International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

# Eval - Automatic Evaluation of Answer Scripts using Deep Learning and Natural Language Processing

**Prerana M S[1], Sagarika M Chavan[2], Ramit Bathula[3], Sreenath Saikumar[4], Dr. Geetha Dayalan[5]**

**Submitted:** 04/11/2022 **Revised:** 19/12/2022 **Accepted:** 04/01/2023

**Abstract:** Professors face a lot of difficulties when it comes to correcting handwritten answer booklets manually. It is both time consuming and labour intensive. As a solution to this problem, the paper proposes a system that automatically evaluates answer booklets, thereby saving time and effort. The proposed method involves using Deep Learning and Natural Language Processing techniques to automate the evaluation process. The first step is to extract the handwriting from input image files using an existing GCP OCR (Google Cloud Platform - Optical character recognition) text extract model, which has superior accuracy and performance to other models. It also uses various Natural Language Processing techniques such as BERT (Bidirectional Encoder Representations from Transformers) to extract keywords; and GPT-3 (Generative Pre-trained Trans- former 3) to summarize long answers. This method has been observed to assign marks that are usually identical to hand- evaluated marks. This paper also proposes a web application that simplifies the process of evaluating answer scripts. The web application generates the text extracted from both the student's answer and the answer key image files, the summary of the student's answer and the marks obtained based on the extracted keywords.

**Keywords:** Natural Language Processing (NLP), Bidirectional Encoder Representations from Transformers (BERT), third generation Generative Pre-trained Transformer (GPT-3), Long short-term memory (LSTM), Support Vector Machine (SVM), Convolutional Neural Network (CNN), Gradient-Boosted Decision Trees (GBDT), Natural Language Toolkit (NLTK)

## 1. INTRODUCTION

In our daily lives, we notice that Professors face a lot of trouble correcting handwritten answer booklets where evaluation is done manually. Manual evaluations may be influenced by mood swings of the evaluator. Furthermore, manual evaluation is a time-consuming and labor-intensive task. It is also difficult to store and access the answer booklets as mentioned in [1]. As a result, the question of how to automate the process of evaluation of answer booklets making it faster, more accurate, and with less effort arises. This research also conducts a performance comparison of various models.

A Handwriting Recognition model is required to extract raw text from answer booklets, which is then passed on to NLP models. According to [2], the handwriting recognition model employs a sequence-to-sequence neural network, which requires robust
hardware (GPUs) to train the CNN model, has a very long training time, and performs poorly. As a result, this paper aims to find a handwriting recognition model that requires less training time, does not require robust GPUs, is trained on a large dataset, and performs well.

The NLP aspect of the work involves keyword extraction, summarization, and computing the marks. According to [3], when words have multiple meanings, it is difficult to extract a summary by understanding the exact context of the text. Therefore, a solution that understands the text and generates a summary is required. This paper also seeks to identify a model that excels at keyword extraction. It is critical to fine-tune the keywords using various Natural Language Processing techniques.

[1] *PES University, Bengaluru, Karnataka, India*
*Email ID: pes2ug19cs296@pesu.pes.edu*
*ORCID ID: 0000-0002-8882-7138*
[2] *PES University, Bengaluru, Karnataka, India*
*Email ID: pes2ug19cs347@pesu.pes.edu*
*ORCID ID: 0000-0003-0406-6712*
[3] *PES University, Bengaluru, Karnataka, India*
*Email ID: pes2ug19cs319@pesu.pes.edu*
*ORCID ID: 0000-0001-7772-6326*
[4] *PES University, Bengaluru, Karnataka, India*
*Email ID: pes2ug19cs406@pesu.pes.edu*
*ORCID ID: 0000-0003-3417-3607*
[5] *PES University, Bengaluru, Karnataka, India*
*Email ID: geethadayalan@pes.edu*
*ORCID ID: 0000-0001-5101-4582*

## 2. RELATED WORK

A technique for extracting terms from sentences has been developed by Himani, et al. [4]. This can be utilized as information retrieval keywords and for effective searching. The automatic keyword extraction process used in this study extracts keywords into three groups: text-based, database-based, and text-and-database-based keywords. The suggested approach seeks to extract keywords (Noun Phrases) from online video lecture transcripts. A named

entity recognition and syntactic grammar technique are used in the study to identify glossary terms. The keywords, along with their PoS Tags and Chunk Tags, were successfully recovered using NP Tag patterns (Grammar) Methods. Due to the complexity of noun phrase construction, it has been difficult to create a set of rules that will capture all NP chunks while excluding verb phrases in the grammar-based approach. The NLTK Chunker's performance is determined by the quality of the manually constructed NP language.

