Proceedings of the Ninth IASTED International Conference
ARTIFICIAL INTELLIGENCE AND SOFT COMPUTING
September 12-14, 2005, Benidorm, Spain

A MORPHOLOGICAL GENERATOR FOR THE INDEXING OF ARABIC AUDIO

Khaled F. Shaalan[12], Habib E. Talhami[12], and Ibrahim H. Kamel[3]
Institute of Informatics
[1]The British University in Dubai
P. O. Box 502216, Dubai, UAE
[2]Honorary Fellow, School of Informatics, University of Edinburgh
{khaled.shaalan,habib.talhami}@buid.ac.ae
[3]College of Information Systems
Zayed University
P. O. Box 19282, Dubai, UAE
Ibrahim.Kamel@zu.ac.ae

ABSTRACT
This paper presents a novel Arabic morphological generator (AMG) for Modern Standard Arabic (MSA) which is designed and implemented using Prolog. The AMG is used to generate inflected forms of words used for the indexing of Arabic audio. These words are also the relevant terms in the Arab authority system (library information retrieval system) used in this study. The AMG generates inflected Arabic words from the root according to pre-specified morphological features that can be extended as needed. The Arabic word is represented as a feature structure which is handled through unification during the morphological generation process. The inflected forms can then be inserted automatically into a speech recognition grammar which is used to identify these words in an audio sequence or utterance.

KEY WORDS
Arabic Morphological Generation and Arabic Audio Indexing

1 Introduction

Arabic morphological analysis has been the focus of researchers in natural language processing for a long time [1]. On the other hand Arabic morphological generation has received little attention in spite of the fact that the types of generation problems can be as complex as those of the analysis [2]. Arabic morphological generation generally suffers from either the lack of coverage or the closeness of lexical and surface levels introduced by using finite state transducers (FSTs). In applications such as indexing, the morphological generation plays an essential role in indexing each occurrence of the inflected words which is generated from the same root. This has led us to design and implement an Arabic morphological generator using Prolog. The Arabic word is represented as a feature structure, a Prolog term, which is handled through unification during the morphological generation process. Hence, our Arabic morphological generator can make use of Prolog's built-in term-unification, instead of the more expensive feature-unification.

The features are selected according to their relevance for the audio indexing of MSA utterances. Audio indexing is done by using a technique developed in [3] which is based on an Arabic authority system (library information retrieval system). At the heart of the technique is a grammar which is designed to identify an authority term and its relevant terms. This is achieved by using a combination of class-based language modeling and robust parsing.

In order to index utterances with the correct authority term, in theory all valid inflected forms should be hand coded into the grammar. This is usually a very large task especially when we are dealing with a large number of authority terms and their corresponding relevant terms (these could be large too). Ideally, we would like to be able to generate these inflected forms automatically and to be able to insert them into our grammar(s). However, we should be able to constrain the generation in order to exclude forms that are invalid or even rarely used. Thus we have designed an AMG that is flexible enough and is able to include any form we deem necessary for audio indexing. The rest of the paper is organized as follows: the audio indexing approach is summarized in section 2; this is followed by a description of the proposed morphological generator in section 3; finally, a conclusion and recommendations for further enhancements are given in section 4.

2 The Audio Indexing Problem

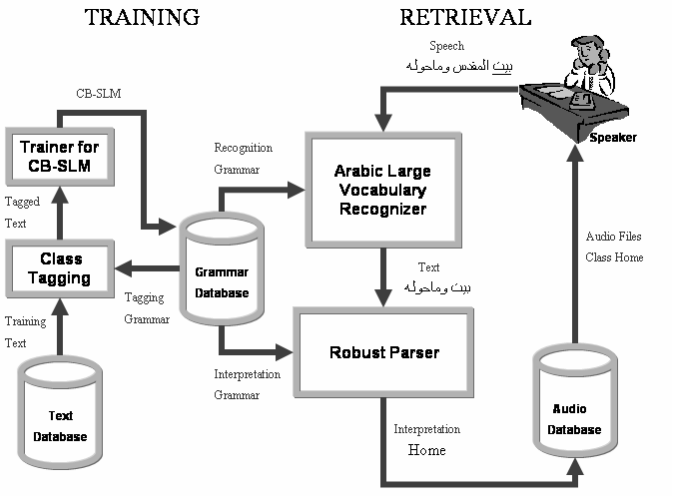
According to [4], approaches to audio access fall under three categories: (a) structural approaches, which allow the indexing of various speech regions such as speaker, emphasis, external events, or even visual events, (b) content-based approaches, where speech recognition is applied and the resulting text is searched for keywords, and (c) surface manipulation [5], where the speech is played back at several times its original rate while maintaining intelligibility. Content-based audio indexing systems such as the one deployed at BBN [6] normally make use of the following technologies: speech recognition, speaker

identification, named-entity extraction, topic classification, and story segmentation. This is particularly so when the area of application is the audio indexing of broadcast news. Central to the content-based systems mentioned above is the idea of keyword detection sometimes referred to as word spotting.

