

On Stochastic Models, Statistical Disambiguation, and Applications on Arabic NLP Problems

Mohamed Attia⁽ⁱ⁾

Research & Development Int'l; RDI

Mohsen A. A. Rashwan⁽ⁱⁱ⁾

Cairo University

Co-Author

Galaal K. Galaal⁽ⁱⁱⁱ⁾

Research & Development Int'l; RDI

ABSTRACT

In this paper, the need for practically solving natural language problems via statistical methods besides the (incomplete) classical closed form ones is manifested and formalized. Also, the statistical features of natural language are extensively discussed. One simple yet effective work frame of this type of processing that relies on long n -grams probability estimation plus powerful and efficient tree search algorithms is presented in detail. Finally, the authors exemplify the above ideas through a practical Arabic text diacritizer (hence a phonetic transcriber for Arabic speech applications especially TTS) that has been built according to that work frame

KEYWORDS

Arabic NLP, Arabic TTS, Arabic, Computational Linguistics, Diacritization, HLT, Human Language Technologies, Linguistics, Natural Language Processing, NLP, Phonetic Transcription, Vowelization, Speech, Statistical Disambiguation, Statistical Modeling, Stochastic Modeling, Text-to-Speech, TTS.

NLP, Stochastic Models, & Statistical Disambiguation

A MAP APPROACH FOR SOLVING NLP PROBLEMS

Natural languages refer to all forms of symbolic messages used in human communication. The language process starts in the brain when the human speaker, writer, etc. composes the message to be delivered to his/her intended listener(s), reader(s), etc. respectively. This message transfer process manifests the communicative nature of the natural languages phenomenon. The ultimate goal of this process is to recognize the message as intended by analyzing the message as observed. This is illustrated by the communication model shown below [11].

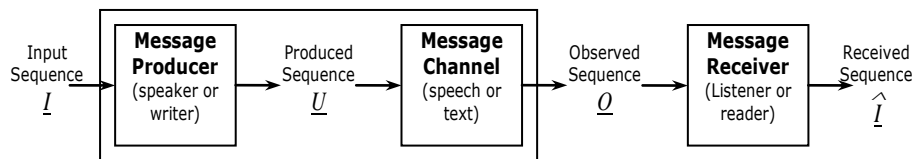


Figure 1: The message processing stages with \hat{I} (the intended pattern that was input by the producer) and \hat{O} (the observed pattern that was output by the producer)

Indeed, the human observer analyzes the messages collectively from the finest linguistic details to the level of their pragmatics. If the observer was a computerized analysis process, this process should do the same to capture the same analytic capability. In fact this is a tough requirement due to our partial ignorance of the laws that govern natural languages and their dynamics! This is why natural languages are studied and practically approached in the light of two models—*formal languages* model and the *stochastic* model.

⁽ⁱ⁾ Muhammad Attiyya is a PhD student in the department of Electronics and Electrical Engineering, Faculty of Engineering, Cairo University. He can be reached at m_Atteya@rdi-eg.com

⁽ⁱⁱ⁾ Dr. Muhsen Rashwan is a professor in the department of Electronics and Electrical Engineering, Faculty of Engineering, Cairo University. He can be reached at mRashwan@rdi-eg.com

⁽ⁱⁱⁱ⁾ Galaal K. Galaal is a software engineer in the Arabic NLP team of RDI. He can be reached at Galal@rdi-eg.com

Formal languages model is addressed by two fundamental fieldworks—the language rules (sometimes referred to as the *grammar*) which is the specification of the permissible language structures, and the parsing process that seizes the compliance of some linguistic structures to the rules [7]. This typically leaves us with the problem of having multiple possible perceived sequences from which we should recognize the intended message sequence. The task of the stochastic model is electing the optimal message sequence probabilistically. The elected sequence \hat{I} can then be stated as

$$\hat{I} = \arg \max_{\forall I} \{P(I | O)\} \quad \text{Eq. 1}$$

The probability of which \hat{I} is elected is known as the *maximum a posteriori probability* or MAP which is the approach typically followed in the work of pattern recognition. The estimation of $P(I | O)$ has a profound basis in modern control theory which states that the longer a system is observed the better its behavior is captured. $P(I|O)$ is hard to be directly computed due to the lack of control over the output sequences. Instead, we make use of Bayes' rule and put

$$\hat{I} = \arg \max_{\forall I} \{P(I | O)\} = \arg \max_{\forall I} \left\{ \frac{P(O | I) \cdot P(I)}{P(O)} \right\} \quad \text{Eq. 2}$$

$P(O)$ is the independent probability of the output and is independent of the maximization function. This simplifies Eq. 2 to

$$\hat{I} = \arg \max_{\forall I} \{P(O | I) \cdot P(I)\} \quad \text{Eq. 3}$$

In other pattern recognition problems like OCR and automatic speech recognition (ASR), the term $P(O | I)$, which is referred to as the likelihood probability, is modeled via probability distributions; e.g., HMM in ASR. In NLP problems, we can do better by viewing the formal available structures, in terms of probabilities, as a binary decision; i.e., a decision of whether the observation obeys the formal rules or not. This simplifies the equation to

$$\hat{I} = \arg \max_{\forall I \in \mathfrak{R}} \{P(I)\} \quad \text{Eq. 4}$$

where \mathfrak{R} is the formal model space. $P(I)$ is the independent probability of the input which is called the statistical language model or **SLM**. $P(I)$ is then what is computed from the training corpus.

STOCHASTIC LANGUAGE MODELS

An SLM is a statistical model for estimating the probability $P(\underline{W})$ for a given linguistic sequence of entities¹

$$\underline{W} \equiv w_1 w_2 \dots w_L \equiv w_1^L$$

More likely sequences typically have higher probabilities. Finding such adequate information would provide wide language coverage and dramatically constrain the search space for higher language layers. The n -gram model is a well known SLM for its relative simplicity and employed reduction of the problem complexity.

