Using Transfer Learning to contextually Optimize Optical Character Recognition (OCR) output and perform new Feature Extraction on a digitized cultural and historical dataset

1st Aravind Inbasekaran
Chalmers University of Technology
Gothenburg, Sweden
arainb@student.chalmers.se

2nd Rajesh Kumar Gnanasekaran
School of Information Studies
The University of Maryland
College Park, USA
rgnanase@umd.edu

3rd Richard Marciano
School of Information Studies
The University of Maryland
College Park, USA
marciano@umd.edu

***Abstract*—Understanding handwritten and printed text is easier for humans but computers do not have the same level of accuracy. While there are many Optical Character Recognition (OCR) tools like PyTesseract[1], Abbyy FineReader[2] which extract the text as digital characters from handwritten or printed text images, none of them are without unrecognizable characters or misspelled words. Spelling correction is one of the well-known tasks in Natural Language Processing. Spelling correction of an individual word could be performed through existing tools, however, correcting a word based on the context of the sentence is a challenging task that requires a human-level understanding of the language. In this paper, we introduce a novel experiment of applying Natural Language Processing using a machine learning concept called Transfer Learning[3] on the text extracted by OCR tools, thereby optimizing the output text by reducing misspelled words. This experiment is conducted on the OCR output of a sample of newspaper images published between the late 18th century to 19th century. These images were obtained from the Maryland State Archives[4] digital archives project named, the Legacy of Slavery[5]. This Natural Language Processing approach uses pre-trained language transformer models like BERT[6] and RoBERTa[7] which are used as word-prediction software for spelling correction based on the context of the words in the OCR output. We compare the performance of BERT and RoBERTa on two OCR tool outputs, namely PyTesseract and Abbyy FineReader. A comparative evaluation shows that both the models work fairly well on correcting misspelled words considering the irregularities in the text data from the OCR output. Additionally, with the Transfer Learning output text, a special process is conducted to create a new feature that originally did not exist in the original dataset dataset using Spacy's Entity Recognizer (ER)[8]. This new extracted values are added to the dataset as a new feature. Also, an existing feature's values are compared to Spacy's ER output and the original hand transcribed data.**

***Index Terms*—BERT, RoBERTa, transfer learning, natural language processing, spelling correction, entity recognition.**

I. Introduction

In the digital archives world, Optical Character Recognition (OCR) plays a vital role in extracting text from archived data like images, scanned documents, etc. PyTesseract is a Python programming language[9] supported library which uses Google's Tesseract OCR Engine[10] [1] to extract text from scanned images passed as input through Python programming language. Abbyy FineReader is a commercial software used by businesses and organizations to extract text for the same purposes. Text extraction outputs from these OCR tools on historical image data such as newspaper images used in this project or handwritten documents often contain unrecognizable characters or misspelled words. To optimize and decrease errors in the natural language character output, there is a need to correct the misspelled words with an automated mechanism that involves understanding the context of words being used in the text output.

Automatic spelling correction is one of the widely used Natural Language Processing (NLP) applications. It's been used in search engines and various applications where a correct word is suggested to the user whenever a mistake is made. Correction of spelling on a sentence requires an understanding of the language and the context. Humans can easily understand the context of the sentence and find the misspelled word but the same cannot be said for computers. Understanding the context of the sentence and correcting the incorrect word has always been one of the challenging tasks in NLP. Recently many language models

[1]https://pypi.org/project/PyTesseract/
[2]https://pdf.Abbyy.com/learning-center/what-is-ocr/
[3]https://www.tensorflow.org/tutorials/images/transfer_learning
[4]https://msa.maryland.gov/
[5]http://slavery.msa.maryland.gov/
[6]https://huggingface.co/transformers/model_doc/bert.html
[7]https://huggingface.co/transformers/model_doc/roberta.html
[8]https://spacy.io/api/entityrecognizer
[9]https://docs.python.org/3/
[10]https://github.com/tesseract-ocr/tesseract

978-1-6654-3902-2/21/$31.00 ©2021 IEEE

have been developed that are being used for spelling correction. But still, they don't perform at a human level.

