Call Routing Based on a Combination of the Construction-Integration Model and Latent Semantic Analysis: A Full System

Guillermo Jorge-Botana[1,3], Ricardo Olmos[2,3] and Alejandro Barroso[3]
[1]Universidad Nacional de Educación a Distancia (UNED), Calle Juan del Rosal, 10, Madrid, Spain.
[2]Universidad Autónoma de Madrid, Campus de Cantoblanco, 28049, Madrid, Spain.
[3]Semantia Lab. c/ Bravo Murillo nº 38, 28015-Madrid, Spain.
E-mail: gdejorge@psi.uned.es

Keywords: call routing, call steering, natural language, latent semantic analysis, construction-integration framework, psycholinguistically motivated algorithms

Received: June 24, 2014

This study stems from a previous article [1] in which we found that a psycholinguistically motivated mechanism based on the Construction-Integration (C-I) model [2,3] could be used for call classifiers in systems based on Latent Semantic Analysis (LSA). In it we showed that with this model more robust results were obtained when categorizing call transcriptions. However, this method was not tested in a context of calls in audio format, where a voice recognition application would be involved. The most direct implication of a voice recognition application is that the text to be categorized may be impoverished and is subject to noise. This impoverishment normally translates into deletions and insertions which are semantically arbitrary but phonetically similar. The aim of this study is to describe the behavior of a complete system, with calls in audio format that are transcribed by a voice recognition application using a Stochastic Language Model (SLM), and then categorized with an LSA model. This process optionally includes a mechanism based on the C-I model. In this study different parameters were analyzed to assess the automatic router's rate of correct choices. The results show that once again the model based on C-I is significantly better, but the benefits are more remarkable when the utterances are long. The paper describes the system and examines both the full results and the interactions in some scenarios. The economy of resources and flexibility of the system are also discussed.

Povzetek: Razvit je sistem za prepoznavanje govora z namenom uspešnega iskanja storitev ali entitet.

1 Introduction

Interactive Voice Response (henceforth, IVR) systems enjoy extremely widespread use today to provide customer service (Self Service). One of their drawbacks is that many of them consist of menus that limit user options to a few items, meaning that many intentions are not described within these items. This happens, for example, when a person wants something but believes that it is not represented in the menu. This constitutes a findability problem, since the category that represents what the person wants is not found. One of the alternatives that have been used is to employ spontaneous speech recognition techniques and subsequently categorize spontaneous utterances[1] into subject areas. These techniques are usually called “call routing” or “call steering” [4]. In them, rather than choosing between several items on a menu, the person simply hears an input such as “what would you like to do?” and responds spontaneously in natural language. What the person says is recognized with the help of an Automatic Speech Recognition (ASR) module, and once converted to text it is categorized and sent to a route where the user's needs are catered for. This is beneficial in terms of busy channels (the call will only be in one of the switchboard channels for a few seconds), and also beneficial in terms of convenience, as callers will be spared the effort associated with linking the representation of what they want to the representation of the categories expressed by the menu items, something which is not always intuitive [5, 6].

Generically speaking, the process of Call-Routing described above involves two steps. The first step, voice recognition, involves phonetic models of the language in which the service is offered, as well as a Stochastic Language Model (henceforth SLM), which is a formal representation of the probabilities of a word occurring if others have occurred beforehand. These SLM are habitually 3-gram or 4-gram models, frequently calculated using Maximum Likelihood Estimation and corrected by means of some Smoothing method (for example Good-Turing). In the end, the ASR module formulates its recognition hypotheses, taking the phonetic models of the language into account, as well as the probabilities provided by the SLM. The relative importance of each element (phonetic model vs. SLM) – in other words, the way of weighting the phonetic model over the SLM or vice-versa – can often be configured in voice recognition devices. Finally, from the scores

[1] Utterance: This is how we habitually refer to the phrase returned by the ASR module. We will also refer to the audio transcription of each call as an utterance.

yielded by the phonetic and SLM models, the ASR module generates a number of recognition hypotheses ordered by the confidence level which it assigns to each of them (a list of possible utterances). What the user has said is already in a text format, and now is the time to assign it to a destination - this will be the second step. To this end, classification techniques will be used to determine when a text (in this case the user request) belongs to one category or another (in this case to one destination or another).

Many classification techniques have been used to this end, with various results, depending also on the text sample and parameters [7]. Some examples are probabilistic models such as Maximum Entropy, artificial neural networks, and high-dimensional spaces, the category to which Latent Semantic Analysis (from now on LSA) would belong. All of the techniques mentioned have been extensively tested, but the current challenge is to achieve a more optimal and less biased representation of the words employed in the utterances, doing so even when data are not labeled or are retrieved and reanalyzed from samples which were not classified due to their difficulty [1, 8, 9].

In this study we have carried out a classification task employing LSA, but, as in these last studies, we have also opted to provide a better vectorial representation of the utterances. To this end, before performing the classification, we pre-processed the utterances using a technique based on a cognitive (or psycholinguistic) model that tries to account for the involvement of prior knowledge in the construction of meaning, and which also perfectly suits the philosophy underlying LSA: the Construction-Integration model [2, 3]. In fact, this study stems from a previous article [1] in which we found that a C-I based technique could be used for call classifiers, but we wanted to test it in a case which involved a voice recognition application, in order to show the performance of the entire system. Both LSA and C-I will be described in more detail in later sections.

2 LSA and Call Routing

LSA was originally described as a method for Information Retrieval [10], although some authors went beyond the original conception and adapted it as a psychological model of the acquisition and representation of lexical knowledge. In recent years its capacity to simulate aspects of human semantics has been widely demonstrated. For example, it adequately reflects why children learn the meanings of words without the need for massive exposure [11]. It is, to summarize, a technique derived from the field of application, but which models some parts of the human linguistic behavior. This in turn can provide benefits for the field of application.

LSA is a vectorial representation of language which is constituted by exploiting word occurrences in different contexts using large linguistic corpora (or text samples). It can be conceived as an automatic sequence of mathematical and computational methods, including pruning, weighting and dimension reduction via Singular Value Decomposition (SVD), to represent the meaning of words and text passages [10, 11]. A key issue is that every term or document (a document is a paragraph or sentence, or in the case of call categorization, a destination cluster or a call) in the corpus is expressed as a k-dimensional vector. Once the whole process is carried out, the cosine of the angle between two vectors is frequently used to evaluate the semantic relationship between the two terms or between the two documents corresponding to the vectors (formula 1). A close semantic relationship between two words, or between two documents, is shown by a high cosine, with a high value that is close to one, whilst two semantically unrelated words or documents have a cosine that is close to zero (orthogonally) or even a slightly negative one. In addition, sentences or texts that are not in the document matrix can be introduced as if they actually were included, using a technique commonly known as Folding-in, which projects this new document into the document matrix (formula 2). In the case of categorization, a vector of the user's utterance will be projected, plus one vector for each of the exemplar texts that represent the destinations, in such a way that if they are similar it will be inferred that what the user wants must be routed to this destination. A new vector d can be created by computing an utterance c (a new vector column in the occurrence matrix including all the terms that occur in it) and then multiplying it by the term matrix, usually called U, and the inverse of the diagonal matrix, usually called S; c is also computed by applying the same weights as in the creation of the original space.

$$Cos(V_1, V_2) = \frac{V_1 \cdot V_2}{|V_1||V_2|} \quad \text{(1) Similarity}$$

$$d = c^T U S^{-1} \quad \text{(2) Folding-in}$$

In the field of Call Routing, LSA has been used mainly with two motivations: correction of speech recognition hypotheses and assignation of utterances to destinations (both are occasionally used in the same router). Concerning the correction of speech recognition output, some studies have tested lists of common potential confusions on the part of speech recognition applications (homophones and near-homophones, {affect, effect}, {quiet, quite}), obtaining good results if such mistakes were corrected by checking the contextual coherence with indices of semantic similarity taken from an LSA model [12]. Satisfactory results were obtained if these confusions were corrected by checking the contextual coherence by means of indices of semantic similarity from an LSA model. These results were better than those of a model that combines a trigram model of the parts of speech with a Bayesian classifier, leading to the conclusion that LSA is a good option in these conditions.

