Medhat, Fady, Mohammadi, Mahnaz, Jaf, Sardar, Willcocks, G. Chris, Breckon, Toby, McGough, Stephen, Theodoropoulos, Georgios and Obara, Boguslaw (2018) TMIXT: A process flow for Transcribing MIXed Handwritten and Machine-printed Text. In: IEEE International Conference on Big Data, US.

Downloaded from: http://sure.sunderland.ac.uk/id/eprint/10447/

Usage guidelines

Please refer to the usage guidelines at http://sure.sunderland.ac.uk/policies.html or alternatively contact sure@sunderland.ac.uk.

TMIXT: A process flow for Transcribing MIXed handwritten and machine-printed Text

Fady Medhat[1], Mahnaz Mohammadi[1], Sardar Jaf[1], Chris G. Willcocks[1], Toby P. Breckon[1], Peter Matthews[1], Andrew Stephen McGough[2], Georgios Theodoropoulos[3], and Boguslaw Obara(✉)[1]

[1]Department of Computer Science, Durham University, UK
[2]School of Computing, Newcastle University, UK
[3]Department of Computer Science and Engineering, Southern University of Science and Technology, China
[1]{fady.medhat, mahnaz.mohammadi, sardar.jaf, christopher.g.willcocks, toby.breckon, p.c.matthews, boguslaw.obara}@durham.ac.uk
[2]stephen.mcgough@newcastle.ac.uk [3]georgios@sustc.edu.cn

***Abstract*—Handling large corpuses of documents is of significant importance in many fields, no more so than in the areas of crime investigation and defence, where an organisation may be presented with a large volume of scanned documents which need to be processed in a finite time. However, this problem is exacerbated both by the volume, in terms of scanned documents and the complexity of the pages, which need to be processed. Often containing many different elements, which each need to be processed and understood. Text recognition, which is a primary task of this process, is usually dependent upon the type of text, being either handwritten or machine-printed. Accordingly, the recognition involves prior classification of the text category, before deciding on the recognition method to be applied. This poses a more challenging task if a document contains both handwritten and machine-printed text. In this work, we present a generic process flow for text recognition in scanned documents containing mixed handwritten and machine-printed text without the need to classify text in advance. We realize the proposed process flow using several open-source image processing and text recognition packages. The evaluation is performed using a specially developed variant, presented in this work, of the IAM handwriting database, where we achieve an average transcription accuracy of nearly 80% for pages containing both printed and handwritten text.**

***Keywords*— big data, unstructured data, Optical Character Recognition (OCR), Handwritten Text Recognition (HTR), machine-printed text recognition, IAM handwriting database, TMIXT**

I. Introduction

Despite the migration to fully electronic administration system across civil and non-civil sectors, most businesses and governmental agencies hold a significant quantity of historical archive material. Originally these documents would have been in the form of printed documents[1], handwritten forms, scanned pages, etc. The availability of such documents is not only confined to closed entities like businesses or agencies, but it also extends to documents proliferating through cyberspace. For example, organizations such as Wikileaks reports that it has collected over 10 million documents, since 2006, involving war, spying and corruption. The sheer volume of documents imposes a need to automate the transcription process to machine readable formats for further security analysis and defence related applications. Since such information is largely made up of text, Natural Language Processing (NLP) for big data presents an opportunity to take advantage of what is contained in these documents and also reveal patterns, connections and trends across disparate sources of data. NLP techniques incorporate a variety of methods, including linguistics, semantics, statistics and machine learning to extract entities, relationships, and context, which enables an understanding of what is being written, in a comprehensive way. Far beyond what could be achieved through analysis by individuals.

A database of scanned text documents is one of the major examples of data sources, where Optical Character Recognition (OCR) systems, being concerned with the semantics recognition of the NLP problem, are used to extract the contents of such documents in order to convert them into a machine readable format to facilitate the retrieval and re-usability of the knowledge held within. These documents may include either handwritten, machine-printed text or both. The respective recognition of handwritten and machine-printed documents are commonly handled separately using dedicated techniques due to the inconsistency in the structure of characters and the style of handwritten text when compared to machine-printed text.

Text localization followed by classification (into handwritten or machine printed) are two important stages before the text recognition phase. Several attempts [1]–[5] that considered these stages depend mostly on handcrafted features and most of these approaches are used in combination with conventional machine learning classifiers such as k-Nearest Neighbor (k-NN) and Support Vector Machines (SVM) [5]. Other attempts exploited the use of statistical models. For example, Cao *et al.* [6] presented a system for identification and classification of handwritten

[1]We do not distinguish here between different printing processes or even typed text as these are handled through the same process.

and machine-printed[2] text from document images using Hidden Markov Models (HMM). Silva *et al.* [7], proposed an automatic discrimination system for identifying handwritten words from machine-printed words. They employed several preprocessing steps for image segmentation, feature extraction and finally classification. The effort of Zagoris *et al.* [8], despite being based on a traditional flow of image preprocessing and feature extraction, differs in being dependent on a Bag of Visual Words (BoVW) model. This model depends on creating a codebook for features extracted from a training dataset that can be further used to generate a code for a new text block. The vectors generated using the code book are further classified using a combination of binary SVM classifiers to decide between handwritten, machine-printed and noise.

Classification methods being rule-based, structural or statistical [9] have been used in combination with hand-engineered features, where HMM were used to capture the feature transition of the handwritten text [10]. Other attempts have tried to combine both HMM with neural networks as in [11]. Later attempts tried to exploit the use of neural networks as feature extractors especially using Recurrent Neural Networks (RNN) and their more advanced counterpart, the Long Short-Term Memory (LSTM). Graves *et al.* [12], [13] extended the use of a single dimensional LSTM to a multidimensional one, where they achieved competitive performance compared to HMM.

