Journal of Theoretical and Applied Information Technology
31st December 2015. Vol.82. No.3

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

SMOOTHING TECHNIQUES EVALUATION OF N-GRAM LANGUAGE MODEL FOR ARABIC OCR POST-PROCESSING

[1]AHMED FARIS RAAID AL-MASOUDI, [2]HISHAM SALAM RAFID AL-OBEIDI
[1]PhD Student, Department of Computer Science, University of Baghdad, Iraq
[2]Prof. Dr., Department of Computer Science, University of Technology, Iraq
E-mail: [1]ahmedfaris35@yahoo.com , [2]hisham_salam_1965@yahoo.com

ABSTRACT

N-gram language model is used to correct the errors that resulted from Optical character recognition process. However, the major problem facing N-gram language modeling is that it depends on finite training corpus. Therefore, some words will be missed from the corpus. They are called unknown words. If any N-gram is missing, then the language model will assign a probability of zero to it. Smoothing is a task used to prevent assigning zero probability for missing N-gram in corpus. Each smoothing technique suffers different limitations, and selecting the appropriate smoothing technique depends on where it will be used. Therefore, the purpose of this study is to test these techniques on Arabic dataset, and determines which one of the techniques is the best for this language. This study evaluates the performance of four main smoothing techniques. The experimental results show that all smoothing techniques can reduce error rate. However, the best among them is the Katz Backoff technique.

Keywords: *Smoothing Techniques, N-gram Language Model, Performance Evaluation, OCR, Arabic Language*

1. INTRODUCTION

N-gram language model is widely used in many applications such as spelling correction, optical character recognition (OCR), and speech recognition [1-3]. N-gram language model is a statistical probabilistic model. It is used to provide probability for sequence of words [4]. The value of "N" in the term "N-gram" can be 1, 2, 3, 4... n. The term itself represents a sequence of N neighboring words in a sentence. N-gram is called unigram when N=1, bigram when N=2 and trigram when N=3 and so on.

The frequency of sentences in large corpus, determines the probability of a language model. This model can be used to detect and correct wrong words in a text. Potential word error will happen if the N-gram that contains this word is missing in large corpus. A correction of this error is based on high probability of other sentences. The most important point in this approach is that it does not require confusion set or predefined rules. However, large corpus is needed in order to build an accurate language model. Equation 1 is used to estimate the probability of an N-gram language model [4].

$$P(W_k | W_{k-N+1}^{k-1}) = \frac{C(W_{k-N+1}^{k})}{C(W_{k-N+1}^{k-1})} \quad (1)$$

Equation 1 shows that the conditional probability of a single word W_k is measured based on the previous history words. The character "*N*" represents the N-gram used, such as bigram or trigram. The term W_k represents a single word that needs to measure its probability in position *k* in a sentence. The symbol "|" denotes the conditional probability, and it means the word *"given"*. The symbol *C* represents a frequency of a sentence. Lastly, the character *k* denotes a number of words in sequence. For example, to measure the probability of the word "*stories*" in the sentence "*Most students love teacher stories*" by using trigram language model, then Equation 1 will be changed as shown below:

$$P(\text{stories} \mid \text{love teacher}) = \frac{C(\text{love teacher stories})}{C(\text{love teacher})}$$

Major problem facing N-gram language model is that it depends on finite training corpus [4].

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Therefore, some words will be missed from the corpus. They are called unknown words. If any N-gram is missing, then the language model will give a probability of zero to it, and infinite value can be resulted. Smoothing is a task used to prevent assigning zero probability for missing N-gram in corpus [5]. Each smoothing technique suffers from different limitations, and selecting the appropriate technique depends on where it will be used in any topic [4]. This study implements a practical performance evaluation of main smoothing techniques that are used in OCR post-processing for Arabic language.

Arabic language over the years is complex when processed by the optical character recognition. This is because Arabic characters are connected, and the shape of characters has a vertical overlapping between them [6-8]. These characteristics cause large error rate during the process of OCR, especially when the images contain noise or their scanning resolution is low [9, 10]. OCR system usually generates two types of errors: non-word errors and real word errors [11]. Non-word error occurred when the word generated from the OCR software does not exist in a specific language, such as the word "*sder*" in English language. The real word error occurred when the word generated from the OCR software exists in a specific language, but unsuitable for the sentence [12, 13]. The purpose of this study is to test smoothing techniques on Arabic dataset, and find which one is the best among them for this language.

