A Strategy for Multilingual Spoken Language Understanding Based on Graphs of Linguistic Units

Marcos Calvo Lance

A thesis submitted in fulfillment of the requirements for the degree of Doctor of Philosophy in the Departament de Sistemes Informàtics i Computació at Universitat Politècnica de València

Supervisors: Dr. Fernando García Granada
Dr. Emilio Sanchis Arnal

December, 2015

UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA

This thesis has been developed under a FPU scholarship (AP2010-4193) granted by the Spanish Government, and in the framework of the projects TIMPANO: Technology for complex human-machine conversational interaction with dynamic learning (TIN2011-28169-C05-01), and ASLP-MULAN: Audio, speech and language processing for multimedia analytics (TIN2014-54288-C4-3-R).

A mi familia.

Agradecimientos

Pensaba que no, que esta parte no me costaría mucho de escribir, pero no podía estar más equivocado. A veces es muy difícil expresar con palabras lo que uno siente, y más cuando una etapa tan importante está a punto de terminar. Y, aunque no suelo ser de agradecimientos públicos, creo que es justo agradecer y reconocer aquí el apoyo y esfuerzo de muchas personas sin las cuales esta tesis probablemente no habría sido posible.

En primer lugar quiero agradecer a mis directores Fernando García y Emilio Sanchis toda la ayuda que me han prestado a lo largo de este tiempo, y el haberse preocupado por que mi formación investigadora se completara. Todo empezó cuando Emilio Sanchis, que había sido mi profesor de algorítmica, me ofreció hacer mi proyecto final de carrera en el grupo de investigación ELiRF. En este grupo he trabajado durante más de seis años y siempre me he sentido valorado. Quiero agradecer a todos los miembros del grupo la confianza que siempre han mostrado en mí, el escuchar mis ideas por extrañas que a veces pudieran parecer, y el haberme ayudado a crecer como persona y como profesional. Pero, aparte de a mis directores, quiero agradecer en especial el apoyo, los conocimientos y la amistad que tres personas me han aportado siempre y que, de una forma u otra, han tenido un impacto decisivo en mi vida: Jon, Lluís y Salva.

El día a día en el laboratorio durante estos años ha sido realmente agradable, y gran parte del mérito lo tienen los que han sido mis compañeros durante todo o parte de este tiempo: Joan, Ximo, Mario, Merche, Lucía, Sergio, Flores, Jorge, Paco... Algunos decían que nos lo pasábamos muy bien en el laboratorio, y la verdad es que tenían razón.

Es muy posible que esta tesis jamás les llegue a ellos, pero me gustaría dar las gracias ahora, al final del camino académico, a algunos profesores que tuve antes de mi época universitaria, y sin los cuales no sería la persona que soy ahora mismo. Carmen, Doña Encarna, Don

Fernando, José (el conserje), Pepe, Ximo, Fernando, María José... Muchísimas gracias a todos vosotros.

Y por último, pero sin duda más importante: a mi familia. A mis padres y a mi hermano, que siempre han estado ahí durante todo este camino, y que seguirán estando siempre. Muchísimas gracias de corazón.

Marcos Calvo Lance
Zürich (Suiza)
14 de diciembre de 2015

Abstract

In this thesis, the problem of multilingual spoken language understanding is addressed using graphs to model and combine the different knowledge sources that take part in the understanding process. As a result of this work, a full multilingual spoken language understanding system has been developed, in which statistical models and graphs of linguistic units are used. One key feature of this system is its ability to combine and process multiple inputs provided by one or more sources such as speech recognizers or machine translators.

A graph-based monolingual spoken language understanding system was developed as a starting point. The input to this system is a set of sentences that is provided by one or more speech recognition systems. First, these sentences are combined by means of a grammatical inference algorithm in order to build a graph of words. Next, the graph of words is processed to construct a graph of concepts by using a dynamic programming algorithm that identifies the lexical structures that represent the different concepts of the task. Finally, the graph of concepts is used to build the best sequence of concepts.

The multilingual case happens when the user speaks a language different to the one natively supported by the system. In this thesis, a test-on-source approach was followed. This means that the input sentences are translated into the system's language, and then they are processed by the monolingual system. For this purpose, two speech translation systems were developed. The output of these speech translation systems are graphs of words that are then processed by the monolingual graph-based spoken language understanding system.

Both in the monolingual case and in the multilingual case, the experimental results show that a combination of several inputs allows to improve the results obtained with a single input. In fact, this approach outperforms the current state of the art in many cases when several inputs are combined.

Resumen

En esta tesis se aborda el problema de la comprensión multilingüe del habla utilizando grafos para modelizar y combinar las diversas fuentes de conocimiento que intervienen en el proceso. Como resultado se ha desarrollado un sistema completo de comprensión multilingüe que utiliza modelos estadísticos y grafos de unidades lingüísticas. El punto fuerte de este sistema es su capacidad para combinar y procesar múltiples entradas proporcionadas por una o varias fuentes, como reconocedores de habla o traductores automáticos.

Como punto de partida se desarrolló un sistema de comprensión multilingüe basado en grafos. La entrada a este sistema es un conjunto de frases obtenido a partir de uno o varios reconocedores de habla. En primer lugar, se aplica un algoritmo de inferencia gramatical que combina estas frases y obtiene un grafo de palabras. A continuación, se analiza el grafo de palabras mediante un algoritmo de programación dinámica que identifica las estructuras léxicas correspondientes a los distintos conceptos de la tarea, de forma que se construye un grafo de conceptos. Finalmente, se procesa el grafo de conceptos para encontrar la mejor secuencia de conceptos.

El caso multilingüe ocurre cuando el usuario habla una lengua distinta a la original del sistema. En este trabajo se ha utilizado una estrategia *test-on-source*, en la cual las frases de entrada se traducen al lenguaje del sistema y éste las trata de forma monolingüe. Para ello se han propuesto dos sistemas de traducción del habla cuya salida son grafos de palabras, los cuales son procesados por el algoritmo de comprensión basado en grafos.

Tanto en la configuración monolingüe como en la multilingüe los resultados muestran que la combinación de varias entradas permite mejorar los resultados obtenidos con una sola entrada. De hecho, esta aproximación consigue en muchos casos mejores resultados que el actual estado del arte cuando se utiliza una combinación de varias entradas.

Resum

Aquesta tesi tracta el problema de la comprensió multilingüe de la parla utilitzant grafs per a modelitzar i combinar les diverses fonts de coneixement que intervenen en el procés. Com a resultat s'ha desenvolupat un sistema complet de comprensió multilingüe de la parla que utilitza models estadístics i grafs d'unitats lingüístiques. El punt fort d'aquest sistema és la seua capacitat per combinar i processar múltiples entrades proporcionades per una o diverses fonts, com reconeixedors de la parla o traductors automàtics.

