Character Mapping for Cross-Language

Mazin Al-Shuaili and Marco Carvalho

***Abstract*—Out-of-vocabulary words are a significant challenge for cross-language information retrieval. Names of people constitute a large portion of out-of-vocabulary words, as there are different methodologies to match names that are written in various languages. Some of the methods convert names to phonetic codes, such as Soundex, or transliterate names from one language to another. We propose a technique to map characters automatically from different languages into English, without human interference and without prior knowledge of the language. This technique can provide a statistical or phonetic model that can be used later for name comparisons or named transliterations into a cross-language. The method also generates Soundex codes for the source language based on English Soundex codes. We implement this technique for five languages: Arabic, Russian, Urdu, Hindi, and Persian. Five Soundex tables are provided as the result of this technique.**

***Index Terms*—CLIR, data linkage, IR, name matching.**

I. Introduction

To manage out-of-vocabulary (OOV) words such as personal names for cross-language information retrieval (CLIR), we need to map characters between languages. There are two different ways to map letters between two languages: 1) direct mapping, in which a letter from one language is mapped directly to a letter of the other language; and 2) indirect mapping, in which a character of the first language is mapped to a symbol or a number that represents the letter sound. Subsequently, a letter with the same sound in the second language is mapped onto the same symbol or number. This mapping helps transliterations of OOV names to have the same or, at least, very similar pronunciations in any language. Character mapping means that one can map $\alpha \longleftrightarrow \beta$, where α is a letter in one language and β is a letter in another language, and each will produce a similar sound.

Handcrafted linguistic rule and automated character mapping exist for name transliteration between languages [1]. Most studies on letter mapping are part of research that is conducted on name transliteration, name extraction, and CLIR; and no comprehensive studies currently focus on letter mapping alone. Therefore, we propose a multilingual character mapping technique that maps letters from one language (the source language) to English (the target language). The experiment was applied to five languages Arabic, Russian, Persian, Hindi, and Urdu, but the method is not limited to these languages. Understanding language types is necessary for implementing this process. Different types of writing systems can affect the results of our process; some of them can be applied to our technique easily, while others may not. Those writing systems will be discussed later in this document.

The central idea of letter mapping is to look at names written in English and their transliteration in the other languages. Then, the algorithm studies the letters of those names in both languages. Of course, these studies must be done on a large number of names for definite accuracy. We have names in English and Arabic, but our aim is beyond these two languages. We include four more languages: Russian, Urdu, Hindi, and Persian. The reverse transliteration is used to transliterate all of the names written in English to the other languages. This will ensure that we have new datasets containing the names in English and their equivalents in the other languages. Bing Translate and Google Translate have been used as a baseline by some researchers [2], [3]. Therefore, we found Google Translate a sufficient tool for translating the English dataset. Fortunately, Google has added many names into Google Translate, including many names that have spelling variations. For instance, Mohd, Mohammed, and Muhammed are all translated to “محمد” in Arabic. Some alternative spellings, such as Mahammed, do not exist in Google Translate, however. We used Google Translate to convert our dataset to the four languages mentioned earlier, excluding Arabic. Our dataset has English names and their equivalent names in the Arabic alphabet. The new dataset contains two fields: English-name (EN) and source language-name (SLN). We use English as a target language and all other languages as source languages.

The experiment shows that our technique provides a phonetic and statistical model of character mapping for five different languages. The technique is not meant for name transliteration, rather this technique is a prior step that provides some statistical relationship in cross-language character mapping.

The implementation of this technique has three main contributions: 1) it can automatically generate statistical analysis for character mapping between languages. The technique can replace handcrafted techniques such as the one that is mentioned in [4]. 2) It can used to map characters from different languages and generate a Soundex table in which all of the characters that have the same sound or a sound that is close enough share the same code. 3) It can be used to find the distance between characters of the same language or between languages. The technique can contribute to CLIR for OOV words, especially for proper names.

II. Background

Most languages have writing systems that transfer a language into a drawing on paper, a computer, or any surface. Dictionary.com defines a writing system as

Manuscript received October 5, 2015; revised December 10, 2015.

The authors are with the Florida Institute of Technology, Melbourne, FL 32901 USA (e-mail: malshuaili1994@my.fit.edu, mcarvalho@cs.fit.edu).

doi: 10.18178/ijfcc.2016.5.1.436

"The set of glyphs used for representing a given human language in written form, generally along with their conventions for use."