The paper is organized in five sections: section 1 presented the introduction. Section 2 discusses the main smoothing techniques that will be used in performance evaluation. Section 3 explains the experimental design, measurements, testing dataset and training dataset. In section 4, developed interface, experimental results and discussion are presented. The last section includes conclusions and future work of our research.

2. SMOOTHING TECHNIQUES

This section will explain four main smoothing techniques that will be used in the performance evaluation. It focuses on how the probability is generated by these techniques, and the strengths and weakness of each technique.

2.1 Laplace Smoothing

Laplace smoothing, also called add-one smoothing belongs to the discounting category. This category consists, in addition to the Laplace smoothing, from Witten-Bell discounting, Good-Turing, and absolute discounting [4]. The approach of discounting category is to transfer some mass of probability from seen N-grams to others that never be seen. This transfer will prevent assigning zero probability for missing N-gram in corpus. The result of this transfer does not reflect the real probability of each N-gram [4, 14]. Techniques of discounting category are simple, but they are not commonly used. They do not perform well in some calculation because much probability mass is moved to all unseen n-grams [2, 4]. Since all techniques under discounting category have the same approach, then this study will select Laplace smoothing as one of the comparative techniques for performance evaluation. If a language model uses Laplace smoothing, then this technique will change Equation 1 of a language model to handle zero probabilities by adding a value to all the counts as shown in an Equation 2 [4].

$$\hat{P}(W_k|W_{k-N+1}^{k-1}) = \frac{\delta + C(W_{k-N+1}^{k})}{\delta * D + C(W_{k-N+1}^{k-1})} \quad (2)$$

Where, the term P^ denotes the probability after using smoothing technique, while the symbol *"D"* is the total number of possible (N–1) grams in corpus. The symbol "d" can take values such as 0.5, 0.9. Whenever the value of "d" is smaller, the results will be better. This study will refer to this technique in experiments as LMULS, which means language model using Laplace smoothing.

2.2 Linear Interpolation

This technique belongs to the hierarchy category that focuses on hierarchy of N-gram orders [2, 4]. In linear interpolation technique, several N-gram orders will be added to handle zero probabilities. It always combines the probabilities calculated from all the N-gram orders. For example, the mathematical expression of this technique for trigram is calculated from combining trigram, bigram, and unigram as shown in Equation 3 [4].

$$\hat{P}(W_3|W_1^2) = \alpha_1 P(W_3|W_1^2) + \alpha_2 P(W_3|W_1) + \alpha_3 P(W_3 \quad (3)$$

Where the symbol ∝ is interpolation constant and it donates the weights of trigram, bigram, and unigram. The values of symbols ∝1, ∝2, and ∝3 can be same if the weights of trigram, bigram, and unigram are equal. Otherwise, they will be different. Furthermore, the sum of values of ∝1,

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

∝2, and ∝3 must be equal to 1. Since large N-gram gives accuracy more than short N-gram, then this study will assign values of 0.5, 0.3, and 0.2 to the trigram, bigram, and unigram respectively. This technique does not perform well in some cases. For example, the bigram language model with interpolation technique will give high probability value to the context "*student student*" even the context "*student student*" cannot come in any meaningful sentence. This is because the unigram "*student*" is very common and interpolation technique will combine both probabilities of bigram and unigram together [15]. This study will refer to this technique in experiments as LMULI, which means language model using linear interpolation.