Two main approaches to word spotting have been identified [7] in the literature: (a) using a large vocabulary speech recognition system (LVCSR) to produce a string, which is then searched for a keyword, and (b) using keyword and filler models, where the models can be whole words or sub-units. Approach (a) seems to be the one that has been achieving the best results [7] and accordingly we have developed a variation to this approach that combines class-based speech recognition with robust parsing (or robust natural language processing) [3].

2.1 Audio Indexing using Class-Based Statistical Language Models and Robust Parsing

Figure 1 shows an architectural outline implementation of the system implemented by [3] for the indexing of Arabic audio databases. It consists of two phases: (a) a training phase for generating class-based statistical language models (CB-SLM) from a training corpus, and (b) a retrieval phase where the caller search phrase is converted to text and then interpreted using a robust parser. The outcome of the retrieval is a set of waveforms belonging to the class identified by the robust parser.



text

Figure 1. The Arabic Indexing System showing both Training and Retrieval Phases

The training process proceeds as follows: some training is retrieved from the text database (Figure 1) and is then tagged by a tagging grammar. The tagged text is then passed to a trainer that produces the class-based statistical language models (CB-SLMs). The CB-SLMs are then passed to the recognizer that also employs the same interpretation grammar as the one used for tagging. The large vocabulary recognizer produces a word sequence, which is used subsequently by the robust parser. The robust parser identifies the tag from the word sequence, which can be used to retrieve all the audio files that have a similar tag from the audio database.

Class-based language models are an extension of the n-gram language models (LM) that are usually used in a large vocabulary speech recognizer. N-gram LM is a statistical model which predicts the next word, w_n, given the previous n-1 words by estimating the conditional probability $P(w_n \mid w_1, \cdots, w_{n-1})$. In most systems and also for this study the trigram, where the word depends on the previous two words, is used. Class-based language models [8] are used when one wants to group words into classes according to certain criteria such as acoustic similarity or common semantic interpretation (which is the case here).

Robust parsing [9] is designed to deal with the imperfections of speech recognition which fall under three main categories: recognition errors made by the recognizer, under-specified grammars: grammars that do not have coverage for all the linguistic constructs needed by the application, and disfluencies and irrelevancies: these are common in natural spoken dialogues.

A robust parser attempts to solve the following three problems: chunking which consists of dividing the text into meaningful sections or chunks, disambiguation or selecting a unique interpretation from a potentially large number of valid options, and undergeneration or dealing with disfluencies and irrelevancies that are outside the coverage of the grammar.

Interpretations in our system are done according to an interpretation grammar which includes the various classes that have been identified as part of an Arabic authority system [3]. So between CB-SLM and robust parsing we are able to use a large vocabulary and a set of classes to identify any tags within any text. The restrictions of having a limited grammar are no longer there and the system can be extended to any audio indexing problem given the appropriate CB-SLM.

2.2 Audio Indexing using an Arabic Authority System

Authority control is a popular term in the library science community that refers to the system for providing consistency in storing author names and titles terms. There are two types of authority files: name authority files, and subject authority files.

Name authority control allows the library user to perform a comprehensive search using one form of the author

name. For example, الجاحظ is a popular poet. But he is also known as أبو عثمان, عمرو بن بحر بن محبوب. The authority control system allows the user to query the database with any form of the name and yet retrieves the books that are indexed under any of the above forms.

Subject authority control is similar to the name authority control but it uses synonyms of the search word. For instance, if the library user wants to search for "cars", the authority control system will search for "cars" plus other related terms such as "automobiles", "vehicles", or "transportation". More over it can retrieve terms in other languages such as "سيارات", "عربات" to retrieve all bibliographic records.

The authority control consists of two main components: the authority file and authority record. Authority record is an entry in the database that gathers all relevant terms together, and consists of two parts: the authorized term and the relevant terms as shown in Figure 2. In this paper, the set of terms that forms the Authority record is also called class. Authorized term is the term that the library cataloguer uses in indexing. An authority file is a set of authority records for every given heading and its corresponding cross-reference.