Reference [5] has another definition of a writing system:

"The process or result of recording spoken language using a system of visual marks on a surface."

Visual marks go beyond spoken languages, and they are a convention that might constitute a particular form of communication, such as traffic symbols or musical notations [6]. Icons in graphical user interfaces are another example of visual marks. These marks refer to specific actions, regardless of the reader's language. Spoken languages are classified into three writing systems: the logographic system, the syllabary system, and the alphabetic system [5]-[7].

Logographic writing systems (ideographic or logosyllabic) use symbols that tell the meanings of words but not the phonemes [5]. They are similar to traffic signs, which represent phrases but not pronunciation. Logography is an ancient writing system that existed in the 4th millennium BC, and some languages using logography include the Sumerians' cuneiform and the Egyptians' hieroglyphics [5]. Most languages that use logographic writing systems have become extinct. The oldest one that still uses this type of system today is Chinese. For instance, the symbol "人" in Chinese means "people" in English.

Syllabary writing systems are phonetic systems in which each symbol represents a different phoneme. A single symbol in a syllabary writing system consists of a consonant (C) followed by vowels (V) or a single vowel. This symbol can also be made up of CVC [5], [8]. For instance, if a language has 15 consonants and 5 vowels, then that forms 75 symbols for CV. It becomes more complicated for CVC languages with the same number of consonants and vowels, which can reach up to 1125 symbols [5]. We can generalize the structure symbols of syllabary writing systems using a regular expression, such as C*VV*C*. The Japanese language uses CV, while Korean and Akkadian use CVC [9]. In contrast, the previous paragraph mentioned that the Chinese writing system is logographic, but [7] revealed that part of Chinese writing system reflects "a unit of pronunciation." Therefore, the Chinese writing system is classified as morpho-syllabic, where "morpho" represents the morpheme, for the logographic writing system, and "syllabic" represents the syllabary writing system. The Korean language builds syllables for all phonemes as blocks from the existing alphabet based on C*VV*C*, for example, a syllable 벚 consists of three other syllables: {{top-left, ㅂ}, {top-right, ㅓ}, {bottom, ㅈ}}.

Alphabetic writing systems consist of letters (symbols) that might each represent different phonemes. This type of writing system is divided into two groups: abjads (consonant alphabets) and alphabets. Abjad writing systems usually have consonants for letters and diacritics for vowels, or a combination of consonants and diacritics [8]. Arabic and Hebrew contain diacritics. Unlike abjads, alphabets have letters that represent both consonants and vowels, for example, the English alphabet has the vowels "A," "E," "I," "O," and "U." Languages that use alphabets might have a single letter with different phonemes or combination of letters that produce a single phoneme. For instance, "ch," "th," and sh have a single sound in English, while "C" and "K" might represent the same phoneme as well. Some other languages, such as Hindi, have diacritics that are attached to letters. They are considered alphabetical languages, but sometimes they are called syllabic alphabets [8].

Alphabetic writing systems and logographic writing systems are phonographic, meaning that each character has its own sound. Most of these languages belong to phonographic writing systems, with some exceptions. Japanese (hiragana), which uses a syllabary writing system, uses about 2000 Chinese characters known as kanji [8]. For instance, the Japanese name 浅原, which is transliterated to ASAHARA by Google, uses Chinese symbols that are logographic. Therefore, the Japanese language uses a combination of syllabary and logographic writing systems, which makes Japanese writing more complex.

Moreover, there are two other ways in which languages might not conform to each other: 1) the writing direction of the languages can vary by language; and 2) the word structure can differ from one language to another. For example, Arabic connects letters to each other to make a word, while English separates words with spaces. Neither word structure nor writing direction add any complexity for character mapping. Consequently, we consider these differences to be outside the scope of this experiment.

III. Experiment

Personal Identity Matching (PIM) is our internal project at Florida Institute of Technology (FIT). The aim of the PIM project is to identify the similarities between personal names when the names can be written in different languages (cross-language). The PIM's preliminary tests are limited to two different types of evaluation: English-to-English and English-to-Arabic. Its results encouraged us to further our investigation and implement other languages. Unfortunately, our knowledge is limited to certain languages. Consequently, we came up with a technique to transliterate names in the English alphabet to another language, so that we could then map the characters in both languages.