2.3 Katz Backoff

This technique also belongs to the hierarchy category [2, 4]. Language model with Backoff is built based on an (N-1) gram model. Its strategy is if N-gram is not found, switch to (N-1) gram and so on. It used lower-order gram when higher-order gram gave zero probability. It recursively decreases a value of N until lower-order gram is available [4, 16]. For example, if trigram gives non- zero probability, then no need to use lower-order gram. Otherwise, if it is not available, then bigram is used. Furthermore, if bigram does not exist, then unigram is used. The mathematical expression of Backoff trigram can be represented in Equation 4 [16].

$$\hat{P}(W_3|W_1^2)=\begin{cases} P(W_3|W_1^2) \text{ if } C(W_1^2)>0 \\ \alpha_1 P(W_3|W_1) \text{ if } C(W_1^2)=0 \text{ and } C(W_1)>0 \\ \alpha_2 P(W_3) \text{ Otherwise} \end{cases} \quad (4)$$

Note α's are back-off constants [3]. This study will assign values of 0.3 and 0.2 to the bigram and unigram respectively. Backoff technique usually works well. However, it also suffers from the same problem of Interpolation technique. For example, suppose that the trigram "x y z" is not seeing due to the rules of the grammar, and the bigram "x y" is very common. The technique will switch to the bigram "x y", which its probability is high, instead of assigning the value of zero to trigram "x y z" [15]. This study will refer to this technique in experiments as LMUKB, which means language model using Katz Backoff.

2.4 Kneyser-Ney

This technique is based on main idea that words that have occurred in more sentences or contexts are more likely to occur in new context or sentence as well [4, 17]. Therefore, it depends on the frequency of different contexts for each word occurred in. For example, by assuming the sentence "*I cannot read without my* _____ ", has two candidates, *glasses* and *Francisco* to complete it. *Francisco* is more common than *glasses*. Therefore, previous smoothing techniques will prefer it. But *Francisco* is only occurring in context "*San Francisco*" and *glasses* are frequent in several contexts. Therefore, this technique will prefer the word *glasses* rather than *Francisco*. Kneyser-Ney technique is built based on Katz Backoff technique or based on linear interpolation technique, by adding context information to their equations [4].

Kneyser-Ney fails in some cases [17]. For example, by assuming the word "*Thai*" in the sentence "*I want Thoi food*", was misspelled as "*Thoi*", and the previous sentence has only two candidates, *Thai* and *Chinese* to complete it, then Kneyser-Ney category will prefer the word *Chinese* rather than the word *Thai* because the word *Chinese* has more contexts than the word *Thai* [4]. This study will implement Kneyser-Ney by adding context information to the Katz Backoff technique. This study will also refer to this technique in experiments as LMUKN, which means language model using Kneyser-Ney.

3. EVALUATION SETTING

This section will explain experimental design, measurement, testing dataset and training dataset.

3.1 Experimental Design

Five experiments will be performed to achieve a single goal of finding out the best smoothing techniques for reducing OCR error rate of Arabic language. This goal will be achieved by comparing output of five experiments in order to identify the best technique. All experiments are performed to convert testing images to a text. Tesseract OCR will be used to extract texts from input images in all experiments. Tesseract OCR is used by many researchers because it is one of the best free OCR engines [18-20]. Figure 1 shows how to conduct the experiments.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

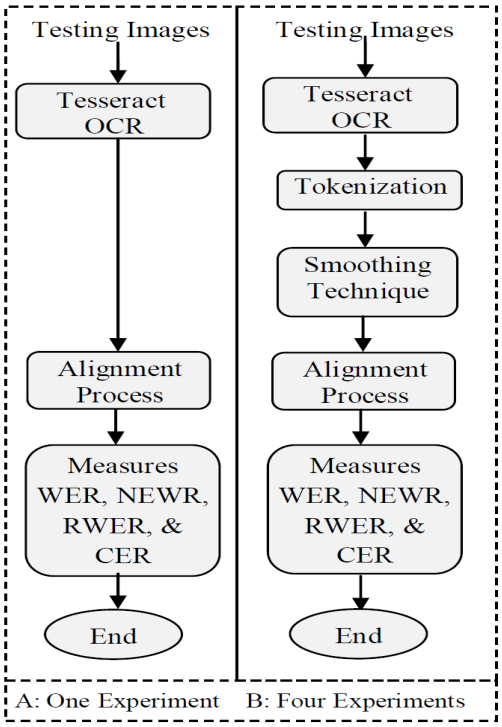

Figure 1: Experimental Design

Figure 1 shows that the result of Tesseract OCR in each experiment is a single output text. Part "*A*" in Figure 1 shows that the first experiment is performed without using any smoothing technique. The purpose of experiment one is to find accuracy of OCR without using any smoothing technique in order to compare it with the accuracy of existing techniques. Part "*B*" in Figure 1 will be implemented four times. However, the smoothing technique for each experiment will be different. Experiments two, three, four, and five will be implemented using LMULS, LMULI, LMUKB, and LMUKN techniques as described in sections 2.1, 2.2, 2.3, and 2.4 respectively. The degree of N-gram in the experiments two, three, four, and five is trigram language model.