Com a punt de partida, es va desenvolupar un sistema de comprensió monolingüe basat en grafs. L'entrada d'aquest sistema és un conjunt de frases obtingut a partir d'un o més reconeixedors de la parla. En primer lloc, s'aplica un algorisme d'inferència gramatical que combina aquestes frases i obté un graf de paraules. A continuació, s'analitza el graf de paraules mitjançant un algorisme de programació dinàmica que identifica les estructures lèxiques corresponents als distints conceptes de la tasca, de forma que es construeix un graf de conceptes. Finalment, es processa aquest graf de conceptes per trobar la millor seqüència de conceptes.

El cas multilingüe ocorre quan l'usuari parla una llengua diferent a l'original del sistema. En aquest treball s'ha utilitzat una estratègia *test-on-source*, en la qual les frases d'entrada es tradueixen a la llengua del sistema, i aquest les tracta de forma monolingüe. Per a fer-ho es proposen dos sistemes de traducció de la parla l'eixida dels quals són grafs de paraules. Aquests grafs són posteriorment processats per l'algorisme de comprensió basat en grafs.

Tant per la configuració monolingüe com per la multilingüe els resultats mostren que la combinació de diverses entrades és capaç de millorar el resultats obtinguts utilitzant una sola entrada. De fet, aquesta aproximació aconsegueix en molts casos millors resultats que l'actual estat de l'art quan s'utilitza una combinació de diverses entrades.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 The problem of multilingual spoken language understanding

Communicative acts are constantly occurring in our lives. We want to share our experiences, feelings, and needs and also receive information about what is happening around us. These and many other information exchanges may be carried out either verbally by using words or by means of non-verbal communication. Both types of communication are important, but most of the conscious communication processes among humans are performed using words, either in a spoken or in a written form. On one hand, speech is the most natural way for humans to communicate with each other, since it is based on sounds that are emitted by our bodies. On the other hand, written language is a human invention which has developed throughout history and which constitutes a graphical representation of spoken language. Even though speech and writing can express the same messages and meanings, they are usually employed differently: written language tends to be more formal and thoughtful, and spoken language is often more informal and spontaneous. However, the massive use of social networks and instant messaging applications in the last few years has brought many of the features of spoken language into written language.

In order to be successful, any communication process requires the following elements:

- A *sender* who sends a *message* that is encoded using a *code*. In some cases, the code is as a language.
- A *receiver* who obtains the message. The receiver must also know the code in order to decode the message.
- A *channel* so that the message is transmitted from the sender to the receiver.
- Information that is external to the message and must be shared by the sender and the receiver. This information, which is called *context*, is needed by the receiver to be able to understand the semantic meaning of the decoded message and the relations that are included in it.

In other words, in a successful communicative process, the message sent by the sender is encoded in a language and arrives to the receiver through a channel; then the receiver decodes the message and can understand it because both the code and the context (which represent linguistic and non-linguistic information, respectively) are shared by both the sender and the receiver. However, some problems may arise. For example, one of these problems is the misrecognition of spoken or written words because they may have been uttered in a noisy environment or written in an illegible script, respectively. Another possible problem is the misunderstanding of the meaning of the message since some words may not have been properly recognized because the context is not completely shared by the sender and the receiver, or because one of them is not a native speaker of the language (i.e., the code is not completely shared).

Humans can naturally handle the complexity of communicative processes and can even ask for clarification if they feel that the communication is not being effective enough. However, in human-computer interaction a large part of this complexity has traditionally been avoided in order to make the interactions successful. For many years, human-computer interaction has been performed in ways that avoid ambiguities, such as commands and menus. The design of all these interactions makes that the intention of the user is impossible to be misunderstood. Nevertheless, these interactions provide a limited amount of information to be communicated to the computer as well as limited expressive flexibility.

In order to enhance human-computer interaction and make it more natural for humans, Natural Language Processing (NLP) has been an ambitious goal of research in computer

science in the last few decades. The purpose of NLP is to develop a set of techniques that allows a computer to analyze a sentence or a discourse that is expressed in a natural language (e.g., English, Spanish, French, etc.) and provide information that is inferred from it. These techniques very often rely on machine learning and pattern recognition fundamentals, such as statistical models that are automatically learnt from data, classification techniques and algorithms, and representations that are based on the features and relations that can be extracted from the input. Some examples of NLP tasks are part-of-speech tagging, which aims to identify the grammatical categories for each word in a text; syntactic analysis, which aims is to identify the syntactic components of a sentence; machine translation, which aims to translate texts from one natural language into another natural language (e.g., English into French); and natural language understanding, which aims to provide a representation of the semantics contained in a written or spoken text.

Many NLP applications usually expect a well-formed written sentence as their input. However, spoken language is a more natural and flexible way for human-computer interaction. Speech technologies attempt to develop a successful interface between humans and machines using spoken language. The first problem in speech processing that was historically addressed was to build an Automatic Speech Recognizer (ASR), in order to transcribe spoken utterances provided by humans in a specific language. Achieving good quality in speech transcription would provide an interface for using text-based applications with speech input. Furthermore, it would enable the construction of more complex voice-activated systems. For example, for many years, one ambitious goal of speech technologies was the development of Spoken Dialogue Systems (SDSs). These systems mimic the understanding and dialogue skills of a human and provide a fully dialogued communication between humans and computers. SDSs open up more possibilities, such as being able to use a computer and receive assistance from it in environments where it is impossible to use your the hands for entering text (for example while driving a car or during surgery). The development of SDSs is an open research field, and many of its aspects must still be improved. This is the reason that most of the SDSs that have been developed to date model less critical tasks such as e-mail applications or help-desk and information services.

Taking a closer look to the understanding abilities of a SDS, they are usually performed by a module devoted to Spoken Language Understanding (SLU), the goal of which is to provide a semantic interpretation of the user's utterance. For this purpose, SLU modules need to have the message that was transmitted by the user (the acoustic signal or its

transcription), information about the language in which the user spoke (the code of the communication process), and all the necessary information that provides meaning to the words of the utterance (the context). Taking this into account, two major problems can be identified when a SLU system is developed: (i) how to adequately model and represent all the information that is involved in the understanding process, and (ii) the development of a semantic decoding algorithm that uses this information and infers a semantic interpretation of the input utterance (De Mori et al. 2007; Tur and De Mori 2011).

SLU has traditionally been divided into three problems which, to some extent, try to mimic the different characteristics of the human understanding process. All of them are usually addressed using machine learning techniques, which, in some cases, may be enriched with expert linguistic knowledge. The three problems are:

- *Domain detection*: In order to understand a sentence, the SLU module must figure out what the user is talking about. For example, it must determine whether the user is speaking about flight tickets, about the weather, or about the actions that can be performed on the user's bank account. Most current SLU systems are specialized in just one domain (restricted-domain SLU systems). This means that if there is an application that supports several actions, the SLU module is different for each of them, and then a decision module determines which of the interpretations is the most suitable (Khan et al. 2015).