Some existing algorithms use distance-based approach by calculating the distance between the misspelled word and the correct word in the dictionary. The minimum distance is the total number of editing operations required to transform the word to the correct word. The word which has the minimum edit distance between the misspelled and correct word is taken. But this method does not consider the context of the sentence.

Misspelled words can be classified into two categories. One could be out of dictionary words, in which the words aren't actual English words. These kinds of misspelled words are easier to correct. The latter would be an actual English word that is out of context. The figure [1] shows both out of vocabulary and out of context words in a sentence. In the first sentence, the misspelled word can be corrected using methods like edit distance and the word with the smallest edit distance will be used as a replacement. In this case, "piece" would be the ideal word. But in the second sentence, we have a word that exists in the vocabulary but is out of context. For correcting these types of errors, understanding the language is a must.

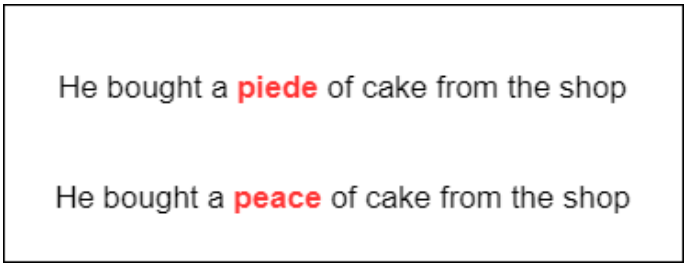


Fig. 1: Example of out of vocabulary and out of context words in a sentence

This paper focuses on correcting misspelled words and optimizing output of OCR output from tools PyTesseract and Abbyy FineReader run on a sample of newspaper images (n=100) published between the late 18th century and 19th century in the United States of America's State of Maryland, which were physically collected and digitally preserved by the Maryland State Archives (MSA). In the present world, these newspaper ads were identified, collected, and digitally transcribed into a dataset collection called as "Runaway Slave Ads" by the MSA through a digital archives project called "The Legacy of Slavery" (LoS). The sample images were passed through Abbyy FineReader tool and PyTesseract to obtain the text extracted output. Figure [3] and [4] shows the digitally extracted text output from PyTeseract and Abbyy respectively. As could be seen, some of the characters and words are unrecognizable or misspelled. To optimize this, in this paper, we perform a novel experiment on passing OCR output to a machine-learning-based natural language processing method using pre-trained language models, BERT and RoBERTa. In this approach, the misspelled words are masked and then the models use their contextual understanding to predict the masked word.

BERT is a language model which is being used for various NLP tasks. BERT's architecture is based on multi-layered bidirectional transformers. BERT is trained on unlabelled text of entire wikipedia and book corpus of 800 million words. BERT [2] is widely used for masked word prediction as it is trained for Masked Language Modeling. Since BERT is undertrained an improved model has been proposed which is called RoBERTa [3]. Both transformer architecture uses attention mechanism to predict the masked word in the sentence. Attention mechanism helps the transformers to have long-term memory.

The digital transcription by volunteers at the MSA captured, among other features, slave owner names, enslaved names, the existence of a reward amount in the ad, etc. These features have been transcribed manually by a human reading these newspaper ads. This paper attempts to use the enhanced text extraction output from pre-trained language models to create new features using ER method that were not originally captured by the volunteers at the MSA. For instance, the MSA volunteers create a feature named "Reward amount Y/N" to indicate if the newspaper ad mentioned a reward amount or not? However, there is no existence of the actual amount if the transcriber captured the value as "Y' to the above existence of the reward amount question. Using ER method on output from Transfer Learning, this paper creates a new feature i.e, the reward amount in dollars. Also, this paper, using the same ER method, recreates the same feature "Reward Amount Y/N", however, using an automated mechanism and a comparative analysis is done between the manually entered data and the automated feature creation data. The following sections explain the Research Questions, Related Work, Dataset used, Methods followed, Experiment, Results, Conclusion and Future Work in that order.

II. Research questions

The main goal of this novel experiment is to find answers to the following research questions.

