Automatic Arabic diacritics restoration based on deep nets

Mohsen A. A. Rashwan
ELC Dept., Cairo University,
mohsen_rashwan@rdi-eg.com

Ahmad A. Al Sallab
ELC Dept., Cairo University
ahmad.elsallab@gmail.com

Hazem M. Raafat
Computer Science Dept., Kuwait University
hazem@cs.ku.edu.kw

Ahmed Rafea
Computer Science Dept., American University in Cairo
Rafea@aucegypt.edu

Abstract

In this paper, Arabic diacritics restoration problem is tackled under the deep learning framework presenting Confused Subset Resolution (CSR) method to improve the classification accuracy, in addition to Arabic Part-of-Speech (PoS) tagging framework using deep neural nets. Special focus is given to syntactic diacritization, which still suffer low accuracy as indicated by related works. Evaluation is done versus state-of-the-art systems reported in literature, with quite challenging datasets, collected from different domains. Standard datasets like LDC Arabic Tree Bank is used in addition to custom ones available online for results replication. Results show significant improvement of the proposed techniques over other approaches, reducing the syntactic classification error to 9.9% and morphological classification error to 3% compared to 12.7% and 3.8% of the best reported results in literature, improving the error by 22% over the best reported systems

1 Introduction

Arabic is a wide spread language spoken by over 350 million people on the planet. Arabic alphabet and vocabulary are very rich, with the same word morphology being a candidate of different meanings and pronunciations. For example the word عمر might bear the meaning of the person name "Omar" عُمَرْ or the meaning of "age" عُمْرْ. What distinguish them is the diacritization signs assigned to each character of the word.

Diacritics are marks added on the character to reflect its correct pronunciation, according to grammatical, syntactical and morphological rules of the language.

Nowadays, Modern Standard Arabic (MSA) transcripts are written without diacritics, left to the ability of the reader to restore them from the context and knowledge. Diacritics restoration is not an easy task even for knowledgeable, native Arabic speakers. On the other hand, there are many machine learning tasks, like Text-To-Speech (TTS), translation, spelling correction, word sense disambiguation,…etc, that require diacritizing the script as a pre-processing step before applying the core application technique.

In its basic form, the problem can be reduced to a pattern classification problem, with seven diacritics classes being the targets. In addition, the diacritics classification can be divided into syntactical diacritization, caring about case-ending and morphological diacritization, caring about the rest of the word diacritics. So far, morphological part of the problem is almost solved, leaving a marginal error of around 3-4%, Rashwan et al. (2009, 2011). On the other hand, syntactical diacritization errors are still high, hitting a ceiling that is claimed to be asymptotic and cannot be squeezed any further, Rashwan et al. (2009, 2011). For this reason, we focus our effort to squeeze this error beyond the least 12.5% error obtained in Rashwan et al. (2009, 2011).

Recently, a significant advancement in the area of deep learning has been witnessed, with the development of a generative model; Deep Belief Nets (DBN), with a fast algorithm for inference of the model parameters. Deep Neural Networks (DNN) shall be the basic machine learning classifier used in this work, employing the latest results reached in the deep learning field. An efficient features' vector is designed under the umbrella of deep learning to distinguish different words diacritics. Features that are tested in the current work are: PoS, morphological quadruple of lexemes, last character and word identity. In addition, context features are essential to the diacritization problem. Context features include, the

Proceedings of the EMNLP 2014 Workshop on Arabic Natural Langauge Processing (ANLP), pages 65–72,
October 25, 2014, Doha, Qatar.

previous word features, as well as the previous word diacritic.

Part-of-Speech (PoS) features are critical to syntactic diacritization, which is the focus of this work. For some datasets PoS tags are manually annotated by professional linguistics, while for the real case and most datasets, they are not available. For this reason, standalone PoS taggers are built under the deep learning framework, which can reused in Arabic PoS tagging systems, needed for many other applications, not only for Arabic diacritization.

The deep learning model often hit a performance barrier which cannot be crossed. Hence, error analysis and diagnosis is run on the confusion matrix results, proposing the Confused Subset Resolution (CSR) method to train sub-classifiers to resolve the identified confusions and automatically generate a deep network-of-networks composed of the main classifier and the sub-classifiers working together to offer improved accuracy system purified of the identified confusions, offering around 2% error enhancement.