Convolutional Neural Networks (CNN) [14] have been used in many text classification tasks. Feng *et al.* [15] have addressed the problem of machine-printed and handwritten text separation using a CNN model. The text-line input to their CNN model captured discriminative content for the classification task. Their experimental validation also showed that integrating their proposed cropping schemes with deep architectures and wider convolutional filters improved the performance significantly.

Most of the referenced attempts are targeted for classifying text as either handwritten or machine-printed text, and the recognition process is applied with the prior knowledge of the type of text under consideration. This imposes an identification overhead especially when the number of documents is large. The presence of both types of text in a document introduces even more challenges for OCR systems, since it requires reliant localization and segmentation methods for different regions within a page upon which a classifier will be able to distinguish between machine-printed and handwritten text for later recognition.

In contrast to most prior works in this field, we propose a generic process flow for Transcribing scanned documents containing MIXed handwritten and machine-printed Text (TMIXT) without the need for any prior knowledge of the type of text within the document, which is the key novelty of this work. TMIXT is implemented using several publicly available packages to fulfill the designated task of each stage within the process flow. The proposed process flow is evaluated using the IAM handwriting database [16], widely used for text recognition research, in combination with a set of specially tailored labels, the MIXED-IAM[3] labels we present in this work, to cope with the nature of mixed text documents.

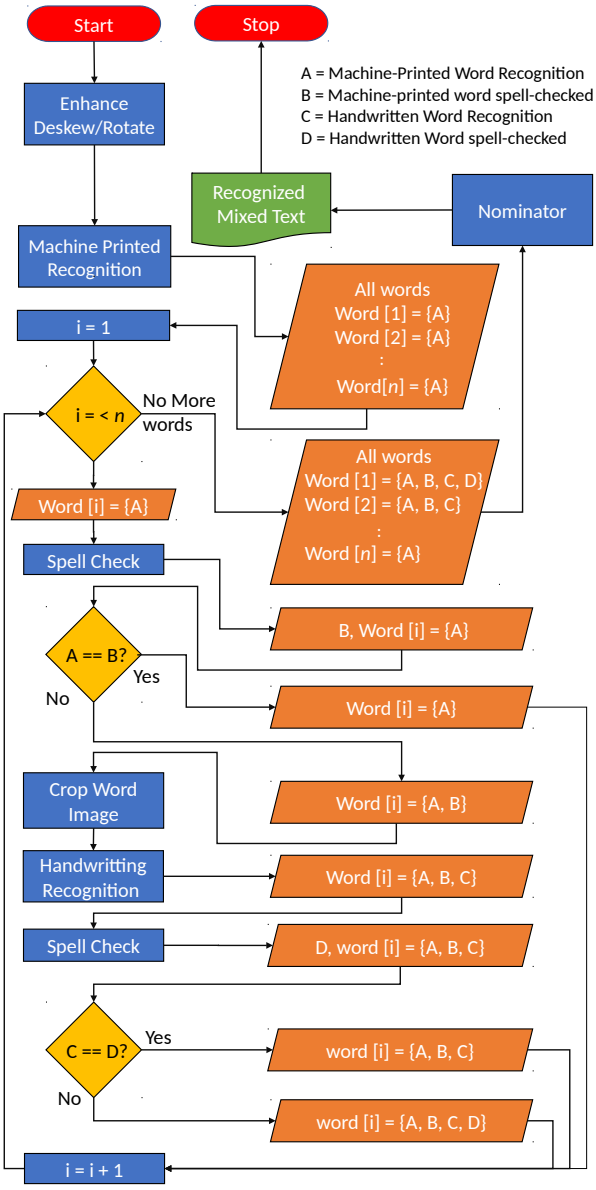

Figure 1: TMIXT process flow for scanned document recognition.

The rest of this paper is organized as follows. Our proposed process flow for scanned text recognition and its constituent modules are discussed in detail in Section II. We present an analysis of different preprocessing stages, and evaluate the performance of our proposed process on the IAM database using different evaluation metrics in Section III, and conclude the paper and discuss future works in Section IV.

[2]Sometimes referred to as typewritten, we will use machine-printed here without loss of generality.

[3]https://bitbucket.org/DBIL/mixed-iamdb

II. Methodology

The TMIXT process flow, we present in this work, allows for transcribing mixed handwritten and machine-printed text without the need to discriminate the handwritten and machine-printed text prior to recognition. This process flow is realized through a suite of open-source software components that can be interchanged with other libraries and packages of similar functionality within the overall architecture of our methodology.

We will discuss the general flow of our process flow before presenting a more detailed discussion of each subpart in the relevant subsection below. Following Fig. 1, a single scanned image of a text document undergoes several phases in the TMIXT process flow. The first phase involves simple image processing and enhancement, that could have direct influence on the recognition performance. The enhanced page is forwarded to the machine-printed text recognition. This stage is applied on the whole page irrespective of the actual text type(s), being handwritten or machine printed, present in the page. Spell-checking is applied on each word generated, where failure to pass the spell-checking validation induces the next phase of the process flow to be performed – the handwriting recognition.

The handwriting recognition is applied at word-level compared to the page-level used in the machine-printed recognition. This is fulfilled by cropping the exact word that fails the spell-checking, and applying the handwriting recognition on this specific word in isolation from the rest of the page. Accordingly, each word in the document could have up to four possible options generated from the machine-printed and the handwriting recognition phases together with their spell-checking. These options are subject to an elimination process to nominate the optimum candidate word based on the context, which is the role of nomination phase – the final phase. Since this work is focusing on presenting the TMIXT flow without constraining the libraries used in implementation, we will avoid details related to training the specific recognition models or the internals of the image enhancement algorithms we adopted for this work. The following subsections discuss each stage of the process flow in more detail.