We apply our technique to languages that use alphabetic writing systems. Of course, it cannot be implemented for logographic languages such as Chinese because they do not have symbols that represent the sounds needed to map with English characters. The second type of languages, which uses syllabary writing systems, may or may not be utilized with this technique. For instance, some Japanese names are written in Chinese symbols that cannot be mapped to English, as mentioned earlier. We have targeted five languages: Arabic, Persian, Russian, Hindi, and Urdu.

TABLE I: SIZE OF SUB-DATASET BEFORE AND AFTER TRANSLATION

Dataset	Original Names	Translated by Google	Mapped Names	Char. / Size
Arabic	50087	N/A	50061	≈1565
Hindi	205488	14417	14385	≈342
Persian	47350	2415	2395	≈73
Russian	3825	2427	2418	≈81
Urdu	47350	3127	2860	≈89

Our aim was to map characters from a source language to English characters. To map between source languages and English, we need a list of names for each source language. Fortunately, we have a dataset containing names in English characters along with nationality. We created sub-datasets from the original dataset based on nationalities. The table below (Table I) shows the size of these sub-datasets in the column “Original Names” for each language. All of the sub-datasets thus contained English names, and we needed to add their equivalent names in the five languages. We used Google Translate to translate these names into our desired languages. Generate Dataset

We used Google Translate (translate.google.com) to find the equivalent names for the names in our datasets. We used different datasets that contained names in the English alphabet, as mentioned earlier. Each dataset was sent to Google Translate in order to generate a new dataset that included the original names in the English alphabet along with their equivalent names in the source language. The number of names in the new dataset generated by Google Translate was lower than the original set due to the limitations of Google Translate. Table I shows the size differences between the original dataset and translated ones. We use Google Translate (translate.google.com) to find the equivalent names for the names that were in our datasets.

Unfortunately, Google Translate could not handle all of the names. For example, the name “Yazen” could not be transliterated into any language; Google Translate returned the same characters, “Yazen.” Thus, the column “Translated Names” in Table I has a lower number than the column “Original Names” because we ignored all names that could not be translated. Also, some of the names, such as the Arabic name, “سعيد,” have meanings that might be transliterated as “Said.” If we use Google to transliterate this name back to Arabic, we will get “قال”, which is the literal meaning of “said”, the past-tense verb form of “say” in English. As a result, “قال” and “Said” appear in the new dataset instead of “سعيد” and “Said”. Since this problem occurred during automatic translation, it was difficult to find at that stage. We consider this an error in literal translation (transliteration), and it will be discussed in the next section.

The column Char/Size in Table-I shows the probability of each character occurrence in some names. For example, the character X can exist in 81 names within the Russian dataset. Of course, a name does not include all characters, and characters are distributed throughout all names. The numbers of character occurrences are good for mapping characters of two languages.

A. Mapping Process

We use two different mapping processes: first we compare only the first characters in both names. The system generates a contingency table that contains rows for the English alphabet and columns for the source language. The number in the cell is the frequency of both characters that appear together as first characters. For instance, if a row “B” and column “β” have a value of five, then that means both B and β exist as first characters in five names. Moreover, a character can be matched with other characters with low frequency due to errors in Google Translate, as explained in the previous section. Therefore, we used the three equations below to reduce such errors. We assume that if the value of E_R or S_R is greater than 30%, then both letters are added to mapping list; otherwise, the harmonic mean (R_{mean}) is calculated, as in (3). If R_{mean} was greater than 10%, we inserted both letters into the mapping list. In the equations below: e_{ij} is the value in the cell of the contingency table, E_j is the total number of English characters in i row, and S_i is the total number of source language characters in the j column.

$$English\ Rate(E_R) = \frac{e_{ij}}{E_j} \tag{1}$$

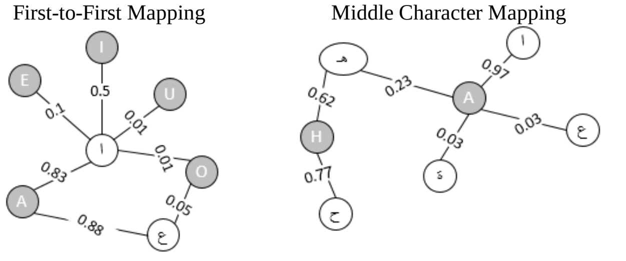
$$Source\ Language\ Rate(S_R) = \frac{e_{ij}}{S_j} \tag{2}$$

$$Mean\ Rate(R_{mean}) = \frac{2(E_R * S_R)}{E_R + S_R} \tag{3}$$

The second mapping process compares middle characters for which all characters ≠ first character. This comparison is more complex, due to the inequality between two name lengths. In other words, if the length of an English name (*EN*) is not equal to the length of a source name (*SN*), then there is a high probability that a character in EN_i does not match a character in SN_i, but it might match with SN_j where j is between 1 and the length of *SN*. Consequently, the frequency number increases based on the existence of both characters in both names, regardless of their positions. For example, “John” is translated to “Джон” Russian; the number in the contingency table increases by 1 for “O” with all characters in “Джон”: {{o, ж}, {o, o}, {o, н}}, {{h, ж}, {h, o}, {h, н}}, and {{n, ж}, {n, o}, {n, н}}. The following steps show the process of mapping middle characters:

1) Generate a contingency table.
2) Find two characters with the highest frequency.
3) If $ER \geq 0.3$, $SR \geq 0.3$, or $R_{mean} \geq 0.1$, then add both characters into the mapping list.
4) Remove both characters from the names where they exist.
5) Exit if the contingency table does not have better value.
6) Otherwise, go to Step 1.

Note: E_j and S_i are the total numbers of the first generated contingency table.


First-to-First Mapping

Middle Character Mapping

Fig. 1. Some of Arabic-English character mapping.

B. Mapping Result

The first language used for character mapping was Arabic because we are familiar with it. The results were excellent for first-to-first characters and middle characters. Fig. 1 shows the mapping result of some characters along with the matching probability. For example, in Fig. 1, first-to-first mapping shows that the Arabic character “ا” is closer to “A” than it is to U, O, E, or I. This is correct because “ا” is mapped to “A” unless it contains a diacritic that changes its sound to be one of the other characters. Arabic speakers can distinguish the pronunciations of names without using diacritics, but an error can occur when people transliterate a name. For instance, the name “أسامة” might be transliterated as Osama, Usama, or Ausama, and these three names are translated to “أسامة” by Google Translate. The name “Ausama” contains an “A” because an alif (أ) is close to “A,” but in this name, which has a diacritic called a dammah (ُ), the sound changes from “A” to be closer to “O.” Diacritics in the Arabic language make each character have multiple sounds, which increases the number of vowels in names when they are transliterated into English.

TABLE II: ARABIC, HINDI, PERSIAN, RUSSIAN, AND URDU SOUND GROUPS BASED ON ENGLISH SOUNDEX

G	English	Arabic	Hindi	Persian	Russian	Urdu
0	A,E,I,O, U,W,H, Y	ا,ي,ه,و,ح,ع,ى,ؤ,ئ,ة	अ,आ,ए,ऐ,औ,ऑ,ई,ह, इ,ऊ,ओ,उ,व,य,	ا,ع,آ,ح,ه,و,ى,	А,И,Е,О,У,В,Й,Х,Э,Я,Ю,Ё,	ا,ع,ح,و,ى,
1	B,F,P,V	ب,ف,	ब,भ,फ,प,व,	ب,ف,پ,و,	В,Б,Ф,П,	ب,ف,پ,و,
2	C,G,J,K, Q,S,X,Z	س,ز,ش,ج,ك,ص,ق,خ,غ ,	च,छ,ग,घ,ज,झ,क,ख,श ,ष,स,	چ,غ,ق,گ,ج,خ,ک, ث,س,ش,ص,ذ,ز, ض,ظ	С,К,Г,Ш,З,Ч,Ж,Ц,	چ,غ,گ,ج,خ,ق,ث,س,ش,ص,ذ,ز, ض,ظ,
3	D, T	د,ت,ط,ث,ض,ذ,ظ,	ड,ढ,द,ध,ट,ठ,त,थ,	د,ت,ط,	Т,Д,	د,ڈ,ت,ٹ,ط,
4	L	ل,	ल,	ل,	Л,	ل,
5	M, N	م,ن,	म,न,	م,ن,	Н,М,	م,ن,
6	R	ر,	र,ऋ,	ر,	Р,	ر,

Surprisingly, we got exactly the same results for Automated Arabic Soundex groups that created by this technique with the results of Arabic Soundex groups that we created manually for the PIM project. Extra characters {ة ,ى ,ؤ} were converted “ة” to “ؤ” “,ه” to “و,” and “ى” to “ي” in our manual mapping for the PIM project [11]. Reference [10] mentioned an exact Arabic Soundex table, and another Arabic Soundex table is called ASoundex, which has one character (ظ) in a different group [12]. Table II represents the Arabic table generated by our automated tool. This table matches with our handcrafted table that was used in PIM and the one mentioned by [10].