Evaluation of the proposed techniques is done on two datasets; the first is a custom one collected from many different sources, which is available online at (http://www.RDI-eg.com/RDI/TrainingData is where to download TRN_DB_II). Manually extracted PoS and morphological quadruples are available for only a part of this dataset. The PoS tags of this part of the dataset were used to build the DNN PoS taggers to tag the rest of the dataset. The corresponding test set is available online at (http://www.RDI-eg.com/RDI/TestData is where to download TST_DB), which is quite challenging and collected from different sources than training ones. The second dataset is the standard LDC Arabic Tree Bank dataset LDC Arabic Tree Bank Part 3, (http://www.ldc.upenn.edu/Catalog/catalogEntry.jsp?catalogId=LDC2005T20) used to bench mark the system against state-of-the art systems in Arabic diacritization area.

The rest of the paper is organized as follows: first the related works in literature are surveyed, followed by a formulation of the CSR method. The next section is dedicated to describing the features used in the system, and how they are encoded and represented in the features' vector followed by the details of building the DNN PoS tagger for Arabic. The datasets used for evaluation are then described. The next section describes the system evaluation experiments. Experimental results include an error analysis study of the effect of each feature and method on the system performance, in addition to benchmarking against state-of-the art systems in literature, evaluated on standard datasets. Finally, the paper is concluded with the main results and conclusion.

2 Related work

There have been many attempts to approach the Arabic diacritization problem by different techniques. Focus will be around three works strongly related to what is proposed here, and having the best results in literature. Zitouni et al. (2006) apply Maximum Entropy classification to the problem taking the advantage of the MaxEnt framework to combine different features together, like lexical, segment-based, and PoS features. Segmentation involves getting the prefix, suffix, and stem of the word. PoS features are also generated under the MaxEnt framework. Habash and Rambow (2007) perform Morphological Analysis and Disambiguation of Arabic (MADA) system, and then apply SVM classification. Last, Rashwan et al. (2009 and 2011) propose a hybrid approach composed of two stages: first, maximum marginal probability via A* lattice search and n-grams probability estimation. When full-form words are OOV, the system switches to the second mode which factorizes each Arabic word into all its possible morphological constituents, then uses also the same techniques used by the first mode to get the most likely sequence of morphemes, hence the most likely diacritization. The latter system shall be our baseline, since it gives the best results in literature so far, and the dataset used to evaluate it is available at our hand, and hence fair comparison is possible. Also, comparison to the three systems is made on the LDC Arabic Tree Bank data set.

3 System architecture

In this section the overall system is presented. The raw text input is fed to the system word by word. According to the configured context depth, a number of succeeding and preceding words are stored in a context memory. In our system the context is experimentally taken as three preceding words and one succeeding word (N=3, M=1), which is found to give the best accuracy results versus other tunings: (N=1, M=1), (N=2, M=2), (N=3, M=2), (N=1, M=3) and (N=1, M=3). If the word is the first or last one in a sentence, the pre-

ceding or succeeding context is zero padded. Word context serves in case of syntactic diacritization, while for morphological case, characters context is also needed, which is directly present in the character sequence of the single input word itself.

Features extraction procedure depends on the feature itself. For PoS tags, a special DNN is trained for that purpose, which also makes use of the context of the word. For other features, like sequence of characters forming the word, the last character of the word and the morphological quadruples are directly extracted from the single word.

The framework in Figure 2 is employed. Three layers network architecture is used for each features extraction subnet or classification network. For the classification network a 20-20-20 architecture was used, while for PoS-tagging networks a 60-60-60 is used. The network architecture is determined empirically. By experiments it was found that the best architecture is the symmetric one, with the same width for all layers. The best width is found to be the same as the average number of ones in the training set features vectors.

The neural network training undergoes DBN pre training as in Hinton et al. (2006) for 20 epochs per layer, with batch size of 1000 examples each without mini batches. Momentum is used initially with 0.5 for the first 5 epochs and then raised to 0.9 for the next epochs. The discriminative fine tuning is performed using conjugate gradient minimization for 30 epochs. For the first 6 epochs, only the upper layer is adjusted, then the rest of the layers are trained for the next epochs.

