Final Report

NLP based Computer assisted language learning of Arabic

Supervised by:

Dr. Violetta Cavalli Sforza

Made by:

Mohamed ElMehdi Amine

Summer 2015

Table of Contents

Introduction

Background Research

Computer assisted language learning (CALL) is a field of investigation that combines research and development. In CALL the aim is to employ computers in the teaching and learning of foreign languages in order to achieve effective language learning. (Heift & Schulze, 2007)

While efficient language learning is what drives CALL, such efficiency cannot be achieved without the contribution of artificial intelligence areas such as natural language processing (NLP). Without intelligence, software meant for language acquisition is only different from traditional workbooks in so far as it presents information in a different way (Heift & Schulze, 2007). Without intelligence, the interaction that can be achieved by using a computer, and from which students benefit the most, is limited to sounds and visual effects instead of constructive feedback. Intelligent CALL, or ICALL, benefits from NLP as it allows for a sophisticated error analysis by which the exercises offered to the learners can move from simple multiple choice and fill in the blanks type of questions, where the system is mostly string matching against the correct answer, to questions where the answers are in the form of freely produced text, where the system is parsing the user input. (Holland et al. , 1993).

ICALL has proven more effective than workbooks in language learning. Studies have shown that "not only do students appreciate the more sophisticated, error specific feedback made possible by NLP, but also perform better on the language skills being taught."(Nagata, 1996). Other than the high degree of interaction and the constructive feedback, employing NLP in CALL is useful as it provides a good mean by which teachers can address one of the biggest problems faced by language learners: grammar. Conveniently for syntactic parsers that cannot detect semantic errors, most of the errors made by students relate to the syntax, morphology, orthography and lexicon: "Syntax is the most problematic area, followed by morphology. These two categories make up for 53% of errors… The study, a contribution to the error analysis element of a syntactic parser of German, indicates that most student errors (80%) are not of semantic origin and therefor, are potentially recognizable by a syntactic parser." (Juozulynas, 1994).

Needless to say, the advantages that NLP brings to ICALL do not come without cost. It requires computational linguists, advanced programmers and resources to build and test parser based CALL applications. (Holland et al., 1993). And even with such costs, parsers, as stated by the authors of "Parsers in Tutors: What Are They Good For?" are not foolproof: "No parser today can accurately analyze all the syntax of a language." (Holland et al., 1993). Aside from the fact that grammatical error detection is to a high degree subjective (Heift & Schulze, 2007), another reason that contributes in making parsers inaccurate is the ambiguities that parsers have to face. To a parser, a sentence is ambiguous when it can have more than one parse tree. Sentences like "Book that flight" contain local ambiguities where one part of the sentence is ambiguous while the whole sentence is not ambiguous. In the case of "Book that flight" book can be parsed as a verb and as a noun, but there is only one parse for the whole phrase. A more complex ambiguity would be a sentence like "we saw the Eiffel tower flying to Paris" where the sentence could either mean that the flying was done by the subject pronoun or the Eiffel tower. NLP systems need to be able to choose the correct parse for the multitude of possible parses through a process known as syntactic disambiguation. Syntactic disambiguation requires statistical, semantic, and pragmatic knowledge that is not readily available during syntactic processing. (Jurafsky, 2009).

"The concept of intelligence as applied to Computer Assisted Language Learning have been a source of contention since the early 1980."(Heift & Schulze, 2007). One reason for the contention and lack of progress in some areas of CALL is due to the limitations faced by parsers: "Complex computational grammars lexicons and parsers underlie these systems [ICALL applications], which as far as I can see do not yet work efficiently." (Wollf, 1993).

Although the limitations of parsers obstruct their efficiency and accuracy, this should not be a reason to obstruct students learning. In fact, this could enhance student learning according to John Higgins: "The parser, however, does not need to be perfect […] Users will be perfectly ready to modify their language and try other formulations to find ones which work as long as they see the machine as basically stupid." (Higgins, 1987). This is not different from the process of learning programming languages and dealing with the alerts and errors faced with the process of building programs. It is during the process of dealing with these errors and alerts that programers have to try different syntax and work their way around the parser's stupidity, which is arguably at the heart of learning programming languages.