Figure 1 also shows that output of all experiments except the first will pass to the tokenization process, which means split text to an array of words using spaces between them as divider. The result of tokenization process is an array of words in each experiment except the first. Tokenization process is important to the smoothing techniques because these techniques cannot deal with output text directly, they can deal with the words of output text [1]. After applying smoothing techniques, array of words in each experiment except the first will be joined as a text. At this point, texts resulting from all experiments will pass to the alignment process.

Alignment process is required because the output text in each experiment will be compared with reference text to calculate four metrics: word error rate (WER), non-word error rate (NWER), real word error rate (RWER), and character error rate (CER). The meaning of each metric and how to measure it is discussed in section 3.2. Alignment process is needed in comparison because some symbols in OCR engine may be deleted, misrecognized, or inserted [21]. For example, the symbol "d" may be recognized by OCR engine as two symbols "cl". Another example, the two symbols "vv" may be recognized by OCR engine as one symbol "w". The deleted, misrecognized, and inserted symbols will make some of the words either be split or merged, which makes the number of words of OCR output text unequal to the reference text [21, 22]. Therefore, this study needs to do alignment process between the OCR text and reference text so that it can measure error rate.

Alignment process means aligning each character in any reference text with similar character in OCR text. It is a complex process due to the high calculations overhead needed, which is causing long processing time, especially when the numbers of the characters are greater than 2500 [20, 23]. Figure 2 shows simple example on an alignment process between OCR output text and reference text.

Reference text	Yahoo mail
OCR output text	Yohoo nnail

Alignment between OCR output and reference text

Y	a	h	o	o		m	-	a	i	l
Y	o	h	o	o		n	n	a	i	l

Figure 2: Simple Example on Alignment Process

The alignment process will be performed by using Levenshtein distance with back trace [24]. This algorithm is the most widely used in OCR post-processing [1, 12]. This is because it is accurate in finding the difference between two sequences. However, it needs much processing time for large sequences [1, 25, 26]. After an alignment process, the four metrics can easily be measured. CER is calculated by comparing each character in output text with each character in reference text.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

The comparison process means examining both characters if they are match or not. WER is calculated by comparing each word in output text with each word in reference text. NWER and RWER are calculated by examining each wrong word if it is in lexicon or not. If it exists in a lexicon, then it considers real word error, otherwise, it considers non-word error [21].

3.2 Measurements

Word error rate (WER) is the main metric that use by most researchers in measuring error rate of OCR post-processing techniques [11, 27, 28]. However, this paper will use in addition to the word error rate, further metrics to check the impact of each smoothing technique on them. The metrics are non-word error rate (NWER), real word error rate (RWER), and character error rate (CER). NWER is a measure of the non-word errors in output OCR text, while RWER is a measure of the real word errors in the output of OCR text. Word error rate (WER) is used to measure the rate of all wrong words in OCR output text. It is a combination of non-word errors and real word errors. Lastly, character error rate (CER) refers to the inserted, deleted, and substituted characters in the output of the resulting text of OCR [1, 21, 27]. Equations 5, 6, 7, and 8 below show how to compute WER, NWER, RWER, and CER respectively [1, 21, 29].

$$WER = \frac{\text{No. of wrong words in output OCR text}}{\text{No. of all words in reference text}} * 100 \quad (5)$$

$$NWER = \frac{\text{No. of non-words in output OCR text}}{\text{No. of all words in reference text}} * 100 \quad (6)$$

$$RWER = \frac{\text{No. of real words in output OCR text}}{\text{No. of all words in reference text}} * 100 \quad (7)$$

$$CER = \frac{\text{No. of wrong characters in output OCR text}}{\text{No. of all characters in reference text}} * 100 \quad (8)$$