THE LANGUAGE PHENOMENON

Let us count the occurrences (frequencies) of single entities $\alpha(\underline{W} = w)$ (composing the vocabulary of a language layer of size V) in some large fair-text corpus of size N , that the assemblage sequence is as simple as a single entity, and let us rank the entities by their frequencies in the corpus. We observe the following:

1. Some part, say of size V' , of the vocabulary is only covered in the corpus, that is

$$N = \sum_{r=1}^V \alpha(\underline{W} = w_r) = \sum_{r=1}^{V'} \alpha(\underline{W} = w_r), \quad V' \leq V \quad \text{Eq. 5}$$

Where r is the rank of entity w_r . We hence define the coverage of the corpus to be

¹ This notation will be used throughout this paper.

$$v = V'/V \quad \text{Eq. 6}$$

2. The rank-frequency relationship roughly follows a hyperbolic curve² as shown in Figure 2. This curve draws an essential understanding to the language use. Zipf, in his principle of least effort, states that the best compromise of both efforts of speaker and listener while communication results in a use of a few very frequent entities and many low-frequency ones³. Also, as a consequence, the coverage increases sluggishly as the corpus gets larger⁴.

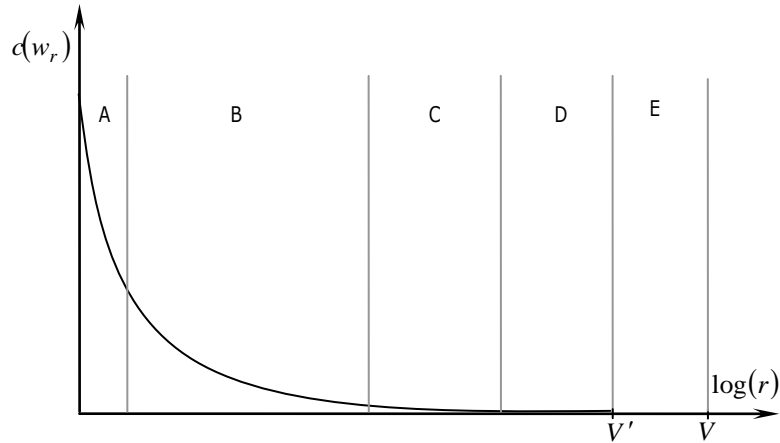


Figure 2: Ranking entities by their frequencies. Curve is coarsely considered as of 5 regions. (A) is the stop list, (B) is the frequent list, (C) is the rare list, (D) is the singletons (read-only-once) list, and (E) is the unattested list.

Empirically from the corpus, we have

$$P(\underline{W} = w_r) = \frac{c(\underline{W} = w_r)}{N} \quad \text{Eq. 7}$$

which yields zero for the unattested entities (uncovered part of the vocabulary) of size $V - V'$. We know that this is incorrect since a better coverage would take place if the corpus is appended some new text. Also for small $c(\underline{W})$ values, $P(\underline{W})$ would be significantly inaccurate⁵ [9],[16]. A solution to this problem is addressed by the so-called *probability estimation smoothing techniques* presented later in this paper.

Ranking longer sequences (assemblages of two entities or more) turns out to apply the same phenomenon with greater ratios of uncovered, singletons (read-only-once), and low-frequency sequences, for the same corpus, which affects the model efficiency. An adequately larger corpus is then needed for achieving the same statistical representation as that of single entities. On the other hand, a very large corpus may be an overfit and adds unnecessary computation and storage cost. The dark areas in the figure below show some sense of adequacy between the sequence length (usually referred to as the gram size) and the corpus size⁶.

² Revise Mandelbrot's correction of the curve, "Structure Formelle des Textes et Communication", 1954.

³ Ref. 16. Zipf's own words on the study of English tokens: "There are a few very common words, a middling number of medium-frequency words, and many low-frequency words", "Human Behavior and the Principle of Least Effort", 1935.

⁴ Remember the math fact that $\int_{x_0}^{\infty} \frac{dx}{x} = \ln(x)|_{x_0}^{\infty} = \infty$

⁵ $P(\underline{W}) \equiv \lim_{\substack{c(\underline{W}) \rightarrow \infty \\ N \rightarrow \infty}} \frac{c(\underline{W})}{N}$ [9]

⁶ Covering V is unnecessary and for unigrams it needs N to be of $O(V \ln V)$ from Zipf's curve, thus taken safely $N=V^2$ as an upper bound. Also, taking $N=V$ introduces considerable chances that some highly expected entities do not occur, so taken safely as a lower bound. For higher grams, combinations are needed and the rest of the curve is easily deduced.

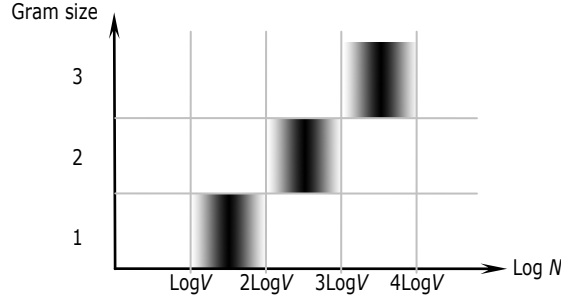


Figure 3: Adequacy regions (in black) of selecting corpus size according to gram size given a fair text.

N-GRAM LANGUAGE MODELS

The sequence \underline{W} of length L has the probability of occurrence $P(\underline{W})$ that can be decomposed using the chain rule as follows [7], [16]:

$$\begin{aligned}
 P(\underline{W}) &\equiv P(w_1^L) \\
 &= P(w_1)P(w_2 | w_1) \dots P(w_L | w_1^{L-1}) \\
 &\equiv \prod_{i=1}^L P(w_i | w_1^{i-1})
 \end{aligned} \tag{Eq. 8}$$

where $P(w_i | w_1^{i-1})$ is the probability that entity w_i will follow the sequence w_1^{i-1} . This resultant conditional form of the previous transformation has a remarkable advantage over the marginal one we started with. Natural languages are said to be a *wide horizon phenomenon* since w_i is bonded to earlier and later entities over the whole sequence L . Yet the expression elegantly dissects the marginal a priori term into sequences mostly shorter than L with causal conditional probabilities estimated from earlier entities only.

For a vocabulary of size V , there will be more than V^i possible different histories to specify $P(w_i | w_1^{i-1})$ completely. Indeed, this kind of estimation is infeasible even for moderate values of i . It is intuitive that, as text of natural language, the present entity w_i mostly has the strongest correlation with its closest past neighbor w_{i-1} and weaker correlations with earlier ones. This is called the *attenuating correlation phenomenon*⁷. We can then approximately assume that $P(w_i | w_1^{i-1})$ depends on some *equivalence class* only, and simply on some previous entities w_{i-h}^{i-1} . This approximation leads to an n -gram model with

$$n = h + 1 \tag{Eq. 9}$$

and h is the history length of the sequence w_{i-h}^{i-1} . Thus

$$P(\underline{W}) \cong \prod_{i=1}^L P(w_i | w_{i-h}^{i-1}) \tag{Eq. 10}$$

⁷ Consider that the human has a limited capacity of memorization and composition that consequently the elements of his/her speech cannot be kept bonded forever.