A. Preprocessing

The preprocessing stage aims to eliminate variations such as deformations and noise in scanned images, which affect the recognition accuracy.

1) Image Enhancement: Enhancing an image is the first stage of the proposed flow as shown in Fig. 1. This stage involves noise and artefacts removal in addition to increasing the contrast between the text and the background. A survey by Jung *et al.* [17] investigated a wide range of proposed approaches for image enhancement to eliminate distortions in images and documents. Filter-based methods are widely used, which are mostly dependent on classical image processing techniques. Fig. 2 shows a sample page before and after enhancement.

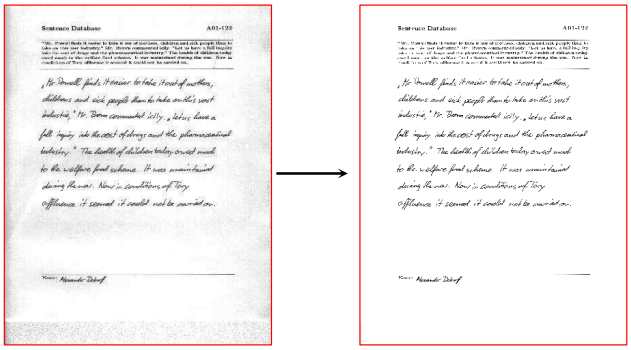

Figure 2: Enhancement of a Scanned Document.

2) Angular Alignment: A page could be scanned with random rotation degrees, which introduces a form of skewness in the scanned image. Loading a skewed image directly to a recognition system degrades the recognition accuracy. A range of attempts have been proposed to tackle this problem [18]. For example, in Profile Projection (PP) analysis, the image is projected to a single vector and further analysis is applied to estimate the skewness angle. The Hough Transform [19] is one of the most widely adopted methods for skew detection and has been especially used for text lines [20], [21].

In our proposed architecture, the angular alignment follows a two-step approach: first we deskew a page to convert a near-vertical (e.g. 13° inclination) or a near-horizontal page to be either strictly vertical or horizontal. We also assume that a page could be rotated by a cardinal angle. Accordingly, the second step involves rotating the image with 0°, 90°, 180° and 270° degrees. For each of the rotated variants of an image, regardless of the content type (handwritten or machine-printed), of the scanned document we apply a machine printed text transcription and score the generated words for each of the four rotations against a dictionary. The rotation with the highest score is assumed to be the optimum rotation of the page, which is further considered for the rest of the processing stages. Fig. 3 shows the angular alignment applied on an input image. We assume here that the page has one dominant rotation for text. However, if this is not the case the page could be divided into regions which are processed independently for rotation.

B. Machine-Printed Text Recognition

Text recognition has been studied widely [22], [23]. The work of Smith [24] proposed a complete framework for text recognition that involved several methods for text detection, localization and recognition. His work became the basis of one of the most successful text recognition frameworks, Tesseract [25], accordingly we used it in our work.

Despite the high accuracy of Tesseract in transcribing the machine-printed text, it fails in transcribing the

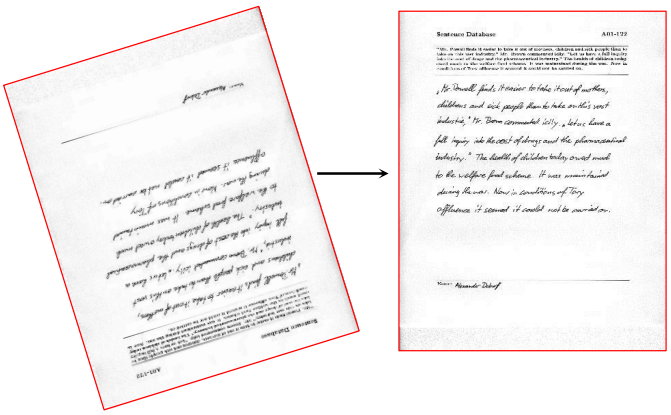

Figure 3: Angular Alignment of a Scanned Document.

handwritten one. For example, Fig. 4 shows a machine-printed word, “Gaitskell”, correctly recognized while its corresponding handwritten recognition is “604'an”. In our process flow, we used the low level transcription accuracy, detected by matching the generated word against the spell-checked version, as an indication of the presence of handwritten text in the input image.

C. Spell Checking

The problem of spell checking and word correction has been considered for several decades [26]. Hodge *et al.* [27] tried to merge the performance of phonetic and associative matching in addition to supervised learning methods to develop a hybrid method for spelling correction. Simple correction methods involve constructing a tree of a language vocabulary with the help of a predefined dictionary, and they proceed by finding matches of the most similar words to the word being corrected. Other methods exploit the use of a language model and the word context [28] to enhance the correction decision.

The spell-checking stage is used in two phases within the process flow. Firstly it is used after the machine-printed text recognition and based on its output, the handwritten text recognition is induced. Secondly, it is also used after the handwritten text recognition stage to validate the handwritten text recognition output. We used a combination of two spell-checkers [29], [30], more could be included to improve the correction decision.

D. Word Cropping

In the cropping stage, words that fail the spell-checking test of the machine-printed text recognition phase are extracted using the bounding box defined for each word. We found through empirical trials that the recognition accuracy is degraded when the boundary characters reside over the border of the cropped image. This is due to the inability of the recognition model to capture the actual vertical and horizontal context of the pixels at boundaries of a word. To address this issue, we padded the cropped word with white space in all directions.