1) RQ1 - Could Transfer Learning be employed on OCR output of printed newspaper images from a cultural and historical dataset to enhance and optimize the natural language text extraction by auto-correcting misspelled words and unrecognizable characters using pre-trained language models, BERT and RoBERTa?
2) RQ2 - Could the optimized output from the pre-trained language models BERT and RoBERTa be used to perform Entity Recognition to create a new feature, "Reward Amount value in dollars", which is non-existent in the original dataset's manually transcribed list of features? And if a comparative analysis be done between the Runaway Slave ad's existing "Reward Amount exists in the Ad Y/N" feature from manual

transcription to the feature created by Entity Recognition?

III. Related work

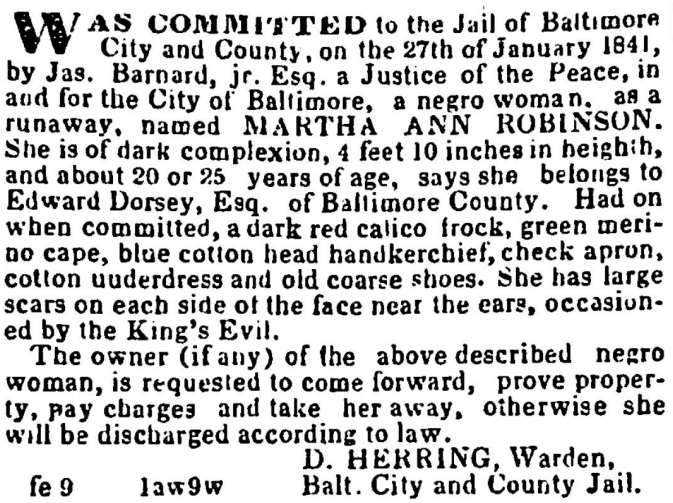
There are various works that have been done previously on spelling correction. Most of them includes probabilistic methods, n-gram based methods, edit-distance, noisy channel and neural networks. Some of these methods, doesn't take context of the sentence into consideration.

Most of the traditional models use the noisy channel model for word correction. Kashyap and Oomen proposed a spelling correction model using probabilistic methods [4] uses the method of generation of a garbled word which can delete or insert characters in the word from which we can derive an estimate of the correct word. Church and Gale proposed a model which takes in rejected words by UNIX spell program and gives a list of corrected words and sorts them by probability score. This is based on the noisy channel model. Andrew and Yves proposed a method [5] called Tribayes which combines trigram-based and feature-based methods for correcting spelling based on context. Trigram is ineffective with words that have the same POS and Bayes performs better than trigram but it is not effective against words that depend on syntactic constraints. Hence they developed a hybrid of Bayes and trigram, which performed better in identifying the correct word based on the context. Recently, a spell checker system [6] has been proposed by Prabhakar Gupta that detects spelling errors and suggests corrections based on the context in real-time. This system can be adapted to any language and it outperforms spell checkers like Hunspell and Aspell. They used n-gram conditional probabilities dictionary to understand the context and suggest words.

In addition to these, a neural network-based model has also been used for spelling correction. A paper by Ghosh and Kristensson [7] proposes sequence to sequence neural attention system for text correction. The model achieved 90% accuracy in the twitter typo dataset. A nested RNN model [8] has been proposed by Li et al. that has been trained on pseudo data based on phonetic similarity. This model doesn't depend on the noisy channel model. The experimental results of this model show that it performs better than Enchant, scRNN, and LSTM-Char RNN models. The model has an accuracy of 69.39% which is around 4% greater than scRNN and LSTM-Char RNN's accuracy. Paper by Hu et al. [9] proposes the use of pre-trained contextual model BERT for correcting out of vocabulary and out-of-context words. They used BERT with edit distance to find the error words in the sentence. They used BERT before edit distance and BERT after edit distance. The model achieved an accuracy of 73.25% after using edit distance with edit BERT. Named Entity Recognition(NER) is one of the most essential tools for NLP tasks. Over the years, NER has changed a lot and many new models for entity recognition have been updated. Paper by Li, Sun, Han, and Li [10] explains the survey on how deep learning is used for NER tasks. Paper by Yan, Deng, Li, and Qiu [11] proposes a NER architecture using transformer encoder. They incorporated transformers with positional encoding and unscaled attention which proved to have better performance than BiLSTM model.