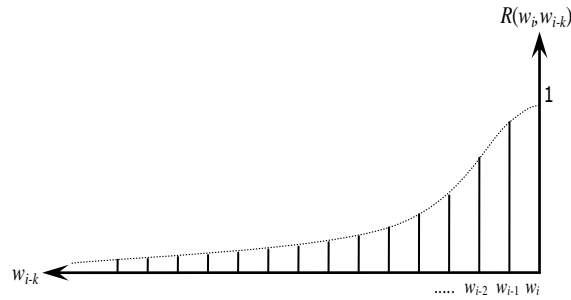


Figure 4: The attenuating correlation phenomenon. The figure displays the correlation between w_i and other previous entities.

Hence, for a unigram ($n=1$:no history) this approximation would yield for example

$$P(w_1 w_2 w_3 w_4) \cong P(w_1)P(w_2)P(w_3)P(w_4)$$

and for bigrams ($n=2$: $h=1$)

$$P(w_1 w_2 w_3 w_4) \cong P(w_1 | \langle s \rangle)P(w_2 | w_1)P(w_3 | w_2)P(w_4 | w_3)P(\langle / s \rangle | w_4)$$

where $\langle s \rangle$ and $\langle / s \rangle$ denote a sequence start and end.

Also for trigrams ($n=3$: $h=2$)

$$P(w_1 w_2 w_3 w_4) \cong P(w_1 | \langle s \rangle)P(w_2 | \langle s \rangle w_1)P(w_3 | w_1 w_2)P(w_4 | w_2 w_3)P(\langle / s \rangle | w_3 w_4)$$

Recall that the above approximation is valid only under the preliminary assumption that the process is random; i.e., the incoming entity w_i is grammatically implied by neither the preceding entities nor the succeeding ones.

DISAMBIGUATION

Each entity of an output sequence \underline{W} has one or more linguistic analyses defined by the language process in hand as illustrated in Figure 5 where a trellis of length equal to the sequence length and of heights corresponding to the possible analyses of each entity.

| | | | | |
|----------------------|----------------------|---------|----------------------|--------------------|
| w_1 | w_2 | \dots | w_L | Output Sequence |
| $q_{1,1} \bullet$ | $q_{2,1} \bullet$ | \dots | $q_{L,1} \bullet$ | |
| $q_{1,2} \bullet$ | $q_{2,2} \bullet$ | \dots | $q_{L,2} \bullet$ | |
| \vdots | \vdots | | \vdots | Solution nodes |
| $q_{1, j_1} \bullet$ | $q_{2, j_2} \bullet$ | \dots | $q_{L, j_L} \bullet$ | |
| \underline{q}_1 | \underline{q}_2 | \dots | \underline{q}_L | |

Figure 5: The input sequence \underline{W} and the solution trellis of analyses $(\underline{q}_1, \underline{q}_2, \dots, \underline{q}_L)$.

Disambiguation is the task of selecting the unique pragmatically correct analysis among the legitimate ones. One then would have to select the best analyses over the whole sequence; one for each entity. This sequence is the best sequence in the sense that it is most relevant to the training corpus. The space of solutions \mathbf{S} to search through is the resultant Décart multiplication

$$\mathbf{S} = \underline{q}_1 \times \underline{q}_2 \times \dots \times \underline{q}_L \quad \text{Eq. 11}$$

and the best sequence can be expressed as

$$\underline{\hat{Q}} \equiv q_{1, \hat{j}_1} q_{2, \hat{j}_2} \dots q_{L, \hat{j}_L} \equiv q_{1, \hat{j}_1}^{L, \hat{j}_L}, \quad 1 \leq \hat{j}_i \leq J_i \quad \text{Eq. 12}$$

Searching for the whole sequence is faced with the following difficulties:

- L is typically long (may extend up to few dozens).
- L is variable for each query sequence.
- J_i , the number of legitimate solutions for entity w_i , is variable for each column i .
- J_{avg} , the geometric mean number of possible solutions per entity⁸, may be large.
- V is large⁹ and consequently V' .
- Vocabulary entities typically occur sparsely.

When cleverly employed, some overcomes can contribute remarkably against the above difficulties as follows:

- Using the n -gram model sets the dependency of the process on n instead of L . This limits the inherent complexity of such problem from $O(J_{avg}^L)$ to $O(J_{avg}^n)$.
- For a context-dependent analysis the search space is constrained by well-defined linguistic rules, thus reducing the legitimate solutions for an entity w_i from J_i to some $k_i J_i$ where k_i is typically less than or equal to one. The average trellis height is then reduced from J_{avg} to $k_{avg} J_{avg}$ which is a direct reduction in the branching factor ahead of each node and therefore a dramatic reduction in the search space. Complexity is then reduced to $O(k_{avg}^n J_{avg}^n)$. Another reduction in the branching factor could be a primary selection of a ratio $r < 1$ of strongly competing candidates among all the candidates residing in column i according to some criteria which reduces complexity to $O(r_{avg}^n k_{avg}^n J_{avg}^n)$.
- Accommodating the storage needed for the occurrences of the n -gram-and-shorter entities could be taken naively of $O(V^n)$. Since data is sparse, some (in fact most) entities have no occurrence and some have useless small occurrences that could be rejected from an adequate storage.