A suggestion from Bojja, et al. [1] conducting research that primarily focuses on how pre-trained models like Tesseract and GTTS operate and how they are used to address issues. These models, in turn, use Decision Trees and Neural Networks to solve and separate the characters and words. APIs such as GTTS rely heavily on recognition (Google text-to-speech). The model identified and processed 17 characters from the input image, with 16 correctly identified and 1 incorrectly identified, yielding an accuracy of 94% for this input and an overall accuracy of 92.7%. Because of these precise and clear voice results, we can hear the recognized text. The recognition procedures are heavily influenced by the type of data that needs to be identified. Cursive writing is more difficult because the letters are frequently connected, written incorrectly, or even absent, making the writing essentially confusing.

The use of a combination of a deep convolutional neural network with an encoder-decoder with an attention mechanism, called sequence to sequence has been proposed by Sueiras, et al. [2] to identify characters and contextualize them with their neighbors to recognize any given word. CNN generates a sequence of visual features from each part of the word image by modeling the visual attributes of a handwritten word from a segmented, preprocessed word patch. This sequence is fed into the Seq2Seq model as an input. The encoder-decoder (LSTMs) functionality of Seq2Seq is used to identify the characters. When compared to competing models at the time of writing, this ensemble network produced lower word errors (19.85) and character errors (6.8). These statistics represent the RIMES and IAM dataset's average errors. Because CNN is used for feature extraction, there is very little preprocessing (skewness/slant removal, word patching, resizing) required. To train the CNN model, powerful hardware (GPUs) is required. Poor performance with words not found in the English lexicon, such as names. Long training time slows down hyperparameter tuning, which is intended to improve model performance.

In [3], the authors have proposed a survey of various text summarization techniques. According to the results of this survey, the seq2seq model, LSTM, and attention mechanism are used to improve accuracy. The work briefly discusses various text summarization forms, including Single Document, Multi-Document, Extractive, Abstractive, Generic, Domain-Specific, and Query Base. To summarize and categorize the data, this paper employs hybrid classifiers such as SVM and Naive Bayes. To summarize, a seq2seq model is suggested. Long Short-Term Memory, a more evolved version of it, is used in tandem with an attention mechanism to improve the accuracy of the created summary. The study concluded that increasing the number of classifiers could improve accuracy. When a word has multiple meanings, use domain-specific text summarization. This paper proposes a co-reference resolution mechanism to address the problem caused by incorrect referencing. Multiple documents demonstrated lower accuracy when compared to summarizing a single document.

Analyzing the answers from the chosen students' University examinations has been suggested by Sanuvala, et al. [5]. The paper describes a strategy for using deep learning to predict a student's paper grade. The evaluated answer text files are used to create the trained model. The human answer key text file is also consulted during training, andeach sentence has a corresponding mark. ML methods suchas NB(Naive Bayes), SVM (Support Vector Machine), and GBDT(Gradient-Boosted Decision Trees) are used. NB: todetermine the maximum power of a lateral potential for agiven input. With Chi-squared enhancement, SVM improved lemmatization results. The fragmentation of the training process is reduced by using GBDT. It can be difficult to determine the appropriate (dis)similarity level between papers.It can be challenging to select the appropriate document features to compare. Only for specific responses does the system provide an accurate estimate.

A Handwritten Text Recognition (HTR) model architecture has been proposed by Singh, et al. [6]. The model is based on neural networks that can be trained to read entire pages of printed or handwritten text without image segmentation. Because it is based on the Image to Sequence architecture, it can extract text from an image and accurately sequence it without regard for the orientation, layout, or size of text and non-text. The paper makes use of character-level vocabulary, allowing any subject's language and terminology to be used. The model achieves a brand-new state-of-the-art paragraph-level recognition of the IAM dataset. When tested on scans of actual handwritten free-form test responses filled with curved and slanted lines, drawings, tables, math, chemistry, and other symbols, it outperforms all commercially available HTR cloud APIs. When tested on Free Form Answers after being trained on all datasets, the model's error rate is 7.6%, compared to 14.4% for the best cloud API. Even though the presented architecture spans numerous tasks, the given datasets are typically significantly biased towards one or two tasks, obscuring the model's performance on outlier tasks. To train the model, text up to 1100 characters long and averaging 360 characters is used. Larger sequence lengths require more

labor to handle if longer texts, such as 10K characters, must be transcribed. According to the study, the Full-Page HTR issue will not be considered "fixed" until the error rate is less than 1%.