IV. Dataset

The images used in this paper for text extraction are from a dataset named "Runaway Slave Ads" which is part of a set of dataset collection created by the MSA. There are 18 datasets in this collection comprising of culturally and historically rich information digitally transcribed into databases by the MSA. In [12] and [13], two of these datasets were used for creating a novel cloud-based learning platform, to perform data science-based analysis that unravels critical insights and to identify networks combining metadata from the two datasets: Certificates of Freedom[11] and Manumissions[12]. In this paper, the images of "runaway slave ads" are used. These newspaper images represent the advertisements placed by the then Slave Owners in the State of Maryland regarding missing or escapement of the enslaved people under their watch. The advertisements detail the enslaved person's appearance, demographic details in the English Language. Some of these ads also specify the reward amount to be awarded to those who could provide information regarding these missing 'runaway' slaves. There are around 12108 such ads documented out of which, about 100 of them were chosen at random for this experiment. These are images of handwritten newspaper ads from Maryland State Archives.


WAS COMMITTED to the Jail of Baltimore City and County, on the 27th of January 1841, by Jas. Barnard, jr. Esq. a Justice of the Peace, in and for the City of Baltimore, a negro woman, as a runaway, named MARTHA ANN ROBINSON. She is of dark complexion, 4 feet 10 inches in heighth, and about 20 or 25 years of age, says she belongs to Edward Dorsey, Esq. of Baltimore County. Had on when committed, a dark red calico frock, green merino cape, blue cotton head handkerchief, check apron, cotton uuderdress and old coarse shoes. She has large scars on each side of the face near the ears, occasioned by the King's Evil.

The owner (if any) of the above described negro woman, is requested to come forward, prove property, pay charges and take her away, otherwise she will be discharged according to law.

D. HERRING, Warden,
fe 9 law9w Balt. City and County Jail.

Fig. 2: An image of one of the newspaper ads from the dataset

The figure [2] is one of the newspaper ads from the dataset. All the data that is being used in the paper are similar to this. The text is extracted from the image using

[11]http://guide.msa.maryland.gov/pages/viewer.aspx?page=afridesc
[12]http://guide.msa.maryland.gov/pages/viewer.aspx?page=afridesc

PyTesseract [4] and Abbyy fine reader. The average length of each extracted text is around 400-500 words. Both the extracted text are used for the transformer models. From the extracted text, we used Stanford's named entity recognizer [14] to find the proper nouns in the text. We then used Brown corpus from the Natural Language Tool Kit [15] to find the misspelled words and out-of-context words. A single sentence may have more than one misspelled word. On average a single text file would have around 15-20 misspelled words or out-of-context words.

For the ad in the figure [2], the output from the PyTesseract [fig:3] and Abbyy fine reader [fig:4] are given below. As it could be seen that the output has lot of misspelled words from the original ad.

```
TAS COMMIS TED to the Jail of Baltimore City and County on the
27th of January 1841  by Jas. Barnard jr.
Esq. a Justice of the Peace in aod for the City of Baltimore a negro woman a8 a
runaway named MARTHA ANN ROBINSON. She is of dark complexion 4 feet 10 inches
in heighih and about 20 or 25 years of age says she betongs to Edward Dorsey Es
q.
of Ballimore County. Had on when committed a dark red calico frock green meri
no cape blue cotton head handkerchief check apron cotton uuderdress and old
coarse shoes. She has large scars On each side ot the face near the ears
occaslun ed by the King's Evil.  The
owner (ifany) of the above described negro woman is requested to come forward
prove proper ty pay charges and take her away otherwise she will be discharged
according to law.  D. HERRING Warden fe 9
law9w Balt. City and County Jail.
```