MAXIMUM LIKELIHOOD ESTIMATION CRITERION

The approach of maximum-likelihood estimation (MLE) for determining the best sequence $\underline{\hat{Q}}$ is intuitive, straight forward, and can show desirable performance provided that an adequate smoothing technique is employed. Hence the sequence $\underline{\hat{Q}}$ is required to satisfy the following criterion:

$$P(\underline{\hat{Q}}) > P(\underline{Q}) \quad \forall \underline{Q} \neq \underline{\hat{Q}} \quad \text{Eq. 13}$$

Using Eq. 8, Eq. 10, and Eq. 13 we can write:

⁸ $J_{avg} = \left(\prod_{i=1}^L J_i \right)^{1/L}$

⁹ For English words V is of $O(10^6)$, and for news sample corpus of size of $O(10^5)$, V' is some $O(10^4)$.

$$\begin{aligned}
\hat{Q} &= \arg \max_{\mathbf{s}} \{P(q_{1,j_1}^{L,j_L})\} \\
&= \arg \max_{\mathbf{s}} \left\{ \prod_{i=1}^L P(q_{i,j_i} \mid q_{(i-h),j_{(i-h)}}^{(i-1),j_{(i-1)}}) \right\} \\
&= \arg \max_{\mathbf{s}} \left\{ \sum_{i=1}^L \log P(q_{i,j_i} \mid q_{(i-h),j_{(i-h)}}^{(i-1),j_{(i-1)}}) \right\}
\end{aligned} \tag{Eq. 14}$$

The probability term in the previous summation can be computed directly from the corpus database

$$P(q_{i,j_i} \mid q_{(i-h),j_{(i-h)}}^{(i-1),j_{(i-1)}}) \cong \frac{c(q_{(i-h),j_{(i-h)}}^{i,j_i})}{c(q_{(i-h),j_{(i-h)}}^{(i-1),j_{(i-1)}})} \tag{Eq. 15}$$

provided that both counts in the numerator and denominator in the RHS of this equation as well as both N_n (the number of n -grams in the corpus) and N_{n-1} (the number of $(n-1)$ -grams in the corpus) are large enough. Otherwise this probability should be calculated using the smoothing technique described in the section below..

"BAYES', TURING-GOOD DISCOUNT, BACKOFF" PROBABILITY ESTIMATION SMOOTHING TECHNIQUE

Any entity in the language vocabulary must have usage in some context, though it seems endless to enlarge some corpus to cover all entities. The curve in Figure 2 will always, in general, have the regions shown, only with different widths. Presumably, a curve without the uncovered set is smoother than the one shown. The process of biasing the uncovered set on the expense of discounting the other regions is called smoothing. A disambiguation system that doesn't employ smoothing would refuse the correct solution if any of its input entities was unattested in the training corpus and consequently yield a sub-optimal solution.

One of the effective techniques, namely Bayes, Turing-Good Discount, Back-Off, is briefed below¹⁰.

The run-time phase (Estimating the m -gram conditional probabilities)

$$\begin{aligned}
&\text{if } \alpha(w_1^m) > k_1, \\
&\text{Apply Bayes' rule} \\
&\{ \\
&\quad \text{if } m > 1, P(w_m \mid w_1^{m-1}) = \frac{\alpha(w_1^m)}{\alpha(w_1^{m-1})} \\
&\quad \text{if } m = 1, P(w_m \mid w_1^{m-1}) = \frac{\alpha(w_1^1)}{N} \\
&\} \\
&\text{If } (m = 1 \text{ AND } \alpha(w_1^m) \leq k_1) \text{ OR } (m > 1 \text{ AND } k_2 < \alpha(w_1^m) \leq k_1), \\
&\text{Apply Turing-Good discount} \\
&\{ \\
&\quad c^*(w_1^m) = \frac{(\alpha(w_1^m) + 1) \cdot n_{\alpha(w_1^m)+1,m}}{n_{\alpha(w_1^m),m}} \\
&\quad d_{\alpha(w_1^m)} = \frac{\frac{c^*(w_1^m)}{\alpha(w_1^m)} - \delta_m}{1 - \delta_m} \\
&\quad \text{if } m > 1, P(w_m \mid w_1^{m-1}) = d_{\alpha(w_1^m)} \cdot \frac{\alpha(w_1^m)}{\alpha(w_1^{m-1})} \\
&\quad \text{if } m = 1 \\
&\quad \{
\end{aligned} \tag{Eq. set 16}$$

¹⁰ This brief is quoted without modification from M. Atteya's master thesis, "A Large-Scale Computational Processor of Arabic Morphology, and Applications", published in the Cairo University Library, January 2000, also found on <http://www.nemlar.org/Publications/index.htm>. In this thesis the "Bayes, Turing-Good Discount, Back-Off" technique is explained. The mathematical derivation of this technique is presented in [10],[12].

$$\text{if } \alpha(w_1^m) \geq k_2, P(w_m | w_1^{m-1}) = d_{\alpha(w_1^m)} \cdot \frac{\alpha(w_1^1)}{N}$$

$$\text{if } \alpha(w_1^1) < k_2, P(w_m | w_1^{m-1}) = \frac{n_{1,1} / n_{0,1}}{N}$$

Eq. set 17

}
 if $(\alpha(w_1^m) \leq k_2)$ AND $(m > 1)$,
 Apply the back-off recursive procedure

$$\{$$

$$\text{if } \alpha(w_1^{m-1}) > k_2$$

$$\{$$

$$\text{if } m > 2, P(w_m | w_1^{m-1}) = \alpha(w_1^{m-1}) \cdot P(w_m | w_2^{m-1})$$

$$\text{if } m = 2, P(w_m | w_1^{m-1}) = \alpha(w_1^1) \cdot P(w_2)$$

$$\}$$

$$\text{if } \alpha(w_1^{m-1}) \leq k_2$$

$$\{$$

$$\text{if } m > 2, P(w_m | w_1^{m-1}) = P(w_m | w_2^{m-1})$$

$$\text{if } m = 2, P(w_m | w_1^{m-1}) = P(w_2)$$

$$\}$$

$$\}$$

Eq. set 18

k_1 & k_2 are constants with the typical values $k_1=5$ & $k_2=1$.
 N =Total number of the occurrences of monograms.
 $n_{r,m}$ =number of m -grams that occurred exactly r times.

The off-line phase (Building the statistical knowledge base)

1. Build the m -grams w_1^m and their counts $\alpha(w_1^m)$ for all $1 \leq m \leq M$.
2. Get the counts $n_{r,m}$ for all $1 \leq r \leq k_1 + 1$ and $1 \leq m \leq M$.
3. Compute

$$\delta_m = \frac{(k_1 + 1) \cdot n_{k_1+1,m}}{n_{1,m}}; 1 \leq m \leq M.$$

Eq. 19

4. Sort all the m -grams in ascending order using the quick-sort algorithm for later fast access using binary search.
5. Compute the α parameters as:

$$\alpha(w_1^{m-1}) = \frac{1 - \sum_{w_m: \alpha(w_1^m) > 0} P(w_m | w_1^{m-1})}{1 - \sum_{w_m: \alpha(w_1^m) > 0} P(w_m | w_2^{m-1})}; m > 1$$

Eq. 20

6. Discard w_1^m and the corresponding $\alpha(w_1^{m-1})$ for which $\alpha(w_1^m) \leq k_2$.

A* SEARCH-BASED DISAMBIGUATION TECHNIQUE

Searching for the best path (the best sequence of analyses) in the sense of Eq. 14 through the solution space \mathbf{S} illustrated in Figure 5, can easily be perceived as a tree search problem. This tree has a root node $\langle /s \rangle$ which spawns the nodes $q_{1,1}, q_{1,2}, \dots, q_{1,J_1}$. Each of these nodes spawns in turn the nodes $q_{2,1}, q_{2,2}, \dots, q_{2,J_2}$, and so on till a depth of L . Obviously, exhaustive search methods, such as breadth-first, through this tree is infeasible due to its exponential complexity; i.e., the number of expanded and examined paths is $O(J_{avg}^L)$.