Another study confirmed the usefulness of LSA in conjunction with a model based on n-grams [13]. This study concludes that the analysis by means of n-grams is limited to a text window that is too local (normally 3 or 4 words), and that the LSA model greatly enriches the results from the router, if the probabilities of the predicted word due some history sequence are also calculated in terms of the LSA model, that is, by means of the similarity between the word and the text of the history, which is projected as a pseudodocument. This probability, based on the Semantic Space, will be highest for words whose meaning aligns most closely with the meaning of the history and lowest for words that do not convey any particular information about it, for example, function terms. It is argued that this fact is complementary to n-grams models, because it is its exact opposite. n-gram models tend to assign higher probabilities to (frequent) function words than to (rarer) content words. So the added value of this study is to integrate both kind of probabilities in the generic LM (Language Model) equations.

Some authors extend this same procedure, improving the representation of the history sequence that acted as a pseudodocument in the original study [14]. Due to the briefness of the history for the utterances (and hence its pseudodocument), they replace the history pseudodocument by a more representative one, extracted from its semantic vicinity. They also use the first nearest semantic neighbors to estimate unseen sequences, as is usually done when using smoothing methods.

Again, other studies also obtained good results by interpolating the LSA model and the n-gram model [15], directly correcting speech recognition hypotheses and reassigning new probabilities to them, taking into account the coherence of the lexical context that accompanies each word [16]. Similarly, a new study performed a series of experiments to correct classification errors [17]. They corrected speech recognition outputs using indices of syntactic and semantic coherence (and a combination of both). As for LSA, the authors reported that it yielded results comparable to those of Word-Net [18, 19] (LSA being more economical and flexible), and that, when certain parameters were used, a combination of the two measures led to an improvement in error correction. This same philosophy has also motivated some studies in which the probabilities of the sequences of SLMs - in other words, the likelihood of a word occurring given a history h (sequence of words that precede it) – are recalculated on the basis of the semantic similarity between the word vector and the vector of its history [20].

As for assigning utterances to destinations, a first study proposed a system in which the user response to the typical “say anything” cue is classified by an LSA module according to candidate destinations [4]. The module will compare the vector representation of what the user has said (utterance vector) with the vectors that represent each of the destinations, made up of a compilation of all the calls that belong to each of those destinations. This module also has a disambiguation mechanism in the event that the utterance vector is similar to several destinations. In this case, terms will be found that represent the difference vectors between the utterance vector and each of the destination vectors. Once found, only the terms that may form bigrams or trigrams with a term from the original request will be used. These terms are used to formulate questions for the users in order to disambiguate. One peculiarity of this study is that they used 4,497 transcriptions from a banking services call-center in the LSA training phase. The occurrence matrix to begin the LSA process consists of terms and routes (rather than terms and transcriptions), and as a result few columns are produced – 23, to be precise. This is precisely the criticism made in a later study [21]: the authors specified that they took the possible destinations rather than call transcriptions as documents, so that the LSA training was quite limited. They obtained better results in their laboratory if the documents of the matrix were composed of call transcriptions. Another study introduced a variant in the preprocessing stage [22]. When the corpus was trained, the Information Gain (IG) was calculated in order to identify the terms that actually contributed useful information to the router. The IG index is based on the variations in document entropy (the amount of information carried by a document) with and without the term analyzed. Good results were later also reported when using a training variant in which the labels flagging the transcriptions (the destinations or routes) were entered as terms [23]. This enabled the authors to bring together transcriptions that had been routed to the same places. Other authors have also obtained improvements by introducing an additional step between recognition and Call Routing [24]. These authors did not enter the “utterance” as collected by the ASR module (with a generated SLM) directly — rather they corrected it by using the LSA model confidence indices. Using this method, calls that were more than eight words long (about 12 words) were routed slightly better, and this improvement is greater if the ASR module has laxer criteria (a lower confidence threshold of acceptance).

As for study [24], the motivation for this study is also to introduce an additional step between recognition and LSA based Call Routing in order to differentially reassign the importance of each term of each utterance. This is done by means of the Construction-Integration model [3,2] (henceforth, C-I). C-I is a psycholinguistically motivated model that seeks to simulate working memory and real-time comprehension mechanisms, but it can be applied to categorization tasks as in the case of Call Routing. We will return to this model in a later section.

3 Objectives

There are two main objectives of this study, a general one and a specific one:

1. To implement (and describe in detail) a real full system in which we used LSA and a mechanism based on the Construction-Integration model (C-I).
2. To study the efficiency of the system in a real speech recognition situation by analyzing the percentage of correct classified utterances using different parameters. The percentage of correct classified utterances, thus, showed the quality of the system.

The manipulated parameters were (1) the use or not of an additional step based in C-I (over a LSA-based Call Routing), (2) the length of utterances, (3) the decrease in recognizer performance, and (4) the number of hypotheses processed from the n-best list derived from the voice recognition application.

4 Functional description of the system

One of the aims of this paper is to present a detailed study of the results obtained in a real call router in which a mechanism based on Kintsch's Construction-Integration model [2] was introduced. This model had already been partially tested in a previous study [1], but in this study the full process is described, including voice recognition for spontaneous speech and classification according to possible destinations. The system comprises several modules and is domain-independent, as it was trained using texts from different subject areas. For this study, the domain subject area was a customer service at a Telco (credit balance, top-ups, complaints, phones, etc.)

4.1 The voice recognition phase

Prior to any semantic interpretation, we need a voice recognition stage. The process consists of several layers. The first layer is responsible for recognizing words using the phonetic model of a language. There are numerous packages that perform voice recognition using a phonetic model of a particular language, based on sequences of letters and their pronunciation. These packages usually have standard dictionaries that specify the pronunciation of very common words, as well as dictionaries for the pronunciations that the integrator itself considers to be correct. In general, for recognition of phrases to occur, we need to explicitly specify which phrases we expect will be uttered. They are specified using deterministic grammars in different formats (abnf, grxml, gsl, etc.).

But if our aim is the recognition of spontaneous speech, we must generate a statistical language model (SLM), which uses a large linguistic corpus to generate a model where the probabilities of some words appearing are specified, given those that have occurred previously. To calculate these probabilities, Maximum Likelihood Estimation (from now on ML) is commonly used, corrected by a smoothing method that estimates the occurrences of words within some ranges [7]. One of these smoothing methods is Good-Turing. The package that we use to calculate probabilities in our model is SRILM [25]. It has Good-Turing as the default method (see http://www.speech.sri.com/projects/srilm/manpages/ngram-discount.7.html). It works as follows: by default, the unigrams that occur more than once and the n-grams that occur more than seven times are considered reliable. For this reason, standard ML is applied to calculate probabilities. But if the n-grams occur less than seven times, a correction is applied to the probability extracted from ML using the Good-Turing smoothing technique. It is also possible to estimate n-grams that do not occur in the reference corpus with the Katz method, using the BOW (back-off weight) of the history of each n-gram and the smoothed probabilities, but for simplicity's sake, this is not implemented in our system. Therefore, our model only contains conditional probabilities of n-grams that appear in the training corpus that have been smoothed (using the Good-Turing method).