Fig. 3: PyTesseract output of the ad

```
TAS COMMIS TED to the Jail of Baltimore City and County on the
27th of January 1841  by Jas. Barnard jr.
Esq. a Justice of the Peace in aod for the City of Baltimore a negro woman a8 a
runaway named MARTHA ANN ROBINSON. She is of dark complexion 4 feet 10 inches
in heighih and about 20 or 25 years of age says she betongs to Edward Dorsey Es
q.
of Ballimore County. Had on when committed a dark red calico frock green meri
no cape blue cotton head handkerchief check apron cotton uuderdress and old
coarse shoes. She has large scars On each side ot the face near the ears
occaslun ed by the King's Evil.  The
owner (ifany) of the above described negro woman is requested to come forward
prove proper ty pay charges and take her away otherwise she will be discharged
according to law.  D. HERRING Warden fe 9
law9w Balt. City and County Jail.
```

Fig. 4: Abbyy fine reader output of the ad

V. METHOD

To address RQ1, we used two pre-trained language models in this paper, BERT and RoBERTa models. Both the models have been trained to predict the misspelled words and out-of-context words in a sentence.

BERT has been trained to predict masked words in the sentence. With masked language, we give an input sentence but a few words are masked and then the model is asked to fill in the masked word. BERT works by giving in a sentence with a masked token and it predicts the word based on the context. Since the BERT is bidirectionally trained, it has a deeper sense of language and can understand the context of the sentence. The figure 5 shows an illustration of how the masked language model works.

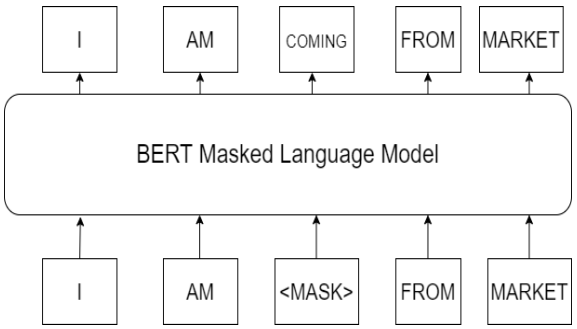


Fig. 5: Schematic illustration of how masked language model works.

The text data is extracted from using PyTesseract and Abbyy fine reader. In this text data, the locations and names are ignored and the rest of the words are checked for misspells and out-of-context words. Such words are then masked. Our data has more than one mask in a sentence and an entire paragraph is sent to the model. The word that has to be predicted is replaced with the tag "MASK". Then the entire text is tokenized and sent to BERT model for prediction. The BERT predicts the word from the vocabulary based on the context of the sentence. In each of the text, [CLS] and [SEP] token is added to separate the text. We use a pre-trained BERT cased model. In addition to BERT, we use RoBERTa on the same dataset and evaluate the performance of the model as well. RoBERTa base uncased model is used for word correction. We then compared the performance of BERT with RoBERTa and the outputs from both models.

We compare the predicted words from both models with misspelled words in the data and compute the edit distance for each of the words. Edit distance is used to find the syntactic similarity between two words. The words with the smallest edit distance are chosen [16]. In addition to this, we counted the total number of misspelled words in the text before masking it and we counted the total number of misspelled words in the text after predicting the words with both the models and compared the difference.

To address RQ2, we use Spacy's ER to extract reward amount and status from the output text from the models and evaluate them against the original dataset. Entity Recognition works by finding named entities in the text. It includes information such as person names, location, monetary values, etc. The figure [6] shows an illustration of how the entity recognition model works. It takes in a sequence of tokens as input and returns the entity in the sentence.

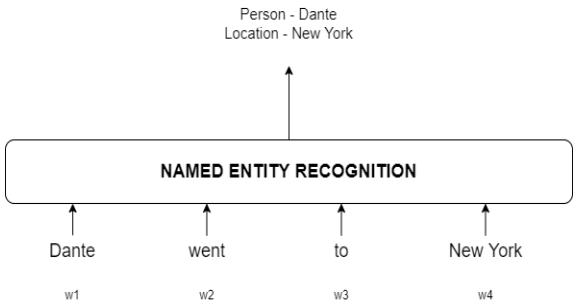


Fig. 6: Illustration of how NER works.

