Speed Up Strategies for the Creation of Multimodal and Multilingual Dialogue Systems

Luis Fernando D'Haro, Ricardo de Cordoba

Speech Technology Group. Dept. of Electronic Engineering. Universidad Politécnica de Madrid
E.T.S.I. Telecomunicación. Ciudad Universitaria s/n, 28040-Madrid, Spain
{lfdharo, cordoba}@die.upm.es

Abstract

In this paper we will summarize the work done in the PhD thesis that follows the same title as this document. In the thesis we propose different innovative, dynamic, and intelligent acceleration strategies applied to a development platform for reducing the design time of multimodal and multilingual dialogue systems and to improve the runtime modules. Throughout the paper we will describe the three different kinds of accelerations proposed, which are innovative with respect to current commercial and research platforms. The first kind of strategies was applied to the design platform in order to allow the prediction of the information required to complete the different aspects of the service. These strategies are mainly based on using the data model structure and database contents, as well as cumulative information obtained from the previous and sequential steps in the design. Thanks to them, the design is reduced, most of the times, to simple confirmations from the designer to the "proposals" that the platform automatically provides. The second kind of strategies is the incorporation of a new adaptation algorithm to the language models used by a machine translation system that automatically translates system's prompts (in audio or text) into an animated sequence in Sign Language for providing the designed service to deaf users using an avatar. Finally, the third kind is an innovative LID technique based on using a discriminative ranking of n-grams that allows the incorporation of contextual longer-span information into the language models used to identify the system needs to use to interact with the users of the service.

Index Terms: Dialogue Systems, Language Identification, Language Model Adaptation, Machine Translation.

1. Introduction

Currently, the growing demand of automatic dialogue services for different domains, user profiles, and languages has led to the development of a large number of sophisticated commercial and research platforms that provide all the necessary components for designing, executing, deploying and maintaining such services with minimum effort and with innovative functions that make them interesting for developers and final users. In general, both commercial and research platforms rely more or less in the same kind of acceleration strategies. For instance, the incorporation of state-of-the-art modules such as language identification, speech recognizers and synthesizers, etc., user-friendly graphical interfaces, inclusion of built-in libraries for typical dialogues, or additional assistants for debugging the service, as well as support for widespread standards such as VoiceXML or CCXML in order to increase the portability and reduce costs. However, surprisingly these platforms do not include any kind of acceleration strategies based on the contents or in the structure of the backend database that, as we will show, can provide important information to the design. The results of both a subjective as an objective evaluation demonstrate the usefulness and high acceptance of the proposed accelerations, while showing that the design time can be reduced on average by more than 56% when compared to a system without them.

Regarding the Language Identification system (LID), in the thesis we have focused on searching solutions for increasing the classification rates of detecting the user's language in the real-time system. In the thesis we proposed to use a discriminative ranking of occurrences of each n-gram with higher n-grams as language model. Our proposed ranking overcomes the state-of-the-art technique Parallel phone recognition followed by language modeling (PPRLM) (13% relative improvement) due to the inclusion of 4-gram and 5-gram in the classifier. Besides, our technique was also combined with acoustic information obtaining better results.

Finally, regarding the machine translation system, it is notorious the significant advances that these systems have reached in the last years, making it possible to face new challenges such as speech-to-speech or speech-to-sign-language translation. The later is especially useful to help deaf people to communicate with hearing people since many of them have problems when reading lips or written texts as they are used to the sign language grammar [1]. In this way, any kind of dialogue service that we can develop for hearing users has to be adapted to deaf users by using an avatar that gestures the system prompts on sign-language. On the other hand, it is well know that an efficient training of any statistical machine translation system requires a big parallel corpus in order to obtain reliable language and translation models. In the thesis we explored solutions for improving the language models (LMs) used to ensure correct grammatical sentences during the translation process. Our technique is based on adapting the original n-gram counts on the target side, using "translated" n-gram counts from the source side retrieved from the web. Our results show relative translation error reductions close to half the maximum performance obtainable when only the LM is optimized (i.e. without optimizing the translation model).

2. Description of the Accelerations

In this section we will describe briefly the main accelerations included in the platform. For further details please refer to the full thesis document[1] or corresponding papers at each section.