Fortunately, the well known best-first strategy can be employed that a list of open nodes (those that are candidates of expansion on the next step) is maintained while the search, and the node to be expanded next is the one that maximizes some likelihood function $f(k, q_{k,j_k}, L)$. One intuitive formulation of this function is

$$f(k, q_{k,j_k}, L) = g(k, q_{k,j_k}) = \sum_{i=1}^k \log P(q_{i,j_i} | q_{(i-N+1), j_{(i-N+1)}}^{(i-1), j_{(i-1)}}) \quad \text{Eq. 21}$$

where N is the maximum order of n -grams used in the language model.

The significance of this formula is that we expand the open node whose path from the root node has a log maximum likelihood over the corresponding paths of all the open nodes. Ultimately when a node at depth level L is chosen for expansion, the search is over and the best (most likely) path is obtained. The virtue of this search technique is that while finding the best path is guaranteed (the admissibility property), visiting all the nodes of the tree is not necessary as is the case with exhaustive search. The ratio of unvisited nodes to the total ones, hence the search efficiency, depends heavily on how competitive are the nodes at each columns (i.e., the possible analyses of each entity) among each other. The closer are the monogram likelihood probabilities of the column nodes, the less efficient is the best-first search and vice versa. Said in other words; the best-first search gets more efficient with search problems when the number of dominant nodes in each column of the trellis gets smaller. Many NLP problems have this feature.

Despite the efficiency improvement of the best-first search over exhaustive ones, one can notice that best paths get fainter (have smaller log likelihood probabilities) at deeper levels than at shallower ones. Consequently, open nodes at shallower levels with unpromising paths may be wastefully elected for expansion only because their log likelihood probabilities are larger than those of open nodes with promising paths at deeper levels. In order to avoid this inefficiency; the likelihood function $f(k, q_{k,j_k}, L)$ should be reformulated to contain a term $h(k, q_{k,j_k}, L)$ of the log likelihood probability of the path from the node on the path next to the open one to its last node at level L as follows

$$f(k, q_{k,j_k}, L) = g(k, q_{k,j_k}) + h(k, q_{k,j_k}, L) \quad \text{Eq. 22}$$

The problem now is that while an open node is being examined, the best path from itself to the by-then unknown last node is simply unknown! So, instead of the unknown exact $h(k, j_k, L)$, an approximate estimate, called a heuristic, $h^*(k, j_k, L)$ is used so that

$$f^*(k, q_{k,j_k}, L) = g(k, q_{k,j_k}) + h^*(k, q_{k,j_k}, L) \quad \text{Eq. 23}$$

The choice of $h^*(k, q_{k,j_k}, L)$ must be done carefully. If the estimation is made with a negative error $h^*(k, q_{k,j_k}, L) - h(k, q_{k,j_k}, L)$, admissibility may be sacrificed; i.e., there is a possibility of terminating on a path whose likelihood is not the maximum. On the other hand, if the estimation error is positive, admissibility is maintained and the search efficiency depends on how small the error is. When $h^*(k, q_{k,j_k}, L) = \log(P = 1) = 0$ at the extreme, the case of Eq. 21 is approached where efficiency is lost.

The perfect case is when one can estimate $h^*(k, q_{k,j_k}, L)$ such that estimation error is zero. If we keep the estimation error as positively minimum as possible, admissibility is maintained and search efficiency is maximized. A neat proof of this statement is presented in chapter 5 of [Nilsson's] and the whole search technique is then called the A^* search algorithm.

While building the n -gram statistical language model, we can easily obtain the quantities $P_{\max,k}$ defined as:

$$P_{\max,k} = \max_{\forall w_1^k} P(w_1^k); \quad 1 \leq k \leq N \quad \text{Eq. 24}$$

then a safe (admissible) heuristic can be easily formulated as:

$$h^*(k, q_{k,j_k}, L) = \begin{cases} \sum_{i=k+1}^L \log(P_{\max,N}) = (L-k) \log(P_{\max,N}); & L \geq N, k \geq N-1 \\ \sum_{i=N}^L \log(P_{\max,N}) + \sum_{i=k+1}^{N-1} \log(P_{\max,i}) & L \geq N, k < N-1 \\ = (L-N+1) \log(P_{\max,N}) + \sum_{i=k+1}^{N-1} \log(P_{\max,i}); \\ \sum_{i=k+1}^L \log(P_{\max,i}); & L < N \end{cases}$$

Eq. 25

The A^* -algorithm is stated below.

A^* -Search algorithm

1. Initialization: Create a stack SS holding nodes of type $Q(\text{path } L, \text{ score } s)$ and push the root node (a fake node) with score $s = \log 1 = 0$ and empty path.
2. Pop up the surface node Q .
3. If length of path in Q is L , exit with the most likely path L .
4. Expand Q to nodes of next column with scores calculated from Eq. 21, Eq. 23, Eq. 24, and Eq. 25, and push them into SS .
5. Reorder SS in a descending order according to s field.
6. Go to step 2.

One of the advantages of this search process is that not only the 1st most likely path can be obtained, but also the 2nd, the 3rd, etc., and in general the m^{th} most likely path can be obtained using the following modification.

m^{th} most likely path A^* -Search algorithm

1. Initialization: Create a stack SS holding nodes of type $Q(\text{path } L, \text{ score } s)$ and push the root node (a fake node) with score $s = \log 1 = 0$ and empty path. Set a counter c to zero.
2. Pop up the surface node Q .
3. If length of path in Q is L , increment c .
4. If c is equal to m , exit.
4. Expand Q to nodes of next column with scores calculated from Eq. 21, Eq. 23, Eq. 24, and Eq. 25, and push them into SS .
5. Reorder SS in a descending order according to s field.
6. Go to step 2.