As was previously mentioned , all these calculations are carried out by the SRILM package, in which individual scripts are used to program SLM generation - in our case with classes that group words (days of the week, months, countries, cities, etc.). In the end, the complete model is output into a file with the .arpa extension (Advanced Research Projects Agency format), whose main use is for exchanging language models. Using this .arpa file, we generate a .grxml file where the corrected probabilities from the input file are recorded several times in the form of a tree, but this time in SGRS (Speech Grammar Recognition Specification) format, which can be read by many commercially available recognition packages. In our case, we will use the grxml of the generated SLM in a Nuance 9 recognition engine.

4.2 The call-routing phase

Once the recognition hypotheses were established - in other words, what was recognized by the recognition engine- we proceeded with the second process: categorization by destination. In the case of this study, as noted earlier, the categorization procedure is implemented using LSA. So we needed a LSA semantic space to project the user's utterance as well as each of the exemplar texts that represent the destinations in it.

4.2.1 Semantic Space

To obtain that semantic space, we used a training corpus of digitized utterances belonging to several phone companies. The LSA was trained and the semantic space was created using this corpus. These utterances were extracted using the Wizard of Oz procedure (a technique where the users are made to believe that they are interacting with a computer but in fact they interact with a person) and the transcriber labeled each of them with the destination to which it was routed. We should note that the labeling was performed using different criteria, was carried out at different times and at different companies, and is not exhaustive. In any case, to provide cohesion for interrelated words, these labels were regarded as additional words, as in Featured LSA [23]. In the end, the LSA comprises the terms and labels that occur in utterances. In a previous study it was demonstrated that retaining these labels boosts categorization performance and produces a positive interaction with usage of C-I, even if they are non-exhaustive [1]. It should also be borne in mind that the only utterances that must be exhaustively labeled are those that are part of the destination sample (see section 4.2).

In the LSA training we used Gallito® (see www.elsemantico.es), a tool that has been used in other occasions for the creation of semantic spaces [26, 27, 28]. The words matching a special stop-list for this domain were eliminated, as were all function words. We also eliminated words that did not occur at least three times. Some words which are relevant within the telephony corpus were artificially grouped into a single class, for example countries, mobile phone brands or telecommunications companies, substituting them with the name of the class. In the end, we obtained a matrix of 1,421 terms and 34,905 utterances. In a next step, log-Entropy is calculated in this matrix. log-Entropy estimates the amount of information that a word carries in the documents where it appears. In this way, the terms that might contain the most information are given a heavier weight. So from this last calculation we got a weighed matrix to which SVD is applied and the three resulting matrices are reduced to 270 dimensions. We chose such a dimensionalization based on the assumptions made in previous studies [28]. In those studies it was suggested that the optimal number of dimensions for specific domain corpora does not have to be extremely low, sometimes even approaching the 300 dimensions recommended by Landauer, et al. [11]. In summary, the result of the entire process, the three reduced matrices (terms, diagonal and utterances matrix), is the semantic space of the mobile telephony domain that will be used as a basis for projecting the user's utterance as well as texts that represent the destinations.

4.2.2 Destination Management

The service we wish to evaluate has 29 basic destinations, covering the needs of a telephone company call center. Previous LSA-based Call Routing research compared the vector representation of what the user says (what the ASR module returns) in real time with each of the utterances used in training that are labeled with a destination. After the comparison is performed, the label of the most similar exemplar is selected [4]. This label will be the destination selected. The philosophy of our system is similar, although three important points should be highlighted:

The first one is that in our system we do not use all utterances from the training corpus as destination exemplars, but rather focus on a representative sample. In particular, a total of 2,329 calls are part of this sample (which are loaded and transformed into a vector of the semantic space only once the router has been launched). This is important because it is not necessary for all the calls from the training corpus to be labeled and participate as exemplars. We manually selected only a few representative calls (an excerpt is shown in Table 1), making the process more economical and the response to changes in the routing model (a change in the definition of the call-center destinations) faster. This is a very important issue if someone wants a successive deployment of a Call Routing service [29]. Massive annotation used to be very time-demanding. We made this pre-selection very carefully given that, as was pointed out in some studies [8], system performance depends on the quality of these destination exemplars.

1	ALTASBAJASC	well I'd like to cancel this phone number
1	ALTASBAJASC	switch to a monthly contract
1	ALTASBAJASC	to cancel a monthly contract
1	ALTASBAJASC	activate a number
1	ALTASBAJASC	cancel the mobile
1	ALTASBAJASC	I'd like to get a phone with you

Table 1: Excerpt from the destination exemplars.

The second point is that to decide what the most credible destination or destinations are we do not simply individually compare the user utterance with each of the utterances in the destination sample. In this system we use a method called average-4 which proved to be better in a previous study [1]. This method averages the four exemplars with the greatest cosine for each destination. The chosen destination is the one where the average of the four exemplars is the highest. In this way, any bias which an anomalous exemplar (seemingly very similar to what the user said) might have is eliminated.

The third point is that after converting the user utterance into a vector and selecting the most likely destinations, a list of the first four destinations is returned in descending order of their cosines. When the user is not sent toward the first option, this allows the first destination and the next most likely destinations to be used as candidates, disambiguating with an explicit question. This will depend on the design of the dialogs.

What is common to earlier studies is that both the destination exemplars and the user utterance will be converted into vectors that can be interpreted by the LSA space (in other words, use of the semantic vector space will be essential). The way the destinations and user utterances are converted into vectors will depend on the effectiveness of the system. In the following section we explain different ways of doing so.

4.2.3 Construction-Integration Model vs. direct method

There are two ways of representing each of the utterances vectorially in the router, whether utterances are destination exemplars or user utterances. One is Direct routing, where the utterances are projected onto the latent semantic space without any kind of additional algorithm. This is the standard LSA method for constructing new documents in the vector space, and is known as Folding-In [10]. The second is C-I routing, which, in contrast to Direct Routing, has an intermediate step in the construction of both destination exemplars and user utterances. This intermediate step is based on a Construction-Integration network (Figure 1). The importance that a Construction-Integration network might have for routers lies in the fact that the words in an utterance are modulated to their correct meaning, taking into account the entire lexical context, which constrains the final meaning. This is particularly relevant for words that are ambiguous (such as “card”) - the meaning that best matches the context will be given priority, thus avoiding predominant sense inundation and other biases frequently observed in vector space models [26, 27]