2.1. Accelerations to the Dialogue Design

The development platform that we have used in the thesis was the result of the European project GEMINI. The platform consists of three main layers integrated into a common graphical user interface (GUI) that guides the designer step-by-step and lets him go back and forth. In the first layer, the designer specifies global aspects related to the service, the data model structure, and the runtime functions to access the backend database. The next layer includes an assistant to define the dialogue flow at an abstract level by specifying the high-level states of the dialogue, plus the slots to ask to the user and the transitions among states, as well an assistant to

[1] http://www-gth.die.upm.es/~lfdharo/index_en.php?status=publications

define, in detail, all the actions to be done in each state (e.g., variables, loops, if-conditions, math or string operations, conditions for making transitions between states, calls to dialogs to provide/obtain information to/from the user). Finally, the third layer contains the assistants that complete the general flow specifying for each dialogue the details that are modality and language dependent. For instance, the prompts and grammars, the definition of user profiles, the error recovery logic for speech or Internet access, the presentation of information on screen or using speech, etc. Finally, the VoiceXML and xHTML scripts used by the real-time system are automatically generated in this layer too.

2.1.1. Accelerations to define the Data Model and Database access

One of the first steps in the design is the definition of the data model and the functions for accessing the backend database at runtime. Regarding the data model, the designer defines it through a visual and object-oriented representation (using classes and attributes) that provides information to following assistants about which fields in the database are relevant for the service (i.e. to provide or request information to/from the users), as well as the relationships between tables and fields.

During this step the assistant extracts information from the database contents such as the name and number of the tables, fields, and records. In addition, the following heuristic information for each field is calculated: a) field type, b) average length, c) number of empty records, d) language dependent fields, and e) proportion of records that are different. This information is used later to simplify the design or to improve the presentation of information in posterior assistants. For instance, we use them to propose which slots can be unify in order to be requested at the same time to the user, for creating automatic dialogue proposals, or to sort by relevance the information displayed in the assistants.

Finally, we have also incorporated an innovative acceleration strategy that simplifies the process of creating the prototypes (API) of the database access functions used by the runtime system, this way reducing the necessity of learning SQL and simplifying the process of adding the query into the real-time modules and scripts. The wizard semi-automatically creates the SQL statement for the given prototype and provides a pre-view of the results that the system would retrieve at runtime. Currently few development platforms include such kind of assistance forcing the designer to use third party software; however, none of these platforms provide such kind of automatic query proposals.

2.1.2. Accelerations to define the Dialogue Flow

The next step in the design is to define clearly the states, data to ask the users (slots), transitions between states, and the actions that make up each state. Since this process is the most complex one, in this layer we have incorporated the most important accelerations. Below we will describe the most interesting ones. For further information about these or other accelerations please refer to [2] or [3].

The first one is that the system automatically suggests the designer when two or more slots must be requested one by one (using directed forms) or at the same time (using mixed initiative forms) according to the VoiceXML standard. The proposal is based on the heuristic information extracted from the database contents related with the corresponding slots to ask to the user and on a set of predefined, but editable, rules. This way, for instance, if we need to ask two numeric data with a proportion of different values close to one, and the total number of records of both fields is high (configurable value), then the system determines that these slots have a large vocabulary and a high probability of misrecognition, therefore it is better to ask one slot at a time (i.e. system initiative). In case there are more than two slots in a state, the system checks different slots combinations in order to find those that can be requested together and those that need to be requested alone.

Another relevant acceleration is the creation of different configurable state/dialogue and action proposals based on the information of the data model and database access functions. This way, for instance, the assistant can propose a complete state for requesting the credit and debit account numbers and the amount of money required to perform a transaction in a banking application. In addition, thanks to the heuristics data and the information from previous assistants, the system is able to propose the designer the most probable actions to be done at each state (see *Figure 1*). In the example, the assistant proposes the following actions to complete the state: a configurable template for requesting the account numbers using mixed-initiative (the account numbers correspond with short-length user-defined aliases), then the dialogue to ask for the amount, the database function to perform the transaction, and finally a built-in dialogue to notify the user with the available balance.

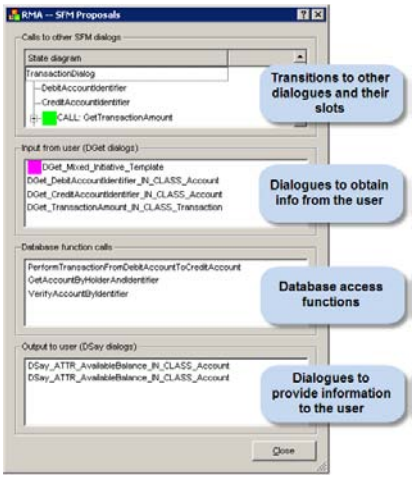

Figure 1: Example of action proposals

Another important contribution is that the platform allows the designer to create over-answering dialogues which are not currently provided by any other platform, since the VoiceXML standard only requires mixed-initiative dialogues. In order to do this, we have designed a special flow using standard elements that overcomes this limitation in the final script. In addition, the creation of these dialogues is very easy since the platform automatically proposes the slots that can be used for over-answering and automatically creates the flow.