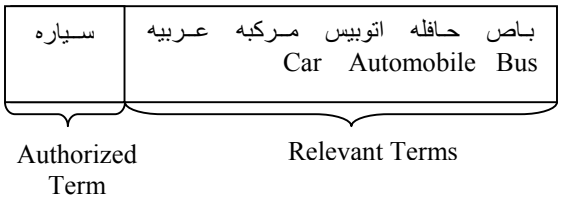


Figure 2. The Arabic Authority Record

In [3] a series of experiments were designed to study the approach depicted in Figure 1. We restricted our experiments to keyword search only, since keyword search is more general and more complex than author name search. A class is a set of words that have the same meaning or referring to the same subject or person. Each class is identified by a given name. For example the class "car" consists of words such as "automobiles", "vehicles", or "transportation". More over it might contain terms in other languages such as "سيارات", "عربات". Ten classes were identified from the Arabic authority database, which are: Car, Field, Home, Phone, President, Power, Sign, Story, Time, and Work. For each class at least five terms have been used, for example, class Home had the following members:

Home

[(دار) (منزل) (ال_منزل) (مكان) (ال_مكان) (مكان ال_معيشة)] {<home (دار_ال) (مسكن_ال) (مسكن) (بيت_ال) (بيت□) "Home">}

The above example is taken from a Nuance GSL grammar [10] that has been used for the tagging and interpretation of text. The grammar contained all ten classes with their associated terms. There is a semantic slot for each class, which was chosen to have the same name as the class. The slot names served also as the tags for tagging the training text needed to produce the class-based statistical language models. Text that consists of the titles of the books and author names is used to train class-based language models (CB-SLM).

Results in [3] have shown that even in the severe case where the language models have not seen the utterances, the identification of semantic classes using class-based language models and robust parsing gave 32% semantic error rate (SER). This is in contrast to the phrase error rate (PER), which was 61.6%, which would have resulted in an equivalent or higher error rate if normal grammar-based parsing had been used. A natural extension to this approach is the inclusion of inflected forms of each of the relevant terms in the grammar. As discussed earlier the manual generation of multiple inflected words is neither practical nor desirable (introduces human error). A better approach would to design a morphological generator that can be both generic and extendable. It also allows the exclusion of forms that are either invalid linguistically or very rarely used.

3 Morphological Generation

The basic principle of morphological generation is to get forms from a root and a set of features (lexical category and morphological properties). Generally, there are two categories of approaches to developing an Arabic morphological generator: approaches that use finite-state transducers (FSTs) and approaches that use rule-based transformations.

FSTs such as Xerox Arabic analyzer [11] are limited to applications that are heavily dependent on morphological generation because the lexical and surface levels are very close. On the contrary, the rule-based transformation approach allows to morphologically generate an Arabic inflected word from the input which is usually a root with a specified feature list. This approach has been used by [2] [12]. The former is a prototype that is restricted in its coverage. The later follows approach of [13] in that morphotactics and orthographic rules are built directly into the lexicon itself. Our approach is rule-based that uses general transformational rules to address the issue of generating inflected Arabic words in various prefix/suffix contexts for facilitating the audio indexing process. Unlike [2], we use general computational rules that interact to realize the output.

3.1 The Lexicon

An Arabic monolingual lexicon was needed for the successful implementation of the Arabic morphological generator. The lexicon is designed to reflect the word categories in Arabic—each with a different set of features. The lexicon stores the root form of a word. This is the uninflected (primitive) form: the past tense of a verb, the singular masculine of a noun, and the particle. In addition, the lexicon holds entries for irregular plurals forms. It

turns out that each irregular entry has room to store the root form too. So, there is enough information in the lexicon to go back from roots to irregulars, and vice versa. Consequently, the problem reduces to one of efficient indexing.

Entries of the lexicon are represented as feature structures that is to say sets of feature-value pairs. Each pair has the form <feature>:<value>. The lexicon entry is represented as a Prolog fact. The first argument is the stem itself and the second argument is a feature structure list. The following describes the forms of the lexicon entry:

1. Verbs: A verb has the following form:
lex('Arabic-word',[stem:'Arabic-verb',cat:verb,gender:female/male,number:sg/dl/pl, tense:past/present/future,case:nom/acc/gen,transitivity: intrans/trans]).
For example, consider the lexicon entry of the verb رغب (desired/ wanted):
lex('رغب',[stem:'رغب ب',cat:verb,gender:male,number:sg,tense:past, case:nom,transitivity:intrans]).