Beside the usefulness of this technique for generating Soundex tables automatically, it also gives more accurate statistical relationships of mapped characters between two languages. Fig. 1 displays numbers representing the probability of two characters occurring together in the entire dataset. This number can be used to define the distance between the letters. A higher number represents a shorter distance between two characters than a lower number. We can formulate the distance as $d = 1 – p$, where p is the probability that two characters appear together in the dataset. The probabilities of character mapping are implemented in many projects for name transliteration; for example, AbdulJaleel and Larkey [4] use statistical translation to convert names from English to Arabic by implementing a handcrafted model of statistical character matching. Their handcrafted model divides character mapping into three parts: the beginning, middle, and last characters of the names. They used n-gram mapping where $n \geq 1$. Our mapping technique provides the probabilities of transliterating the source language to English or vice versa. This automated technique can be utilized to generate such statistical models for names transliteration in an efficient and faster way.

Table III shows a comparison between handcrafted mappings of the one in [4] and our auto-mapping. The comparison displays the mapping of a character “A” with Arabic characters. We used two datasets that both have two attributes: Arabic name and English name. The first dataset contains names that have been transliterated from Arabic to English. The column Ar-Eg shows the results of this dataset. The dataset contains names that are transliterated from English. The Eg-Ar shows the results of the second dataset. This dataset produces slightly different results that are closer to those created by [4]. They wanted to transliterate English names to Arabic, so their mapping focused on non-Arabic names. The character “A” was matched 82% and 17% for first character mapping with “أ” and “ع,” respectively. The results in the Eg-Ar column are closer to their results than the results from the Ar-Eg column, especially for first-character mapping. In this dataset, “A” is mapped to “أ” with a probability of 0.60, and it is mapped to ع with a probability of 0.10. This result is logical because English names do not have character “ع.” It is possible that some of the names are non-English names, which may have caused this 0.1 in their statistics. The originality of names also affects the result of character mapping, as we explained earlier in this paragraph.

TABLE III: SIZE [4] (HANDCRAFTED) VS. OUR MAPPING (AUTO)

Position	Arabic Character	A&L [4]	Ar-Eg	Eg-Ar
First	ا	0.9	0.68	0.82
	ع	0.1	0.32	0.17
Middle	ا	0.6	0.5	.77
	ي	0.1	0	.001
	ع	0.1	0.20	0
	هـ	N/A	0.3	0
End	هـ	0.6	N/A	N/A
	ا	0.4	N/A	N/A

Table II contains the Soundex codes for the other languages that were generated automatically by our algorithm. Their characters are classified based on English Soundex. These codes were generated after characters from each language were mapped with English characters. The following are some observations for each language:

- There is a Hindi Soundex Table that exists with 21 characters, while Hindi has 11 vowels and 33 consonants. This Hindi Soundex table misses more than half of the characters. The existing Hindi Soundex table cannot convert all names because it is missing these characters [13]. The Soundex table created by our algorithm has 42 characters. The character “व” overlaps between two groups. Google translates names that start with “V” and “W” to “व.” The result shows “W” and “V” having probabilities of 0.89 and 0.07 respectively. Therefore, “व” is closer to being a member of group one than group zero.

- The algorithm uses the second dataset for Persian and Urdu to map both languages characters to English. Both languages have similar characters to Arabic with extra symbols. Some of the characters look like Arabic characters, but they are pronounced differently. Again, “W” and “V” were mapped to the same character, “و,” in both languages. The probabilities for the Persian “W” and “V” were 0.76 and 0.18, while Urdu had 0.86 and 0.10 for “W” and “V” respectively. The results for both languages agreed with Hindi.
- The Russian language did not differ much from the languages that preceded it. There was also a character overlap between the two groups for “W” and “V”. The Russian “B” character had a probability of 0.70 for “V” and 0.17 for “W.” The Soundex codes for Russian are the result of middle character mapping, unlike the others using first-character mapping. The Russian first character mapping has one more overlap character, “Д.” This character was mapped with “D” and “J” with probabilities of 0.60 and 0.31, respectively.

IV. Conclusion

Proper names are a major challenge for CLIR because they are OOV words [4]. Yet, these names represent approximately 30% of the content in the news [1]. We cannot depend on a dictionary alone because it is impossible to locate all names that have spelling variations. Hence, we need to deal with these names in a dynamic way, especially since personal names continue to increase. There are different types of algorithms to solve this problem by matching names phonetically or statistically. In both ways, systems need to understand the relationships between characters in different languages.

