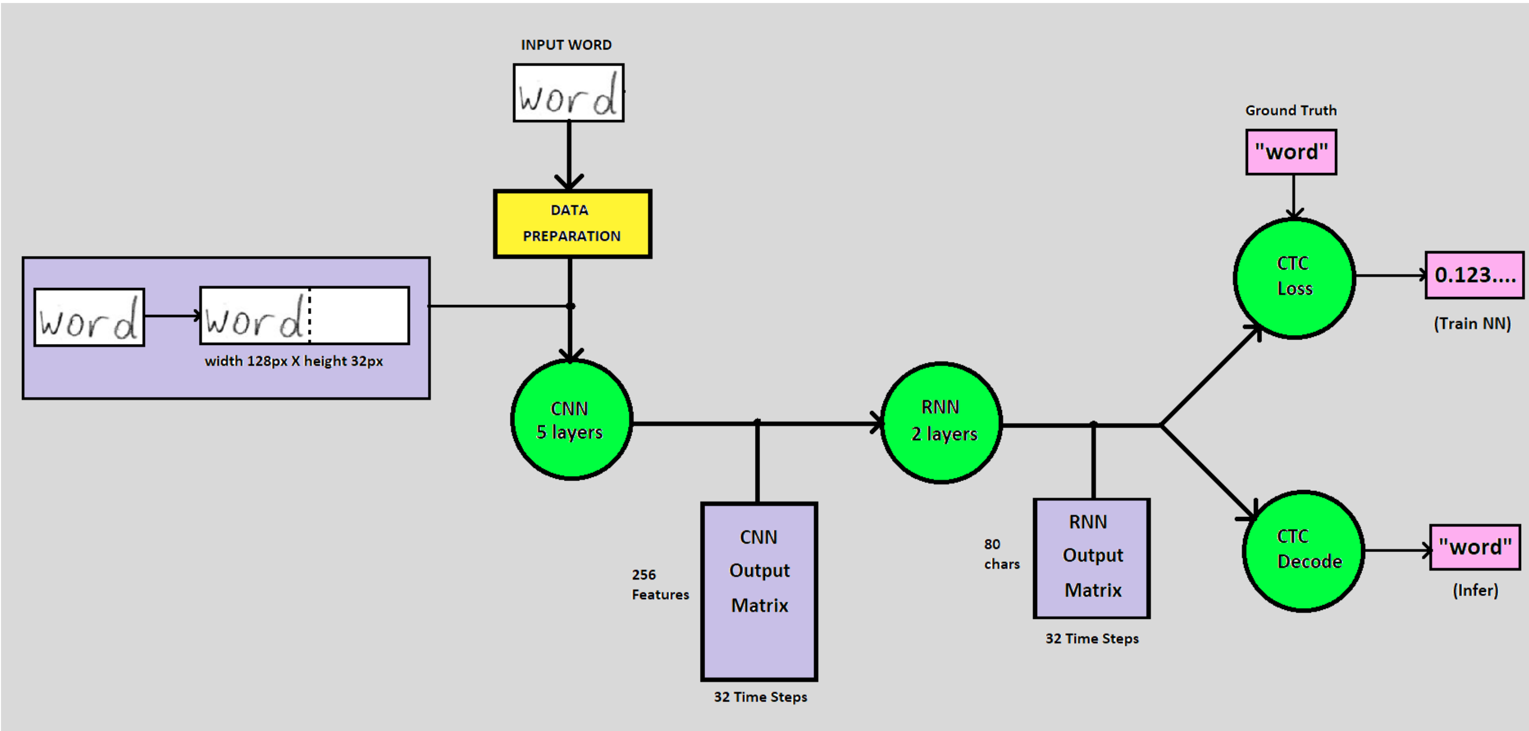
- editdistance==0.5.2
- lmdb==1.0.0
- matplotlib==3.2.1
- numpy==1.19.5
- opencv-python==4.4.0.46
- path==15.0.0
- tensorflow==2.4.0
- Visual Studio

HARDWARE REQUIREMENTS

Ideal conditions:

- -8gb ram
- -A graphic processing unit GTX 1050 Ti

SYSTEM DESIGN



OVERVIEW OF SYSTEM

DATA PREPARATION: File concept is used in dataloader.py to load data from our dataset and converted into array format for easy traversing. All input images are cropped closely and fit to dimensions width 128 px X 32 px. The input images should have high contrast and words handwritten, so that it's easy to differentiate between the data points.