2. Nouns: A noun has the following form:
lex('Arabic-word',[stem:'Arabic-noun',cat:noun, gender:female/male/neut,number:sg/dl/pl, sub_cat:infinitive/demonstrative/proper_noun/ common_noun/adverb/ausative_object/accusative/adjective/ question,definition:yes/no,irr_form: [irr_pl:'Broken_pl',…]]).
For example, consider the lexicon entry of the noun بيت (house) and its irregular plural بيوت (houses):

lex('بيت',[stem:'بيت',cat:noun,gender:male,number:sg, sub_cat:common_noun,definition:no,case:nom,irr_form: [irr_pl:'بيوت']]).

lex('بيوت',[stem:'بيوت',cat:noun,gender:female,number:pl, sub_cat:common_noun,definition:no,case:nom,irr_form:[singular: 'بيت']]).

3. Particle: A particle has the following form:
lex('Arabic-word',[stem:'Arabic-noun',cat:particle, sub_cat:conjunct/preposition...])
For example, consider the lexicon entry of the preposition في (in):
lex('في',[stem:'في',cat:particle,sub_cat:preposition, connect:no]).

The predicates unify_feature/2, unify_features/2, update_feature/2, and update_features/2 are used to get/give values to a feature(s), through unification, during the morphological generation process.

3.2 The Arabic Morphological Generator (AMG)

In our Arabic morphological generation, the inflected Arabic words are synthesized from a given root according to a combination of morphological properties that include definition (article "ال"), gender (masculine, feminine), number (singular, dual, plural), case (nominative, genitive, accusative), and person (first, second, third). This generation process is implemented such that not every combination generates a correct inflected from. So, a verification of prefix-stem, prefix-suffix, and stem-suffix compatibility has to be done before applying the generation process in order to ensure which morphological properties are allowed to occur. For example, the prefix "ت" is not compatible with the noun category such as "وقت" (time). Similarly, the prefix article "ال" (the) is not compatible with the suffix second person pronoun "ك". Nevertheless, the stem "بيت" (home) is not compatible with the feminine marker "ة".

Within the morphological generation, there are three main problems: regular words, irregular words, and compounds.

Generating regular words from a root:
Primitive rules that generate regular inflected Arabic words are implemented in Prolog (see Figure 3). These primitive rules are grouped into clusters according to the Arabic morphological rules. There are two ways to generate the inflected Arabic words: either by calling clusters that matches the specified morphological features or by using the built-in predicated bagof/3 to exhaustively generate every possible inflected Arabic word from the given root.

```
synthesize_noun(number:plural,case:_,FS0,FS):-
   unify_features([number:sg,gender:female,
      stem:SgWord],FS0),!,
   name(SgWord,W1),
   append(W2,"ة",W1),
   append(W2,"ات",W3),
   name(PlWord,W3),
   update_features([stem:PlWord,number:pl],FS0,FS).
synthesize_noun(number:plural,case:nom,FS0,FS):-
   unify_features([number:sg,gender:male,
      stem:SgWord],FS0),!,
   name(SgWord,W1),
   append(W1,"ون",W2),
   name(PlWord,W2),
   update_features([stem:PlWord,number:pl,case:nom],
     FS0,FS).
```

Figure 3. Rules for generating Sound Plurals

Generating irregular words from a root:
Irregular words are those which don't obey the normal rules of inflection. Classes of irregulars include:

- Irregular plurals: Arabic has lots of irregular plurals- known as broken plurals such as فأر/فئران (mouse/mice) and طفل/أطفال (child/children). For people nouns that have both masculine and feminine forms, often the feminine plural is regular and the masculine plural is irregular such as صديقات

(friends, feminine) and أصدقاء (friends, masculine).

- Abbreviated noun (المنقوص Al-Mankous): Nouns whose last radical letter Yeh "ى" may be curtailed (محذوف) in some cases such as محامى/محامون (lawyer/lawyers).
- Prolonged noun (الممدود Al-Mamdoud): nouns that its last radical letter Hamza "ء" changes to Waw "و" in some cases such as صحراء/صحراوان (desert/two deserts).

In order to generate irregular plurals from its root form, AMG looks up the root form entry and gets this form. Then, another look up for the irregular plural is performed to get its morphological features, see Fig 4. This information is used with the input morphological properties to generate the correct irregular form. For example, to generate بيوتهم (their homes, feminine) from the singular noun بيت (home, masculine), AMG looks the irregular plural in the lexicon and then generates the plural of the third person suffix pronoun that will be attached to the irregular plural. For both the abbreviated and prolonged nouns, AMG applies the irregular generation rule(s) that corresponds to the input morphological properties.

```
synthesize_noun(number:plural,case:Any,FS0,FS):-
   unify_features([stem:X,number:sg,irr_form:(irr_pl:
     Broken_pl)],FS0),
   lex(X,FS0),
   \+ Broken_pl = [],!,
   lex(Broken_pl,FS).
```