- *Intent determination*: It is important to capture the goal of the user in order to provide the information that the user is asking for. For example, in the context of a flight booking system, it is necessary to determine if the user wants to book a ticket, know what the possible direct flights that depart from a given city are, or obtain information about the price of a specific flight. A nice example of an intent determination system is the *How may I help you?* system by AT&T (Gorin, Riccardi, and Wright 1997). This system was implemented for the customer care service in that company. When the user called the customer care service, the user was prompted with the question "How may I help you?". Then, the system extracted the intent of the user from the answer, and the call was forwarded to a human operator based on the inferred intent.

- *Slot filling*: The goal that is pursued here is to identify the semantic concepts that appear in the user's utterance, along with their associated values. The set of concepts that may appear within a sentence in the scope of a SLU task must be established beforehand, during the definition of the SLU task, and these concepts constitute an ontology for the task. For example, in a task that models a flight booking system, concepts such as *departure city*, *arrival city*, *date*, *price*, and *airline* can be defined. This way, in the sentence *"I'd like to travel from Rome to Athens tomorrow"* three concepts can be found with associated values: a *departure city*, which is Rome; a *destination city*, which is Athens; and a *date*, which is tomorrow. Each concept is supported by a set of words of the sentence. For example, the concept *departure city* is substantiated by the words *from Rome*, and the value Rome can be extracted from it. Furthermore, sometimes additional concepts are defined in order to express the user's intent, despite the fact that they do not usually have an associated value and they are not directly related to the task-dependent semantic concepts. This allows the intent determination problem to be embedded into the slot filling problem. For example, the concept *query* may be added to the flight booking task in order to express that the user is asking for some information. Then, this concept *query* can be found in the former sentence supported by the words *I'd like to travel*. For this reason, the slot filling problem is often modelled as a sequence labelling problem in which each word or set of words of the sentence is assigned a semantic label that corresponds to the concept that the words support. This constitutes a semantic representation of the user's utterance.

It is worth highlighting that domain detection can be skipped if the system does not need to choose between different domains, and the intent determination problem can be embedded into slot filling if an adequate semantic model is used. For this reason, in the remainder of this thesis, the problem of SLU will be restricted to the slot filling problem. However, this definition of the SLU problem still makes it very challenging.

Based on this definition of SLU, one of the problems that must be confronted is that SLU is described in terms of linguistic units (words), but the user's input is an acoustic signal (an utterance). One possible solution for this problem is to perform a speech recognition process before SLU so that the input to the SLU method is a transcription of the user's utterance. Therefore, the ASR must have very good accuracy to ensure that no relevant information is lost in the recognition process. However, current technology for speech recognition is still

far from perfect, achieving word error rates of around 15% in the transcription of broadcast news (Hinton et al. 2012). Moreover, these errors can completely change the meaning of the original utterance. This means that an in-cascade architecture where a SLU module is fed with a single transcription provided by an ASR is probably not the best setting. However, if several hypotheses were delivered to the SLU module, it could choose among more possibilities in order to find the most suitable semantic interpretation. Two formats for conveying several transcriptions to the SLU module are a list of transcriptions (n-best list) and a graph structure (word lattice or graph of words).

Graphs have been successfully used in many NLP applications, and they have proven to be powerful and effective representations for linguistic information (Mihalcea and Radev 2011). However, the problem with graph structures for SLU is that the usual sequence labeling methods are not able to naturally process them since they do not represent a single sequence to be labeled, but rather a set that is represented as a structure that is more complex than a single sequence. Two of the contributions of this thesis are the modelling of the SLU problem as a graph search problem and the development of a SLU algorithm that is able to naturally and efficiently process several transcriptions that are represented as a graph structure.

The several transcriptions that arrive to the SLU system can be delivered either by a single ASR or by a set of ASRs that work in parallel. Actually, in the last few years it has been shown that the use of an appropriate combination of machine learning systems (also known as ensembles of classifiers) outperforms the results obtained using just one system. There has been a large amount of research on how the different systems can be combined, and many fusion methods have been developed (Xu, Krzyżak, and Suen 1992; Kittler et al. 1998; Brill and Wu 1998; Ruta and Gabrys 2000; Florian et al. 2003; Rokach 2010). Furthermore, not only outputs from different classifiers can be fused, but inputs from several sources can also be merged. This idea allows data from several sources or multimodal data (e.g., images and speech) to be combined to develop more robust systems (Ross and Jain 2003) or to analyze multimedia (Atrey et al. 2010). In fact, some evaluation campaigns such as the MediaEval Benchmarking Initiative for Multimedia Evaluation[1] try to advance the research on the combination of multimodal data for a number of tasks (Eskevich et al. 2015; Poignant, Bredin, and Barras 2015). Inspired by this outlook, and exploiting the fact that several ASR outputs (from one or more ASRs) can be combined into a graph that can be analyzed

[1] www.multimediaeval.org

using a graph-based SLU algorithm, another contribution of this thesis is the combination of multiple ASR outputs into a graph of words for SLU purposes.

Due to the challenging nature of the SLU problem, it would be very helpful if, to some extent, the effort that is required to develop a SLU system for one language or task could be re-used for other languages or other tasks. This process is known as language portability or domain portability of SLU systems, respectively. In the case of language portability, it means that the *code* (i.e., the language) is not shared between the user and the original SLU system. This implies that a translation process is needed so that both of them use the same language. In the case of domain portability, the SLU system is going to be used in a new task and therefore must acquire some knowledge about it. In other words, the SLU system must learn about a new *context*.

This thesis addresses the problem of language portability, also known as multilingual SLU. As mentioned above, a translation process is needed in order to equalize the languages of the user and the system. Two methodologies can be used for this purpose:

- Translating the SLU system or the data from which it was learnt to the user's language. This strategy is known as *train-on-target*.
- Translating the user's utterances into the language of the SLU system, and then processing them with this SLU system. This strategy is called *test-on-source*.

Even though results may be dependent on the corpus and the pair of languages that are considered, works like (Lefèvre, Mairesse, and Young 2010; Jabaian, Besacier, and Lefèvre 2013) suggest that the test-on-source methodology leads to better results than the train-on-target methodology. This is why the multilingual SLU system developed in this work uses a test-on-source methodology. Two speech translation systems have been developed for this purpose. In addition, different translations are combined in a graph of words to be processed by the graph-based SLU method.