3.3 Testing and Training Dataset

Most researchers used different sizes and types of testing dataset for Arabic language [1, 11, 29]. This paper will follow the same procedure used by [29] to produce testing images. This dataset has five characteristics. The first is that it contains 101258 symbols within Arabic images. The symbol means the smallest meaningful unit within a writing system. The second is that, text is chosen randomly from formal Arabic websites on the Internet. The third is that it contains in addition to characters of a text, the special symbols, such as commas, brackets, etc. The fourth is that, it includes eight different Arabic fonts. The names of these fonts are Adobe Arabic, Simplified Arabic, Courier new, Tahoma, Traditional Arabic, Times New Roman, Arial, and Microsoft sans Serif. Lastly, each font will consist of six different sizes ranging from 10 to 20. The texts in these images act as reference during the testing process. In addition to that, reference text will be used to produce testing images. Arabic text is first printed on papers, and then the hardcopy are scanned at 300 dpi with a grey level in a modern scanner to generate testing dataset images.

The corpus named "*Arabic Gigaword Fourth Edition*" will be used as training dataset for building the database of trigram language model. This corpus is produced by the linguistic data consortium at the Pennsylvania University [30]. It was collected over many years from Arabic news websites, and it contains more than 850 million of Arabic words. It works well for building the database of trigram language model because format structure of this corpus is not based on single words, but it is based on paragraphs [30, 31].

4. EXPERIMENTS RESULTS

This section presents the results of the five experiments performed in this study. The goal of these experiments is to measure the performance of each smoothing technique. This can be achieved by making comparison of results of them. To implement the experiments, an interface is designed using VB.NET under MS Visual Studio.net 2012. The term "LMWUS" will be used to refer to the experiments 1 and it will mean results of OCR system without using any smoothing technique. Furthermore, this study will present four figures to represents the testing results. Each figure represents the results of comparison of a single metric. Testing results are shown in Figures 3 to 6.

From Figures 3, 4, 5, and 6, it can be seen clearly that values of the metrics for each smoothing technique are different from each other. Overall, Figures 3, 4, 5, and 6 show that results of OCR system without using any smoothing technique (LMWUS) had the highest percentage values of WER, NWER, RWER, and CER than the others, with rates of 47.88%, 34.80%, 13.08% and 20.60%

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

respectively. This means OCR accuracy is still low for Arabic language.