Algorithm 1 Rule-Based Nomination to Retrieve the Optimum Candidate Word

```
CO : Candiate Option
Options List : [A, B, C, D]

if Options Count == 1 then
    CO = A
else if Options Count == 3 then
    CO = C
else if Options Count == 4 then
    if D != <UNK> then
        CO = D
    else
        if B != <UNK> then
            CO = B
        else
            CO = A
        end if
    end if
end if
```

E. Handwritten Text Recognition

Statistical methods such as HMM have been used for handwritten text recognition. The HMM was also considered in combination with neural networks in [31]. A two dimensional HMM to capture the changes in both the horizontal and vertical axes of a text line was investigated in [32]. Neural network based models have also been applied to the problem. Long Short-Term Memory (LSTM) [33] is one of the successful neural based models that is adapted for temporal sequences especially text, where Graves *et al.* [12], [13] proposed a multidimensional LSTM for text recognition. Motivated by the success of RNN in text recognition, we adopted models dependant on LSTM for this stage.

F. Word Nomination

Following Fig. 1, at the nomination stage we are aiming to retrieve the optimum candidate word from the list of options generated through the process flow for each single word present in the document. For this task, we propose a rule-based and a context-based method.

Each potential word in the document at this stage will have a maximum of four options (A, B, C, D), which are the machine-printed text recognition output (A) and its spell-checked version (B), and the handwritten text recognition version (C) together with its spell-checked output (D). The list of options could be of size one (if B matches A) or three (if D matches C) or four (if D does not match C).

In the rule-based nomination, Alg. 1, Option A is considered as the candidate option by default, and based on the list size it could be either substituted or chosen as the candidate option. For example, if the list has three options, it means that the machine-printed version has failed the spell-checking stage accordingly the system had to retrieve the handwritten variant of the word, which passed the spell-checking. In this case, the third option (C) for the handwritten version is the chosen one. A similar

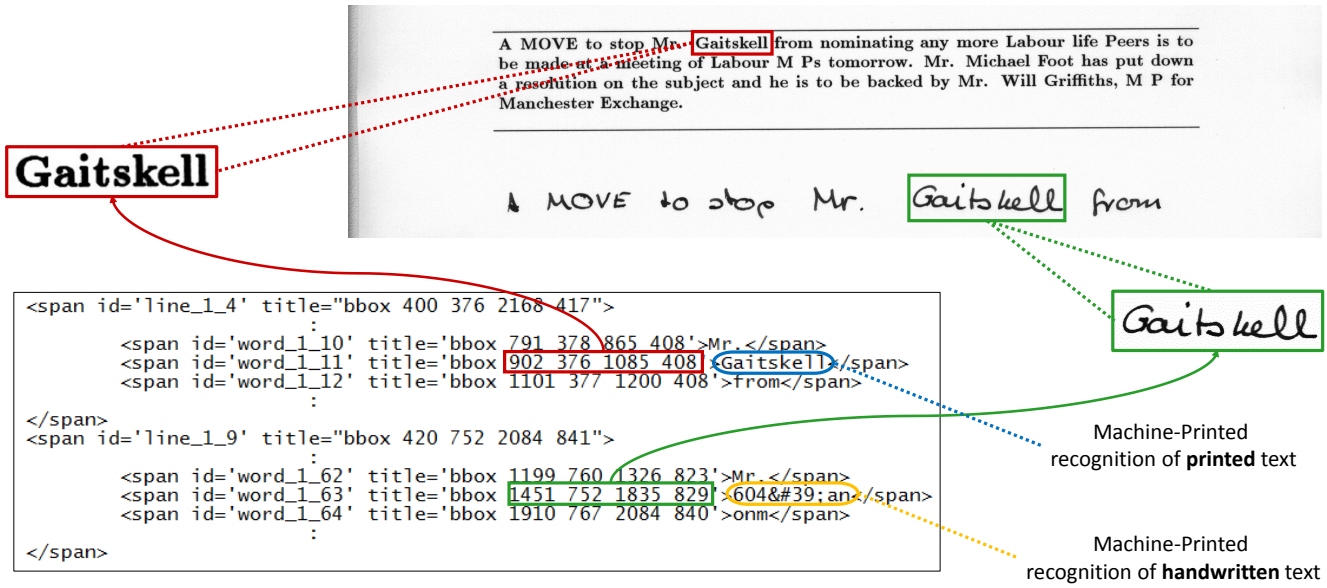

Figure 4: An hOCR File Containing Recognized Text and The Bounding-box for Each Word.

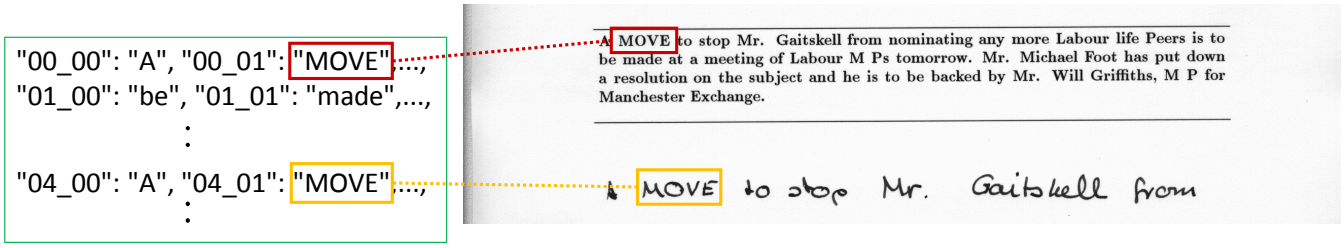

Figure 5: The format of the MIXED-IAM labels for a sample page of the IAM database.