1.2 Objectives and contributions of this thesis

The main objective of this thesis is the development of a SLU system that is able to work in a multilingual environment and that has the capability of processing uncertain input that is represented as a graph of words in a natural and efficient way. To confront this challenge, the SLU problem was first modelled as a graph search problem within a statistical framework. This formalism is able to naturally accept graph inputs instead of just strings. A grammatical inference algorithm for combining several sentences (e.g., transcriptions from an ASR) into a graph of words was studied and was then used to generate the graph inputs to the SLU module. Then, a set of dynamic programming algorithms was developed, analyzed, and evaluated to solve the graph search description of the SLU problem. Finally, this monolingual method was ported to a multilingual environment by incorporating a speech translation method. Two ways of implementing the speech translation step have been studied. In order to carry out an experimental evaluation of the multilingual system, a low-cost strategy for language portability of SLU corpora was developed.

The development of this thesis has led to the following contributions:

- The use of a grammatical inference algorithm to combine and generalize several transcriptions or translations of the same utterance according to their lexical and syntactic structures, with the goal of being further processed by a SLU module. The result of this combination is a left-to-right stochastic automaton or graph of words.

- The modelling of the SLU problem as a graph search problem. This graph search combines the information represented in a graph of words with a semantic model of the SLU task, which is also represented as a set of graph structures. Furthermore, this graph representation of the problem fits into a statistical framework, where the models represent probability distributions, and the goal is to search for the sequence of concepts with maximal probability given an utterance provided by a user.

- The development of a strategy that allows the best sequence of concepts for a given utterance to be computed using algorithms on graphs. First, the input is represented as a graph. Then, the SLU strategy is divided into two stages, in which different linguistic units are represented in the graphs. The first stage processes a graph of words and builds a graph of concepts which has lexical, syntactic, and semantic

information attached to its arcs. The algorithm used to build this graph of concepts is a Dynamic Programming algorithm that, for a given concept, searches for all the best sub-paths in the graph of words between any pair of nodes, taking into account the semantic model for that specific concept. This procedure is repeated for all the concepts of the task. The second stage uses another dynamic programming algorithm to find the best path in the graph of concepts, also taking into account the information contained in a semantic model of sequences of concepts. This procedure provides the best sequence of concepts; its underlying sequence of words, which is a transcription of the user's utterance; and a segmentation of the sequence of words in terms of the sequence of concepts.

- The construction of a multilingual SLU corpus by porting the Spanish DIHANA corpus to English and French, and the development of a low-cost language portability process for SLU corpora in which both human and machine translation are used.

- An extension of the graph-based SLU algorithm to test-on-source multilingual SLU. In this case, the outputs provided by one or more automatic translators are combined using the graph builder algorithm. Two methods have been proposed for the speech translation process: (i) a combination of several general-purpose freely-available web translators and (ii) an unsupervised acquisition of a task-dependent parallel corpus to train a phrase-based statistical machine translation system. The output of this system is analogous to the monolingual setting, but, in this case, the sequence of words corresponds to a translation of the user's utterance into the system's language. The influence of the combination of translations and the domain-specific semantic information in the quality of the resulting translation has also been analyzed.

1.3 Structure of the thesis

The content of this thesis is structured in six chapters. After the introduction to the topic of the thesis, the Chapter 2 provides a more detailed explanation of the historical evolution of SLU.An in-depth analysis of several statistical approaches to SLU is provided as well as some extensions to accept uncertain input that is represented as graphs of words is discussed. The problem of multilingual SLU is also examined, and the two main approaches for solving this problem (train-on-target and test-on-source) are presented.

Chapter 3 introduces the corpora that were used in the different experiments presented in this thesis. For Monolingual SLU, the chosen corpora are the Spanish DIHANA corpus and the French MEDIA corpus. Furthermore, the Spanish DIHANA corpus was translated into French and English using different translation methods (machine translation and human translation) and was partially uttered by native users. This way, the multilingual DIHANA corpus was obtained. The multilingual DIHANA corpus was the one used for the multilingual SLU experiments.

Chapters 4 and 5 describe the main contributions that have been developed in this thesis: a graph-based SLU algorithm that is able to process graphs of words obtained from one or more sources and its extension to a multilingual setting. Chapter 4 describes this graph-based SLU algorithm for a common monolingual environment. This algorithm has the ability to process uncertain input represented as graphs of words in a natural and efficient way by means of several dynamic programming algorithms. Furthermore, the use of a grammatical inference algorithm to build a graph of words from a set of sentences for SLU purposes is also explored. An experimental evaluation of the method is carried out using both speech and text input, and a comparison with other state-of-the-art SLU methods is performed.

In Chapter 5, the monolingual model is extended to build a test-on-source multilingual SLU system, where speech translation is performed before the SLU module. Two different methods were developed to carry out the speech translation process. Both of them provide several outputs, which are combined in a graph of words in order to provide the SLU module with more possible translations. These two speech translation methods lead to two different multilingual SLU systems which are thoroughly evaluated, and they are also compared to other in-cascade approaches to multilingual SLU.

Finally, Chapter 6 presents the conclusions that have been drawn and some lines of future work.

Chapter 2

State of the art

2.1 Spoken language understanding in the context of spoken dialogue systems

Speech technologies have a fairly short, but rich history. One of the first goals of the research on speech technologies was the development of a continuous automatic speech recognition software that works with a reasonable recognition error rate. Once this goal was achieved, then the first research projects and proposals of SLU systems emerged. One milestone during the initial years of the research on SLU was the creation of the ATIS corpus (Hemphill, Godfrey, and Doddington 1990), which represented the domain of a flight information system. This corpus was the first to incorporate a semantic labeling to the information of the utterances.

The first SLU systems were conceived as components of a spoken dialogue system. SDSs constitute a natural interface for human-machine interaction, which is performed through conversational speech. Therefore, a good SLU system is crucial for a SDS to work correctly since the user's utterances must be properly understood before providing the user with an answer (see Figure 2.1). Initially, the development of open-domain SDSs that could interact with the user and provide the user with any information seemed to be a too ambitious goal. For example, one important problem that still had to be solved was the construction of a high-performance very large vocabulary open-domain ASR. Therefore, the most reasonable

option was to develop restricted-domain SDSs (Walker, Passonneau, and Boland 2001), which could work with a much simpler task-dependent ASR. Some examples of restricted-domain spoken dialogue systems are:

- *VOYAGER* (Zue et al. 1992), which was to provide dialogued access to geographical information and other services as well as telephone numbers within a restricted area. This system originally worked in English, but was later ported to other languages such as Japanese and Italian (Glass et al. 1995).
- The *Carnegie Mellon Communicator* system (Rudnicky et al. 1999), which allows the user to book flight tickets, hotels, and car rentals using the English language.
- *JUPITER* (Zue et al. 2000), which works in English and provides weather information.
- *Let's go* (Raux et al. 2005), which is a spoken dialogue system in English to access bus schedule information in Pittsburgh.
- A SDS for the DIHANA task (Griol et al. 2006), where the user can obtain information about railway timetables and fares in Spanish.
- *DS-UCAT* (López-Cózar, Callejas, and Montoro 2006), which works in English and Spanish and has the goal of assisting students and teachers in common activities in an academic environment.
- A dialogue system for the EDECAN-SPORT task (Planells et al. 2012), where the user's goal is to perform booking operations on sports facilities in Spanish.