DATA SET

We will be using the IAM dataset. The IAM Handwriting Database contains forms of handwritten English text which can be used to train and test handwritten text recognizers and to perform writer identification and verification experiments.

We will split the dataset into two subsets with a 95:5 ratio (train : validation).
Total training samples: 91633

Total validation samples: 4823

No. of epochs: 45

CNN: The input image is used to train the CNN levels. These layers are trained to extract relevant image features. Each level is made up of three operations. First, there is the convolution operation, which uses a 5 x 5 filter kernel in the first two layers and a 3x 3 filter kernel in the last three layers at the input. The nonlinear RELU function is then used. Finally, a layer pooling summarizes the image regions, and the entry will be published in a resize version.

RNN: the feature sequence has 256 features per time step, and the RNN propagates relevant information through it. The popular Long Short-Term Memory (LSTM) RNN implementation is used because it can propagate information over longer distances and has more robust training characteristics than vanilla RNN. The RNN output sequence is mapped to a 32X80 matrix. The IAM dataset contains 79 distinct characters, plus one additional character required for the CTC operation (CTC blank label), for a total of 80 entries for each of the 32 time-steps.

CTC: The CTC is given the RNN output matrix and the ground truth text while training the NN and computes the loss value. The CTC is only given the matrix while inferring and decodes it into the final text. Both the ground truth text and the recognised text can have a maximum length of 32 characters.

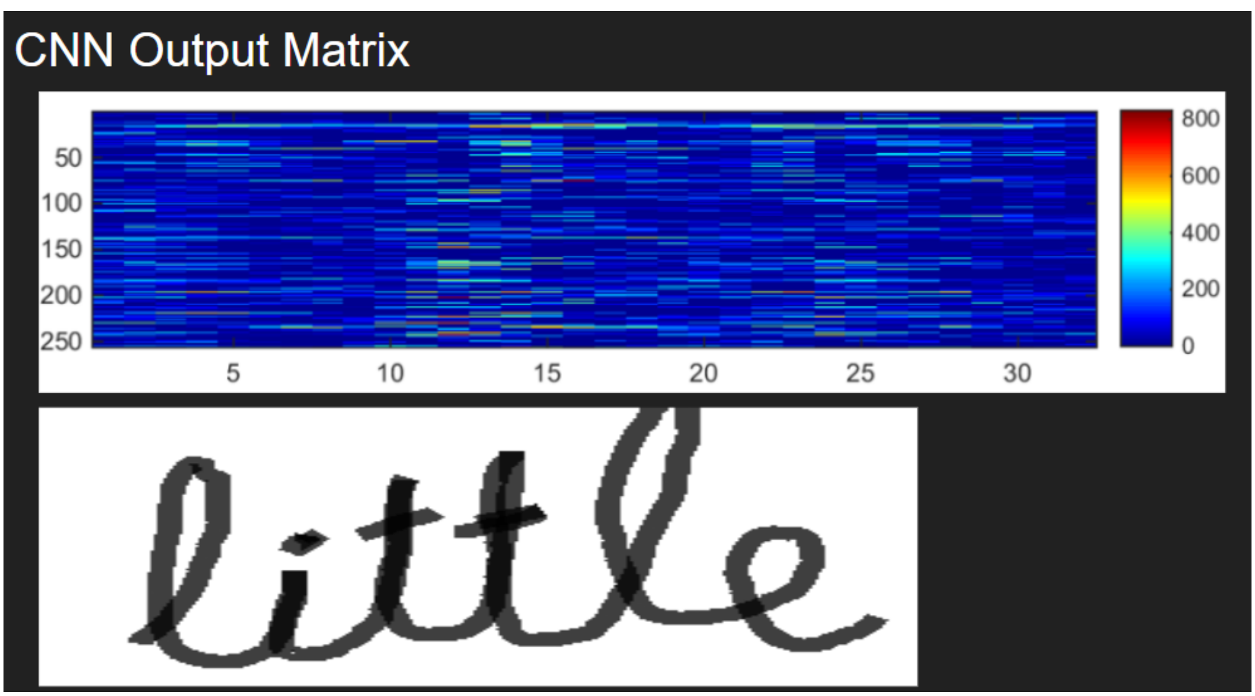
ALGORITHM

CNN

The function CONV2D is used to calculate the dot product of the input matrix and the kernel matrix.