Once the features are ready of a certain raw word it is fed to the DNN classifier. The resulting confusion matrix from the training phase is then fed to the CSR method to generate the tree structure that improves the system accuracy. During testing phase, the raw input features are fed to the DNN classifier to obtain an initial guess of the target diacritic. This guess is then improved in the next CSR stage to obtain the final diacritic decision.

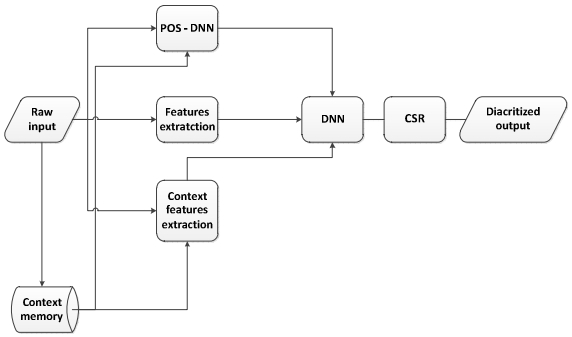

Figure 1 Overall Arabic diacritization system

4 Deep learning framework

The Arabic diacritics restoration task can be formulated as pattern classification problem. The target classes shall be the diacritics themselves, described in TABLE I. The input is the raw MSA transcript. The task is to classify the input based on well-designed features' vector and restore the original diacritics of the raw text. The output shall be the full diacritized text. All these diacritics can exist on case-ending character, while Fathten, Dammeten and Kasreten can never occur on non-ending character of the word root.

TABLE I ARABIC DIACRITICS CLASSES

Diacritics form on Arabic letter	Class name	Pronunciation
بَ	Fatha فتحة	/a/
بُ	Damma ضمة	/u/
بِ	Kasra كسرة	/i/
بًا	Fathten فتحتين	/an/
بٌ	Dammeten ضمتين	/un/
بٍ	Kasreten كسرتين	/in/
بْ	Sukun سكون	No vowel
بّ	Shadda شدّة	Double consonant

The machine learning classifier tool chosen in this paper is the Deep Neural Network (DNN), under the framework of learning deep architecture proposed by Hinton et al. (2006). The raw text is presented to the classifier, and a group of sub-nets work to extract the desired features, like PoS tags. The network architecture is shown in Figure 2. Each sub-net is trained to extract a certain kind of features, and the obtained features' vectors are concatenated together to form the input that is represented to the classifier network. In fact the training of features extraction nets is guided by certain desired features, like PoS tags.

This enables building a standalone system that operates on the raw text only.

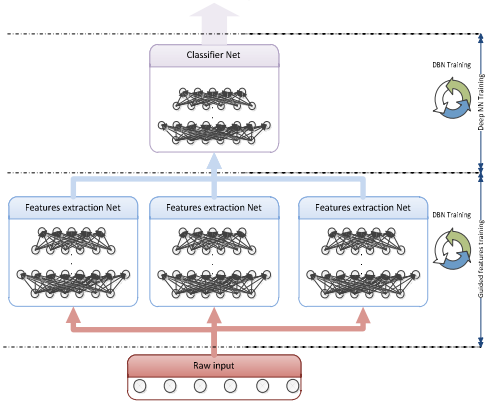

Figure 2 Deep network framework

5 Confused sub-set resolution method

The Confused Sub-Classes Resolution (CSR) is based on confusion matrix analysis and the method by Raafat and Rashwan (1993). The output of this analysis shall be a network architecture composed of the original classifier operating with sub-classifiers to resolve confusions that were identified through confusion matrix analysis.

The method starts with training a global classifier, then evaluating its performance. To enhance its accuracy, the sources of errors are analyzed by building the confusion matrix for the training set. The position of the off diagonal element identifies the pair of classes that are confused together.