The technique that we provided in this document is a useful tool for exploring the relationships between characters across languages. The mapping between two languages is done without prior knowledge of any language; therefore, there is no obstacle to implementing name matching or transliteration for personal names in any language that we do not know. This tool helped us in our internal project to define the distance between characters and generate standard Soundex codes for multiple languages.

This technique can be improved by using a bigger dataset. The dataset must include English, which is the target language, names that have spelling variations and spelling variations in the source languages. Moreover, we can enhance character mapping by converting “CH,” “SH,” and “TH” to special characters, which are not in the English alphabet, before mapping them to a source language. This group of characters produces different sounds that do not exist with a single character in English, while many other languages match these sounds with a single character.

References

[1] B. Pouliquen, R. Steinberger, C. Ignat, I. Temnikova, A. Widiger, W. Zaghouani, and J. Zizka, “Multilingual person name recognition and transliteration,” *CORELA — Cogn. Represent. Lang.*, vol. 3, no. 2, pp. 115–123, 2006.

[2] F. Alshuwaier and A. Areshey, “Translating English names to Arabic using phonotactic rules,” in *Proc. the 25th Pacific Asia Conference on Language, Information and Computation*, 2011, pp. 485–492.

[3] P. M. and S. P. M. Faruqui, “Soundex-based translation correction in Urdu–English cross-language information retrieval,” in *Proc. Workshop on CLIA at IJCNLP*, 2011.

[4] N. AbdulJaleel and L. S. Larkey, “Statistical transliteration for english-arabic cross language information retrieval,” in *Proc. the Twelfth International Conference on Information and Knowledge Management*, 2003, p. 139.

[5] J. Fon. (2015). Writing systems. [Online]. Available: http://www.ling.ohio-state.edu/~jfon/ling201/writing_system.pdf

[6] H. Eifring and R. Theil, *Linguistics for Students of Asian and African Languages* 2005.

[7] M. Wang, K. Koda, and C. A. Perfetti, “Alphabetic and nonalphabetic L1 effects in English word identification: A comparison of Korean and Chinese English L2 learners,” *Cognition*, vol. 87, no. 2, pp. 129–149, 2003.

[8] S. Ager. (2015). The online of encyclopedia of writing system & language. [Online]. Available: http://www.omniglot.com/

[9] Lawrence Lo. (2015). A compendium of world-wide writing systems from prehistory to today. [Online]. Available: http://www.ancientscripts.com/

[10] A. H. Yousef, “Cross-language personal name mapping,” *Int. J. Comput. Linguist. Res.*, vol. 4, no. 4, 2013.

[11] M. Al-Shuaili and M. Carvalho “Personal idintity matching,” Report, Florida Institute of Technology, Melbourne, USA, 2014.

[12] S. U. Aqeel, S. Beitzel, E. Jensen, D. Grossman, and O. Frieder, “On the development of name search techniques for arabic,” *J. Am. Soc. Inf. Sci. Technol.*, vol. 57, no. 6, pp. 728–739, 2006.

[13] S. Chaware and S. Rao, “Rule-based phonetic matching approach for Hindi and Marathi,” *Comput. Sci. Eng. An Int. J.*, vol. 1, no. 3, pp. 13–24, 2011.

Mazin H. Al-Shuaili is currently a Ph.D. candidate at Florida Institute of Technology (FIT), in Melbourne, FL, USA. In 1998, he graduated in computer science from FIT. In 2000, he obtained his master’s degree in software engineering from FIT. His master had focused on software test automation. From 2000 till 2012, he was a system analyst and project manager at Omani government. In May 2016, he is going to graduate and he going back to work with Omani government, Muscat. His research interests are in general areas of Natural language processing (NLP), social network, and data mining.

Marco M. Carvalho is an associated professor at the Florida Institute of Technology, in Melbourne, FL, USA. He graduated in mechanical engineering at the University Brasilia (UnB – Brazil), where he also completed his M.Sc. in mechanical engineering with specialization in dynamic systems. Marco Carvalho also holds a M.Sc. in computer science from the University of West Florida and a Ph.D. in computer science from Tulane University, with specialization in machine learning and data mining. At Florida Tech, Dr. Carvalho is the executive director of the Harris Institute for Assured Information, and the principal investigator of several research projects in the areas of cyber security, information management, networks, and tactical communication systems.