A project that tries to classify each solitary handwritten word to digitize handwritten writing has been proposed by Balci, et al. [7]. The study uses two fundamental techniques to accomplish this task: direct word categorization and character segmentation. The study employs Convolutional Neural Networks (CNN) with a range of topologies to train a model that can accurately categorize words. First, as part of the investigation, the Word-level classification model using VGG-19 was trained. However, considering the number of parameters required, it was found that the model required a lengthy training period. The test accuracy with VGG-19 was found to be 20%. RESNET-18 could produce comparable findings at even faster rates. The RESNET-18 test accuracy is 22%. Following a results analysis, RESNET-34 was used in the study. For RESNET-34, training accuracy was 35% and validation accuracy was 27%. For the latter, the study uses convolution and Long Short-Term Memory networks (LSTM) to build bounding boxes for each character. The study submits the segmented characters to CNN for categorization. Based on the results of categorization and segmentation, CNN reconstructs each word. Due to time and resource constraints, the study has only been able to use 20 training samples for each phrase provided to analyze and enhance the model. The model's training is unreliable and limited to the dataset.

According to Bluche, et al. [8], state-of-the-art Word error rates (WERs) can be attained using both Recurrent Neural Networks (RNNs), a common method for handwriting recognition, and Deep Multi-Layer Perceptrons (DeepMLPs), a typical way for speech recognition. The purpose of the paper is to demonstrate that the suggested hybrid systems, regardless of the type of features (hand-crafted or pixel values) and the neural network optical model, produce performance comparable to the state-of-the-art (DeepMLP or RNN). The claim stands that while DeepMLPs, which are now common in hybrid voice recognition systems, can perform just as well, RNNs, which have become a standard component of handwriting recognition systems, can. The proposed models' robustness hasn't been evaluated in this study, therefore it's impossible to know how well they might perform when used with fresh databases that weren't used for training.

## 3. MATERIALS AND METHODS

### 3. 1. Tools and Techniques

Due to a lack of commercially available data, most hand-writing recognition models produce low accuracy scores and high error rates. The performance of the Cloud Vision and Textract OCR models were significantly better due to the vast difference in training data in terms of quantity, variety, and computational power leveraged.

Cloud-based solutions such as the Google Cloud Vision API and AWS Textract APIs have been investigated from a commercial and performance standpoint. The ability to process pages asynchronously will result in considerable performance increases due to the capacity to process many pages at once. Since these models were trained on millions of publicly available documents and internal data, it makes sense that they would have a significantly shorter prediction time and perform considerably better on unseen data. The use case diagram for the Handwriting Recognition model has been shown in Fig.1.

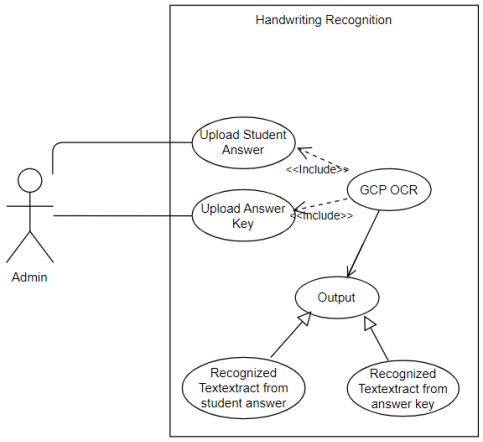


Fig 1. Use Case Diagram for Handwriting Recognition model

At first, the paper investigated keyword extraction and summarization using both BERT and GPT3. It has been discovered that BERT has a more significant encoder capability for generating contextual embedding from a sequence, making it a better model for keyword extraction. Whereas GPT-3 is stronger on the decoder side for taking in context and generating new text, making it the best model for summarization. Based on the performance comparison of the two models for keyword extraction and summarization, this paper proposes BERT model for keyword extraction and the GPT-3 model for summary. A use case diagram of the NLP aspect of the work has been shown in Fig.2.

For keyword extraction, the BERT 'all-mpnet-base-v2' model has been used. This model takes raw text as input. To begin with, the input will be tokenized. This text will then be sent to the BERT model, which recognizes and extracts keywords from both the student answer and the answer key provided by the teacher. The keywords with a cosine distance greater than 0.2 will then be chosen. These keywords are then case-folded, POS-tagged, and submitted to the WordNet lemmatizer. The grading process begins once all of the lemmatized keywords from the teacher's

and students' answers have been collected. For grading, a similarity metric has been employed to compare the quantity of matched lemmatized case-folded keywords between the two sets of keywords.