Finally, several other accelerations are available such as a mechanism to automate the process of passing information among actions/dialogues by proposing the variables that best match the connections. This is a critical aspect of dialogue design since several actions and states have to be 'connected' as they use the information from the preceding dialogues. In addition, the platform supports the creation of different kind of dialogues, easy definition of dialogue variables, calls to other dialogs, variable assignments, a mathematical and strings assistant for including procedures, among others.

2.1.3. Other accelerations

The following step in the design is to complete the general flow specifying for each dialogue the details that are modality and language dependent. In this case, we have incorporated a wizard window that semi-automatically generates the dialogue flow for showing the lists of results after querying the database and to confirm user's answers. Another assistant

allows the designer to specify the prompts and grammars used at runtime. Here, we have incorporated an assistant that helps in the creation of stochastic language models and debugging of JSGF grammars, and that automatically creates the pronunciation dictionaries used by the speech recognizer. Finally, the platform automatically generates the runtime VoiceXML script that can be run using any voice browser or using our own runtime modules. In the last case, the runtime system uses a distributed running platform similar to Galaxy [4] and the script is interpreted using OpenVXI [5].

2.1.4. Reported Results

In order to evaluate the platform and the accelerations, we carried out a subjective and objective evaluation where several developers, with different experience levels, were requested to fulfill typical design tasks covering each assistant and the proposed accelerations. For the subjective evaluation, the participants were asked to answer several questions about the platform and the strategies. The results confirm the usability of the accelerations and designer-friendliness of the platform since all of them were marked over 8.0. For the objective evaluation, we collected the following metrics: elapsed time, number of clicks, number of keystrokes, and number of keystroke errors. We compared these metrics obtained when using our assistants with a low-level accelerated editor included in the platform. The results confirm that the design time can be reduced, in average for all the assistants and tasks, in more than 45%, the number of keystrokes in 81%, and the number of clicks in 40%.

2.2. Improvements to Language Identification

Currently one of the most used technique for LID is PPRLM [6]. In this technique, the language is classified based on statistical characteristics extracted from the sequence of recognized allophones. In spite of the good results obtained by PPRLM, one of its main problems is that the accuracy is reduced due to an unreliable estimation of the LM. In order to reduce this problem, in the thesis we proposed a new algorithm for creating and using as LM a ranking of discriminative n-grams for each language to identify. Our proposed ranking resulted in a 15% relative improvement over PPRLM due to the inclusion of 4-gram and 5-gram in the classifier. Additional improvements were also obtained by including acoustic information into the GMM classifier.

In [7] the original ranking algorithm for a text-categorization task is described. In our system, we have incorporated innovative modifications to this technique. In summary the most relevant were: a) A new definition of the rank position following what we call "golf score" i.e. all n-grams that have the same number of occurrences share the same position in the rank, b) the creation of specific rankings for each n-gram order in order to avoid to take only the n-grams in the top positions that are always devoted to the unigrams, bigrams, etc., which are less discriminative, and c) the definition of a new training procedure for ranking first those n-grams that appear most in a particular language than in the others (i.e. discriminative). Finally, in the thesis we also investigated the performance of the new ranking-based system when incorporating additional information into the classifier. In detail, we tested the following features: a) Sentence acoustic score provided by the ASR, b) Phoneme acoustic score, and c) Duration for each phoneme.

2.2.1. Reported Results

Figure 2 and *Figure 3* show the cumulative results in LID error rate for the Invoca database obtained with the previous mentioned rank modifications and acoustic information. As we can see, in both cases the n-gram ranking outperforms the PPRLM system. Our final system is the integration of both systems (PPRLM and Ranking) and all the acoustic information which resulted in a significant reduction from 3.69% to 2.52% (31,7%). Further details in [8] and [9].