More recently, systems that are able to provide users with information about different domains have been built; these are called multi-domain spoken dialogue systems. This is the case of the system presented in (Planells et al. 2013), where three single-domain SDSs in Spanish are integrated into a single SDS (one for booking sports facilities, one that provides weather information, and one that manages a personal calendar).

As Figure 2.1 shows, the input to a usual spoken dialogue system is a spoken utterance that encodes information provided by the user. This utterance is first processed by an ASR that transcribes it using information from acoustic and language models. Then the sequence of words that the ASR outputs is analyzed by a SLU module which provides a

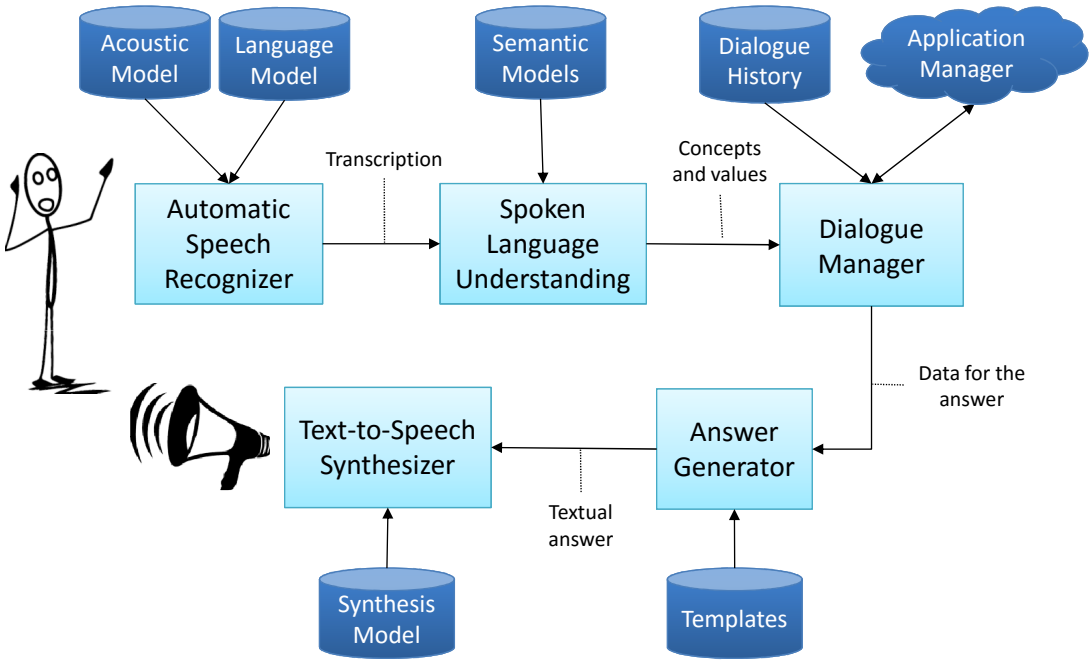

Figure 2.1: Scheme of the modules of the usual architecture of a spoken dialogue system.

semantic interpretation of the sentence by using one or more semantic models. In order to provide a complete semantic analysis of the utterance, the different concepts that are found in it must be instantiated with their corresponding values. One possible way to do this is to use a frame representation. In a frame representation the goal of the user is identified, as well as the different attribute-value pairs that summarize the semantic information of the utterance. This information is provided to a dialogue manager that takes into account the previous history of the dialogue and the semantic information to build a query to a structured information repository, such as a database, if the query is needed to provide the user with an answer. Then, the dialogue manager determines what is going to be said in the system's answer, and this information is lexicalized by an answer generator module. Finally, this sequence of words is produced by means of a text-to-speech synthesizer, in order to deliver it to the user in a more natural way. After the answer is received, the user can decide whether or not the information provided by the system fulfills the user's objective. If the system's answer is satisfactory, then the user finishes the dialogue; otherwise the user can provide the system with more information in order to fulfill the user's goal.

To develop restricted-domain SDSs, several SLU systems and corpora with semantic annotations were developed in the framework of a number of research projects for a wide range of languages and tasks. Some examples are described below.

- The ATIS corpus (Hemphill, Godfrey, and Doddington 1990) was the first dataset that incorporated semantic annotations. This corpus was developed in the US, and it represents the domain of an information system for airlines (in fact, ATIS stands for *Air Travel Information System*).
- In the framework of the ARISE project (Den Os et al. 1999), four prototypes of SDSs for different train information systems were developed. These prototypes worked in French (two systems), Italian and Dutch.
- The French MEDIA/EVALDA project (Devillers et al. 2004) specifically aimed to assess the performance of SLU systems in the context of SDSs. For this purpose, the MEDIA corpus (Bonneau-Maynard et al. 2005; Bonneau-Maynard, Quignard, and Denis 2009) was developed. The content of this corpus is in French and represents a tourist information task.
- The Spanish DIHANA project, had the goal of developing a SDS to access a train information service by phone using spontaneous speech in Spanish. The DIHANA corpus (Benedí et al. 2006) was built in the framework of this project
- The European LUNA project had the research goals of improving language and semantic modelling for SLU, enhancing the robustness of SLU systems, and exploring the multilingual portability of the SLU components. Three corpora were developed for this purpose: one in Italian for a computer help-desk service (Dinarelli et al. 2009); another in Polish for a public transportation information service (Marasek and Gubrynowicz 2008); and a third one in French for stock exchange queries.

More details on corpora for SLU, specifically on the DIHANA and MEDIA corpora, are provided in Chapter 3.

In addition to academic research, recent advances in speech technologies and natural language processing have enabled the creation of commercial personal assistants for smartphones, with one of their features being the possibility to request them information by

means of speech. Examples of these assistants are Google Now[1], Apple's Siri[2], Microsoft's Cortana[3], and Sherpa[4]. In these systems the idea of spoken language understanding plays a very important role since the system must extract the relevant information from the user's query in order to provide the user with the requested information or to ask for more data if it is needed.

2.2 Statistical approaches to spoken language understanding

From a methodological point of view, the different approaches to SLU can be divided into two types: rule-based and statistical-based. Some of the first approaches to SLU were rule-based, which were based on hand-crafted grammars that could be enriched with probabilities learnt from training data. Two examples of systems of this kind are TINA (Seneff 1989; Seneff 1992) and Phoenix (Issar and Ward 1993). Furthermore, the statistical information that was added to the rules allowed to better handle the usual errors that occur in systems that depend on a previous ASR, even though the robustness to ASR errors was still very low. Also, the manual design of these grammar-based SLU systems involved a great linguistic and engineering expertise in order to avoid coverage leaks and to achieve good performance. Hence, the development of these systems was difficult and time-consuming, and the final system was likely to be error-prone and problematic to maintain.