5.1 Algorithm

The flow chart of the CSR method is shown in Figure 3. The following steps describe the algorithm:

1. Train a basic global classifier in DNN framework and obtain the confusion matrix C on the training set
2. Identify the confusion domains $D = \{D^i\}$ that have confusions more than a threshold δ, which is a parameter of the algorithm obtained from confusion matrix analysis. It can be set to the highest confusion figures in the off diagonal elements of the confusion matrix.
3. Train sub-classifiers for each confusion domain D^i.
4. Determine the architecture of the model having N^{nm} sub-classifiers. The superscript n denote the index in the layer, while m denotes the layer depth in which this domain is resolved. When a sub classifier is a complete subset of another one, it is placed in a deeper layer of the architecture. In this case, the m superscript is incremented to denote extra depth in the model.

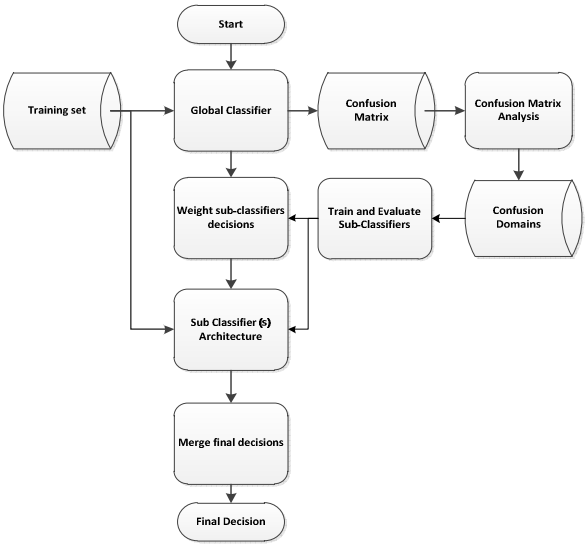

Figure 3 CSR algorithm flow chart

TABLE II shows the confusion results for DNN classifier (vertically: true, horizontally: predicted).

1. Fatha, Damma, Kasra: $D^{11} = \{4,5,6\} \rightarrow \mathrm{N}^{11}$
2. Fathten, Dammeten, Kasreten: $D^{21} = \{1,2,3\} \rightarrow \mathrm{N}^{21}$
3. Kasra, Kasreten: $D^{12} = \{3,6\} \rightarrow \mathrm{N}^{12}$

Each domain has its own N^{nm} classifier to resolve its confusion. The final model shall be as shown in Figure 4.

TABLE II CONFUSION MATRIX RESULTS FOR DNN CLASSIFIER ON SYNTACTIC DIACRITIZATION

	Fathten	Dammeten	Kasreten	Fatha	Damma	Kasra	Shadda	Sukkun
Fathten	4762	2179	2455	336	389	197	0	120
Dammeten	2647	6976	2720	660	1144	408	0	231
Kasreten	4560	3378	32588	801	303	**4868**	0	951
Fatha	438	475	1458	92755	11671	8340	0	1980
Damma	262	727	579	5858	72994	14995	0	952
Kasra	59	184	**3275**	2682	3657	220357	0	1970
Shadda	2	78	86	51	75	0	416	4
Sukkun	3	128	271	1150	630	1565	0	73983