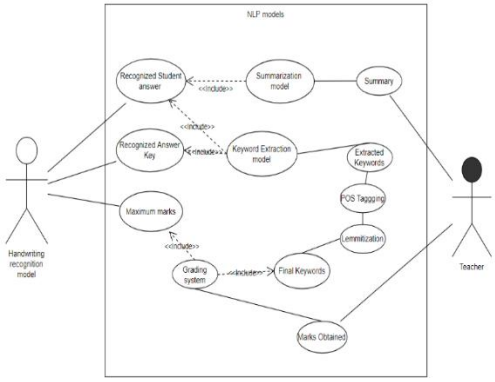


Fig 2. Use Case Diagram for NLP aspect of the system

Generative Pre-trained Transformer 3 is an autoregressive language model that produces text using a pretrained generative model. For text summarization, the GPT3 'text- davinci-002' model has been used. The input as raw text will be passed to the GPT3 model with a temperature of around 0.7 to summarize the student's answer. By increasing the temperature, the GPT-3 model comprehends the long paragraph provided by the student, summarizes based on domain knowledge, and generates a summary that may not contain the exact words from the answer, but with the variation required for summarizing a long paragraph in a shorter version. The temperature parameter thus determines how greedy the model is. If the temperature is low, the model will most likely produce the most accurate text with little variation. If the temperature is high, the model can output other words with a high probability, resulting in more diverse text. As a result, the GPT-3 model by Open AI provides a summary of the answer given by the student primarily to assist the teacher in evaluation, if the teacher is dissatisfied with the auto-evaluated marks. It also assists in recognizing the key concepts in a text filtering out unimportant information and effectively incorporating the key concepts.

The work proposes a web application Eval that teachers can use to evaluate. Next.js has been used for the frontend, al- lowing teachers to upload student answers and answer keys as jpg, jpeg, or png files and specify the maximum marks. Python FastAPI, a modern, fast (high-performance) web framework for building APIs, has been used for the backend. The image file sent to the backend will first be recognized by the OCR Cloud Vision API and stored in a text file before being fed to the BERT model and the GPT3 model for keyword extraction and summarization, respectively. The user can view the marks obtained, summary, and output from the handwritten recognition model so that the teacher could evaluate manually if the marks obtained, and summary are not satisfactory. Fig.3 and Fig.4 show the expected answer and the answer provided by the student to the web application, respectively, based on a real-world scenario in which the student provided answer is very close to the answer key but does not contain all of the required keywords. The marks allotted (for maximum marks of 10), along with a short summary of the answer provided by the student for the same can be seen in Fig.5.

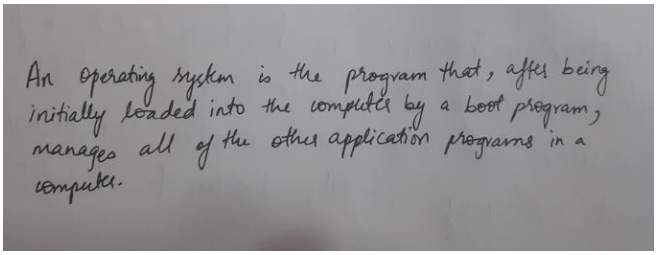


Fig 3. Input 1: Answer Key or Expected answer

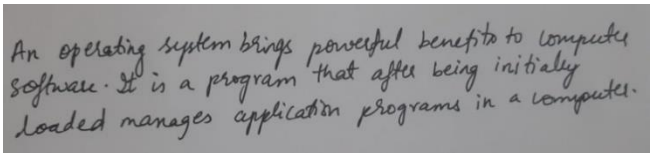


Fig 4. Input 2: Answer given by Student

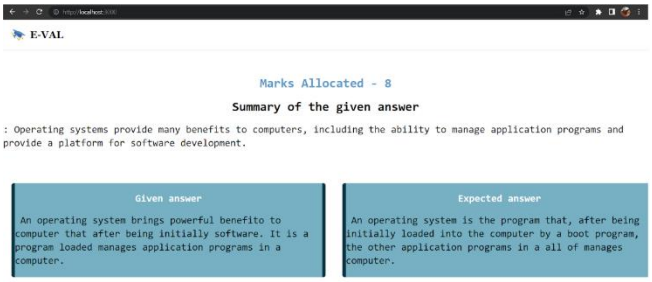


Fig 5. Web Application output given 10 as total marks

### 3. 2. Algorithm

function preprocess
{
apply padding if necessary to input image resize input image
apply whitening/normalization to input image call label encoder to send to model
return the preprocessed image
}