Figure. 4 A Rule for generating Irregular Plurals

Generating compound words from a root:
Compound words such as برنامج شتوي (winter program), ساحة انتظار (park), خمس نجوم (five star), معالم سياحية (landmarks), أقامة كاملة (full board) need special handling. A compound word usually consists of one or more words that have entries in the lexicon. For example, to generate برامج شتوية (winter programs, plural) from برنامج شتوي (winter program, singular), we first get the features of each of the constituent words: برنامج (program, singular, masculine, noun) شتوي (winter, singular, masculine, adjective) from the lexicon. Second, we apply the generation rules of irregular plural to the noun yielding برامج (programs, plural, feminine, noun). Third, we apply the agreement of the (described) noun and its adjective yielding شتوية (winter, plural, feminine, adjective). In our implementation, we can handle the compound words that consist of up to three words and form a noun phrase or adjective phrase. We do not handle the case of preposition phrase as this is rarely the case with automatic indexing, which is our target application.

4 Conclusions and Future Work

In this paper, we described the development of a novel Arabic morphological generator for MSA. The morphological generator is implemented in SICStus Prolog and takes the advantage of Prolog's built-in term-unification. The rule-based approach is used to give some flexibility in the indexing of Arabic Audio. An inflected Arabic word is generated from the root according to pre-specified morphological features. Features are selected according to their relevance and validity for the audio indexing of MSA utterances. The inflected forms are used to replace the relevant terms in the grammar which used for both class tagging and speech recognition.

Future work will include a large data capture, both speech and text to determine the effect of these inflections on perplexity and recognition accuracy. This will be followed by extending the AMG to colloquial dialects of Arabic. Another interesting challenge would be to introduce diacritics into the lexicon. Text in Arabic is generally written without the diacritics (or vowels) and these are sometimes essential for the disambiguation of words. The system will also be extended to include diagnostic information especially when invalid inflections are generated.

References

[1] Sughaiyer I., Al-Kharashi ,I. (2004). Arabic Morphological Analysis Techniques: A Comprehensive Survey. *Journal of The American Society for Information Science and Technology*, 55(3):189-213.

[2] Nizar Habash. Large scale lexeme based Arabic morphological generation. In *Proceedings of Traitement Automatique du Langage Naturel (TALN-04)*. Fez, Morocco, 2004.

[3] Habib Talhami and Ibrahim Kamel. Identifying semantically similar Arabic words using a large vocabulary speech recognition system. In *Proceedings of the Ninth IASTED International Conference on Internet and Multimedia Systems and Applications*, Grindelwald, Switzerland February 21 - 23, 2005.

[4] S. Whittaker, J. Hirschberg, B. Amento, L. Stark, M. Bacchiani, P. Isenhour, L. Stead, G. Zamchick, & A. Rosenberg. SCANMail: a voicemail interface that makes speech browsable, readable and searchable. In *Proceedings of CHI2002*, 2002, 275-282.

[5] B. Arons, SpeechSkimmer: A System for Interactively Skimming Recorded Speech, *ACM Transactions on Computer-Human Interaction, 4*(1), March 1997, 3–38.

[6] J. Makhoul, F. Kubala, T. Leek, D. Liu, L. Nguyen, R. Schwartz, and A. Srivastava. Speech and language technologies for audio indexing and retrieval. In *Proceedings of the IEEE*, 88(8), August 2000.

[7] I. Magrin-Chagnolleau, and N. Parlangeau-Vallès. Audio indexing: what has been accomplished and the road ahead. *Proceedings of JCIS*. 2002, 911-914.

[8] X. Huang, A. Acero, and H.-W. Hon. *Spoken language processing: a guide to theory, algorithm, and system development.* Prentice Hall, New Jersey: 2001.

[9] Y. Wang. A robust parser for spoken language understanding. In *Proceedings of Eurospeech*, Budapest, 1999, 2055-2058.

[10] Nuance Communications Inc. *Nuance system grammar developer's guide, Version 8.5* Nuance Communications, Inc.: 2003.

[11] K. Beesley. Arabic finite-state morphological analysis and generation. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING-96)*, volume 1, pp. 89–94, Copenhagen, Denmark, 1996.

[12] V. Cavalli-Sforza, A. Soudi, and T. Mitamura. Arabic morphology generation using a concatenative strategy. In *Proceedings of the 6th Applied Natural Language Processing Conference (ANLP 2000)*, PP. 86–93, Seattle, Washington, USA, 2000.

[13] T. Buckwalter. Buckwalter Arabic Morphological Analyzer Version 1.0. *Linguistic Data Consortium*, University of Pennsylvania, 2002. LDC Catalog No.: LDC2002L49, 2002.