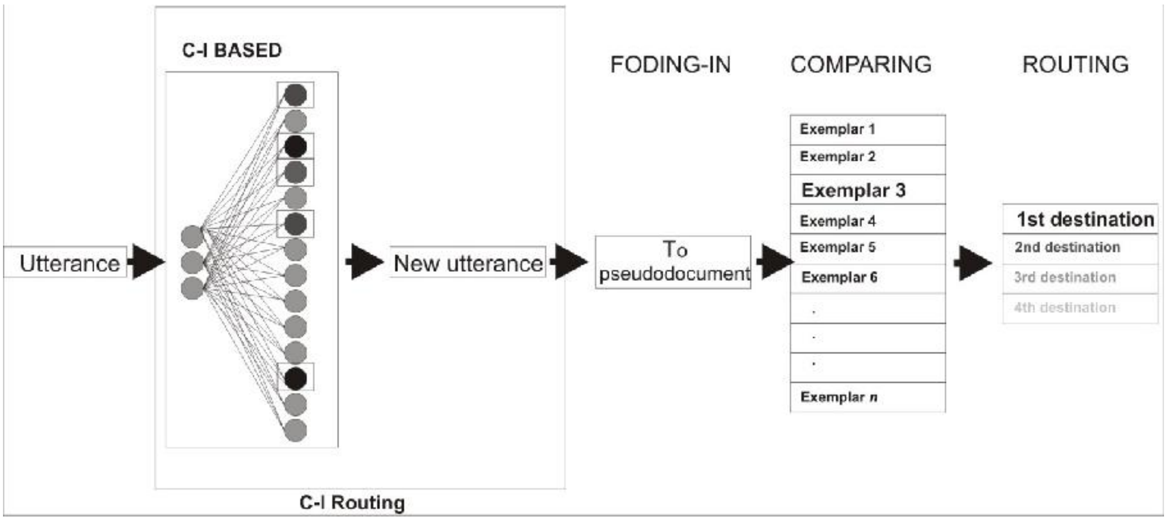

Figure 1: Graphical view of the Call Routing process for Direct routing and C-I routing.

Computational models based on Construction-Integration [2, 3] are based on the idea that the meaning of a text is constructed and integrated in the working memory. The mechanism retrieves the content linked to every word of a text from long-term memory, and, once retrieved, ensures that all this content is integrated in real time using a mutual constraint mechanism in the working memory. Therefore, it is a model that seeks to simulate working memory and real-time comprehension mechanisms [30]. Looking at the model details, interpretation of a text is carried out in two phases. In the first phase, all terms related to each of the words that make up the text or phrase are evoked. A network is constructed from all of them, where each one is joined to all the others - this phase is known as the Construction phase. In the second phase, known as the integration phase, each of these terms receives activation from the others proportionally with respect to the similarity they have with each other. After this phase, the most activated terms are those related to the main idea in the text [3,30]. This takes into account the fact that a text is not the sum of the terms included in it, but rather the integration of all of them into a final idea. Each term is constrained by the meaning of the other terms, thus generating the meaning in real time. This type of mechanism has been used on predicate and metaphorical structures [31, 32, 33] and on structures enriched with syntactic dependency relationships, thus demonstrating that the meaning of complex phrases can be modeled [34]. In operational terms, this mechanism makes it possible to differentially reassign the importance of each term. Ultimately, it is an estimate of the relevance of the utterance terms, as was done in previous studies [8].

How is this model implemented in our system? For each utterance, whether it is a user utterance or destination exemplars, a network is constructed based on the Construction-Integration network whose launch will lead to the extraction of new terms (see Figure 2). We might say that these terms produce a better definition of the utterance, an idea common to all the words contained in the original utterance. The procedure, in terms of its functionality, is analogous to that used by a study [35] implemented to improve classical methods for evaluating text with LSA. Figure 2 provides a graphic description of the procedure. Firstly (in the Construction phase), each term of the utterance is compared to all of the terms present in the semantic space, and the 200 neighbors most similar to each of them are extracted (this similarity is calculated using the cosine). A connectionist network is created between all these neighbors (neighbor node layer) and each of the original terms of the utterance (utterance node layer), where the weight of each connection is given by the cosines[2] between each of the connected terms (figure 2). Once the weights of the connections have been assigned, the activation of each node is calculated based on the connections received, in the second Integration phase (see formula 3). Thus, the greater the weight of the connections received, the greater the activation. The activation function also favors instances where the source of activation derives from several terms in the utterance, and not just one or two (due to the parameter δ in formula 3).

$$A_i = \text{¢}_i \sum_{j=1}^{n} \log(1 + C_{ij}) \quad \text{(3) Activation function}$$

Where *j* is the sub-index of the utterance layer, *i* is the sub-index of the neighbors layer, C_{ij} is the strength of the connection that node *i* received from node *j* (the latter node belonging to the first layer), and *δ* is a correction factor to avoid unilateral activation (based on the standard deviation of the connections received), as defined in formula 4:

$$\text{¢}_i = \frac{1}{\sqrt{\frac{\sum_{j=1}^{n}(C_{ij} - \overline{C}_i)^2}{n}} + 1} \quad \text{(4) Correction factor}$$

[2] The cosines are calculated using the previously trained semantic space, in other words each of the terms to be compared is represented by a vector in this space. Any term vector might then be compared to another term vector using the cosine.

Where n is the total number of connections received by i (or the number of nodes in the neighbors layer) and C_i is the mean of all the strengths that node i received.

Finally, the 20 most highly activated neighbors are chosen from all of those activated in the Integration phase, and a new utterance is constructed with them. In other studies an utterance has been replaced by a more representative one [14], but now the aim is also for this new utterance to contain terms which are closer to the meaning originally intended by the user. These 20 terms are used to form a new pseudodocument, this time using Folding-in (see Figure 1), and they will be compared with the destination exemplars (created in the same way) in order to assign them a destination.

As the reader can realize, there are a few differences between the procedure to perform C-I used in this study and the original mechanism proposed by Kintsch [3]. Because call utterances are shorter and simpler than propositions within colloquial language, the algorithm used here is not exactly the original construction-integration algorithm. The integration part proposed by Kintsch is a spreading activation algorithm which is iterative until the net is stable (the cycle when the change in the mean activation is lower than a parameterized value), whereas our algorithm is a “one-shot” mechanism. The activation of each node is calculated based on the connections received. Another difference with the C-I algorithm as proposed by Kintsch is that we only consider words and not propositions nor situations. In any case, note that the original C-I is more complete and fine-grained, but our mechanism is sufficient for our purposes and may be more flexibly programmed, because an OOP (Object Oriented Programming) paradigm has been used, with classes such as net, layer, node, connection, etc., instead of the iterative vector × matrix multiplication in the original (see [39] for details of the original conception).

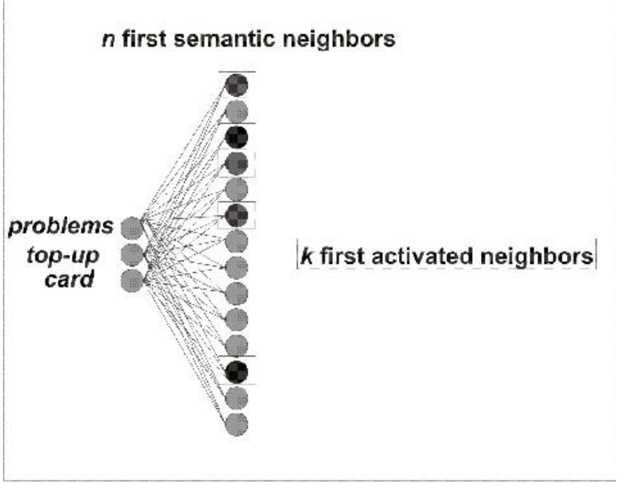

Figure 2: C-I based net implemented in this study. The K most strongly activated neighbors (with a square) will be represented in a pseudodocument.

5 Software and Architecture used

This section describes the application that we have implemented in our lab, and the auxiliary Software and technologies that has been used. In the last part, we describe the place of each module in the global architecture.