VI. Experiment

For this paper, we used a pre-trained model of BERT and RoBERTa without performing any fine-tuning. The dataset is first obtained in form of images. A sample of 100 images of newspaper ads is taken. We perform text extraction from the image using PyTesseract and Abbyy fine reader. The extracted text from both of them contains a lot of misspelled words as the images are handwritten text from the 1800s. Before sending it to the transformer model, some of the unwanted characters have been removed. Then the Stanford NER tagger is used to find the person names in the text as these might be classified as misspelled words. These names are added to a list. Then Spacy's ER is used to find locations and these are added to the list as well. We used both brown corpus and spellchecker to find the misspelled words in the text and mask them.

A. Model 1: BERT

The first model to be used for prediction is the BERT-base-cased model. A single piece of text contains many sentences. The entire text is sent to the model without splitting them into individual sentences. BERT can take up to tokens of length 512 and for data which has a token length of greater than 512 and split into chunks and sent to the transformer model. The model returns the N-words for each of the masked positions. We compare each of these words with the misspelled words using edit distance. The word with the smallest edit distance from the generated words is then selected as the final word to be filled in. Then the final text which has the predicted word is again checked for any incorrect words in the text.

Once the final output is obtained, this is used to extract the reward amount from the text. This is compared with the original data which has a list of whether the ad has a reward or not. For addressing RQ2, Spacy's ER is used to extract reward amounts from the optimized output text data.

B. Model 2: RoBERTa

Another model we used for word prediction is the RoBERTa-base model. The same steps that have been followed for BERT are followed for RoBERTa as well. RoBERTa can take up to tokens of length 512 and data that has a size greater than 512 has to be divided into chunks. RoBERTa model returns N number of words for each of the masked positions and edit distance is used to find the best candidate word for each of the positions. For each of the models, text extracted from PyTesseract and Abbyy fine reader has been used. The final output from the RoBERTa model is used to extract reward amounts. This is then compared with the output from BERT and our original dataset.

VII. Results

The result section includes the output from BERT and RoBERTa for text data extracted using PyTesseract and Abbyy fine reader. We compare the performance both the models on both the datasets. We calculate the accuracy of the model on correcting total number of misspelled words.

	BERT-PyTesseract	BERT-Abbyy
Accuracy	76%	62%

TABLE I: Accuracy of BERT on both the datasets

	RoBERTa-PyTesseract	RoBERTa-Abbyy
Accuracy	87%	78%

TABLE II: Accuracy of RoBERTa on both the datasets

The table [I] and [II] show the accuracy of the model on both datasets. BERT has an accuracy of 76% on correcting the errors in the text extracted using PyTesseract and 62% accuracy on correcting the errors in the text extracted using Abbyy fine reader. As for RoBERTa, it has better accuracy than the BERT model. RoBERTa has an accuracy of 87% on PyTesseract data and 78% accuracy on Abbyy fine reader data. For the image [2], the BERT and RoBERTa output is obtained as given in the figure [7] and [8]. The model was able to correct some of the misspelled words in the output of PyTesseract.

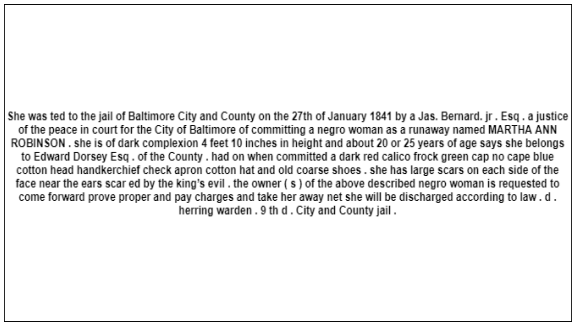


Fig. 7: Output of BERT model.