It was at the beginning of the 1990s when the first statistical-based models for spoken language understanding started to be developed. These models have the advantage that they can be learnt from corpora, instead of requiring manual effort from experts in linguistics. Two examples of these early statistical-based systems are CHRONUS (Pieraccini et al. 1992; Pieraccini and Levin 1995), which was developed by AT&T and was based on Hidden Markov Models; and the system developed by the BBN company, which was based on Hidden Understanding Models (Miller et al. 1996; Schwartz et al. 1996).

In a statistical framework, the problem of restricted-domain spoken language understanding can be stated as the search for a sequence of concepts $\hat{C}$ that best indicates the meaning of an utterance A. Since this is a restricted-domain task, a set of concepts $\mathscr{C}$ should be

[1] www.google.com/landing/now/
[2] www.apple.com/ios/siri/
[3] windows.microsoft.com/en-us/windows-10/getstarted-what-is-cortana
[4] sher.pa

defined beforehand, usually by a human expert, thereby creating an ontology for the task. These concepts identify the pieces of information that are contained in a sentence in the context of the task and that are relevant in its domain. For example, in the context of a train-schedule information system, some possible concepts that can be identified are the origin and destination city of a trip, the departure and arrival times, and the services that are offered during the trip (e.g., if it is possible to smoke on the train, if there is a restaurant carriage, etc.). In addition, a *dummy*, *null* or *void* concept that represents *filler words* is often added. Filler words are the words that are not attached to any relevant concept. Therefore, according to this scheme, all the words of a sentence will be attached to some concept, using the filler concept when a word is not relevant to any of the other concepts. Also, each word is attached to a single concept since each word is part of a single idea being communicated in the utterance. In many Western languages, the words that shape a concept are adjacent to each other in a sentence. The resulting structure is called a segmentation of the sentence in terms of a sequence C of concepts of the task. Table 2.1 shows an example of a segmentation for a public transport information system.

Transcription of the user's utterance	Good morning, I'd like to know the timetables to go to Liverpool since I have a meeting there.
Semantic segmentation	Good morning: courtesy I'd like to know: query the timetables: time? *(object of the query)* to go to Liverpool: destination_city since I have a meeting there: null

Table 2.1: Example of a segmentation for a train-schedule information system.

When training statistical models for SLU, it is very important that the training corpus is segmented and labelled with the concepts of the task in order to properly estimate the probabilities or features that take part in the computations. The process of segmenting and labeling a corpus for SLU purposes is usually a time-consuming task that must be carried out by experts in linguistics. However, some approaches have recently been proposed to assuage the cost of this task. For example, active learning strategies allow bootstraping from a smaller amount of samples (Gupta et al. 2006; García et al. 2011). It is also possible to train models for SLU when the set of concepts that appear within a sentence is known, but the segmentation of the sentence in terms of those concepts is unknown (Ortega et al. 2010). Another option is to use targeted crowdsourcing techniques to have the corpus segmented

and labelled at a lower monetary cost (Chowdhury et al. 2014). All these approaches make use of human knowledge to some extent and need the active participation of a human to provide providing certain (possibly small) amount of information. However, there are also works that aim at unsupervisedly acquiring all the semantic information that is needed to train a SLU system (Tur et al. 2012; Chen, Wang, and Rudnicky 2015).

Since SLU is defined as the search for a sequence $\hat{C}$ that best fits an utterance A, it stands that, in principle, $\hat{C}$ may be any sequence of concepts that is built from the elements in $\mathscr{C}$. In other words,

$$\hat{C} = \underset{C \in \mathscr{C}^*}{\operatorname{argmax}}\, p(C|A) \tag{2.1}$$

where $\mathscr{C}^*$ is the Kleene star closure of $\mathscr{C}$. However, due to the existence of the *null* concept, every non-empty sequence of words will have a non-empty sequence of concepts attached to it. Hence, it is more appropriate to substitute the Kleene star closure by the Kleene plus closure (i.e., $C \in \mathscr{C}^+$). Nevertheless, for the sake of readability, the subindex of the argmax operation will simply be C in the remainder of this thesis.

One possible solution for the search of the best sequence of concepts given an acoustic signal would be to integrate in a single unified system or model all the knowledge sources that take part in the spoken language understanding process (acoustic, lexical, syntactic and semantic) and apply all the constraints at once. Unfortunately, this solution generates an excessively large search space, thus increasing the difficulty of the task. In this unified model, the different individual models should be properly weighted during their combination, in order to improve the performance of the system. The computation of these weights makes the application of the unified model even harder. For this reason, a more realistic option is to use a modular sequential architecture, in which the information that is transmitted from one module to the following one is a set of hypotheses. This set usually only contains the 1-best transcription, which allows the SLU module to work on a text string. However, the best transcription provided by the ASR may not be the most semantically complete sentence of the hypotheses the ASR built (for example due to the existence of an error in a semantically relevant word). Therefore, it seems an advantage that this set contains multiple transcriptions, either in the form of a n-best list or as a word lattice.

In the last few years, the idea of transmitting more than one hypothesis to the SLU module has gained notable interest, as shown in works such as (Tur et al. 2002; Hakkani-Tür et al. 2006; Henderson et al. 2012; Bayer and Riccardi 2012; Svec, Ircing, and Smidl 2013; Svec et al. 2015), among others. Being able to take into account more than one transcription when performing the semantic decoding has the main advantage that ASR errors, which can spoil the meaning of the sentence completely, can be alleviated if the SLU module is able to adequately search among the different options provided. This trend can also be observed in other speech applications such as speech translation (Bertoldi and Federico 2005; Bertoldi, Zens, and Federico 2007; Dyer, Muresan, and Resnik 2008). Another variant that is also explored in this thesis is to use a graph representation to fuse the information provided by one or more systems and to build a final sentence that represents a consensus among them. This approach has been successfully used for mixing the outputs from several ASRs (Fiscus 1997; Schwenk and Gauvain 2000), as well as from several machine translation systems (Bangalore, Bordel, and Riccardi 2001; Freitag, Huck, and Ney 2014).

Other options aim at a tighter coupling of the ASR and SLU processes. One early approach was to estimate confidence scores during the ASR process and use them at the SLU stage (Hazen, Seneff, and Polifroni 2002). This way, the SLU module can have information on how well the system estimates that a word has been recognized and use this information to try to overcome the harmful effects of misrecognitions in the semantic decoding. More recently, in (Zamora-Martínez et al. 2012), the authors attempted to enrich the ASR module with semantic knowledge, and achieved promising results. Other works such as (Tur, Deoras, and Hakkani-Tur 2013; García et al. 2015a) have explored the possibility of learning the semantic models for SLU by using the automatic transcriptions of the training sentences instead of the orthographically correct sentences. This way, the variability introduced by ASR errors is conveyed to the SLU system.