Algorithm 2 Context-Based Nomination to Retrieve the Optimum Candidate Word

$CO : Candiate\,Option$
$PoL : Previous\,options\,List$
$CoL : Current\,options\,List$
$NoL : Next\,options\,List$
$SM : Similarity\,Matrix$

$PCoL = CrossJoin(PoL, CoL)$
$CNoL = CrossJoin(CoL, NoL)$
$PCoEL = Embeddings(PCoL)$
$CNoEL = Embeddings(CNoL)$

for $i < len(PCoEL)$ **do**
  for $j < len(CNoEL)$ **do**
    $SM[i][j] = Cosine(PCoEL[i], CNoEL[j])$
  end for
end for

$index = argmax(argmax(SM))$
$CO = CoL[index]$

rule applies to a list of four options, but in this case we have to consider the possibility of having an unknown flag generated from the spell-checking of either the machine-printed and the handwritten variant as listed in Alg. 1.

In the context-based nomination, Alg. 2, choosing from one of the possible four options of each word is considered within a context of one preceding and one succeeding word. The elimination stage would have been straight forward if the neighbouring words in a context do not have an option list, but the case could be that each of these words have an option list on its own, which introduces additional complexity to the decision process. To fulfill the nomination process in such scenario, bi-grams are constructed from each of the words in the current options list and all the words in the options list of the preceding and succeeding words. For example, for a current word options list of size 4 and a succeeding word options list of size 3 (noting that the preceeding options list will always contain a single word due to the progression order of the transcription), we will have a total of 4 bi-grams between the current and the preceding words and 12 bi-grams between the current and the succeeding words. Bi-gram embedding vectors are retrieved using pre-trained models base on [34]. The distances between all combinations of vectors of the previous embeddings set and the succeeding embeddings set of vectors are retrieved. The smaller the

distance, the more it indicates a higher probability of a pair of bi-grams to reside in proximity to each other in the feature space. The smallest distance is used in choosing the optimum candidate from the options list. Alg. 2 summarizes the operation of the context-based nomination, where a cross join operation is performed between the options list of a preceding word (PoL) and the options list of the current word (CoL). A similar cross join is applied to current options list and the succeeding options list (NoL). Following the generation of the cross join, the embeddings are generated for each bi-gram item. The embeddings of the previous-current options (PCoEL in Alg. 2) are compared against each item in the current-next (CNoEL in Alg. 2) options using cosine similarity between each pair of vectors. The index of the vector with maximum similarity value is the index of the optimum candidate word in the current option list.

III. Experiments

In this section, we evaluate the accuracy of our proposed flow and provide analysis to the effect of different stages in the process flow.

A. MIXED-IAM labels

The IAM handwriting database [16] is composed of 1539 forms of handwritten English text written by 600 writers. A single page of the IAM database is split into two sections, machine-printed text in the upper section and the bottom section of the page is used for a writer to fill with their style of handwriting – transcribing the machine-printed text above[4]. The dataset is released with tokenized transcription for each text line of the handwriting section. Paragraph-level transcription for the machine-printed section of the page is also provided along with the database.

We used Natural Language Processing Toolkit (NLTK) [35] to tokenize the machine printed section and merged both sections into a single transcription represented by the lines and word order in the page disregarding the word type as shown in Fig. 5.

B. Evaluation Metrics

The output transcription of the TMIXT process flow is concatenated into a single paragraph, and similarly the corresponding target transcription within the MIXED-IAM labels. Both represent the predicted and target transcription for an image of a document retrieved from the IAM database, respectively.

The evaluation we applied is based on:

- Character level evaluation using the Levenshtein distance [36] to measure the number of deletions, insertions, or substitutions required to transform the predicted document to the target document.
- Word level evaluation using a Bag-of-Word (BoW) [37] representation, which involves comparing the frequency of occurrences of words in either the predicted or the target transcription disregarding the syntax structure.
- Document similarity, which involves generating an embedding vector [38] for the complete document for either the prediction or the target transcription, and using a similarity measure between the generated embeddings.

[4]Although we know what the handwritten text is supposed to say we do not use that information in our work.

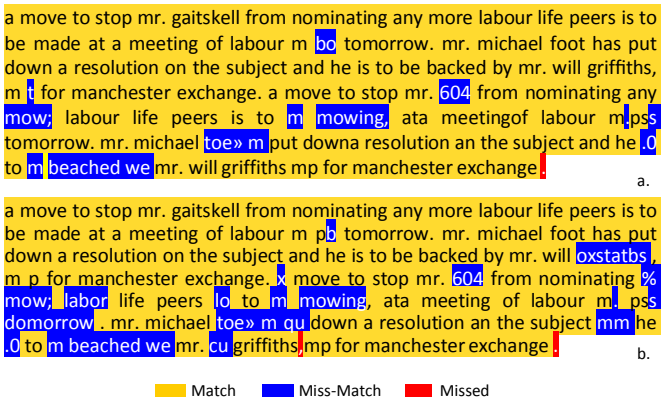

Figure 6: Effect of enhancement on text recognition: a. with enhancement, b. without enhancement.

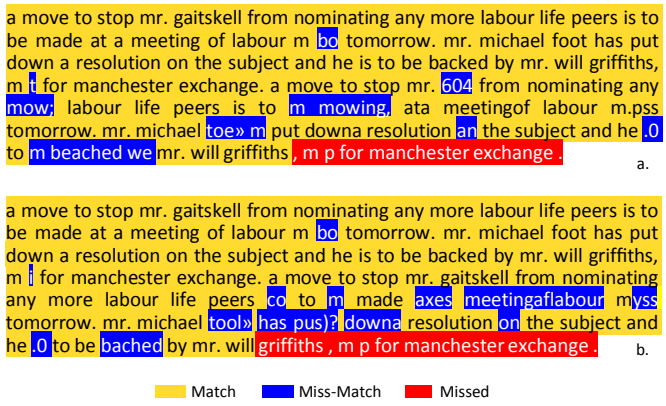

Figure 7: Effect of deskewness on text recognition: a. deskewed, b. without deskewness.