The non linear function RELU is used to convert the negative data points to zero and keep positive data points the same.

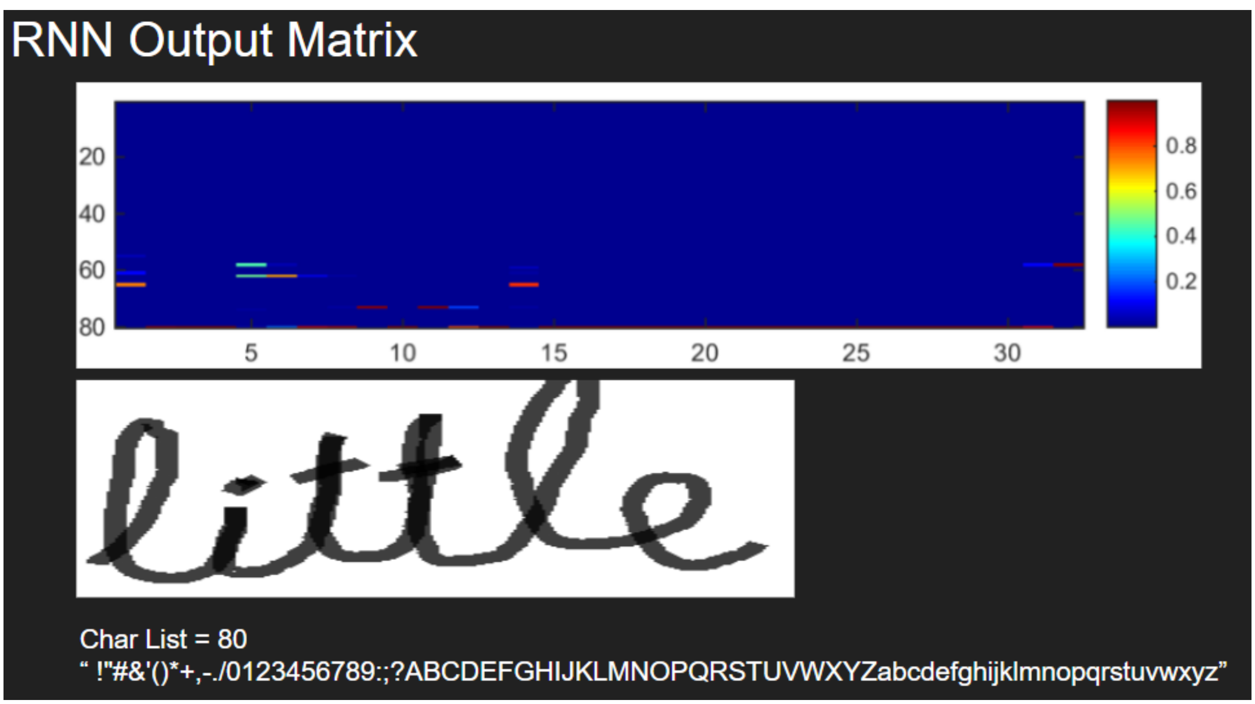
Next , the function max pool is used to summarize image regions.



Above is the rainbow color plot of the CNN output matrix. Although the above matrix will be further processed by RNN we can already see similarities and co-relations with the input word.

RNN:

2 RNN layers of 256 units each were created and stacked. Bi-directionsal RNN was created from this , so that the input word is traversed front to the end and from the back to the front. Bi-directional RNN is used so that at a particular timestamp ,we have information about both from the front and back to predict at that point. Finally it is mapped to a matrix of 32 X 80.