As it turns out that the limitations of parsers could be actually seen as beneficial for learners, parsers have more to offer. In parser based CALL, learners are allowed to practice

their production skills by entering a huge variety of sentences. (Holland et al. 1993). The freely produced text of the learners is analyzed and the system can provide in return valuable, evaluative feedback. As the system is able to recognise the learners' errors, it can adapt to their weaknesses by selecting exercises according to the classes of errors faced. Parsing alone, however, is not enough for this. To achieve the necessary intelligence for ICALL, parsers should be combined with error analysis. We call such parsers that can recognise errors in addition to detecting them, robust parsers. (Heift & Schulze, 2007).

For robust parsers to perform error analysis three steps are required (Corder, 1981):

1. Recognize errors: simple parsers can detect errors in the grammar of learners' texts, but while it is enough to know whether errors exist or not for systems that can provide the only feedback of "wrong" or "correct," ICALL aims for constructive feedback and so needs to do better than just detecting the presence of errors. To recognize errors, corpora of text produced by students is used to retrieve their errors. Once collected, these errors are classified. Classes serve to group sets of errors under the same theme, examples of error classes could be gender agreement, number agreement, appropriate verb tense, article and preposition errors. (Al-Yaari & al., 2013). Recognising errors is crucial for constructive feedback, and classifying errors is useful for the system to be able to suggest exercises for the learners according to their errors.
2. Describe errors: once errors are recognised, they should be described. The description of an error is important in so far as it lets the learner identify what the error is and where it occurred in his/her text.
3. Explain errors: explaining errors helps the learners to go beyond identifying their errors. The explanation of an error should contain information like what might have caused the learner making such error. Studies that aim to understand the influence of L1 (first language acquired) can provide explanation for learners' errors. As an example one such study shows that there are four ways an L1 can influence learners of English (Leacock et al. 2010):

    3.1. Learning will be difficult if the learner has no close equivalent feature. For example, the native speakers of Japanese and Russian will have particular difficulty mastering the use of articles.

    3.2. Learning will be facilitated if the L1 does have an equivalent feature. For example, native speakers of French or German will find the English article system relatively easy to learn.

3.3. When two languages are similar, but not identical, there will be transfer problems, since two languages will rarely have exact equivalents. French and German learners will make article errors where their L1's article system differs from that of English.

3.4. Speakers of unrelated languages will have fewer transfer problems than speakers of a closely related language, but they will make more errors due to the difficulty of complex English structures.

Being able to recognize errors, classify them, describe and explain them would be all wasted if it never reaches the learner. Hence feedback comes in as a way to deliver the results achieved by robust parsing. Garrett (1987) describes four kinds of feedback for CALL:

1. Only the correct answer is provided as feedback.
2. The feedback points to the location of the error.
3. Based on analysis of the wrong answers, error messages associated with possible errors are stored in the computer and are presented if the student's response matches those possible errors.
4. The system uses an intelligent, NLP approach in which the feedback is provided as a linguistic analysis of the student's response.

From the four types of feedback described by Garret, only the forth takes full advantage of the results of the robust parser. However, providing different levels of feedback should be considered. The system could for example suggest hints by using the second type of feedback. If the student is still unable to come up with the correct answer, the correct answer could be provided with the possibility of the student to choose to learn more, in which case the third and forth types of feedback could be provided. Working with levels of feedback is good because it is concise therefore not overwhelming for the student, but still offers the possibility for more information to be provided. It is important for the feedback not to be long. Long feedback is not read therefore not useful.

Aljaafreh and Lantolf (1994), provide a detailed approach that uses different levels of feedback to guide the learner to the right answer. In their approach, there are eleven levels:

1. Feedback asks the student to read and find the errors and correct them.
2. Feedback indicates that something may be wrong in a segment.
3. Feedback rejects unsuccessful attempts at recognising the error.
4. Feedback narrows down the location of the error.
5. Feedback indicates the nature of the error, but does not identify the error.

6. Feedback identifies the error.
7. Feedback rejects unsuccessful attempts at correcting the error.
8. Feedback provides clues to help the learner arrive at the correct form.
9. Feedback provides the correct form.
10. Feedback provides some explanation for use of the correct form.
11. Feedback provides examples of the correct pattern when other forms of help fail to produce an appropriate responsive action.

Human-computer interaction (HCI) research provides some guidance concerning the feedback. Regarding the timing of the feedback, only one problem at a time should be presented to the user. Short response times for the feedback are quick and enjoyable while longer response times motivate the students to exercise more care and make fewer errors. (Shneider , 1998).