C. Libraries and packages

We used several open-source efforts to implement the TMIXT process flow. For the image enhancement and skewness stage we used the work in [39] and [40], respectively. We used Pillow [41] for the cropping and padding of images. We used Tesseract [25] for the machine-printed and Laia [42] for the handwriting text recognition. Tesseract was also used to generate the hOCR files containing the bounding boxes of the words to be cropped. We used the work of Pagliardini *et al.* [38] for the embeddings generation of the bi-grams and for the spell-checking we used [29] and [30].

D. Analysis

In this subsection, we investigate the impact of the pre-processing stages proposed in the process flow as discussed

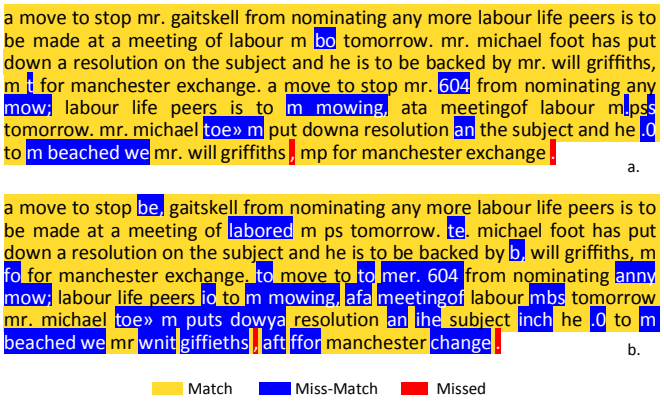

Figure 8: Effect of padding on text recognition: a. with padding, b. without padding.

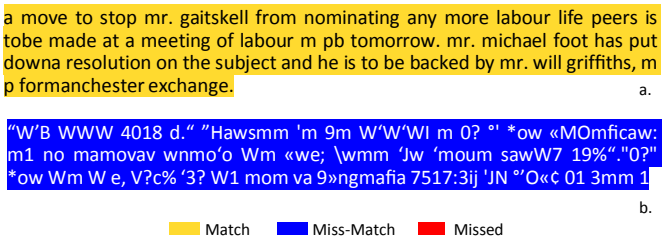

Figure 9: Effect of rotation on text recognition: a. with rotation, b. without rotation

in the Section II. The analysis is applied on a single page on the IAM database with the MIXED-IAM labels. Fig. 6 shows the generated transcription with and without the image enhancement compared to the target transcription. The Levenshtein distance normalized by the length of the longer document between the prediction and the target achieved 89.27% with enhancement. On the other hand, without enhancement the accuracy is drastically degraded to 54.79%.

Fig. 7 shows the effect of the deskewness (vertically aligned) on a single image transcription. As the figure illustrates, some parts of the text are missed in the OCR process if page is skewed. The truncated words degrade the overall accuracy of the transcription, where the recognition accuracy of this sample page with deskewness was 89.27% and 85.44% without it.

Through our recognition process, words which fail the machine-printed recognition stage are cropped from the image and forwarded to the handwritten text recognition for further processing. During cropping, the text may fall at the image boundaries, which make it difficult for the handwriting text recognition to capture the pixels context and consequently the word boundaries accurately. To resolve this issue we pad the cropped images with white spaces in each direction. The effect of the padding is shown in Fig. 8, where the padded version achieved an accuracy of 89.27% compared an accuracy of 82.57% without the padding.

Fig. 9 shows the transcription of the system for a vertically oriented page against a 180° rotation for the same page. It is clear from the figure that the transcription failed to generate a single correct word compared to the correctly aligned variant.

Table I: Performance Evaluation using Character Level and Document Level Comparison

Evaluation Metric	Context-Based Accuracy(%)	Rule-Based Accuracy(%)
Levenshtein	77.17	79.38
Document Similarity	75.06	69.77

Table II: Precision, Recall and F-Score of Proposed Architecture over IAM database

Method	Precision(%)	Recall(%)	F-Score(%)
Context-Based	65.97	67.31	66.46
Rule-Based	70.17	68.14	68.72

E. Results

The evaluation of our proposed flow use the IAM database with the MIXED-IAM labels. We use several metrics to measure the performance of the system over different transcription levels, i.e. character, word and document.

For the character level evaluation, the Levenshtein distance between the predicted raw string and the target label is computed. The edit distance is then normalized using the longest length of the two strings. To evaluate the document level similarity, we extract the document embedding for both the target and the prediction, where the cosine distance is used to measure the similarity between the generated vectors. It is worth mentioning that the preprocessing stages execution time approach approaches 38 seconds on average per page, which is an optimization consideration for future work.

Table I shows the performance evaluation of Levenshtein distance and document embeddings similarity for our proposed architecture using both the rule-based and context-based nomination. The table shows a transcription accuracy of 79.38% using Levenshtein distance in combination with Rule-based nomination, with a comparable accuracy of 77.17% for Context-based nomination. Higher accuracy could be achieved with Context-based nomination through considering other variants of pre-trained embedding models, used in the bi-gram embeddings generation, or a combination of them, which will be considered further in future work.

The BoW evaluation involved creating a set of distinct words using both the target and the predicted documents. The set acts as a dictionary, where each word in the vocabulary is assigned a unique identifier. Each of the two documents are further transformed using the established dictionary into a histogram composed of the unique identifiers present in the document and the frequency of their occurrences disregarding the syntax structure of the document. At this stage, the generated histograms for both the predicted and target documents can be compared against each other. Table II shows the average precision,

Table III: Number of Documents in Different Accuracy Ranges Using Context-Based Elimination Method.