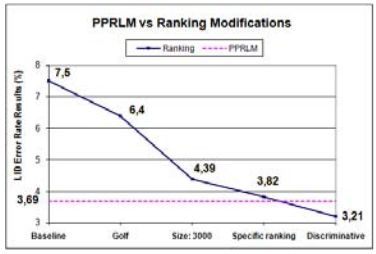

Figure 2: LID error rate results for the different changes in the original ranking algorithm in comparison with PPRLM

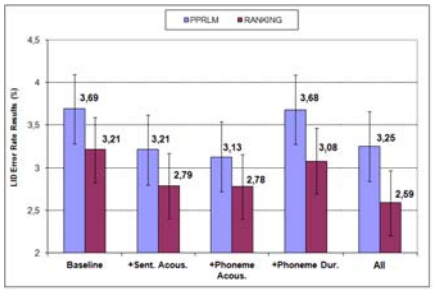

Figure 3: Comparative results between PPRLM and Ranking for adding acoustic scores into the classifier

2.3. Improvements to Machine Translation

Nowadays, most machine translation systems are trained using statistical-based algorithms that require big parallel corpora in order to guarantee a correct estimation of the translation and language models. In our case, this requirement could not be fulfilled since our target language is Sign Language (SL) and most of the currently available SL corpora are very small. For instance, [10] considers a corpus of about 2000 sentences while [11] uses a corpus of only few hundred sentences (in our case we had 266 sentences for training and 150 for test and dev). In addition, there is not any available corpus from online content as is usual in spoken languages. However, our proposed technique takes advantage of using the "source-side" language (in our case, Spanish) and from the phrase-based translation table created during the training of the MT model, in order to collect web frequency counts for the "source-side" language, using information retrieval techniques as reported in [12], and then "translating" them into "target-side" counts. The proposed technique is done in the following three steps (more details can be found in [13]):

Backward: The system uses the phrase pairs table created during the training of the translation probability $Pr(f_1^J | e_1^I)$. This table consists of a list of n-gram pairs that are consistent translations between the source and target language, with their probabilities $p(\bar{f}_i|\bar{e}_i)$ and $p(\bar{e}_i|\bar{f}_i)$, and lexical weights [14]. Using this table, the system creates a list of source-side n-grams that satisfy $p(\bar{f}_i|\bar{e}_i) \geq \theta$. Here the threshold θ reduces the number of n-gram pairs to be queried in the web, so that they are more reliable. In our experiments, θ was set as a function of the number of reverse translations for $\bar{f}_i$.

Information Retrieval (IR): Using the list of previous selected n-gram, the system queries the internet to obtain web frequency counts using the Google-API[2].

[2] http://code.google.com/apis/ajaxsearch/

3.) Forward: Finally, the translation table is applied on the opposite direction to obtain the "translated" n-gram frequency counts on the target side. The conversion is done taking each n-gram pair in the list, $\bar{f}_i$, multiplying the retrieved web count, $N^{web}(\bar{f}_i)$, by the phrase translation probability, $p(\bar{e}_i|\bar{f}_i)$, and summing up all the contributions that satisfy $p(\bar{e}_i|\bar{f}_i) \geq \delta$, with $\delta = 1/n_i$, to obtain the counts for the target n-gram, $N(\bar{e}_i)$ (see Eq. 1). Then, MAP is applied to merge the counts from the original sign corpus with the converted counts. Finally, a new target LM is created form the linear interpolation of the original LM and the adapted one.

$$N(\bar{e}_i) = \frac{\sum_{\forall \bar{e}_i : p(\bar{e}_i|\bar{f}_i) \geq \delta} N^{Web}(\bar{f}_i) * p(\bar{e}_i \mid \bar{f}_i)}{\sum_{\forall \bar{e}_i : p(\bar{e}_i|\bar{f}_i) \geq \delta} p(\bar{e}_i \mid \bar{f}_i)} \qquad \text{Eq. 1}$$

2.3.1. Reported Results

Table 1 shows the perplexities results provided by the baseline LMs and the adapted ones on train, dev, and test sets. The results for the test and dev sets correspond to the averaged perplexities of a three-fold cross validation test. The baseline LM is a backoff trigram with Good-Turing discount. The perplexities on both sides correspond to the adapted LMs. Values in parenthesis are relative improvements over the baseline perplexities. As we can see, the proposed technique provides a 15.5% relative improvement in the test set.

Table 1: *Perplexity results*

	Train	**Dev**	**Test**
Baseline	5.02	10.8	10.7
Adapted	3.16 (37.1%)	8.75 (18.7%)	9.04 (15.5%)

Table 2 shows the averaged MT results for text-to-sign translation on the test set. For the oracle experiment the LM is trained considering all sentences (train, development, and test sets). Since this model has all the available information, it corresponds to the top performance that it is possible to obtain only due to the LM component. As we can see, the results show that the proposed technique is able to reach approximately half (2.73%) of the maximum improvement (6.1%) in WER that it is possible to obtain when only the LM is improved (i.e. without improving the translation model).