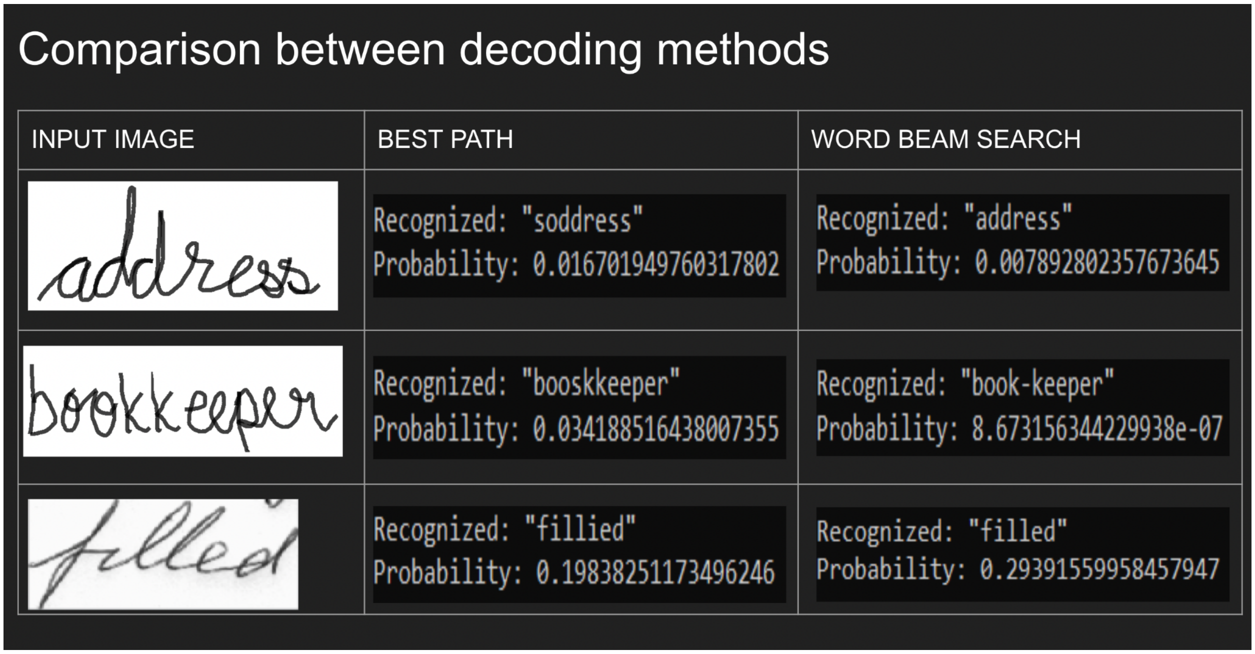
Above is the rainbow color plot of the RNN output matrix. X-axis has a 32 timesteps matrix and Y-axis has the index char list[] array. We can see the occurrence of double *T* in the plot.

DECODING ALGORITHMS

Best Path (Word accuracy: 73.482%) It calculates the best path by taking the most likely character per time-step. It undoes the encoding by first removing duplicate characters and then removing all blanks from the path. What remains represents the recognized text.

WordBeamSearch (Word accuracy: 84.304%) The output words are constrained by a dictionary, but all the other characters are just added in the way the NN sees them. If the output is not in the dictionary, then the inferred word is the same as the best path.

OUTPUT SNAPSHOTS



COMPARISON WITH EXISTING SYSTEMS

HTR	Handwritten English character recognition using a neural network. (*VIJAY P & SANJAY S*)	MetaHTR: Towards Writer-Adaptive Handwritten Text Recognition. (*AYAN K , AMAN K*)	**HTR USING DEEP NET**	Handwritten Text Recognition and Conversion using CNN based Deep Learning Model. (*Jebaveerasingh, Immanuel Johnraja*)
MODEL	Neural network with 40 inputs, using Error back propagation training(EBPT) and feed forward recall.	CNN - 2 layer Bi-LSTM - Attention module (recursive) - GRU decoder.	**5 CNN layers - 2 layer Bi-LSTM (RNN) - CTC loss, decoding layer (WordBeamSearch)**	CNN layers - LSTM layers - CTC loss layer.(Word segmentation algorithm)
ACCURACY	70%	81.3%	**84.3%**	85%