Feedback of a certain kind can act as reinforcer. Reinforcement, to be powerful, should abid by a set of principles (Lieberman, 1990):

1. Reinforcement should not be given after each answer. It should be used intermittently. This reminds me of something that has been said when I took the psychology course: employees work with more motivation with a boss who gives promotions on a random schedule as opposed to one who gives promotions after every constant amount of time.
2. For simple tasks, high motivation is helpful; on more difficult tasks, strong motivation can narrow attention in a way that interferes with learning.
3. In case where a specific response is targeted through reinforcement and that response is difficult to learn, it may be possible to shape it by first reinforcing a simpler response and only gradually requiring closer approximations to the target behavior.

Reinforcement strengthens feedback, but only when used wisely: “positive reinforcement with value-judgement phrases (excellent …) did not improve performance or satisfaction in an arithmetic drill and practice lesson. On the other hand, the presence of a simple numerical counter (6 correct, 2 incorrect) improved learning.” (Shneiderman, 1998)

It should be mentioned that on the opposite side of reinforcement, punishment does not contribute in effective language learning: “reinforcement changes the probability of a response’s recurring, but punishment does not.” (Thorndike, 1913).

Once the system is able to perform error analysis and give feedback, it makes sense to evaluate its consistency and efficiency. In evaluating error detection systems, a set of evaluation measures must be considered (Leacock et al. 2010):

1. Precision: Precision is the proportion of flagged items that are, in fact, usage errors. It measures how often the system is correct when it reports that an error has been found.

Precision = Hits/(Hits + False Positives)

• A hit occurs when the system detects an actual usage error —> detects *a* missing in *I am going for walk.*

• A false positive occurs when a non-error is flagged —> suggests deleting *a* from *I am going for a walk.*

2. Recall: Recall is the proportion of actual usage errors that have been flagged. It measures the system's coverage, i.e, the fraction of errors that the system has detected.

Recall = Hits/(Hits+Misses)

• A miss occurs when the system fails to flag a usage error —> No error reported in *I am going for walk.*

3. F-score: Researchers often combine precision and recall into an F-score, which is the harmonic mean of the two values.

F-score = 2*Precision*Recall/(Precision+Recall)

Evaluations of different systems should be consistent and comparable, hence the evaluation should follow a standard checklist for consistent reporting of the system results (Leacock et al. 2010):

~ Report Precision and Recall, not just F-scores, as they are easier to interpret. F-scores are harder to interpret as they make the assumption that precision and recall have the same importance.

~ Use more than one evaluator/annotator whenever possible.

~ Since few researchers use the same evaluation data, describe the properties of your test corpus including: the population from which the learners were drawn, their proficiency levels, the size of the corpus, and the frequencies of the error types.

~ Be explicit about the usage errors your system targets. For example, for preposition correction, list all of the prepositions it targets.

~ Be explicit about what usage contexts have been included/excluded. For example, if a noun phrase already has a non-article determiner, like some, this automatically precludes adding an article. When reporting on article detection, make it clear whether you are counting non-insertion of articles in these noun phrases as a correct prediction.

Problem Addressed by the Research

The research was suggested as a reaction to the absence of work done for NLP based CALL for the Arabic language. While English and German were both the target of many research conducted in this field, Arabic did not receive as much attention. Among the first points mentioned in the web page of the Stanford NLP group is that:

"Arabic is the largest member of the Semitic language family and is spoken by nearly 500 million people worldwide. It is one of the six official UN languages. Despite its cultural, religious, and political significance, Arabic has received comparatively little attention in modern computational linguistics."

http://nlp.stanford.edu/projects/arabic.shtml

Before finding out about the Arabic GramCheck paper (Shaalan, 2005), evidence seemed to point towards a total absence of work done concerning an Arabic NLP based CALL application. Even the work done in the Arabic GramCheck is only part of what is done in NLP based CALL as it focuses on general grammar checking for Arabic rather than grammar checking for Arabic that is meant for an educational purpose. The difference is that the Arabic GramCheck is concerned about the practice of NLP of Arabic in word processors rather than CALL applications. Not surprisingly, the word "feedback" is only mentioned once in Shaalan's paper as feedback is certainly not as important in a word processor as it is when it comes to helping users learn a new language.

The lack of work done for NLP based CALL in Arabic should not be confused with a lack of work in CALL concerning Arabic. A quick search on the net is enough to find that there are lots of applications available for learning Arabic. The problem with these applications is the same that has been mentioned about all CALL applications: the lack of intelligence. They are CALL applications but not ICALL applications. The only difference between them and exercises written in workbooks is that they display exercises from a screen and need answers to be typed instead of written, they have no interaction besides maybe the visual effects and the sounds they produce and therefore the purpose of taking advantage of the computer is not fulfilled.