Accuracy Range	Levenshtein Distance	Document Embedding Similarity
90%-100%	15	45
80%-90%	496	491
70%-80%	861	579
60%-70%	147	304
50%-60%	16	92
40%-50%	4	25
30%-40%	0	2
20%-30%	0	1

Table IV: Number of Documents in Different Accuracy Ranges Using Rule-Based Elimination Method.

Accuracy Range	Levenshtein Distance	Document Embedding Similarity
90%-100%	48	24
80%-90%	758	300
70%-80%	592	547
60%-70%	110	326
50%-60%	24	215
40%-50%	4	99
30%-40%	0	26
20%-30%	0	2

recall and F-score of our architecture for transcription of all documents of the IAM database using either the Rule-Based or the Context-Based nomination. The close values of the precision and recall listed in Table II, validates the stability of the reported accuracies of our proposed process flow.

Tables III and IV show the distribution of the 1539 documents of IAM database across different accuracy ranges using Context-Based and Rule-Based option nomination, respectively. Using the Levenshtein distance as a distance measure, 758 documents were transcribed with an average accuracy of 85% compared to 496 documents using the Context-Based elimination. Despite the higher number of documents residing in the 80%-90% range for the Context-Based elimination compared to the Rule-Based using the embedding similarity as a distance measure, this is not indicative of better performance compared to the Levenshtein evaluation, but rather it is showing a variant evaluation representation that depends on the distance measure between the vector representation of documents. Future work will consider optimizing the Context-Based elimination to further exploit the context window to decide on the optimum candidate word.

The recognition of each document through the process flow, results in creating option lists with one, three or four words for each word in a document based on the success or failure of machine-printed or handwritten recognition and the spell-checkers throughout the process. Retrieving the optimum candidate word from the list generated for each word, occurs at the nomination stage, and has considerable impact on the resulting transcription. Following our analysis of the portion of words falling in each category of lists over the whole IAM database, we found that 65% of the words are transcribed correctly through the machine-printed recognition (options list contains one word), 16% of the words passed the handwritten recognition spell-checking (options list contains three words) and 19% failed the spell-checking of hand-written recognized words (options list contain four words). These statistics are primarily dependent on the modules integrated in the process flow. Future work will consider more optimized recognizers and spell-checkers in addition to enhancing the context-based elimination.

IV. Conclusions and Future Work

We have proposed a process flow for Transcribing MIXed handwriting and machine-printed Text (TMIXT) in scanned text documents. The TMIXT process flow allows recognizing text in an image of a document without the need for prior text categorization. The proposed process exploits several open-source libraries to investigate the feasibility of its implementation. However, the flow is generic enough to allow substituting any of the libraries used with different ones fulfilling the same task at the relevant stage, which extends the ability of the TMIXT flow to be applied to other languages, or make use of more powerful tooling as they become available. We have evaluated the accuracy of TMIXT using different evaluation metrics based on character and word level in addition to document embeddings using the widely adapted IAM database. Future work, will investigate enhancing the preprocessing stages for the input image and further optimization for the models used in text recognition either machine-printed or handwriting. We will also consider adapting more enhanced methods for the spell-checking and different pre-trained models to generate the bi-gram embeddings used in the context-based nomination.

V. Acknowledgement

This work was funded by Applications Service Development Operations Team, Joint Forces Command - Information Systems and Services (ISS), UK.

References

[1] J. K. Guo and M. Y. Ma, "Separating handwritten material from machine printed text using hidden markov models," in *International Conference on Document Analysis and Recognition*, Seattle, WA, USA, September 2001, pp. 439–443.

[2] Y. Zheng, H. Li, and D. Doermann, "Machine printed text and handwriting identification in noisy document images," *Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 3, pp. 337–353, 2004.

[3] R. Kandan, N. K. Reddy, K. Arvind, and A. Ramakrishnan, "A robust two level classification algorithm for text localization in documents," in *International Symposium on Visual Computing*, Berlin, Heidelberg, November 2007, pp. 96–105.

[4] S. Chanda, K. Franke, and U. Pal, "Structural handwritten and machine print classification for sparse content and arbitrary oriented document fragments," in *ACM Symposium on Applied Computing*, New York, USA, March 2010, pp. 18–22.

[5] K. S. Abdel Belaïd and V. P. d'Andecy, "Handwritten and printed text separation in real document," in *International Conference on Machine Vision Applications*, Kyoto, Japan, May 2013.

[6] H. Cao, R. Prasad, and P. Natarajan, "Handwritten and typewritten text identification and recognition using hidden markov models," in *International Conference on Document Analysis and Recognition*, Beijing, China, September 2011, pp. 744–748.

[7] L. F. da Silva, A. Conci, and A. Sanchez, "Automatic discrimination between printed and handwritten text in documents," in *Brazilian Symposium on Computer Graphics and Image Processing*, Rio de Janiero, Brazil, October 2009, pp. 261–267.

[8] K. Zagoris, I. Pratikakis, A. Antonacopoulos, B. Gatos, and N. Papamarkos, "Distinction between handwritten and machine-printed text based on the bag of visual words model," *Pattern Recognition*, vol. 47, no. 3, pp. 1051–1062, 2014.

[9] R. Plamondon and S. N. Srihari, "Online and off-line handwriting recognition: a comprehensive survey," *Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 1, pp. 63–84, 2000.

[10] A. El-Yacoubi, M. Gilloux, R. Sabourin, and C. Y. Suen, "An hmm-based approach for off-line unconstrained handwritten word modeling and recognition," *Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 8, pp. 752–760, 1999.