PERFORMANCE METRICS

Best Path (Word accuracy: 73.482%)

```
(venv) C:\Users\Varun\Desktop\Handwritten Text Recognizer\venv\Scripts\src>py main.py
 --mode validate --decoder bestpath --data_dir ../dataset/
```

```
[OK] "'s" -> "'s"
[OK] "awfully" -> "awfully"
[ERR:1] "lucky" -> "lucdy"
[OK] "." -> "."
[ERR:1] "I" -> ","
[OK] "wish" -> "wish"
[OK] "I" -> "I"
[OK] "went" -> "went"
[OK] "to" -> "to"
[OK] "that" -> "that"
[ERR:1] "school" -> "sthool"
[OK] "." -> "."
[ERR:1] "Did" -> "Aid"
[ERR:1] "you" -> "You"
[OK] "notice" -> "notice"
[OK] "that" -> "that"
[ERR:3] "girl" -> "gd"
[OK] "who" -> "who"
[ERR:1] "said" -> "sad"
[ERR:1] "hullo" -> "Lullo"
[OK] "to" -> "to"
[ERR:1] "him" -> "hin"
[OK] "in" -> "in"
[OK] "the" -> "the"
[OK] "garden" -> "garden"
[ERR:1] "?" -> "!"
Character error rate: 10.914349276974416%. Word accuracy: 73.48248352410684%.
```

WordBeamSearch (Word accuracy: 84.304%)

```
(venv) C:\Users\Varun\Desktop\Handwritten Text Recognizer\venv\Scripts\src>py main.py
 --mode validate --decoder wordbeamsearch --data_dir ../dataset/
```

```
[OK] "'s" -> "'s"
[OK] "awfully" -> "awfully"
[OK] "lucky" -> "lucky"
[OK] "." -> "."
[ERR:1] "I" -> ","
[OK] "wish" -> "wish"
[OK] "I" -> "I"
[OK] "went" -> "went"
[OK] "to" -> "to"
[OK] "that" -> "that"
[OK] "school" -> "school"
[OK] "." -> "."
[OK] "Did" -> "Did"
[ERR:1] "you" -> "You"
[OK] "notice" -> "notice"
[OK] "that" -> "that"
[ERR:2] "girl" -> "gr"
[OK] "who" -> "who"
[ERR:1] "said" -> "sad"
[OK] "hullo" -> "hullo"
[OK] "to" -> "to"
[ERR:1] "him" -> "hin"
[OK] "in" -> "in"
[OK] "the" -> "the"
[OK] "garden" -> "garden"
[ERR:1] "?" -> "!"
Character error rate: 8.431590656284762%. Word accuracy: 84.30454387790496%.
```

CONCLUSION

We discussed a neural network that can recognise text in photos. The Neural Network outputs a character-probability matrix and has 5 Convolutional Neural Networks(CNN) and 2 Recurrent Neural Network(RNN) layers. This matrix is either used to calculate Connectionist Temporal Classification(CTC) loss or to decode CTC. An implementation using Tensor Flow is supplied, as well as certain key elements of the code. Finally, suggestions for improving recognition accuracy were offered.

REFERENCES

(IEEE)

[1]Memon, J, Sami, M, Khan, RA & Uddin, M 2020, 'Handwritten Optical Character Recognition (OCR): A Comprehensive Systematic Literature Review (SLR)', IEEE Access, Access, IEEE, vol. 8, pp. 142642–142668, viewed 21 August 2021, <https://search.ebscohost.com/login.aspx?direct=true&db=edseee&AN=edseee.9151144&site=eds-live>.

(Science Direct)

[2]Ptucha, R, Petroski Such, F, Pillai, S, Brockler, F, Singh, V & Hutkowski, P 2019, 'Intelligent character recognition using fully convolutional neural networks', Pattern Recognition, vol. 88, pp. 604–613, viewed 21 August 2021, <https://search.ebscohost.com/login.aspx?direct=true&db=edselp&AN=S0031320318304370&site=eds-live>.

(Springer)

[3]Kumar, M(1), Jindal, MK(2), Sharma, RK(3) & Jindal, SR(4) n.d., 'Offline Handwritten Numeral Recognition using Combination of Different Feature Extraction Techniques', National Academy Science Letters, vol. 41, no. 1, pp. 29–33, viewed 21 August 2021, <https://search.ebscohost.com/login.aspx?direct=true&db=edselc&AN=edselc.2-52.0-85044395850&site=eds-live>

(IEEE)

[4]Scheidl, H. and Fiel, S. and Sablatnig, R., 'Word Beam Search: A Connectionist Temporal Classification Decoding Algorithm', 16th International Conference on Frontiers in Handwriting Recognition, IEEE, 2018 <https://repositum.tuwien.at/retrieve/1835>