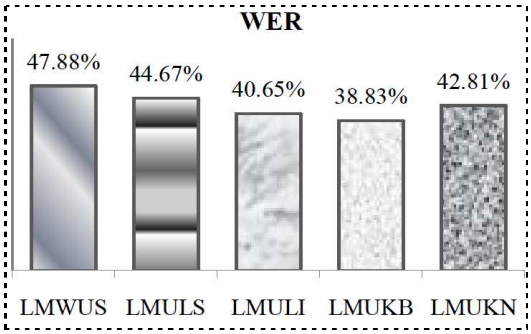

Figure 3: WER for Five Experiments

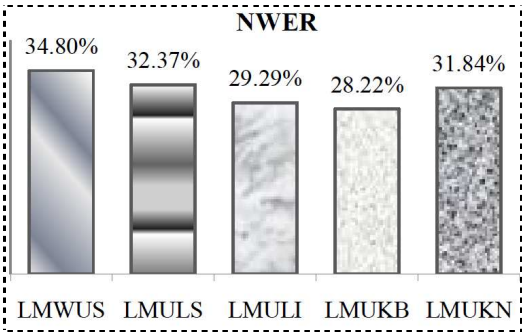

Figure 4: NWER for Five Experiments

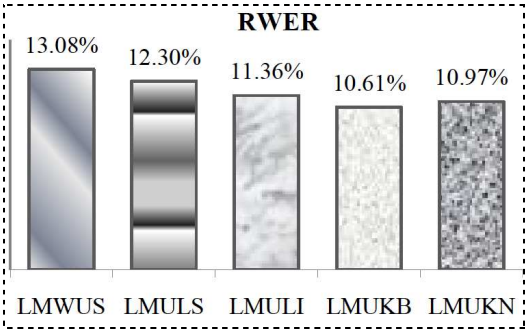

Figure 5: RWER for Five Experiments

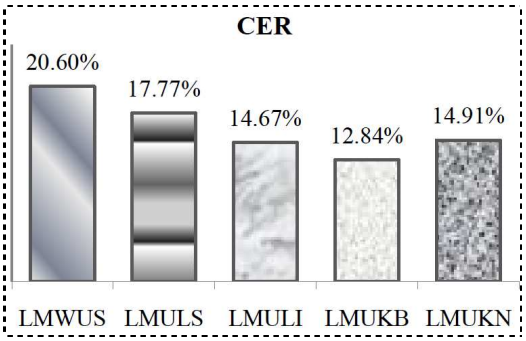

Figure 6: CER for Five Experiments

On the other hand, it can be seen that WER, NWER, RWER, and CER of LMUKB had lowest percentage values than the others, with rates of 38.83%, 28.22%, 10.61% and 12.84% respectively. This implies that LMUKB is the best smoothing technique for correcting errors of Arabic OCR. Figure 3, 4, 5, and 6 also show that LMULI had a significant reduction in WER, NWER, RWER, and CER, with rates of 40.65%, 29.29%, 11.36% and 17.77% respectively. However, LMULI is less efficiency than LMUKB in smoothing of OCR errors. This is because all values of metrics WER, NWER, RWER, and CER for LMULI are greater than values of LMUKB. Lastly, Figure 3, 4, 5, and 6 show that LMULS had the least efficiency in WER, NWER, RWER, and CER than the others, with rates of 44.67%, 32.37%, 12.30% and 17.77% respectively. To sum up, results from five previous experiments show that it is difficult for accuracy of OCR engines to be 100% for Arabic language. Furthermore, they show LMUKB is the best smoothing technique for correcting errors of Arabic OCR.

5. CONCLUSION AND FUTURE WORK

This study presents the details of the performance evaluation of main smoothing techniques for Arabic OCR errors. The evaluation process of this study is performed based on experimental approach. All techniques were tested using the same testing dataset. As with any research, if a testing dataset is large, then the validity and reliability of the research are higher, while if testing dataset is too small, then it will be inappropriate to be used to determine the strength of each smoothing technique. Therefore, this study used large testing dataset in the evaluation process.

The experimental results show that using any smoothing technique can reduce error rate. However, the best technique among them is LMUKB. Further research can be done to improve existing smoothing techniques so that they can handle high error rate of Arabic OCR. In addition to that, it can combine some existing OCR post-processing techniques in a hybrid way to benefits from strengths of them.

REFRENCES:

[1] I. Q. Habeeb, S. A. Yusof, and F. B. Ahmad, "Two Bigrams Based Language Model for Auto Correction of Arabic OCR Errors," *International Journal of Digital Content Technology and its Applications,* vol. 8, pp. 72 - 80, February 28 2014.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[2] A. Islam and D. Inkpen, "Real-word spelling correction using Google Web 1T n-gram with backoff," in *Natural Language Processing and Knowledge Engineering, 2009. NLP-KE 2009. International Conference on*, 2009, pp. 1-8.
[3] V. Gupta, M. Lennig, and P. Mermelstein, "A language model for very large-vocabulary speech recognition," *Computer Speech & Language,* vol. 6, pp. 331-344, 1992.
[4] D. Jurafsky and J. H. Martin, *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition*, 2nd ed.: Pearson Education India, 2009.
[5] D. Jurafsky and J. H. Martin, *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition*: Pearson Education India, 2000.
[6] I. Aljarrah, O. Al-Khaleel, K. Mhaidat, M. a. Alrefai, A. Alzu'bi, and M. Rabab'ah, "Automated System for Arabic Optical Character Recognition with Lookup Dictionary," *Journal of Emerging Technologies in Web Intelligence,* vol. 4, pp. 362-370, 2012.
[7] M. Labidi, M. Khemakhem, and M. Jemni, "Grid'5000 Based Large Scale OCR Using the DTW Algorithm: Case of the Arabic Cursive Writing," *Recent Advances in Document Recognition and Understanding,* p. 73, 2011.
[8] M. Oujaoura, R. El Ayachi, M. Fakir, B. Bouikhalene, and B. Minaoui, "Zernike moments and neural networks for recognition of isolated Arabic characters," *International Journal of Computer Engineering Science,* vol. 2, pp. 17-25, 2012.
[9] H. Al-Rashaideh, "Preprocessing phase for Arabic Word Handwritten Recognition," *Информационные процессы,* vol. 6, 2006.
[10] M. S. Khorsheed, "Off-line Arabic character recognition–a review," *Pattern analysis & applications,* vol. 5, pp. 31-45, 2002.
[11] Y. Bassil and M. Alwani, "Ocr post-processing error correction algorithm using google online spelling suggestion," *arXiv preprint arXiv:1204.0191,* 2012.
[12] J. F. Daðason, "Post-Correction of Icelandic OCR Text," (Master's thesis, University of Iceland, Reykjavik, Iceland), 2012.
[13] K. Kukich, "Techniques for automatically correcting words in text," *ACM Computing Surveys (CSUR),* vol. 24, pp. 377-439, 1992.
[14] W. Gale and K. Church, "What is wrong with adding one," *Corpus-based research into language,* pp. 189-198, 1994.
[15] C. D. Manning and H. Schütze, *Foundations of statistical natural language processing*: MIT press, 1999.
[16] S. Katz, "Estimation of probabilities from sparse data for the language model component of a speech recognizer," *Acoustics, Speech and Signal Processing, IEEE Transactions on,* vol. 35, pp. 400-401, 1987.
[17] B.-J. P. Hsu, "Language modeling for limited-data domains," Massachusetts Institute of Technology, 2009.
[18] C. Patel, A. Patel, and D. Patel, "Optical character recognition by open source OCR tool tesseract: A case study," *International Journal of Computer Applications,* vol. 55, pp. 50-56, 2012.
[19] Google Inc. (2015, January 02). *Tesseract-ocr v3.02*. Available: https://code.google.com/p/tesseract-ocr/
[20] W. B. Lund, D. J. Kennard, and E. K. Ringger, "Combining multiple thresholding binarization values to improve OCR output," in *IS&T/SPIE Electronic Imaging*, 2013, pp. 86580R-86580R-11.
[21] I. Q. Habeeb, S. A. Yusof, and F. B. Ahmad, "Improving Optical Character Recognition Process for Low Resolution Images," *International Journal of Advancements in Computing Technology,* vol. 6, pp. 13 - 21, May 30 2014.
[22] W. B. Lund, D. D. Walker, and E. K. Ringger, "Progressive alignment and discriminative error correction for multiple OCR engines," in *Document Analysis and Recognition (ICDAR), 2011 International Conference on*, 2011, pp. 764-768.
[23] C. Notredame, "Recent evolutions of multiple sequence alignment algorithms," *PLoS computational biology,* vol. 3, p. e123, 2007.
[24] X. Cai, "Approximate Sequence Alignment," Peking University, 2013.
[25] K. U. Schulz and S. Mihov, "Fast string correction with Levenshtein automata," *International Journal on Document Analysis and Recognition,* vol. 5, pp. 67-85, 2002.
[26] P. Mitankin, "Universal levenshtein automata. building and properties," Master's thesis, Sofia University, Bulgaria, 2005.
[27] W. B. Lund, E. K. Ringger, and D. D. Walker, "How well does multiple OCR error correction generalize?," in *IS&T/SPIE Electronic Imaging*, 2014, pp. 90210A-90210A-13.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[28] W. B. Lund and E. K. Ringger, "Improving optical character recognition through efficient multiple system alignment," in *Proceedings of the 9th ACM/IEEE-CS joint conference on Digital libraries*, 2009, pp. 231-240.

[29] M. S. M. El-Mahallawy, "A large scale HMM-based omni front-written OCR system for cursive scripts," (PhD thesis, Cairo University, Cairo, Egypt), 2008.

[30] A. AbdelRaouf, C. A. Higgins, T. Pridmore, and M. Khalil, "Building a multi-modal Arabic corpus (MMAC)," *International Journal on Document Analysis and Recognition (IJDAR),* vol. 13, pp. 285-302, 2010.

[31] R. Parker, D. Graff, K. Chen, J. Kong, and K. Maeda, "Arabic Gigaword," *Linguistic Data Consortium, University of Pennsylvania, Philadelphia,* 2009.