function handwriting recognition
{
preprocess the input image
load model (CNN + Encoder-Decoder) from checkpoint input to CNN(ResNet) returns feature maps.
feature maps input to Encoder-Decoder
decode the recognized text output from Encoder-Decoder using label encoder.
return decoded text.
}
function alternate handwriting recognition GCPVision

{
receive input image via FastAPI endpoint encode and convert image into base64 notation initialize the Vision API server
send encoded image via requestOCR
receive json response object
concatenate detected text from json object
return detected text
}

function answer summary
{
input recognized text from handwriting recognition model input to GPT3 model text-davinci-002 for summarization
return shortened summary of input
}

function text preprocess
{
tokenize each sentence in the input text remove stop words from input text case fold input text
return preprocessed text
}

function keyword extraction
{
input recognized text from handwriting recognition model text preprocess recognized text
text preprocess answer key
input preprocessed recognized text to BERT model input preprocessed answer key to BERT model return extracted keywords from recognized text return extracted keywords from answer key
}

function scoring
{
input extracted keywords from recognized text input extracted keywords from answer key lemmatize extracted keywords from recognized text
lemmatize extracted keywords from answer key input maximum marks
calculate maximum marks based on number of matching keywords
return maximum marks
}

## 4. PROPOSED SYSTEM

This paper proposes a solution to assist professors by automating the evaluation process. The solution has been implemented using a combination of Deep Learning and Natural Language Processing. The model incorporates a scanned copy of the answer booklets as well as the expected answer.

The model's output is the suggested marks and the summary of the given answer. Fig.6 shows the system from a black-box perspective (high level design of the system).

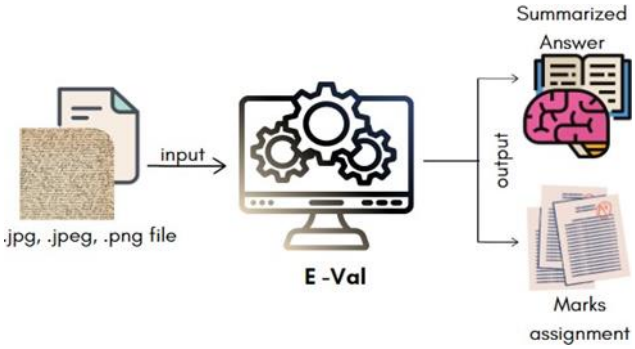


Fig. 6: System at a black-box point of view

Reliability: Attempt to reduce any downtime caused by hardware or software failures

Performance: Ideally, the model should be optimized to run through an entire answer script in 300-350 seconds.

Maintainability: Clear code structure to facilitate change integration and feature additions.

Safety requirements are not as important because most failures (hardware or software) are assumed to have no significant impact on the safety or well-being of any person.

### 4.1. Stage 1: Image to Text Conversion

The scanned answer sheets are fed into the system as image files. These image files are parsed using the Cloud Vision API, which extracts text from them. The extracted text will then be saved locally as text files. Fig.7 shows the high-level diagram of stage 1.

### 4.2. Stage 2: Processing the answer

The extracted text saved in text files is fed into the NLP model. The text will be processed here, and then keyword extraction and summarization (only for student answers) will be performed. These extracted keywords will be used to evaluate the answers. The extracted keywords and a summary of the answer will be the NLP model's overall output. Fig.8 shows the high-level design for stage 2.

### 4.3. Stage 3: Marks allocation

The extracted keywords from Stage 2 will now be used in Stage 3 (Marks allocation) to assign marks to the student's answer. The keywords from the expected answer and the given answer will be sent to a similarity checking algorithm as a pair. The algorithm's similarity index will be used to calculate the plausible marks for the answer. Fig.9 shows the high-level design for stage 3.

The system's overall output will now be the answer extracted from Stage 1, the summary from Stage 2 and the final allocated marks from Stage 3.