The significance of this research is in being a contribution to fight the silence that reigns over NLP based CALL in Arabic, be it with an inaudible little whisper. And so to produce such an insignificant whisper, a prototype that detects gender and number agreement errors in the Arabic Grammar and provide feedback for these errors will be implemented.

Problem Presentation

The problem, as already mentioned, is the lack of progress made in Arabic language learning that uses NLP. Also as already mentioned, NLP based CALL proved beneficial and enjoyable for learning languages, for a language spoken by hundreds of millions of people to lack behind in this field should be a topic of concern. Grammar checkers are far from perfect in all languages (Shaalan, 2005), but it is by far better to have imperfect work progressing towards perfection than not having any work at all.

The lack of progress concerning Arabic is to high degree due to the complexity of its grammar. The Arabic GramCheck paper describes a set of sources for this complexity (Shaalan, 2005):

1. The length of the sentence and the complex Arabic syntax.
2. The omission of diacritics (vowels) in written Arabic 'at-tasˇk¯ıl'.
3. The free word order nature of Arabic sentence.
4. The presence of an elliptic personal pronoun 'ad-damir al-mustatir'

Enough have already been mentioned about the problem addressed, for any clarification please refer to the previous section.

Procedure of the Research

Having in mind that building a prototype that recognises gender and number agreement errors in the Arabic grammar, and ICALL being a field of investigation that combines research and development, it makes sense to split the procedure as part research and part development.

In the theoretical part of the research, documenting has been done concerning the context of the research. The documenting covered previous work and examples of such work like the German Tutor (Heift & Schulze, 2007) and the Arabic GramCheck (Shaalan, 2005). Enough understanding was developed concerning the contributions of NLP in language learning, the costs involved with combining NLP with CALL. Additional reading enabled the understanding of the error analysis process and the criteria involved in providing good feedback. Details about the theoretical information gathered is mentioned in the "Background Research" section.

Considerable amount of information has been acquired throughout the theoretical part of the research. Still, developing the prototype requires more than theoretical knowledge, and so transitioning to practical knowledge was needed.

In the development part of the research, the first step was to learn Python. The choice of Python as programming language is due to the availability of resources like the natural language tool kit (NLTK), the simplicity of the language and the ease of use were a great plus. Building a parser involves parsing algorithms and construction of grammar rules. The second step was to understand algorithms like the Earley, the CKY and the Chart algorithms. To be able to parse some text, there is a need for grammar rules, this is where treebanks come in. Treebanks are corpora of syntactically annotated texts, they are used to extract the grammatical rules of languages. An example of a famous treebank projects is the Penn Treebank project which produced treebanks for Arabic among several other languages. Context-free Grammars can also be generated manually, that is from an understanding of the grammar of the language. (Jurafsky & Martin, 2009) A third step was to write a BNF for Arabic, which took a while as every rule I came up with was either too general or too narrow as to find exceptions for those rules. One of the attempts to produce the BNF was:

Jumla —> Jumla Ismia | Jumla Fialiya

Jumla Ismia —> Mubtada | Xabar

Mubtada —> Ism Mubtada | Damir Mubtada | ε

Xabar —> Naat | Fial | Ism

Jumla Fialia —> Fial | Fial Jumla Ismia

Other attempts were made, but none was deemed good enough to rely on. I was trying to do so many things from scratch and all of those things probably took years to develop for the already available resources.

The Arabic language is a rich language and recognizing the words of a sentence is not intuitive, a morphological analyzer is used for this purpose. The Buckwalter Arabic morphological analyzer was a good one to look at. Transliteration is part of what the Buckwalter analyzer provides and it is useful as far as it prevents some of the burden of working with Arabic characters by using equivalent latin letters for them. Arabic is a language where different parts of speech can be combined in one word, this calls for the need of a stemmer. Some of the available stemmer for Arabic are the "Khoja", "ISRI" and "Light" stemmers. To be able to parse a sentence you also need a sentence splitter to split sentences from a given text. It was the fourth step to get familiar with the available tools for stemming, morphological analyzing, transliterating, sentence splitting. And this is the step where I started getting aware of how ignorant I am compared to the task at hand. It was intimidating to face all these tools, the results of the Buckwalter morphological analyzer alone were enough to feel very far away from the implementation of the prototype given the time of the summer semester. So it was the fifth step to brainstorm how to figure out a way around the obstacle of dealing with so many tools in such few time. I could not read enough to be able to comfortably integrate the tools involved, but I read enough to get inspired to implement a quick and dirty solution. Very dirty.