5.1 IVR Application

The IVR Application is implemented using VXML technology (Voice Extensible Markup Language), located on a Tomcat application server (icon D in Figure 3). The server dynamically generates VXML files and sends them to the VXML interpreter, in this case VoxPilot (icon B in Figure 3), as it requests them and according to the application flow. First, the interpreter requests the start and welcome VXML files, then the VXML where the user is asked to request a service (“say anything”). This second VXML has voice recognition and therefore requires a grammar - in this case a grxml grammar generated using the SLM. In this way, Voxpilot sends the user response to the voice recognition application (icon C in Figure 3) along with the route where the grammar is located on the application server. Finally, the ASR module returns the text recognized with its confidence interval. This recognized text is sent in the actual request to the next VXML located on the application server, using the SOAP protocol[3], to the semantic router (icon E in Figure 3). The semantic router returns a list with the four most likely destinations. The first destination from the list will be inserted into the VXML that is sent once again to Voxpilot for it to run the definitive Call Routing routine.

5.2 ASR Module

The recognition engine used is Nuance 9® (icon C in Figure 3), located on the Speech Server®, also by Nuance, with a test license that allows the use of 4 channels in non-production environments. This recognition engine accepts grammars programmed in the standard .grxml (SGRS Speech Grammar Recognition Specification) so it is a good match for our SLM.

5.3 Semantic Router

The router is basically dedicated to the task of receiving texts from the application server (icon D in Figure 3) and returning a list with the 4 most likely routes, or simply returning a rejection if it does not reach a confidence threshold. As we explained above, the router application is based in the LSA and C-I framework and was programmed in VB.NET using the object-oriented programming (OOP) paradigm. It is accessible as a web service on a Microsoft IIS server, which offers a series of functions and procedures such as loading the destination exemplars and converting them into vectors, and loading the reference semantic space (the semantic space

[3] **SOAP** (Simple Object Access Protocols) is a standard protocol that defines how two objects in different processes can communicate by exchanging XML data.

previously generated with the Gallito® tool mentioned above). It also has a configuration file which specifies the acceptance thresholds of the logistic function, the method of converting utterances into pseudodocuments (C-I Routing and its parameters or Direct Routing) and the classes that will be used according to the training (months, days, mobile phone brands, countries, etc.). The fact that it is implemented as a Web Service means it can be integrated into a SaaS (Software as a Service) structure that allows it to route service utterances in a centralized manner. When the destination has been assigned, this module returns a list with the four most likely destinations and their cosines.

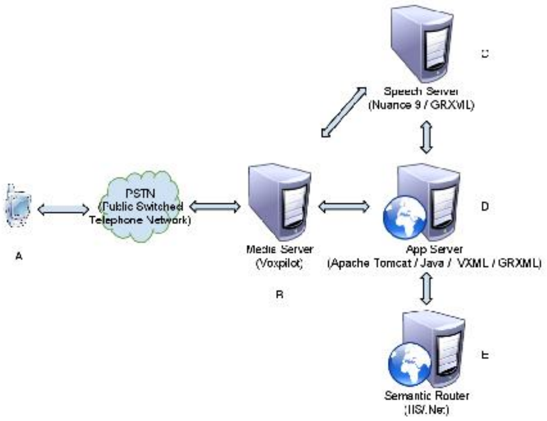

Figure 3: Software, technologies, modules and system architecture used in our lab.

6 Evaluation of the system

In this section we describe the procedure that we followed to examine the efficiency of the system in different configurations. First we will present the variables that were manipulated. Second, we will explain the procedure and the method to test them.

6.1 Dependent variable and independent variables

The dependent variable used to assess the efficiency of the system was whether the destination was correctly classified or not.

The first independent variable is called Routing Method and has to do with how LSA vectorially represents the test utterances and the utterances that represent the destinations. There are two methods (see section 4.2.3): Direct routing and C-I routing.

The second independent variable is the Number of Captures, the number of utterances captured by the voice recognition application that were passed as input to the categorization module. The voice recognition application can be programmed to return only one text containing what it has recognized of the participant's voice message. However, as this captured text derives from a probability-based model, it can also be programmed to return the second, or the third most probable text, and so on. Bearing in mind the idea that by combining the first two options we can create a text including more correctly recognized words than using only the first option, the independent variable contains two quantities: the first recognized utterance and the concatenation of the first and second utterance. We named the two levels Captur1 and Captur2.

A third independent variable is Accuracy, which is measured by a value F’ (see later in the method section). This value F’ is broken down into two groups: high and low accuracy (see later for a more detailed description).

Lastly, another independent variable is the number of tokens that is the length in words of transcriptions of the utterances. Two groups are again formed: Short and long utterances.

6.2 Procedure and Method

The evaluation method is as follows: 1,872 (randomized) audio files are obtained from real calls to several telecommunications companies. All of these audio files were then transcribed, to create the ideal condition where the ASR module captures the utterance perfectly. Recognition of each of these audio files was forced using the Nuance® ACC_TEST tool. The outputs from this tool provided the first two captures of what had been recognized (the two first outputs from the n-best list).

To obtain objective measures of the third independent variable, Accuracy, measured as the deviation between what is recognized and what is transcribed, Information Retrieval measures were used. Their usage is justified by some studies in substitution of the WER (Word Error Rate) [36]. The measures used for IR were Recall, Precision and the combination of both in F'. Recall is the proportion of the transcription that is present in recognition; Precision is the proportion of the recognition that is present in transcription. These two measures range from 0 to 1, and are combined in an index, F´. The latter is the harmonic mean of the precision and the recall measures. To calculate them, we used the most popular natural language package in the Python environment, the Natural Language Toolkit (NLTK) [37] available at http://www.nltk.org/. Using this tool we extracted recall, precision and F' of each recognition compared to its transcription as well as revealing the recognition loss, and these were introduced as variables in the overall analysis of the router (becoming the Accuracy variable). It is important to note that both precision and recall have been calculated using only relevant terms, meaning those that will be considered by the router. Function words and words from the stop-list, for example, will be excluded from the calculations. At the same time, we also counted the number of tokens in each transcription to obtain a measure of the fourth independent variable, which gave us an idea of the length of the phrases uttered by users to make system requests.

In the second phase the router is forced to categorize the text of each audio clip, beginning with the first recognition capture (Captur1), followed by the first two captures concatenated together (Captur2). This operation is repeated twice, once with the router working in Direct Routing mode, and once with C-I Routing. As a result we obtain all the combinations of independent variables (Routing Method × Number of Captures to route). In addition the router is forced to categorize the transcriptions, simply to have a baseline, but without introducing these results into the analysis matrix. The Call Routing is considered correct if the first destination returned by the router coincides with the ideal destination previously assigned to the utterance by a human[4].

With this data we now have all the necessary conditions and all the grouping variables. A matrix is formed and we proceeded to carry out a Repeated Measures ANOVA (Routing Method × Number of Captures to route) with two grouping variables (Accuracy – high or low, and Nº of tokens – long or short). In summary, a 2×2×2×2 ANOVA. The results will be extracted from this analysis, and their implications will be examined in the discussion.

At the end of the analysis, a logistic function is proposed to create acceptance zones in the router. There is a first zone where the first hypothesis returned and routed to this destination is accepted as valid; then a second disambiguation zone, where we might disambiguate between four hypotheses returned by the router, asking the user to choose;, and a third rejection zone , where any hypothesis is assumed to be erroneous and is not routed. The results of this mechanism will be shown as the percentage of correct call routings.

7 Results

In this section we will present the results of the study. First we will present the percentages of correctly routed calls under all possible conditions, including those which involve voice recognition. In addition, we also present the results of an ANOVA which displays the interactions between the variables involved: Routing Method, Accuracy, number of tokens and Number of Captures. Finally, the results of entering an acceptance criterion by means of a logistical function are presented, as well as the application of various confidence levels for a potential disambiguation strategy.