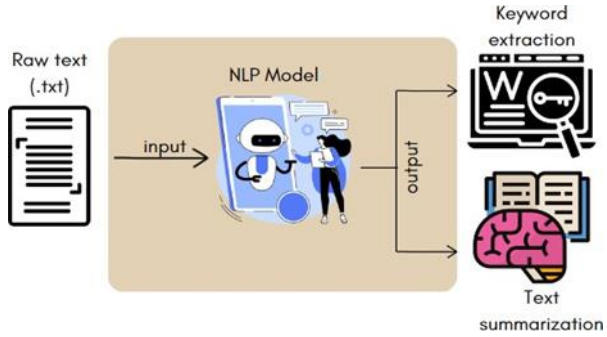


Fig. 7: Stage1: Image to Text Conversion - applies to both, teacher's, and students' answers

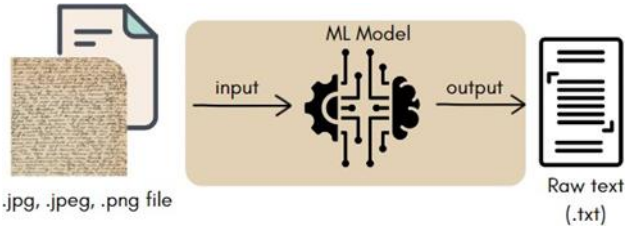


Fig. 8: Stage 2: Processing the answer - appliesto both, teacher's and students' answers

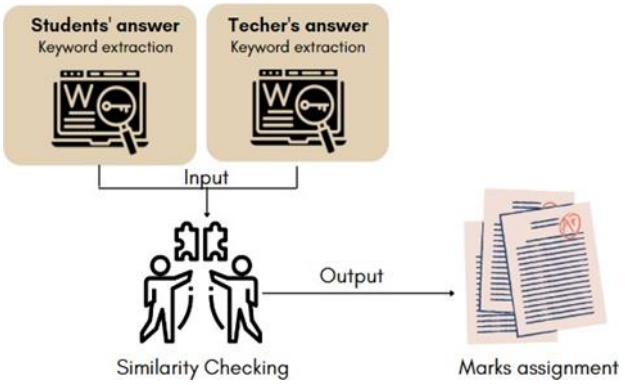


Fig. 9: Stage 3: Marks allocation - applies only to students' answers

## 5. IMPLEMENTATION AND RESULTS

### 5.1. Data Collection

The following data sources contribute to the dataset for testing the application:

1) The well-known IAM dataset containing handwritten English text.

2) Crowd-sourced handwritten samples from the University peers and students by sending out a form for handwritten samples containing the required bi-grams and trigrams.

3) Pesuacademy ESA manuscript samples.

As long as the handwritten scripts are not too sloppy, Google Cloud Vision OCR provides adequate accuracy. The model's accuracy on a legible (best-case) document has been observed to be approximately 98.6%, and on a sloppy (worst-case)

document has been observed to be approximately 66.66%. Fig.10 and Fig.11 show the handwriting recognition model's input and output with legible and sloppy (illegible) handwritings, respectively. Many processes that previously required a person to read images or documents repeatedly can now be automated.

The language processing model allocated the given answer full marks because its keywords matched those of the expected answer, as shown in Fig.12.