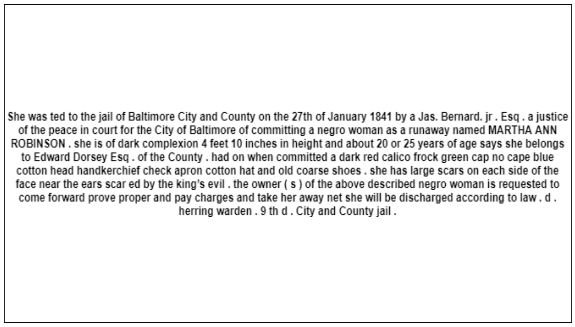


Fig. 8: Ouput of RoBERTa model.

We use the outputs from BERT and RoBERTa with PyTesseract data for extracting reward features from them. As the original dataset doesn't contain the reward amounts, this information will be useful as it adds new features to the dataset. The reward details in the original dataset have been entered manually. This information is compared with the output from our model.

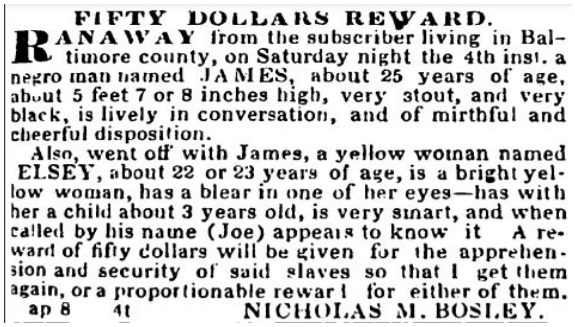
	Original	BERT-PyTesseract	BERT-Abbyy
Yes	71	46	40
No	29	54	60

TABLE III: Comparison of BERT reward status with original dataset

	Original	RoBERTa-PyTesseract	RoBERTa-Abbyy
Yes	71	45	43
No	29	55	57

TABLE IV: Comparison of RoBERTa reward status with original dataset

The table [III] shows the output obtained from Spacy's ER for PyTesseract and Abbyy data with the BERT model. From the table, it could be seen that it has worked fairly well on extracting most of the reward feature from the output text. The table [IV] shows the output obtained with RoBERTa model. It couldn't be concluded that if Spacy's performance on text from any one model is better than the other as the overall accuracy is almost equal for both model.


FIFTY DOLLARS REWARD.

RANAWAY from the subscriber living in Baltimore county, on Saturday night the 4th inst. a negro man named JAMES, about 25 years of age, about 5 feet 7 or 8 inches high, very stout, and very black, is lively in conversation, and of mirthful and cheerful disposition.

Also, went off with James, a yellow woman named ELSEY, about 22 or 23 years of age, is a bright yellow woman, has a blear in one of her eyes—has with her a child about 3 years old, is very smart, and when called by his name (Joe) appears to know it A reward of fifty dollars will be given for the apprehension and security of said slaves so that I get them again, or a proportionable reward for either of them.

ap 8 4t NICHOLAS M. BOSLEY.

Fig. 9: Example image with reward amount.

The figure [9] is one of the text data that has been used for this paper. This data has a reward amount of 50$. The Spacy's ER has worked perfectly for this case in extracting the reward amount as it is. We were able to extract rewards from most of the data

Reward Amount exists in the Ad Y/N	**Reward Amount status BERT model**	**Reward Amount value in dollars**
Yes	No	None
Yes	Yes	200$

TABLE V: Comparison of reward status from original dataset with status from BERT model and extracted feature

The table [V] shows the existing reward status and the reward status obtained from the model and the extracted new feature, reward amount in dollars. We could see that the model sometimes outputs the reward amount as none and status as 'No' but it was able to extract reward amount from more than half of the enhanced output text.

VIII. Conclusion and Future work

The main aim of this paper was to answer the research questions and we were able to achieve our goals. From our results, we could see that both the pre-trained transformer models worked fairly well in correcting the misspelled words and unrecognizable characters in the input text. From our results, RoBERTa model provided better results in correcting the misspelled words in our data. As it is trained on more data and with improved training methodology than BERT, it is expected to give better results. Even though the words that both the models replaced aren't exact same words from the ad, it is trying to correct the misspelling based on the context of the sentence. BERT and RoBERTa being two most widely used language models, we wanted to evaluate the performance of the model on this kind of historical dataset.