The most common interpretation of the SLU problem is to see it as a Sequence Labeling Problem, specially when text is taken as its input. This means that, given an input (a sequence of words W) the goal is to provide a sequence of labels C (concepts) that segments the initial sequence. There are two major approaches to sequence labeling: establishing a label for each word, in such a way that adjacent equal labels represent the same segment; and directly finding a segmentation of the input sequence. The approach that is used depends on the particular statistical model.

There are two main families of approaches to statistical-based SLU: generative and discriminative. Both types of approaches are often considered to be independent because they express different points of view on how the sequence of concepts should be obtained, even though both of them come from the same initial maximization.

Discriminative approaches (Wang and Acero 2006) aim at directly maximizing the probability of the sequence of concepts C given a sequence of words W:

$$\hat{C} = \underset{C}{\operatorname{argmax}}\, p(C|W) \tag{2.2}$$

Discriminative approaches consider the sequence of words W as granted, and so they can use all the information in it at any time during the search for the best sequence of concepts. For example, when deciding the concept to which the i-th word w_i belongs, information about the preceding words w_{i-x} and the following words w_{i+y} can be used. The information on the concepts that have appeared until then can also be used. Some discriminative approaches to SLU are support vector machines (Haffner 2006; Mairesse et al. 2009), linear chain conditional random fields (Raymond and Riccardi 2007), log-linear models for statistical machine translation (Macherey, Bender, and Ney 2009; Jabaian, Besacier, and Lefèvre 2011; Jabaian, Lefèvre, and Besacier 2014), and recurrent artificial neural networks (Mesnil et al. 2013; Yao et al. 2013; Yao et al. 2014; Vukotic, Raymond, and Gravier 2015).

Generative methods can be derived from Equation 2.2 as shown in Equation 2.3. This means that generative methods look for the sequence of concepts $\hat{C}$ that maximizes the joint probability of $\hat{C}$ and the sequence of words W provided by the ASR.

$$\hat{C} = \underset{C}{\operatorname{argmax}}\, p(C|W) = \underset{C}{\operatorname{argmax}}\, \frac{p(W|C) \cdot p(C)}{p(W)} = \underset{C}{\operatorname{argmax}}\, p(W, C) \tag{2.3}$$

Generative methods build the sequence of concepts taking the words in W consecutively, as if the sequence of words was being built at the same time as the sequence of concepts. This implies that only information on the words and concepts that have appeared until then can be taken into account when establishing the concept that a word belongs to. Historically, this family of approaches appeared before the discriminative family since generative algorithms are inspired on the equations that are traditionally used for speech recognition,

which was a very important research topic in the 1990s. The relation to speech recognition is established by considering the similarity of $p(W|C)$ to an acoustic model and the similarity of $p(C)$ to a language model. Hence, it is not surprising that the first statistical-based systems for SLU were based on hidden Markov models (Pieraccini, Levin, and Lee 1991; Pieraccini et al. 1992; Pieraccini and Levin 1995; Johnsen et al. 2000) since they were the state-of-the-art models for speech recognition. Other generative methods for SLU that have appeared since then are the hidden vector state models (He and Young 2003; He and Young 2005), the models based on composition of stochastic finite state automata (Sanchis et al. 2000; Segarra et al. 2002) or composition of finite state transducers (Raymond et al. 2006), and the use of dynamic Bayesian networks (Lefèvre 2007).

The existence of so many statistical methods for SLU make it very useful in practice to be able to perform comparisons between them. There are several works in this line, for example (Raymond and Riccardi 2007; Hahn et al. 2008; Dinarelli 2010). The work by (Hahn et al. 2011) provides a very comprehensive comparison between generative and discriminative methods, and the authors report that the best result on the French MEDIA corpus was achieved by using linear chain conditional random fields with a specific set of feature functions.

The following sections provide a detailed explanation of several of the major generative and discriminative approaches to SLU. Furthermore, some of these approaches have been extended to be able to deal with uncertain input, which can be represented as a graph of words.

2.2.1 Hidden Markov models

First-order Hidden Markov Models (HMMs) are a powerful generative formalism that are able to model short-term relations between elements of a sequence. Since they had been successfully applied to speech recognition tasks, they were one of the first statistical formalisms that were used for spoken language understanding. However, in the case of speech recognition they are continuous HMMs since the acoustic features are real numbers, while in the case of SLU they are discrete HMMs because the input is composed of words.

The use of HMMs for SLU was first proposed in (Pieraccini, Levin, and Lee 1991) and later used by the same authors in the CHRONUS system (Pieraccini et al. 1992; Pieraccini and

Levin 1995). Other spoken dialogue systems have also used HMM-based SLU modules, for example the one presented in (Johnsen et al. 2000) for a bus information service in Norwegian language.

The reason why HMM are feasible models for SLU tasks is because generative methods for SLU can be described as the combination of the prior probability of a sequence of concepts and the conditional probability of the sequence of words given that sequence of concepts:

$$\hat{C} = \underset{C}{\operatorname{argmax}}\, p(W, C) = \underset{C}{\operatorname{argmax}}\, p(W|C) \cdot p(C) \tag{2.4}$$

Let C and W be sequences of the same length ($|W| = |C| = K$) such that the i-th element in C, c_i, corresponds to the concept tag that is associated to the i-th word in W, w_i. Applying the chain rule, Equation 2.4 can be decomposed into:

$$p(W|C) \cdot p(C) = [p(w_1|C) \cdot \prod_{i=2}^{K} p(w_i|w_{i-1}...w_1, C)] \cdot [p(c_1) \cdot \prod_{i=2}^{K} p(c_i|c_{i-1}...c_1)] \tag{2.5}$$

Both product terms in this equation denote the probability of long-term dependencies, either at the word level or at the concept level. However, these dependencies can be reasonably approximated by short-term dependencies, as is done in language modelling for speech recognition. It can also be legitimately assumed that when the i-th word of the input is considered the concept tag for it is independent of the concept tags of the other words in W. Thus, Equation 2.5 can be rewritten as:

$$p(W|C) \cdot p(C) = [p(w_1|c_1) \cdot \prod_{i=2}^{K} p(w_i|w_{i-1}...w_{i-m}, c_i)] \cdot [p(c_1) \cdot \prod_{i=2}^{K} p(c_i|c_{i-1}...c_{i-n})] \tag{2.6}$$

This equation denotes two Markov processes: the part that calculates the conditional probabilities is a Markov process of order m; and the terms that compute the prior probability

of the sequence of concepts is a Markov process of order n. Both processes can be approximated by first-order Markov processes, resulting in the following:

$$p(W|C) \cdot p(C) = [p(w_1|c_1) \cdot \prod_{i=2}^{K} p(w_i|w_{i-1}, c_i)] \cdot [p(c_1) \cdot \prod_{i=2}^{K} p(c_i|c_{i-1})] \tag{2.7}$$

By making this approximation, the resulting equation describes a first-order HMM, where the states denote the concepts of the task and each state has an estimate of the emision probabilities for each word of the vocabulary. It is worth noting that the topology of a HMM that fits Equation 2.7 represents a bigram language model of the concepts of the task, based on a training corpus. Higher order HMM are also possible, but first-order HMM are the most widely used in the literature. Once the HMM is learnt, for example using a segmented and labelled training corpus, then the best sequence of concepts can be obtained by means of a Viterbi search on the HMM according to the input sequence W.