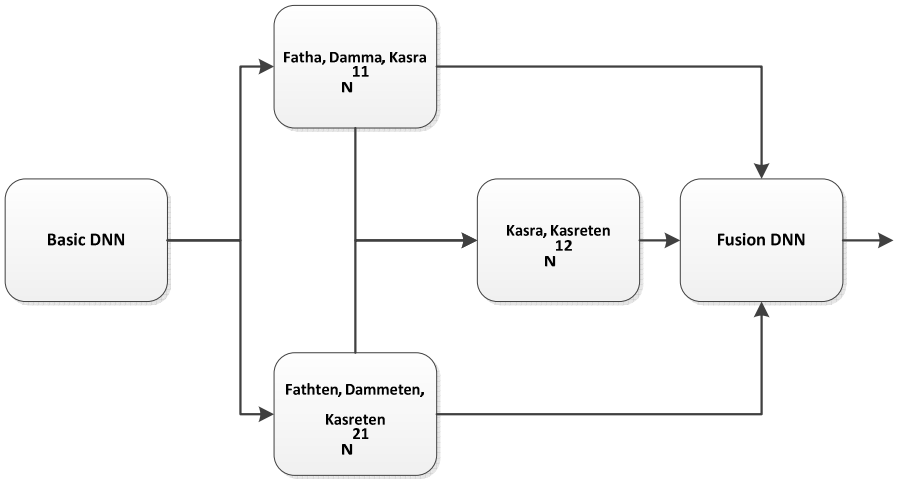

Figure 4 CSR model for syntactic Arabic diacritization task

6 Features

The input to text processing tasks is a transcript or document containing raw text. For Arabic diacritization task specifically, a set of features have proved good performance in literature, such as morphological lexemes, PoS, word identity,…etc see Rashwan et al. (2009, 2011), Zitouni et al. (2006) and Habash and Rambow (2007). In this section the features employed in our features vector are described.

Last character identity: case-ending diacritization is about adding diacritics on the last character of the word. Arabic language prohibits some diacritics from being placed over some characters. For example fatha on"و" is phonetically forbidden. Also, it favors some diacritics over some character like fatheten on "ا". A rule based system would have set a rule for that, however, in DNN framework, the system is left to learn such rules. Hence, the last character identity is an effective feature for syntactic diacritization task.

The raw word identity: is another type of possible features. The system proposed by Rasshwan et al. (2009, 2011) uses this feature. There are two possibilities of encoding such feature, the first would be to use a raw index representing the word index from a vocabulary vector built from training set. However, this could lead to many out of vocabulary (OOV) cases, in addition to long vector. On the other hand, a word can be encoded as sequence of the identities of its composing characters, which is more efficient under the DNN framework to avoid OOV, because even if a word is not encountered during training, at least a similar one with a character less or more was encountered during training, generating nearly similar activation of the stochastic binary units and leading to similar result as the original word. The same exact word need not be present during training phase, instead only a similar word is enough so that the word is not considered OOV. This is a direct result of encoding words as sequence of their constituting characters.

Context features: Considering the features vector of the preceding and/or succeeding words or characters can improve significantly the classification accuracy. This is what we refer to as context features. Context features are essential to syntactic and morphological diacritization tasks. For morphological diacritization context is just the surrounding characters, while for syntactic diacritization context is represented by the sur-

rounding words. We denote the depth of the preceding context by N and the succeeding context elements by M.

Context class labels: The context does not only include the features' vectors of inputs, it can also include the context of class labels. For example, the decision of the classifier for the previous diacritic can be re-considered as an input feature for the current diacritic classification. This results in something like an auto-regressive model, where the previous decisions affect the next one recursively

Part-of-Speech tags: are essential features to discriminate syntactic diacritics cases, where syntactic diacritics restoration is strongly related to grammatically parsing and analyzing the sentence into its syntactic language units or PoS. There are many models for Arabic PoS tags. In this work we adopt the one in Rashwan et al. (2011), which sets 62 context-free atomic units to represent all possible Arabic language PoS tags. A very rich dataset of Arabic words, extracted from different sources, is used to train the system (available on http://www.RDI-eg.com/RDI/TrainingData is where to download TRN_DB_II). PoS tags are manually annotated for this dataset by expert Arabic linguistics. A DNN is trained on this dataset to identify different PoS tags.

7 Datasets

In all the coming experiments one of the following datasets is used:

- ***TRN_DB_I:*** This is a 750,000 words dataset, collected from different sources and manually annotated by expert linguistics with every word PoS and Morphological quadruples.
- ***TRN_DB_II:*** This is 2500,000 words train set.
- ***TST_DB:*** This is 11,000 words test data set. For more information refer to Rashwan et al. (2009, 2011).
- ***ATB:*** LDC Arabic Tree Bank.

For TRN_DB_I, PoS tags are available as ready features added manually. When the manually PoS tags are used as input features, the dataset is referred to as TRN_DB_I – Ready PoS. While, when our PoS-DNN nets are used, a derivative dataset with only raw text is referred as TRN_DB_I – Raw text.

8 System evaluation

8.1 Effect of CSR method

The objective of this experiment is to show the effect of CSR method. The test set is TST_DB. Results in TABLE III show improvement around 2% in all tested datasets. This represents 17.09% improvement of error.