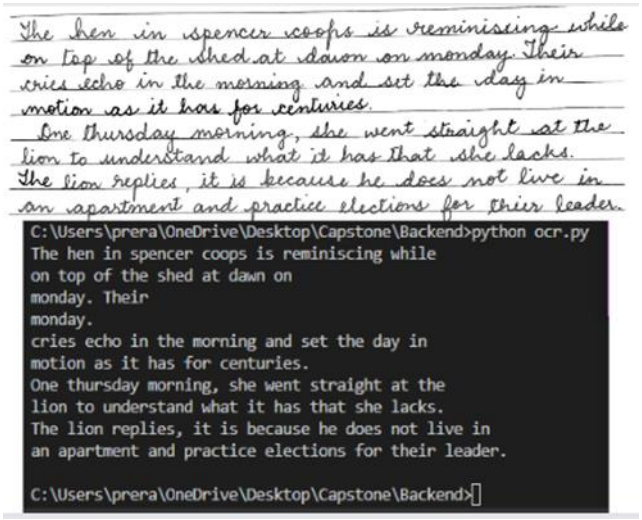


Fig. 10: Input and Output for legible handwriting

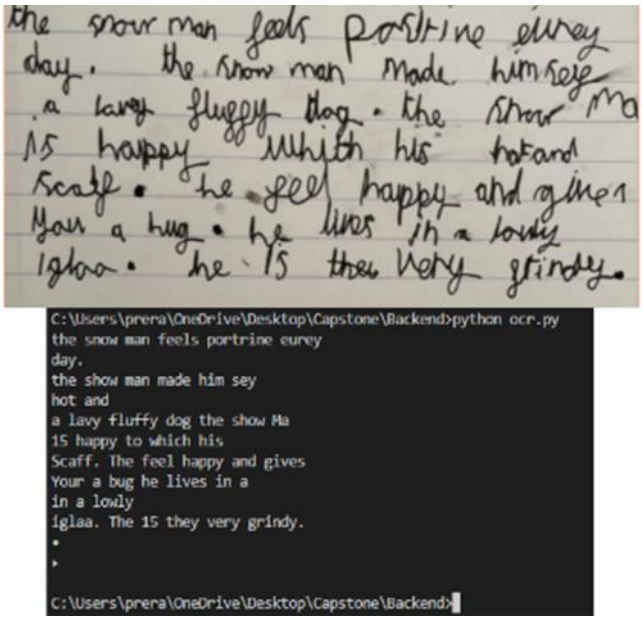


Fig. 11: Input and Output for messy handwriting

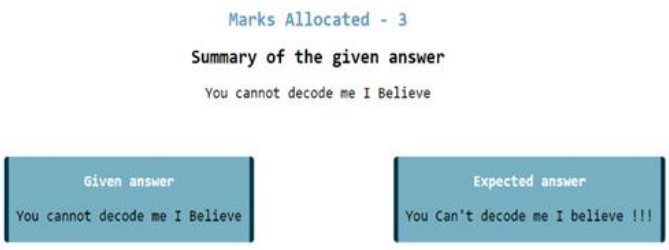


Fig. 12: Output given for maximum marks 3

## 6. CONCLUSION

According to [9], Convolutional Neural Network (CNN) has a good accuracy for recognizing handwriting, but the main disadvantage of this approach is that training the model takes a long time due to the large number of image samples used. SVM and KNN (K-Nearest Neighbors) classifiers can also be used for character recognition but again, have their respective limitations [10] Handwriting recognition is difficult due to the wide range of handwritings encountered by the model. This necessitates a diverse set of datasets for training the model. This paper proposes the use of a GCP Optical Character Recognition text extract model that is trained on vast amounts of publicly available data, making the model more accurate and avoiding the time-consuming task of training the model on large datasets.

The proposed language processing algorithm outperformed all the referenced algorithms in terms of marking accuracy. This algorithm is unique in that the keywords are lemmatized, case-folded, stop-words eliminated, and duplicate checked be- fore direct comparison.

This paper proposes using GPT-3 text-davinci-002 to summarize the students' answers. The GPT-3 models are convincing, demonstrating how powerful cloud AI is becoming. GPT-3 is more powerful on the decoder side, which takes context and generates new text, making it ideal for summarizing.

### 6.1. Future Scope

1. Expanding the range of features the website offers. For example, accepting PDF inputs.
2. Optimize the code to improve performance and render time.
3. Taking sentence grammar into account when allocating marks by including contextual relationships between words, using a LSTM as in [11].

### 6.2. Appendix

1. CNN: Convolutional Neural Networks, is a type of artificial neural network used in image recognition and processing that is specifically designed to process pixel data.

2. RNN: Recurrent Neural Networks, is a class of artificial neural networks where connections between nodes form a directed or undirected graph along a temporal sequence.

3. ResNet: A 34-layer plain Neural Network in the architecture that is inspired by VGG-19 in which the shortcut connection or the skip connection is added to reduce underfitting over time.

4. BERT: Bidirectional Encoder Representation Transformer, is a transformer-based machine learning technique for Natural Language Processing pre-training developed by Google.

5. LSTM: Long short-term memory is an artificial recurrent neural network architecture used in the field of deep learning.

6. API: Acronym for Application Programming Interface, which is a software intermediary that allows two applications to talk to each other.

7. IAM: 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.

8. GPT3: Generative Pre-trained Transformer 3 is an autoregressive language model that uses deep learning to produce human-like text.

9. Encoder-Decoder: RNN Encoder-Decoder, consists of two recurrent neural networks (RNN) that act as an encoder and a decoder pair. The encoder maps a variable-length source sequence to a fixed-length vector, and the decoder maps the vector representation back to a variable-length target sequence.