[11] E. Krevat and E. Cuzzillo, "Improving off-line handwritten character recognition with hidden markov models," *Transaction on Pattern Analysis and Machine Learning*, vol. 33, 2006.

[12] A. Graves and J. Schmidhuber, "Offline handwriting recognition with multidimensional recurrent neural networks," in *Neural Information Processing Systems*, Vancouver, Canada.

[13] A. Graves, M. Liwicki, S. Fernández, R. Bertolami, H. Bunke, and J. Schmidhuber, "A novel connectionist system for unconstrained handwriting recognition," *Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 5, pp. 855–868, 2009.

[14] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[15] Z. Feng, Z. Yang, L. Jin, S. Huang, and J. Sun, "Robust shared feature learning for script and handwritten/machine-printed identification," *Pattern Recognition Letters*, vol. 100, pp. 6–13, 2017.

[16] U.-V. Marti and H. Bunke, "The iam-database: an english sentence database for offline handwriting recognition," *International Journal on Document Analysis and Recognition*, vol. 05, no. 1, pp. 39–46, 2002.

[17] K. Jung, K. I. Kim, and A. K. Jain, "Text information extraction in images and video: a survey," *Pattern Recognition*, vol. 37, no. 5, pp. 977–997, 2004.

[18] A. Al-Khatatneh, S. A. Pitchay, and M. Al-qudah, "A review of skew detection techniques for document," in *International Conference on Modelling and Simulation*, Cambridge, UK, March 2015, pp. 316–321.

[19] P. V. Hough, "Method and means for recognizing complex patterns," *United States Patent*, 1962.

[20] L. Likforman-Sulem, A. Hanimyan, and C. Faure, "A hough based algorithm for extracting text lines in handwritten documents," in *International Conference on Document Analysis and Recognition*, Montreal, Quebec, Canada,, August 1995, pp. 774–777.

[21] G. Louloudis, B. Gatos, I. Pratikakis, and K. Halatsis, "A block-based hough transform mapping for text line detection in handwritten documents," in *Workshop on Frontiers in Handwriting Recognition*, France, October 2006.

[22] S. Impedovo, L. Ottaviano, and S. Occhinegro, "Optical character recognition:a survey," *Pattern Recognition and Artificial Intelligence*, vol. 5, no. 01n02, pp. 1–24, 1991.

[23] Q. Ye and D. Doermann, "Text detection and recognition in imagery: A survey," *Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 7, pp. 1480–1500, 2015.

[24] R. W. Smith, "The extraction and recognition of text from multimedia document images," Ph.D. dissertation, University of Bristol, 1987.

[25] R. Smith, "An overview of the tesseract ocr engine," in *International Conference on Document Analysis and Recognition*, Parana, Brazil, November 2007, pp. 629–633.

[26] K. Kukich, "Techniques for automatically correcting words in text," *ACM Computing Surveys*, vol. 24, no. 4, pp. 377–439, 1992.

[27] V. J. Hodge and J. Austin, "A comparison of standard spell checking algorithms and a novel binary neural approach," *IEEE Transactions on Knowledge and Data Engineering*, vol. 15, no. 5, pp. 1073–1081, 2003.

[28] A. Carlson and I. Fette, "Memory-based context-sensitive spelling correction at web scale," in *International Conference on Machine Learning and Applications*, Cincinnati, OH, USA, December 2007, pp. 166–171.

[29] Will Sentance, "spellchecker-autocorrect," https://github.com/WillSen/spellchecker-autocorrect, [Online; accessed November-2018].

[30] AbiWord, "enchant," https://github.com/AbiWord/enchant, [Online; accessed November-2018].

[31] S. Espana-Boquera, M. J. Castro-Bleda, J. Gorbe-Moya, and F. Zamora-Martinez, "Improving offline handwritten text recognition with hybrid hmm/ann models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 4, pp. 767–779, 2011.

[32] H.-S. Park and S.-W. Lee, "A truly 2-d hidden markov model for off-line handwritten character recognition," *Pattern Recognition*, vol. 31, no. 12, pp. 1849–1864, 1998.

[33] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 09, no. 8, pp. 1735–1780, 1997.

[34] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *International Conference on Learning Representations*, 2013.

[35] S. Bird and E. Loper, "Nltk: the natural language toolkit," in *Association for Computational Linguistics on Interactive poster and demonstration sessions*, Sydney, Australia, July 2004, pp. 69–72.

[36] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," *Soviet Physics Doklady*, vol. 10, pp. 707–710, 1966.

[37] G. Salton, A. Wong, and C.-S. Yang, "A vector space model for automatic indexing," *Communications of the ACM*, vol. 18, no. 11, pp. 613–620, 1975.

[38] M. Pagliardini, P. Gupta, and M. Jaggi, "Unsupervised learning of sentence embeddings using compositional n-gram features," *Transactions of the Association for Computational Linguistics*, vol. 1, pp. 528–540, 2018.

[39] M. Villegas, V. Romero, and J. A. Sánchez, "On the modification of binarization algorithms to retain grayscale information for handwritten tet recognition," in *Iberian Conference on Pattern Recognition and Image Analysis*, Santiago de Compostela, Spain, June 2015, pp. 208–215.

[40] Marek Mauder, "App-deskew," https://bitbucket.org/galfar/app-deskew, [Online; accessed November-2018].

[41] Alex Clark and Contributors, "Pillow," https://github.com/python-pillow/Pillow, [Online; accessed November-2018].

[42] Joan Puigcerver, Daniel Martin-Albo and Mauricio Villegas, "Laia: A deep learning toolkit for htr," https://github.com/jpuigcerver/Laia, [Online; accessed November-2018].