TABLE III EFFECT OF CSR

Dataset	Accuracy with CSR (%)	Accuracy without CSR (%)
TRN_DB_I – Ready PoS	90.2	88.2
TRN_DB_I – Raw text	88.2	86.2

8.2 Effect of class context learning

The objective of this experiment is to evaluate the effect of employing sequential class labels model. Test set is TST_DB. The results in TABLE IV show that employing this feature offers 1% to 2% improvement of accuracy over basic DBN model alone. This represents 15.625% improvement of error.

TABLE IV EFFECT OF CLASS LABELS CONTEXT ON SYNTACTIC DIACRITIZATION

Dataset	Accuracy with class labels context (%)	Accuracy without class labels context (%)
TRN_DB_I – Ready PoS	88.3	87.2
TRN_DB_I – Raw text	86.7	85.1
TRN_DB_I + TRN_DB_II / TST_DB	86.3	84.3

8.3 Effect of last character feature for syntactic case

The identity of the last character of a word is a critical feature for syntactic diacritization task. The dataset used for training is TRN_DB_I and for testing TST_DB. TABLE V shows the effect of utilizing this feature. A significant error improvement of about 4% is witnessed with this new feature.

TABLE V EFFECT OF LAST CHARACTER FEATURE ON SYNTACTIC DIACRITIZATION

	Accuracy (%)
With last character	88.2
Without last character	84.5

Justification to this strong improvement is that; Arabic language prohibits some diacritics from being placed over some characters. For example fatha on"و" is prohibited phonetically. Also, it favors some diacritics over some character like fatheten on "ا". A rule based system would have set a rule for that, however, in DNN framework, the system is left to learn such rules.

8.4 Effect of character level encoding of the word

The word identity is an important feature for diacritization task. The dataset used for training is TRN_DB_I and for testing TST_DB. TABLE VI shows the effect of utilizing this feature. A significant error improvement of about 2% is witnessed with this feature.

TABLE VI EFFECT OF CHARACTER LEVEL WORD ENCODING ON SYNTACTIC DIACRITIZATION

Encoding	Accuracy (%)
Word level	88.2
Character level	86.3

"Word level" could lead to many out of vocabulary (OOV) cases, in addition to long vector. On the other hand, "Character level" is more efficient under the DNN framework to avoid OOV suffered in Rashwan et al. (2009, 2011), because even if a word is not encountered during training, but a similar one with a character less or more was encountered, then a nearly similar activation of the stochastic binary units would be generated, leading to similar result to the most similar word existing in training data set.

8.5 Comparison to other systems

The objective of this experiment is to evaluate the performance of the proposed system for Arabic diacritization versus the architecture in Rashwan et al. (2009, 2011)., the MaxEnt model proposed in Zitouni et al. (2006) and the MADA system Habash and Rambow (2007). These systems represent the state of the art Arabic diacritization systems, with the best reported accuracy in literature. The evaluation was done on all the datasets as explained in Rashwan et al. (2011). The PoS features are extracted using the DNN-PoS tagger, since TRN_DB_II / TST_DB dataset contains only raw text without ready PoS features.

Results in TABLE VIII show that the proposed system achieves improved performance by around 1.2% over the system in 0Rashwan et al. (2011), which represents 9.23% of the error, evaluated on the (TRN_DB_I + TRN_DB_II / TST_DB) dataset. Also, on ATB standard dataset, the proposed system achieves 0.9% improvement over the best result in literature using the same training and testing data same as evaluation in Rashwan et al. (2011) was done.

Another comparison is done when the dataset TRN_DB_I is used with ready PoS features. Results in show that the proposed system achieves better performance by 3.2% over the system in Rashwan et al. (2011), which represents 24.6% of the error. The importance of this experiment is to isolate the automatic PoS tagging errors from the evaluation.