The solution was to build a small dictionary with enough words to allow testing of the prototype. To get around morphological analysis, the dictionary contained different forms of every word as an alternative to only including the stem of words and leaving it to the program to figure what form of the word it is dealing with from the parts of speech attached to the stem. For example, the dictionary contains the word ولد and الولد and ولدان and الولدان instead of only having the word ولد in the dictionary and figuring the features of the word from "ال", and "ان" in the program. Again, I wasted a lot of time in trying to learn to use the available tools that is why I had to shortcut my way, and even such a thorny way seemed better than remaining stuck trying to learn to use all the tools.

Once the dictionary was filled enough, the implementation of the program used it to identify the structure of the phrases entered, then used it to determine the features associated with the tokens of these phrases. The structures that can be recognised are:

- Ism Naat
- Damir Ism Naat
- Fial Ism
- Fial Ism Ism
- Fial Ism Naat
- Fial Ism Naat Ism

The features include the gender and number of the words as well as whether they are determined or not. In cases of nouns, part of the features is whether they are animate (relate to humans) or inanimate (relate to animals or objects). In cases of verbs, the features include whether they are personal, accusative, or absent. The personal form of a verb accords with pronouns like انا and نحن. The accusative form accords with انتم انتما انت انتن. The absent form accords with هو هي هما هم هن.

After the implementation of error recognition of gender and number agreement, next was to provide feedback. For the feedback, different sentences were stored to be displayed according to the errors made, with the special case of inanimate nouns which included a reminder to the rule of the agreement with the plural form of inanimate nouns.

Interpretation of Results

It was part of the research that the parser should be either found or built. I decided I won't even search for a parser because I would build my own Arabic parser. Later on in the midst of the research, I came upon the Stanford parser. I played around with it and came to conclude that it works marvellously well. But it's hard to let go of ambitions especially when you are ignorant about your own ignorance, so I walked away waving towards the Stanford parser convincing myself that I would reinvent the wheel of the Arabic parser and take pride in reinventing such a complex wheel, that's how ignorant I was. I don't regret my decision even now when I look at the mess I made of recognising some small Arabic phrases with plenty if statements that could be considered a crime against CPUs.

During a class of the NLP summer school Dr. Karim Bouzoubaa, when testing the SAFAR project kept replacing long sentences with shorter ones so that results won't take a long time to be generated. The SAFAR project integrates different available tools like the Stanford parser, the Buckwalter analyzer, the ISRI stemmer and others, and even with those tools one has to worry about the processing time. My prototype would take more than eternity to generate results if it grew enough to include different words and different sentence structures. This is due to the huge length of the dictionary which resulted from getting rid of the need to determine the features of the words from the parts of speech attached to them and filling the dictionary with many forms of the same word. Besides, the prototype uses cases to determine the structures of the phrases instead of using a CFG to do the same work more efficiently and with less processing time. Again, this is due to the fact that it would have taken me a lot more time than available to implement things the way I think they should be implemented and to the fact that I lost a lot of time aiming towards the ideal implementation.

The following table illustrates some sentences with the feedback generated for them:

Sentences	Generated Feedback
الكتب صغيرون	You made a mistake in the gender and number agreement between الـكتب and صغيـرون Remember: الـكتب is inanimate so صغيـرون should be singular female
هم بنت صغيرون	You made a mistake in the gender and number agreement between هم and بـنت You made a mistake in the gender and number agreement between بـنت and صغيـرون
رسمتِ الطفل الشارع	You made a mistake in the gender agreement between رسمتِ and الـطفـل
رسمتِ طفل الشارع	No errors detected

In the case of الكتب صغيرون, the word الكتب is inanimate and so the adjective following it should agree with its plural form by being in a singular feminine form.

For هم بنت صغيرون, the disagreement is in gender and number between the Damir and the Ism as well as between the Ism and the Naat.

Because الطفل is determined in رسمتِ الطفل الشارع, it is concluded by the program that الطفل is the subject of the verb رسمتِ. Thus the feedback is generating an error in the gender disagreement between الطفل and رسمتِ.

In رسمتِ طفل الشارع however, طفل is not determined. The program concludes that طفل here is the object while تِ is the subject. Thus no disagreement is detected.