### Acknowledgements

This work was supported by Dr. Geetha Dayalan, Associate Professor, Department of Computer Science and Engineering, PES University. We would like to express our gratitude for her continuous guidance, assistance, and encouragement throughout the development of this paper. We are grateful to the Capstone Coordinators, Dr.Saraswathi V, Associate Professor, and Dr. Sudeepa Roy Dey, Assistant Professor, for organizing, managing and helping with the entire process. We take this opportunity to thank Dr. Sandesh B J, Chairperson, Professor, Department of Computer Science and Engineering, PES University, for all the knowledge and support we have received from the department. We would like to thank Dr. B.K. Keshavan, Dean of Faculty, PES University for his help. We are deeply grateful to Dr. M. R. Doreswamy, Chancellor, PES University, Prof. Jawahar Doreswamy, Pro-Chancellor – PES University, Dr. Suryaprasad J, Vice-Chancellor, PES University for providing us with various opportunities and enlightenment every step of the way. Finally, this paper would not have been completed without the continual support and encouragement we have received from our family and friends.


### Author contributions

**Prerana M S:** Literature Survey, Keyword Extraction using BERT, Text Summarization using BERT, Comparative Analysis between BERT, and GTP-3; Handwriting Recognition using GCP Cloud-vison API; Middleware between Frontend and Backend, Website Frontend and Integration with Backend, Website Backend for Batch Processing, Download report(xlsx) functionality for Batch Processing.

**Sagarika M Chavan:** Literature Survey, Keyword Extraction using GTP-3, Text Summarization using GPT-

3, Comparative Analysis between BERT, and GTP-3; Website Backend, Frontend with ReactJS for uploading files, Download report(pdf) functionality for Batch Processing.

**Ramit Bathula:** Literature Survey, Website Frontend for Batch Processing**.**

**Sreenath Saikumar:** Literature Survey, Download report(pdf) functionality for Batch Processing.

**Geetha Dayalayan:** Literature Survey, Constant Guidance.

### Conflicts of interest

The authors declare no conflicts of interest.

### References


[1] P. Bojja, N. Velpuri, G. K. Pandala, S. Polavarapu, and P. Ku- mari, "Handwritten text recognition using machine learning techniques in application of nlp," International Journal of In- novative Technology and Exploring Engineering (IJITEE) ISSN, pp. 2278–3075, 2019.

[2] J. Sueiras, V. Ruiz, A. Sanchez, and J. F. Velez, "Offline continuous handwriting recognition using sequence to sequence neural networks," Neurocomputing, vol. 289, pp. 119–128, 2018.

[3] R. Boorugu and G. Ramesh, "A survey on nlp based text sum- marization for summarizing product reviews," in 2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA). IEEE, 2020, pp. 352–356.

[4] H. Shukla and M. Kakkar, "Keyword extraction from educational video transcripts using nlp techniques," in 2016 6th International Conference-Cloud System and Big Data Engineering (Conflu- ence). IEEE, 2016, pp. 105–108.

[5] G. Sanuvala and S. S. Fatima, "A study of automated evalu- ation of student's examination paper using machine learning techniques," in 2021 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS). IEEE, 2021, pp. 1049–1054.

[6] S. S. Singh and S. Karayev, "Full page handwriting recognition via image to sequence extraction," in International Conference on Document Analysis and Recognition. Springer, 2021, pp. 55–69.

[7] B. Balci, D. Saadati, and D. Shiferaw, "Handwritten text recog- nition using deep learning," CS231n: Convolutional Neural Networks for Visual Recognition, Stanford University, Course Project Report, Spring, pp. 752–759, 2017.

[8] T. Bluche, H. Ney, and C. Kermorvant, "A comparison of sequence-trained deep neural networks and recurrent neural net- works optical modeling for handwriting recognition," in Interna- tional conference on statistical language and speech processing. Springer, 2014, pp. 199–210.

[9] S. Preetha, I. Afrid, S. Nishchay et al., "Machine learning for handwriting recognition," International Journal of Computer (IJC), vol. 38, no. 1, pp. 93–101, 2020.

[10] A. Garg, M. K. Jindal, and A. Singh, "Offline handwritten gurmukhi character recognition: k-nn vs. svm classifier," Inter- national Journal of Information Technology, vol. 13, no. 6, pp. 2389–2396, 2021.

[11] S. Meshram and M. Anand Kumar, "Long short-term memory network for learning sentences similarity using deep contextual embeddings," International Journal of Information Technology, vol. 13, no. 4, pp. 1633–1641, 2021.