7.1 Performance of the ASR module (F')

As suggested above, one of the proposed means of measuring the ASR module's performance is applying Information Recall indices. To be precise, our system provides the following indices: Recall=.912, Precision=.959, F'=.932.

7.2 Decrease in performance caused by voice recognition

Another of the relevant issues to be examined is the lowered performance of the whole system when speech recognition output utterances are routed rather than transcriptions. If we consider only the router's first hypothesis (Table 1), the results show that between the best condition with transcriptions (.75 with C-I) and the best condition with recognition (.67 also with C-I), we lose 8 percentage points in the rate of correct choices. If we consider the first four hypotheses returned (Table 2), between the best condition with transcriptions, also using C-I (.91), and the best with recognition, again using C-I (.85), the difference is reduced to 6 percentage points. We should remember, though, that our application has not undergone any optimization process for either voice recognition or the model of categories and that the main aim of this study is to check the scenarios in which C-I might behave more productively. We will look at this in the following section.

	1st hypothesis		
	Trans	Captur1	Captur2
No C-I	.69	.63	.62
C-I	.75	.67	.66

Table 1: Percentages of correctly routed calls, considering only the first hypothesis returned by the router. Captur1 and Captur2 represent the conditions where one or two speech recognition hypotheses respectively are used.

	Accumulated first 4 hypotheses		
	Trans	Captur1	Captur2
No C-I	.89	.83	.83
C-I	.91	.85	.84

Table 2: Percentages of correctly routed calls, considering the four hypotheses returned by the router. Captur1 and Captur2 represent the conditions in which one or two speech recognition hypotheses respectively are used.

7.3 Results of the ANOVA

Another of the aims of this study was to analyze the performance of each method in different scenarios - specifically bearing in mind the Accuracy of recognition measured with F', and the number of tokens in user utterances. For this purpose the ANOVA described in the method section was carried out, extracting both main effects and interactions.

[4] Very often, many categories are interrelated or overlap, such that not even a human classifier could be sure which category to assign. As a last resort, whilst both may even be correct, the choice between one or another is binary and exclusive, so the results may be understated. Some evaluations of Call Routing have used not only binary coincidence to correct this but also ratings of the fitness of the destination returned, or also a scale of possible success [38]. We have not used this type of evaluation due to the high cost involved, which lessens information in the results and implies that they may be understated.

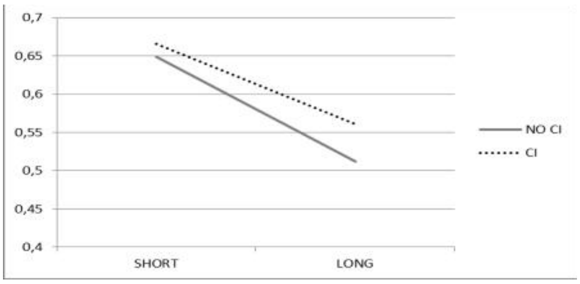

Figure 4: Interaction between Routing Method and Number of Tokens.

Three significant main effects were in fact found. The first of them relates to Routing Method (F(1,1735)=18.13, MSE=.087, p < .001). C-I is better than Direct Routing as a general effect. The second is the Nº of tokens (F(1,1735)=34.07, MSE=.637, p < .001). Short phrases boost the router's effectiveness. The third is F' (F(1,1735)=128.14, MSE=.637, p < .001). The Accuracy of voice recognition also increases the router's performance.

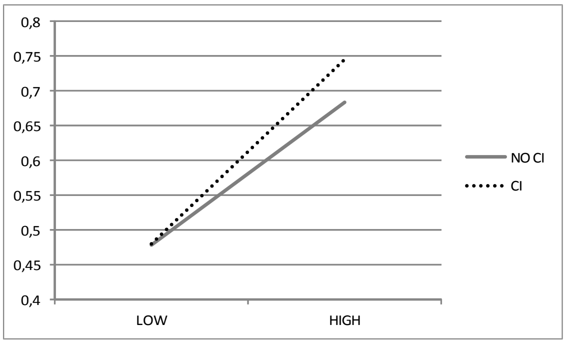

Figure 5: Interaction between Routing Method and Accuracy.

Given that we also found significant interaction effects with the variables above, we will focus on these, leaving the main effects as supplementary information. Firstly, we find a two-way interaction effect between the factors Routing Method and Nº of tokens (F(1.,1735)=4.45, MSE=.087, p = .035). The loss of effectiveness observed when the user utters long phrases is greater with Direct Routing than with C-I (Figure 4). Both methods show reduced effectiveness with long utterances, but C-I less so, hence we could say it has a corrective effect. We also found a significant effect for the two-way interaction between Routing Method and Accuracy (F(1,1735)=15.41, MSE=.087, p < .001). C-I shows beneficial effects if the quality of the recognition (measured by F') is good (Figure 5); otherwise, C-I works the same as Direct Routing. Lastly, we found a significant effect for the two-way interaction between Number of Captures and Accuracy (F(1,1735)=8.60, MSE=.087, p = .003). The results of this last interaction show how concatenating the first two speech recognition Captures is advantageous compared to using only one, if the quality of the recognition is low (Figure 6).

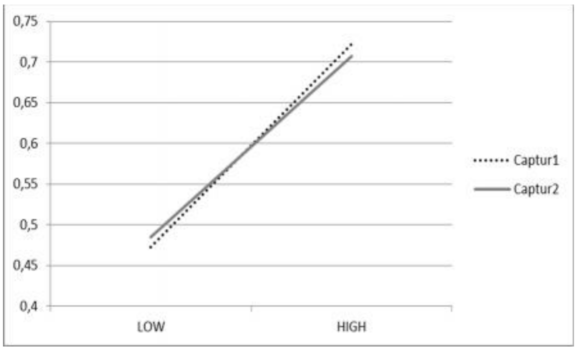

Figure 6: Interaction between Number of Captures and Accuracy.

7.4 Acceptance levels

Beyond straight performance data - in other words, the percentage of occasions a hypothesis returned is correct - we must introduce acceptance levels into the process. These maximize the correct choices and minimize errors by using a confidence threshold above which the route proposed by the system is accepted (formula 5). Previous studies have often introduced objective acceptance criteria into the process, such as logistic functions of acceptance [4]. In our case we will use the same parameters used by a similar study [1] in the logistic function, as they gave good results. In this function two parameters act as predictor variables, and the correct choice / error rate as our dependent variable. The parameters are as follows: firstly we use the cosine from the first hypothesis (average of the 4 largest cosines with the exemplars of this destination). As the second parameter, we use the difference between this first cosine and the cosine from the second hypothesis. The latter parameter takes into account the level of certainty about whether the first hypothesis should be returned rather than the second.

$$P(Y=1)=\frac{1}{1+e^{2.17-3.02d\cos-2.97\cos}}$$

(Formula 5)

With the output from the logistic function, we define three zones of acceptance. The first one (0.5 - 1), above which we accept the label directly; the second one (0.4 – 0.5), where we then disambiguate (using a question) between the four destinations proposed by the router (those that have the greatest cosine); and the third one (0 – 0.4), where it is directly rejected. In this way, not only does the percentage of correct choices fall within the acceptance zone, but it can also retrieve calls that are in the intermediate zone. Disambiguation mechanisms, such as preparing questions using the four hypotheses returned, could be built in this zone. For example, if the logistic function returned 0.46, the hypothesis returned would not be rejected, but rather disambiguated using the four hypotheses returned by the router, in the hope that the correct route would be among them, which is very likely. The results (Table 3) show that this disambiguation mechanism would be productive. In the acceptance zone (0.5 - 1), the percentage of correct destinations would be 76.54%. If we also disambiguated in the next zone of acceptance (0.4 – 0.5), we would guarantee that 81% of the time the correct destination would be among the four hypotheses used to ask the question. In addition, our data would contain 215 calls that would be rejected directly as they are in the rejection zone (0 - 0.4). This amounts to only 12% of the calls.