The number of nodes in the stack could increase quickly that it may become impractical to hold them all in. Fortunately, we know that only a few best-scoring nodes at any column will most probably contribute to the most likely path, hence deleting some least-scoring nodes from the stack is very unlikely to miss the most likely solution. This practical modification leads to the Beam search listed below.

Beam-Search algorithm

1. Initialization: Create a stack SS holding nodes of type $Q(\text{path } L, \text{ score } s)$ and push the root node (a fake node) with score $s = \log 1 = 0$ and empty path.
2. Pop up the surface node Q .
3. If length of path in Q is L , exit with the most likely path L .
4. Expand Q to nodes of next column with scores calculated from Eq. 21, Eq. 23, Eq. 24, and Eq. 25, and push them into SS .
5. Reorder SS in a descending order according to s field.
6. Truncate bottom nodes from SS according to some criteria R so that the number of remaining nodes equals the maximum allowable stack size (beam size).
7. Go to step 2.

Applications on the Arabic NLP Problems



JOSEPH, 12, 1-4

The wide range of the applications based on software digital speech processing technologies are growing to fuel multibillion dollars markets of telecommunications, mass media, desktop and mobile computing, security, etc. [7], [8], [17],[18]

Automatic phonetic transcription is one common infrastructure of numerous digital speech processing technologies such as text-to-speech, speech recognition, speech verification, speaker verification, etc. The problem of automatic phonetic transcription is a language dependent one. In specific; the problem of (reliable) Arabic phonetic transcription is a painfully tough one. This is true due to the following reasons:

1. Arabic text is typically written without diacritics. [2],[13]
2. Arabic text is frequently written with many common spelling mistakes. [2]
3. Look-up tables (or dictionaries) of phonetic transcription is not a feasible solution because Arabic is not a vocabulary oriented language, instead it is a highly derivative and inflective one. Phonetic transcription look-up tables would have to cover a prohibitively huge size of possible vocabulary words. On the other hand, inducing the diacritization (vowelization) hence the phonetic transcription of the given words through a full fledged morphological analyzer is a much better approach.
4. The morphological analyses of Arabic words are dramatically ambiguous; i.e., many possible morphological analyses are possible for the same given word. [2]
5. For about 60%¹¹ of the Arabic text; part of the phonetic transcription of the words is not statically dependent on their correct morphological analyses, but dynamically dependent on their correct syntactic analyses. What makes matters harder is the highly involved and ambiguous nature of Arabic grammar.
6. Like other languages, transliterated words (mostly foreign nouns) often exist in Arabic text. Arabic news domain text contains about 7.5% transliterated words¹¹. These words do not obey the Arabic morphology. All what is obeyed here is the Arabic phonetic grammar which leaves us again with a high degree of diacritization ambiguity.
7. Some transliterated words are analyzable as ordinary Arabic derivative ones, which adds another ambiguity problem of whether a given word should be morphologically analyzed (hence diacritized) as an ordinary Arabic word or be treated as just a transliterated one.
8. In text-to-speech systems, segmenting the text to be converted to speech into short phrases without ruining the meaning of the intended speech is a highly important feature. This feature is called prosodic phrasing whose absence leads to a dramatic deterioration of the resulting quality - especially the pitch contour - of the synthesized speech if long text is occasionally fed to the system. This adds one more ambiguity question of whether each word is followed by a pause or not.
9. All the aforementioned ambiguities can theoretically be solved combinatorially using the analysis at all the linguistic layers (see Figure 6) such as the semantic analysis for filtering away “none sense” solutions and exiting with the sole solution that corresponds to the intentions of the speaker or the writer. [4] Unfortunately, the current state of the art of computational linguistics has not grown powerful enough to let us produce reliable analyzers at the higher layers. Moreover, the forward and backward interactions between the successive language layers are still far from being clearly understood and modeled reliably.

¹¹ These statistics are extracted from a modern news domain Arabic corpus of size about 500,000 words. The author and his colleagues did these experiments in RDI's NLP labs.

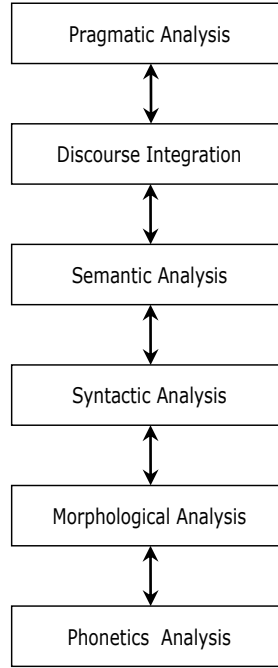


Figure 6: The language layers.

Consequently, producing a practically reliable automatic Arabic phonetic transcriber may employ formal (closed form and deterministic) language models for the well understood parts of the language phenomenon and substitutes the ill-understood parts with some statistical disambiguation technique such as the one we described earlier through this paper.

The author, with the help of other colleagues, have built a practical automatic Arabic diacritizer under the name ArabDiac© to support advanced Arabic speech applications; especially Arabic Text-to-Speech. The main components of that system are representative to the mixed rule-based/stochastic work frame discussed in this paper as follows:

RULE BASED COMPONENTS

1. Arabic morphological analyzer that produces a set of possible analyses per each input string w .

$$\underline{q} = q_1 q_2 \dots q_J; \quad J \geq 0$$

Each possible analysis has the same canonical structure

$$\underline{q} = (t : p, r, f, s)$$

where t is TypeCode, p is PrefixCode, r is RootCode, f is FormCode (or PatternCode), and s is Suffix code. If \underline{q} is disambiguated to get \hat{q} , the diacritization of w is directly extracted from the triple

$$\hat{q} = (t : p, f, s)$$

Revise [2] for a full detailed discussion of this Arabic morphological analyzer¹².

The morphological analysis of an input word also suggests the set of allowed syntactic diacritics due to this analysis; $\underline{d} = d_1 d_2 \dots d_{J_i}; J_i \geq 1$. Producing the syntactic analysis formally requires a reliable Arabic syntactic analyzer that produces a unique (unambiguous) parsing tree. Unfortunately, such an analyzer does not exist so far. In fact building such analyzer is much more difficult than the problem of Arabic phonetic transcription. Following our arguments in this paper, the absence of this analyzer had to be stochastically substituted as Figure 7 implies.