As for RQ2, we were able to perform entity recognition on the optimized output from the pre-trained models. We were able to extract more than half of the reward amount from the output. This could be added to the original dataset as a new feature. Even though, they are not better than the inputs entered by humans, this feature extractions from enhanced contextual output could be used to capture capture some of the key details from the text that may have been overlooked by the human transcription. We were successfully able to answer both of our research questions but further improvement could be made.

For this paper, we used a sample of 100 data. In the future, we will be using the entire dataset to evaluate the performance of the model. The pre-trained model has been used for both BERT and RoBERTa. And, if we have more historical data, we can could fine tune the BERT on the data and could achieve better results. Another interesting area that could be explored in this paper would be using BERT as NER to extract the reward amount and check how it performs against the Spacy's ER. For this, the model has to be fine-tuned with a custom dataset similar to our dataset with custom tags added to them.

IX. Acknowledgments

We wish to acknowledge two funded grants that have contributed to this work: (1) the 2020-2022 IMLS Laura Bush 21st Century Librarian Program: "Piloting an Online Collaborative Network for Integrating Computational Thinking into Library and Archival Education and Practice." In addition, a deep appreciation for the continued support and collaboration of the staff from the Maryland State Archives who have created these incredible historical resources, Christopher Haley (Director of the Study of Legacy of Slavery in Maryland) and Maya Davis (Research Archivist and Legislative Liaison).

References

[1] R. Smith, "An overview of the tesseract ocr engine," in *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, vol. 2, 2007, pp. 629–633.

[2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2019.

[3] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," 2019.

[4] R. Kashyap and B. Oommen, "Spelling correction using probabilistic methods," *Pattern Recognition Letters*, vol. 2, no. 3, pp. 147–154, 1984.

[5] A. Golding and Y. Schabes, "Combining trigram-based and feature-based methods for context-sensitive spelling correction," in *34th Annual Meeting of the Association for Computational Linguistics*. Santa Cruz, California, USA: Association for Computational Linguistics, Jun. 1996, pp. 71–78. [Online]. Available: https://aclanthology.org/P96-1010

[6] P. Gupta, "A context sensitive real-time spell checker with language adaptability," 2019.

[7] S. Ghosh and P. O. Kristensson, "Neural networks for text correction and completion in keyboard decoding," 2017.

[8] H. Li, Y. Wang, X. Liu, Z. Sheng, and S. Wei, "Spelling error correction using a nested rnn model and pseudo training data," 2018.

[9] Y. Hu, X. Jing, Y. Ko, and J. T. Rayz, "Misspelling correction with pre-trained contextual language model," 2021.

[10] J. Li, A. Sun, J. Han, and C. Li, "A survey on deep learning for named entity recognition," *ArXiv*, vol. abs/1812.09449, 2018.

[11] H. Yan, B. Deng, X. Li, and X. Qiu, "Tener: Adapting transformer encoder for named entity recognition," 2019.

[12] L. A. Perine, R. K. Gnanasekaran, P. Nicholas, A. Hill, and R. Marciano, "Computational treatments to recover erased heritage: A legacy of slavery case study (ct-los)," in *2020 IEEE International Conference on Big Data (Big Data)*, 2020, pp. 1894–1903.

[13] R. K. Gnanasekaran and R. Marciano, "Piloting Data Science Learning Platforms through the Development of Cloud-based interactive Digital Computational Notebooks," in *Proceedings of International Symposium on Grids & Clouds 2021 — PoS(ISGC2021)*, vol. 378, 2021, p. 018.

[14] J. R. Finkel, T. Grenager, and C. Manning, "Incorporating non-local information into information extraction systems by gibbs sampling." 2005, pp. 363–370.

[15] S. Bird, "Nltk: The natural language toolkit," 01 2006.

[16] F. J. Damerau, "A technique for computer detection and correction of spelling errors," *Commun. ACM*, vol. 7, no. 3, p. 171–176, Mar. 1964. [Online]. Available: https://doi.org/10.1145/363958.363994