	Error		Correct choice	
	N	%	N	%
0.4 to 0.5 (disambiguation)	39	18.14	176	81.86
0.5 to 1 (1st hypothesis)	308	23.46	1005	76.54

Table 3: Correct choice rates in the two zones of acceptance.

8 Discussion

The overall results of this system are encouraging. Considering that it is a pilot study performed without optimizing voice recognition, and using a hypothetical customer service operation, the results are very satisfactory. To offer some raw data, 67% of utterances were assigned a destination where router and human agreed (with spontaneous speech recognition and categorization). As explained above, this finding is cautious if we consider the possibility that the destinations overlap or that there is ambiguity in the human assignment of an utterance to a destination. In addition, by introducing an acceptance criterion and using disambiguation mechanisms, a large proportion of the remaining utterances (81% of the total utterances) can be assigned to a correct destination. In any case, since this is an academic study, our aim was to focus on checking the performance of some methods in certain scenarios and not so much on the overall results, which could be improved upon.

Our system brings several features together. One of them is that the router is based on LSA, a technique that is independent from the subject domain and is also fairly economical in terms of implementation. We have inserted an additional layer in this router, based on the assumption that the meaning of an utterance is not the sum of its words. Rather it is important to take into account how these words activate others that are not explicit, but participate in the final meaning. Whilst it is not identical, this layer is based on the Construction-Integration model, which makes the same assumption. We have also decided to summarize the final routing destinations in a file of sample utterances that represent them (golden destinations), thus avoiding any change in the organization of destinations, leading to a need to reclassify all utterances in the training corpus. This provides added flexibility to the system and reduces the response times to changes. All this has been integrated with a voice recognition engine (supported by a Stochastic Language Model or SLM), whose outputs are passed to the router in order to assign destinations. In order to maximize the number of correctly recognized words, not only the first hypothesis output from the ASR module is routed: there is an option where the concatenated first and second hypotheses can also be routed. This was tested on audio clips of real calls and an experimental design that allowed us to evaluate performance in some scenarios depending on length of utterances and accuracy of voice recognition.

The analysis performed yielded various findings. Firstly that C-I is better than the direct method, in particular when it comes to cushioning the drop in performance in certain scenarios. It is true that in conditions where recognition has greatly declined, the contribution of C-I is not important, but when recognition is not bad, the C-I method seems to behave best with long utterances. Whilst this is not the case in a model like C-I, some previous studies have found benefits in long utterances if the speech recognition outputs are corrected by means of similarities extracted from LSA [24]. It should come as no surprise that in our system the C-I method performed best for long utterances, given that the original C-I model was created to account for longer propositions [2] and that the presence of a number of terms in the phrase facilitates the building of a context. This helps to over-weight the words that are within this context and to under-weight those that are not - for example, substitution or insertion errors. It also biases the meaning of ambiguous terms toward a meaning coherent with this context. The great contribution of this type of models is that they objectively mimic the process carried out in working memory while processing texts. As a text is being read or listened to, it is available in working memory, which retrieves content related with each word in the text from long-term memory. This will be the construction phase. In the integration phase, a mutual constraint mechanism is applied to this linked content [34] in order to extract the key idea. Therefore it is only to be expected that the higher the number of words in working memory (up to a threshold for simultaneous processing), the more data will be available to carry out this integration in a more correct way.

Secondly, although this is approximate complementary data, we have also tried using two voice recognition captures in the Call Routing process. The results are modest, although they show a subtle trend. When recognition is expected to decline, there is a tendency that taking two hypotheses rather than one improves the results. We sense that occasionally the errors committed in the first hypothesis are not committed in the second, and vice-versa.

Thirdly, we have seen that introducing a logistic function with some parameters helps to form acceptance criteria, above which correct choices are maximized, either by correctly rejecting the label or by accepting a label that later proves to be correct. By doing so, the results rise to 76.54% accuracy (either correct choices or correct rejections). We have also shown that improved performance results from setting up three zones of acceptance: the first one (0.5 - 1), above which we accept the label directly; the second one (0.4 – 0.5), where we then disambiguate (in the form of a question) between the four destinations proposed by the router; and the third one (0 - 0.4), where the label is rejected directly. In this way this, we achieve 79.3% correct Call Routing, and the correct destination of the utterances that remain in the intermediate zone for disambiguation would be among the four labels proposed in the question (the four returned by the router) in 81.56% of cases. Thus we have not only the percentage of correct choices in the acceptance zone, but also those that arise from disambiguation. It is clear that this requires a cost in terms of disambiguation question design, and a cost in terms of satisfaction, but it might be a good way to gradually implement the system.

There is one last thing to note: this system's adaptability to change. On the one hand, although in this study we have used labels to identify the destinations for utterances in the training corpus, acceptable results can be obtained without them. We could even extend the training corpus, combining labeled with unlabeled parts, or parts labeled using different criteria. If we did these things, we would find a small, controlled reduction in performance, but not an abrupt drop [1]. On the other hand, the only labels that need to be exhaustive are those that identify the utterances that act as destination exemplars. These exemplars form part of a chosen sample of training utterances, but represent only a small percentage of them. Since there are few of them, they can be examined, changed or expanded quickly. Thus any change in the routing model (for example, a re-dimensioning of skills) can be dealt with, with acceptable response times, and with no need to retrain all utterances, thereby slowing down the process.

9 Conclusion

In view of these results, the possibility of using psycholinguistic models in information recovery or utterance categorization systems is encouraging. Simulating human processing is not an easy task. It is not even easy to describe, but it is useful to reflect upon it in order to find possible improvements to current systems. In summary, LSA represents a very flexible and economical means of implementing Call Routing, and also allows us to explore algorithms and methods derived from research in Cognitive Science that may prove very promising. In this case, we have presented a system that combines LSA with a network based on Construction-Integration. The results obtained are good, although fine tuning is needed to optimize the voice recognition process, as well as more coherent organization of the destinations (which would be the case in a working production system). In any case, what has proved most interesting about this study is not the overall results in absolute terms, but rather testing the C-I layer and how it works. This has offered quite promising results, demonstrating superiority in some scenarios and stability in others. We believe that establishing links between computational science and psycholinguistics can help to find ways to optimize current categorization systems.

Acknowledgements

The authors wish to thank Airenas Vaiciunas from the Department of Applied Informatics at Vytautas Magnus University (Lithuania) for his useful comments about SLMs and also to Semantia Lab (www.semantialab.es) for supporting the logistic of this research.

References

[1] Jorge-Botana, G., Olmos, R. & Barroso, A. (2012). The Construction-Integration algorithm: A means to diminish bias in LSA-based Call Routing. *International Journal of Speech Technologies, 15(2)*: pp. 151-164.

[2] Kintsch, W. (1998). The use of knowledge in discourse processing: A construction integration model. *Psychological Review*, 95: 163-182

[3] Kintsch, W., & Welsch, D. *The construction-integration model: A framework for studying memory for text*. In W.E. Hockley & S. Lewandowsky (Eds.), Relating theory and data: Essays on human memory in honor of Bennet B. Murdock. Hillsdale, NJ: Erlbaum. 1991: 367-385.