Other words could be used to construct the same phrase structures as long as they are added to the dictionary with their types and features.

Conclusions

The work done in terms of documenting was enough to develop good understanding of NLP based CALL and assemble a detailed picture of what needs to be done in terms of the implementation and the application of the concepts learned about for Arabic. While there is a gap between the ideal picture of how the implementation should be compared to how the prototype was implemented, the gap is due to decisions made to provide a working prototype within the deadlines granted.

Clearly the prototype is limited in recognizing the gender and number agreement errors since it fails in terms of generalising error detection to a large set of sentence structures. I admit that between research and development, most of the time had to be devoted for research even when the development phase started, as I had to learn about plenty of tools that are all necessary for ideal implementation.

I don't attribute the weaknesses of the prototype to my inability in developing something better, but rather to my inability in developing something better given the provided time. I believe so mostly because I either chose to or had to work from scratch in different steps (learning Python, trying to develop a CFG for Arabic, trying to implement a parser, …) which slowed me down and consumed a lot of time, and partly because of ego.

The prototype was an experiment good enough to see with clarity an ideal architecture of the implementation, and to also see the different difficulties that might be faced as well as how to overcome them. The results I was able to produce could have very few significance, but building enough confidence to be able to believe that I can do a lot better is for me a non negligible (even though selfish) significance.

As a last word, I would like to thank you professor Cavalli. You provided me with very good guidance and very good books. You made me see how research can be enjoyable.

Future Work

I believe I acquired enough information that, with enough time, I can build something a lot more efficient. I have a lot of free time ahead of me and can easily predict that I will be bored during most of it, I'm eager to use that time to continue working on the ideal implementation of the program.

Some of the ideas I can think of for future work are:

- A full implementation that uses available tools to recognize gender and number agreement errors for longer and more varied and complex sentences.
- Recognizing other errors.
- Exercises shaped to take advantage of this work.
- Implementation of different levels of feedback.
- Using reinforcement for feedback.
- Developing a graphical user interface.

References

Aljaafreh A. Lantolf J. (1994). "Negative Feedback as Regulation and Second Language Learning in the Zone of Proximal Development." *The Modern Language Journal.* 465-483.

Al-Yaari, Al Hammadi and Alyami. Written Grammatical Errors of Arabic as Second Language (ASL) Learners: An Evaluative Study. *International Journal of English Language Education.* ISSN 2325-0887. 2013, Vol. 1, No. 2, Special Issue.

Corder, Steven Pit. (1981). "Error Analysis and Interlanguage." Oxford University Press.

Garret N. (1987). "A Psycholinguistic Perspective on Grammar and CALL." *Modern Media in Foreign Language Education: Theory and Implementation.*

Holland, V. Mellissa, Maisano, Richard, Alderks, Cathie, & Martin, Jefferey. (1993). "Parsers in Tutors: What Are They Good For?" *CALICO Journal*, 11(1), 28-46.

Heift, Trude, and Mathias Schulze. "Errors and Intelligence in Computer-assisted Language Learning: Parsers and Pedagogues." New York: Routledge, 2007. N. pag. Print.

Higgins J. (1987). "Can Computers Teach?" *CALICO Journal, 1(2)*, 4-6.

Leacock C. Chodorow M. Gamon M. Tetreault J. (2010). "Automated grammatical error detection for language learners." Morgan and Claypool Publishers.

Levy M. (1997). "CALL: context and conceptualisation." Oxford: Oxford University Press.

Nagata N. (1996). "Computer vs. Workbook Instruction in Second Language Acquisition." *CALICO Journal, 14(1),* 53-75.

Juozulinas H. (1994). "Errors in the composition of second year German students."

Jurafsky D. & Martin J. (2009)."Speech and Language Processing. An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition." Second edition. Pearson Education.

Shaalan. K. (2005) "Arabic GramCheck: a grammar checker for Arabic."

Schneider, David, & McCoy, Kathleen F. (1998). "Recognizing Syntactic Errors in the Writing of Second Language Learners." (Vol. 2, pp. 1198-1204)

"The Stanford NLP Group." The Stanford NLP (Natural Language Processing) Group. Web. 14 June 2015.

Thorndike E. (1913). "Educational Psychology. The Psychology of Learning." New York: Teachers College Press.

Wolf D. (1993). "New Technologies for Foreign Language Teaching." *In Foreign Language Learning and the use of New Technologies.* Conference Proceedings. London 1993. Brussels: Bureau Lingua.