2. Arabic phonetic grammar (APG) that can deterministically judge whether a given diacritized sequence of Arabic characters are phonetically valid or not. This phonetic grammar is formalized as a set of BNF rules. This grammar expresses a lower linguistic layer of Arabic, as shown in Figure 6, and constitutes the sole constraint on Arabic phonetic

¹² The Arabic morphological analyzer has other important applications than automatic Arabic phonetic transcription, e.g. Arabic POS tagging as input features for higher linguistic layers; esp. syntax analysis, Arabic derivative search engines for IR systems, Arabic spell checkers, etc. [2], [13]

transcription when morphological analysis does not apply; e.g., while diacritizing transliterated names. Although being a linguistically interesting work, presenting the APG and its formalization in detail needs a considerable space and may be the subject of a later paper of the authors¹³.

STOCHASTIC COMPONENTS

1. Collecting and indexing the frequencies of all n -grams starting from monos to n^{th} order grams of relevant linguistic entities (morphological quadruples, diacritics, ordinary-Arabic/transliterated markers, pause/no-pause markers) from a large annotated Arabic text corpus, statistical models can be built for estimating $P(I)$ as required by Eq. 1. Then, building our disambiguation trellis as shown in Figure 7, and using the A^* search algorithm as detailed earlier through this trellis, the following two sub-solutions are obtained:

- 1) The best (most likely) sequence of linguistic entities are obtained, hence the most likely diacritics of the input Arabic string (except for the words judged as transliterated) can directly be obtained.
- 2) The transliterated words are spotted and sent to the next component for inferring their most likely diacritization.

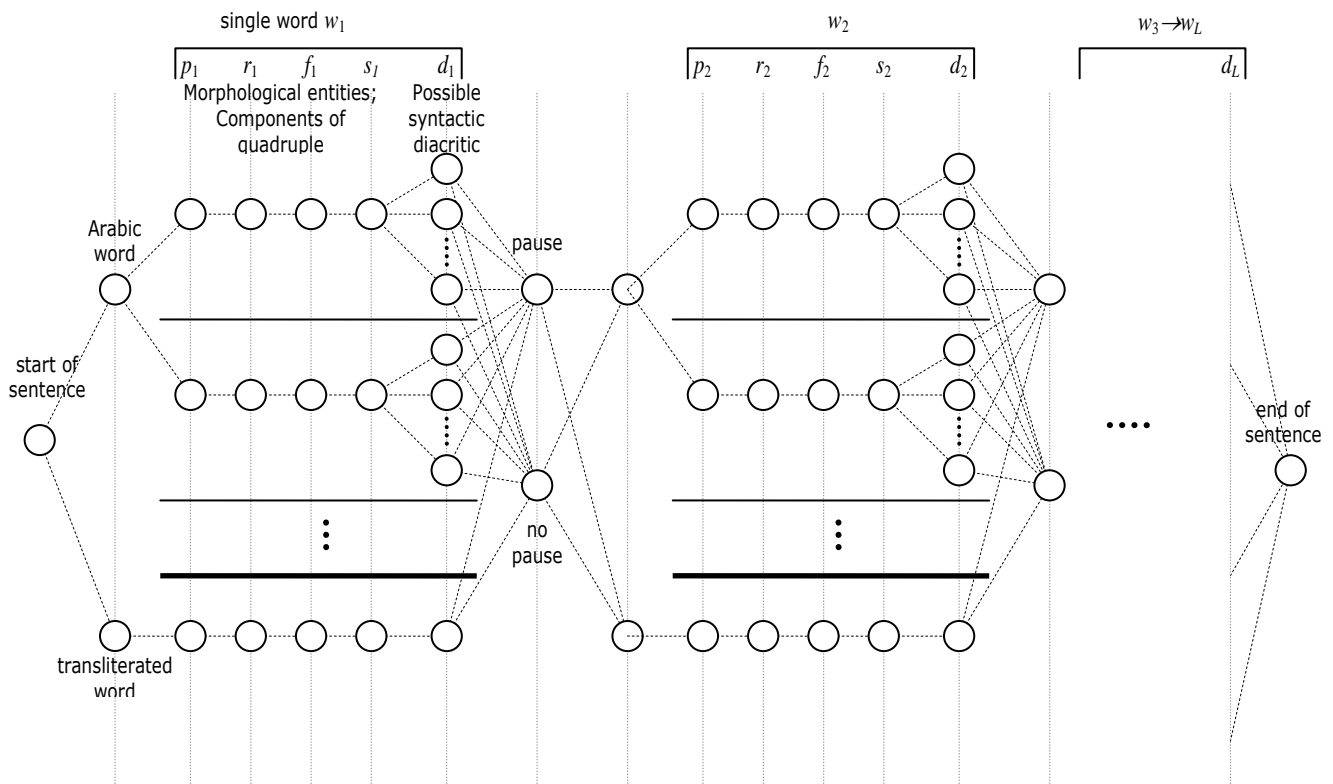


Figure 7: Disambiguation lattice for morphological disambiguation, syntactic diacritization, & prosodic phrasing.

2. Collecting and indexing the frequencies of all n -grams starting from monograms to n^{th} order grams of “character, shadda-state, diacritic” sequences from a large corpus of diacritized transliterated words, statistical models can be built for estimating the likelihood of possible suggested diacritics. Then, building our disambiguation trellis as shown in Figure 8, and using the A^* search algorithm as detailed earlier through this trellis, the most likely diacritization of the transliterated word is obtained.

It should be noted here that the disambiguation trellis alone of Figure 8 below may generate diacritics invalid according to the APG. So, the A^* algorithm here is modified such that as soon as any open path is chosen for expansion, the corresponding partial diacritization is tested against the APG and is eliminated if the test result is false.

¹³ APG is readily formalized and developed in RDI's Arabic NLP labs.