[4] Chu-Carroll, J., y Carpenter, B. (1999). Vector-based natural language call routing, *Computational Linguistics, 25(3)*: pp. 361-388.

[5] Brumby, D. & Howes, A. Good enough but I´ll just check: Web page search as attentional refocusing. *Procedings of the 6th International Conference on Cognitive Modelling*, 46-51. 2004.

[6] Brumby,D. & Howes, A. Interdependence and past experience in menu choice assessment. In: Alterman, R., Kirsh, D. (Eds.), *Proceedings of the 25thAnnualConference of the Cognitive Society*. Lawrence Erlbaum Associates, Mahwah, NJ, p. 1320. 2003.

[7] Jurafsky, D. & and Martin, J. Speech and Language Processing: *An Introduction to Natural Language Processing, Speech Recognition, and Computational Linguistics*. 2nd edition. Prentice-Hall. 2009.

[8] Gasanova, T., Zhukov E., Sergienko, R., Semenkin, E. & Minker, W. A Semi-supervised Approach for Natural Language Call Routing. *Proceedings of the 14th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*. August 2013. Metz, France.

[9] Sarikaya, R., Hinton, G. & Ramabhadran, B. Deep Belief Nets for Natural Language Call-Routing. *In ICASSP-2011*.

[10] Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshman, R. Indexing. (1990). By Latent Semantic Analysis. *Journal of the American Society For Information Science, 41*: pp. 391-407.

[11] Landauer, T. K., & Dumais, S. T. A solution to Plato's problem: the Latent Semantic Analysis theory of the acquisition, induction, and representation of knowledge. (1997). *Psychological Review, 104*, pp. 211-240.

[12] Jones, M.P., & Martin, J.H. Contextual spelling correction using latent semantic analysis. *In Proceedings of the Fifth Conference on Applied Natural Language Processing*. pp. 163-176. 1997.

[13] Bellegarda, J.R. Exploiting latent semantic information in statistical language modeling. *In Proceedings of the IEEE. 88 (8)*, 1279– 1296. 2000

[14] Jen-Tzung Chien, Meng-Sung Wu and Hua-Jui Peng. (2004). Latent semantic language modeling and smoothing, *International Journal of Computational Linguistics and Chinese Language Processing, vol. 9*, no. 2, pp. 29-44.

[15] Pucher, M., Y. Huang, Y. Combination of latent semantic analysis based language models for meeting recognition. *In Proceedings of the Second IASTED International Conference on Computational Intelligence*, San Francisco, California, USA, November 20-22, 2006

[16] Lim, B.P, Ma, B., & Li, H. Using Semantic Context to Improve Voice Keyword Mining, *In Proceedings of the International Conference on Chinese Computing (ICCC 2005), 2005*; 21-23, March, Singapore

[17] Shi, Y. *An Investigation of Linguistic Information for Speech Recognition Error Detection*, Ph.D. University of Maryland, Baltimore County, Baltimore. 2008.

[18] Miller, G.A. (1995). WordNet: A Lexical Database for English. *Communications of the ACM Vol. 38*, No. 11: 39-41.

[19] Fellbaum, C. (1998, ed.) *WordNet: An Electronic Lexical Database*. Cambridge, MA: MIT Press.

[20] Wandmacher, T. & Antoine, J. Methods to Integrate a Language Model with Semantic Information for a Word Prediction Component. *In Proceedings of EMNLP-CoNLL*, 506-513. 2007.

[21] Cox, S. & Shahshahani, B. A Comparison of some Different Techniques for Vector Based Call-Routing. *In Proceedings of 7th European Conf. on Speech Communication and Technology*, Aalborg. 2001.

[22] Li, L. & Chou, W. Improving latent semantic indexing based classifier with information gain, *In Proceedings of the 7th International Conference on Spoken Language Processing, ICSLP-2002*, 1141-1144. September 16-20, 2002 Denver, Colorado, USA.

[23] Serafin, R. & Di Eugenio. B. (2004). FLSA: Extending Latent Semantic Analysis with features for dialogue act classification. *In Proceedings of ACL04, 42nd Annual Meeting of the Association for Computational Linguistics*, Barcelona, Spain, July.

[24] Tyson, N. & Matula, V.C. Improved LSI-Based Natural Language Call Routing Using Speech Recognition Confidence Scores, *In Proceedings of EMNLP*. 2004.

[25] Stolcke, A. SRILM - An Extensible Language Modeling Toolkit, *In Proceedings of Intl. Conf. Spoken Language Processing*, Denver, Colorado, September 2002.

[26] Jorge-Botana, G., León, J.A., Olmos, R. & Hassan-Montero, Y. (2010). Visualizing polysemy using LSA and the predication algorithm. *Journal of the American Society for Information Science and Technology*, 61(8), pp. 1706–1724.

[27] Jorge-Botana, G., Olmos, R., León J.A. (2009) Using LSA and the predication algorithm to improve extraction of meanings from a diagnostic corpus. *Spanish Journal of Psychology*, 12(2), pp. 424-440.

[28] Jorge-Botana, G., León, J.A.,Olmos, R., & Escudero, I. (2010). Latent Semantic Analysis Parameters for Essay Evaluation using Small-Scale Corpora. *Journal of Quantitative Linguistics, 17(1)*, pp. 1–29

[29] Suendermann, D., Hunter, P., Pieraccini, R., Call Classification with Hundreds of Classes and Hundred Thousands of Training Utterances ... and No Target Domain Data, in Perception in Multimodal Dialog Systems, *Proceedings of the 4th IEEE Tutorial and Research Workshop on Perception and Interactive Technologies for Speech-Based Systems*, PIT 2008, Kloster Irsee, Germany, June 16-18, 2008, Springer-Verlag, pp. 81-87.

[30] Kintsch, W., Patel, V., & Ericsson, K. A. (1999) The role of Long-term working memory in text comprehension. *Psychologia*, 42: 186-198.

[31] Kintsch, W. (2000) Metaphor comprehension: A computational theory. *Psychonomic Bulletin & Review, 7*, pp. 257-266.

[32] Kintsch, W. (2001). Predication. *Cognitive Science, 25*, pp. 173-202.

[33] Kintsch, W., & Bowles, A. (2002). Metaphor comprehension: What makes a metaphor difficult to understand? *Metaphor and Symbol, 17*, pp. 249-262.

[34] Kintsch, W. *Meaning in context*. In Landauer, T. K, McNamara, D., Dennis, S. & Kintsch, W. (Eds.) Handbook of Latent Semantic Analysis. Mahwah, NJ: Erlbaum. 2007: 89-105.

[35] Olmos, R., León, J.A., Jorge-Botana, G.,& Escudero I. (2009). New algorithms assessing short summaries in expository texts using Latent Semantic Analysis. *Behavior Research Methods*, 41, pp. 944-950.

[36] McCowan, Moore, Dines, Gatica-Perez, Flynn, Wellner & Bourlard. *On the use of information retrieval measures for speech recognition evaluation*. Technical Report. IAIDP. 2005.

[37] Bird, S. and Loper, E. NLTK: The Natural Language Toolkit. *In Proceedings of ACL04, 42nd Annual Meeting of the Association for Computational Linguistics*. Barcelona, Spain, July. 2004.

[38] Malmström P. E. *Methods for Evaluating a Natural Language Call Routing Application: A Case Study.* Master's thesis of the Uppsala University. Department of Linguistics and Philology. 2010