Table 2: Machine translation results

		WER	PER	BLEU	NIST
Text-to-Sign	Baseline	34.74	29.59	0.50	6.30
	Adapted	**33.79** (2.73%)	**29.1** (1.68%)	**0.51** (2.61%)	**6.36** (1.06%)
	Oracle	32.62 (6.1%)	28.06 (5.48%)	0.55 (9.91%)	6.57 (4.23%)

3. Conclusions

In this paper, we have summarized the most important contributions of the PhD thesis that consist of different kinds of acceleration strategies applied to a complete development platform for designing and running dialogue applications.

The first kind of strategies are based on using heuristic information extracted from the backend database and on cumulative information obtained from the previous and sequential steps in the design. Our proposals include the unification of slots to be requested using mixed-initiative dialogues, the semi-automatic creation and debugging of SQL statements, as well as automatic action proposals for each dialogue. Subjective and objective evaluations confirm that the proposed strategies are useful and contribute to simplify and accelerate the design.

The second kind of strategy was applied to a language identification system that allows the dialogue system at runtime to detect the language to interact with the user. In this topic we have proposed a novel algorithm to create ranking templates with the most discriminative language-dependent n-grams which are then integrated into a state-of-the-art LID system for recognizing the user's language. The results show that this technique when unified with PPRLM and acoustic information results in a relative improvement of 31,7%.

Finally, the third kind of strategies was a LM adaptation technique successfully applied to a machine translation system that allows designers to translate automatically the prompts created for a traditional speech-based dialogue system into a visual sign language representation that can be used to offer the same dialogue service to deaf users.

4. Acknowledgements

In first place I want to thank God for His continuous direction and to my family for their support along my career. Special thanks to my advisor and members of my research group for the opportunity of doing the thesis, and to the different students that I have supervised in the development of the accelerations described above. And thanks to my friends for staying close to help me to never give up.

5. References

[1] Zhao, L., Kipper, K., et al. "A machine translation system from English to American Sign Language". AMTA, 2000. pp. 54-67.

[2] L. F. D'Haro, R. Cordoba, et al. 2006. "An advanced platform to speed up the design of multilingual dialog applications for multiple modalities". Speech Communication Vol. 48, Issue 8, pp. 863-887. July 2006.

[3] L .F. D'Haro, R. Cordoba, et al. 2009. "Speeding up the design of dialogue applications by using database contents and structure information". SigDial, pp. 160-169. London, UK.

[4] Seneff, S., Hurley, E., et al. 1998. "Galaxy-II: A reference architecture for conversational system development". ICSLP 1998, 931-934.

[5] Cordoba, R., Fernández, F., Sama, V., D'Haro, L. F., et al. 2004. "Implementation of Dialogue Applications in an Open-Source VoiceXML Platform". ICSLP 2004, pp. I-257-260.

[6] Zissman, M.A., 1996. "Comparison of four approaches to automatic language identification of telephone speech," IEEE Trans. Speech & Audio Proc., v. 4, pp. 31-44.

[7] Cavnar, W. B. and Trenkle, J. M., 1994. "N-Gram-Based Text Categorization". 3rd Symposium on Document Analysis & Information Retrieval, pp. 161-175.

[8] Cordoba, R., D'Haro, L. F., et al. 2007. "Language Identification based on n-gram Frequency Ranking". Interspeech 2007, pp. 354-357.

[9] Cordoba, R., D'Haro, L. F., et al. 2007. "Language Identification using several sources of information with a multiple-Gaussian classifier". Interspeech 2007, pp. 2137-2140.

[10] Chiu, Y.-H., Wu, C.-H., et al. 2007. "Joint Optimization of Word Alignment and Epenthesis Generation for Chinese to Taiwanese Sign Synthesis", IEEE Trans. Pattern Analysis and Machine Intelligence, 29(1):28–39.

[11] Stein, D., Dreuw, P., Ney, H., et al. 2007. "Hand in hand: Automatic Sign Language to English Translation". TMI 2007, pp. 214-220.

[12] Keller, F., and Lapata, M. 2003. "Using the Web to Obtain Frequencies for Unseen Bigrams", Computational Linguistics, 29(3):459–484.

[13] D'Haro, L. F., Ney., H. et al. 2008. "Language Model Adaptation for a Speech to Sign Language Translation System Using Web Frequencies and a MAP framework". Interspeech 2008, pp. 2119-2202.

[14] Koehn, P., Och, F. J., and Marcu, D. 2003. "Statistical Phrase-Based Translation", HLT/NAACL 2003, pp. 48-54, Canada.