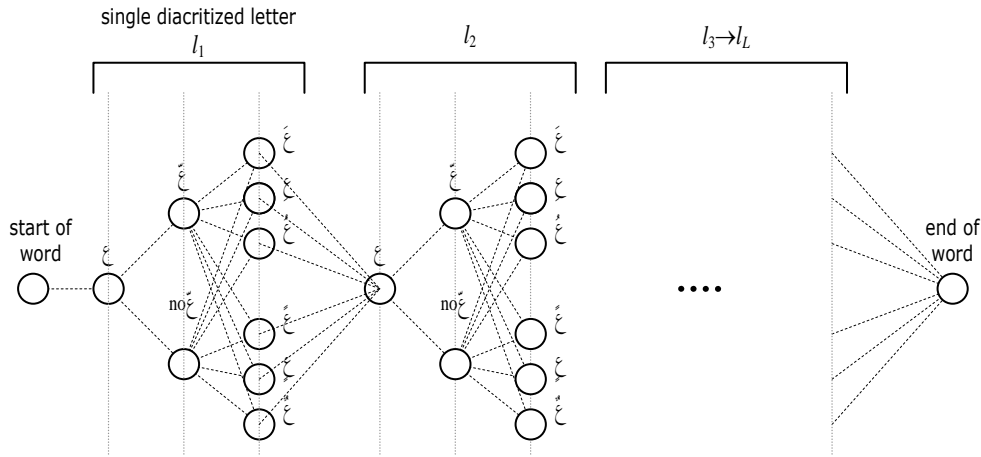


Figure 8: Disambiguation lattice for Arabic word diacritics.

CONCLUSION

ArabDiac has been trained by annotated Arabic text corpora including the Holy Qur'an, news domain text, business domain text, and Arabic dictionary text. The total size of these corpora was about 0.5M words. A test sample of news domain text corpus of size 30K words shows an overall diacritization word-accuracy of about 91.8%; i.e., about 11 correctly diacritized words of each 12 ones. The accuracy of prosodic phrasing was about 90.6%. It is to be noted that the same solution may be applied with one or more elements of ambiguity excluded, say the syntactic diacritics. In fact, when only the morphological disambiguation alone was tested versus the same sample, a much higher accuracy of about 96.4% has been recorded.

Finally, it should be noted that the more the training corpora, the better is the accuracy of the system. Despite the saturation at huge annotated training corpora, the author thinks that this will not occur before exceeding a size of about 10 M annotated corpus, so there is a room for a considerable improvement in the system accuracy.

Acknowledgement

The authors pose all the sincere thanks to the software engineering team of Research & Development Int'l; RDI whose contributions, extending from material affordance, to technical discussions, criticism, explanations, software applications development, and even to the editing of this paper - were all essential for this work to see the sun light. We pose many gratitudes in specific to the current colleague Ibraheem Subh as well as the former ones Yasir Hifny and Wa'il Hamza. May Allah reward them all the best.

References

1. Al-Wa'er, M., "Toward a Modern Theory of Basic Structures in Arabic", Ed. Arab School of Science and Technology, Damascus, Syria, 1983.
2. Attia, M., "A Large-Scale Computational Processor of The Arabic Morphology, and Applications.", MSc. thesis, Dept. of Computer Engineering, Faculty of Engineering, Cairo University, 2000. This thesis is downloadable from <http://www.emlar.org/publications/idex.htm>
3. Dutoit, T., "An Introduction to Text-to-Speech Synthesis", Kluwer Academic Publishers, 1996.
4. Grosz, B.J., Jones, K.S., and Webber, B.L., "Readings in Natural Language Processing", Morgan Kauffman publishers, 1986.
5. Hamza, W.M., "A Large Database Concatenative Approach for Arabic Speech Synthesis", PhD. thesis, Dept. of Electronics and Electrical Communications, Faculty of Engineering, Cairo University, 2000.
6. Hassan, H.M., "Maximum Entropy Framework for Natural Language Processing Applied on Arabic Text Classifications", MSc. thesis, Dept. of Electronics and Electrical Communications, Faculty of Engineering, Cairo University, 2002
7. Haung, X., Acero, A., Hon, H.W., "Spoken Language Processing", Prentice Hall PTR, 2001.
8. Jurafsky, D., Martin, J., H., "Speech and Language Processing; An introduction to natural language processing, computational linguistics, and speech processing", Prentice Hall, 2000.
9. Kapur, J.N., Saxena, .H.C., "Mathematical Statistics", 7th edition, S. Chand & Co. (Pvt.) LTD, 1972.

10. Katz, S.M., “*Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer*”, IEEE Transactions on Acoustics, Speech and Signal Processing, vol. ASSP-35 no. 3, March 1987.
11. Lathi, B.P., “*Modern Digital and Analog Communication systems*”, 2nd edition, Holt, Rinehart and Winston Inc.
12. Nadas, A., “*On Turing's Formula for Word Probabilities*”, IEEE Transactions on Acoustics, Speech and Signal Processing, vol. ASSP-33 no. 6, December 1985.
13. Nagy Fatehy Mohammad, “*An Integrated Morphological and Syntactic Arabic Language Processor Based on a Novel Lexicon Search Technique*”, MSc. thesis, Faculty of Engineering, Cairo University, 1995.
14. Nilsson, N.J., “*Problem Solving Methods in Artificial Intelligence*”, McGraw-Hill, 1971.
15. Ratenaparkhi, A., “*Maximum Entropy Models for Natural Language Ambiguity Resolutions*”, PhD thesis in Computer and Information Science, Pennsylvania University, 1998.
16. Schutze, H., Manning, C.D., “*Foundations of Statistical Natural Language Processing*”, the MIT Press, 2000.
17. Sproat, R., “*Multilingual Text-to-Speech Synthesis*”, Kluwer Academic Publishers, 1998.
18. Van Santen, J.P.H., Sproat R.W., Olive, J.P., Hirschberg, J., “*Progress in Speech Synthesis*”, Springer, 1996.

About the Authors

Mohamed Attia has been graduated from the department of Electronics and Electrical Engineering, Faculty of Engineering, Cairo University, in 1995. He has been awarded MSc. degree from the department of Computer Engineering of the same faculty, in 2000. He is now a senior software engineer and in same time an Arabic NLP researcher in Research & Development Int'l; RDI which is a software house located in Giza, Egypt. Now, he is a PhD. Student in the department of Electronics and Electrical Engineering, Faculty of Engineering, Cairo University. His research interests mainly focus on software intelligent systems that aim for more natural human/machine interaction especially via written and spoken natural languages.

Dr. Mohsen Ali Abdel Raziq Rashwan is a professor in the department of Electronics and Electrical Engineering, Faculty of Engineering, Cairo University. He is also the CEO of Research & Development Int'l.; RDI which is an Egyptian s/w house specialized in developing spoken and written Arabic language processing technologies and applications. His research interests mainly focus on advanced speech processing fields with all variants such as speech compression, speech recognition, speech verification, and speech synthesis. Now, he is also highly interested in Arabic NLP technologies and its employment in speech processing.

Galaal K. Galaal is a software engineer in RDI's Arabic NLP team where he shared in the R&D activities towards many of RDI's Arabic HLT products.