TABLE VII COMPARISON TO HYBRID ARCHITECTURE WITH READY PoS FEATURES

System	Syntactical accuracy (%)
Deep network + CSR	90.2
Hybrid Architecture 0Rashwan et al. (2011)	88.3

TABLE VIII COMPARISON TO OTHER SYSTEMS

System	Dataset	Case-ending accuracy (%)	Morphological accuracy (%)
Deep network + CSR (This paper)	TRN_DB_I + TRN_DB_II / TST_DB	**88.2**	**97**
	ATB	**88.4**	**97**
Hybrid Architecture – Rashwan et al. (2009, 2011)	TRN_DB_I + TRN_DB_II / TST_DB	87	96.4
	ATB	87.5	96.2
MaxEnt - Zitouni et al. (2006)	ATB	82	94.5
MADA - Habash and Rambow (2007)	ATB	85.1	95.2

9 Conclusion

In this paper the problem of Arabic diacritization restoration is tackled under the deep learning framework taking advantage of DBN model training. As part of the proposed deep system, a PoS tagger for Arabic transcript is proposed as well using deep networks. The first contribution is the introduction of the Confused Sub-set Resolution (CSR) architecture to enhance the accuracy.

Design of features vector played a key role in error improvement. Specifically, using features like last character identity had valuable contribution to error improvement by about 4%. Including class labels context features in auto-regressive fashion has also good impact of 1.1% on error improvement. Finally, encoding of word as sequence of characters enables to reduce OOV cases and enhance the accuracy.

CSR enables to purify the cross confusions between diacritics. A network-of-network architecture formed of group of classifiers, each working to resolve a set of confusions, is directly generated to enhance the overall accuracy by about 2%. The identified confusions and the architecture go smoothly with the grammatical and syntactical rules of the Arabic language.

Evaluation of the proposed system is made on two different datasets; custom and standard, both available online to enable replicating the experiments. Details of features vectors formatting and the used features are presented to facilitate results re-generation. The standard LDC Arabic Tree Bank dataset is used to bench mark the system against the best three systems in literature, showing that our system outperforms all previously published baselines. The effect of each proposed method is presented separately. Results show improvements ranging from 1.2% to 2.8% over the best reported results representing 22% improvement of the error.

Reference

Rashwan, Mohsen A A; Attia, Mohamed; Abdou, Sherif M.; Abdou, S.; Rafea, Ahmed A. 2009. “A Hybrid System for Automatic Arabic Diacritization”, International Conference on Natural Language Processing and Knowledge Engineering, pp 1-8, 24-27.

Rashwan, Mohsen A A , Al-Badrashiny, Mohamed A S A A; Attia, Mohamed; Abdou, Sherif M.; Rafea, Ahmed A. 2011. “A Stochastic Arabic Diacritizer Based on a Hybrid of Factorized and Unfactorized Textual Features”, IEEE Transactions on Audio, Speech, and Language Processing, vol. 19, issue 1, pp 166-175.

I. Zitouni; J. S. Sorensen; R. Sarikaya, 2006. “Maximum Entropy Based Restoration of Arabic Diacritics”, Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (ACL); Workshop on Computational Approaches to Semitic Languages; Sydney-Australia

N. Habash; O. Rambo. 2007. “Arabic Diacritization through Full Morphological Tagging”, Proceedings of the 8th Meeting of the North American Chapter of the Association for Computational Linguistics (ACL); Human Language Technologies Conference (HLT-NAACL).

G. E. Hinton; S. Osindero; Y. Teh, “A fast learning algorithm for deep belief nets” Neural Computation, vol. 18, pp. 1527–1554, 2006.

Ruslan Salakhutdinov. 2009.“Learning Deep Generative Models” PhD thesis, Graduate Department of Computer Science, University of Toronto,

Raafat, H.; Rashwan, M.A.A. 1993. “A tree structured neural network, Proceedings of the Second International Conference on Document Analysis and Recognition” , pp. 939 – 941, ISBN: 0-8186-4960-7

Hai-Son Le ; Oparin, I. ; Allauzen, A. ; Gauvain, J., 2011. “Structured Output Layer neural network language model” 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 5524 - 5527, ISBN: 978-1-4577-0537-3

http://www.RDI-eg.com/RDI/TrainingData is where to download TRN_DB_II.

http://www.RDI-eg.com/RDI/TestData is where to download TST_DB

LDC Arabic Tree Bank Part 3, http://www.ldc.upenn.edu/Catalog/catalogEntry.jsp?catalogId=LDC2005T20