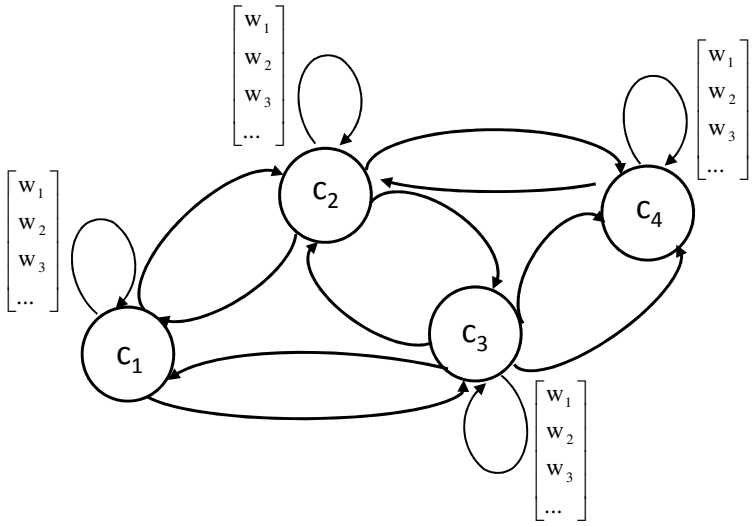

Figure 2.2: Example of a Hidden Markov Model for SLU.

Multiple variants on this scheme have been proposed in the literature. For example, the Hidden Understanding Models (Miller et al. 1996; Schwartz et al. 1996) substitute the transitions of the HMM with a stochastic grammar. Another variant is the Hidden Vector State model (He and Young 2003; He and Young 2005), which aimed at enhancing the

expressivity of HMM by introducing hyerarchical relationships that can better model long-term dependencies among the elements of the sequence of concepts. Another important variant of HMMs is their extension to discriminative modelling. These are called maximum entropy hidden Markov models and were proposed in (McCallum, Freitag, and Pereira 2000). In this model the emision probabilities are substituted by feature functions that allow decisions to be taken grounded on the full sequence of words W and on the previous concept in the sequence c_{i-1}. This model has been used for SLU purposes in works like (Bender et al. 2003). Maximum entropy hidden Markov models are the predecessors of linear chain conditional random fields, which are explained in Section 2.2.3.

2.2.2 Stochastic finite state machines

Another generative approach to SLU that has been widely used in the literature is Stochastic Finite State Machines (SFSMs). Similarly to HMMs, SFSMs were successfully used in speech recognition before their application to SLU. The use of SFSMs for SLU is explained by the fact that it is a very powerful mechanism for analyzing sequences, and the input to SLU is either a sequence of words or a sequence of vectors of acoustic features that represent the user's utterance. However, since the expressive power of SFSMs is restricted to regular languages, they are only able to capture short-term relations among the components of the sequence.

From a transduction point of view, SLU can be interpreted as a process that analyzes an utterance A or a sequence of words W and converts it into a sequence of concepts C. Among the finite-state formalisms that generate outputs as a function of an input, Moore and Mealy machines are two well-known models that provide different perspectives of the transduction process: in Moore machines the output symbols are determined at the states; but in Mealy machines the output symbols are imposed by the transitions (i.e., by the current state and the input symbol).

First, let us consider Moore machines, and their relation with Equation 2.6. In this equation, it was stated that the generation of the words that compose a concept can be generated by a Markov process of order m, and the sequence of concepts can be generated by another Markov process of order n. However, these Markov processes can also be interpreted as the probability provided by an $(m+1)$-gram language model and an $(n+1)$-gram language model, respectively. Works such as (Riccardi, Pieraccini, and Bocchieri 1996; Torres and

Varona 2001) have shown that n-gram language models can be represented as stochastic finite state automata (SFSA), where the states determine the last unit (word or concept) that has been analyzed. Hence, for each concept, it is possible to build one SFSA that represents the conditional probabilities of a sequence of words given that concept, and another SFSA that represents how the concepts are chained.

For each concept c_i of the task, let A_{c_i} be the SFSA that represents the language model corresponding to c_i, and let A_C be the SFSA that provides the probability of a sequence of concepts. Let σ be a regular substitution such that $\forall c_i \; \sigma(c_i) = L(A_{c_i})$, where $L(A_{c_i})$ denotes the stochastic language represented by the automaton A_{c_i}. Let us define $A_{SLU} = \sigma(A_C)$. Since each state of A_C is determined by a concept c_i, it is equivalent to applying the substitution σ to each of the states of A_C by exploding each state into the corresponding SFSA A_{c_i}. Then, each state of A_{SLU} can be labelled with the concept c_i that it represents, thereby making A_{SLU} into a Moore machine since the concept is determined only by the current state.

The semantic analysis of a sequence of words W is performed via a Viterbi search on the automaton A_{SLU}. The best sequence of concepts $\hat{C}$ for this input is the one determined by the states of the best path according to this Viterbi search.

This approach is known as the 2-level approach (Sanchis et al. 2000; Segarra et al. 2002) since the automaton A_C may be seen as an *upper* level and each of the automata A_{c_i} can be seen as a member of a *lower* level. Moreover, the formulation of this approach allows each of the automata to be learnt using a different approach, as long as the automata are consistent with the finite substitution σ. For example, in (Pla et al. 2001), the authors propose a system where the models for each concept are based on HMMs that use information from part-of-speech tags. Also, the 2-level approach was eventually extended to 2+1-level (Bonneau-Maynard and Lefèvre 2005), where the extra level performs the task of extracting the value that is associated to a concept (e.g., the hour attached to a concept *time*) and normalizes it according to some criteria. Another implementation possibility for a 2+1-level system comes from the point of view of a multi-stage SLU approach, where the two lower models are of the generative type, and the higher level is either generative (Lefèvre 2006) or discriminative (Lefèvre 2007).

Another 2-level approach but in the fashion of a Mealy machine is the one presented in (Raymond et al. 2006). In this case, the different SFSMs that are composed are Stochastic

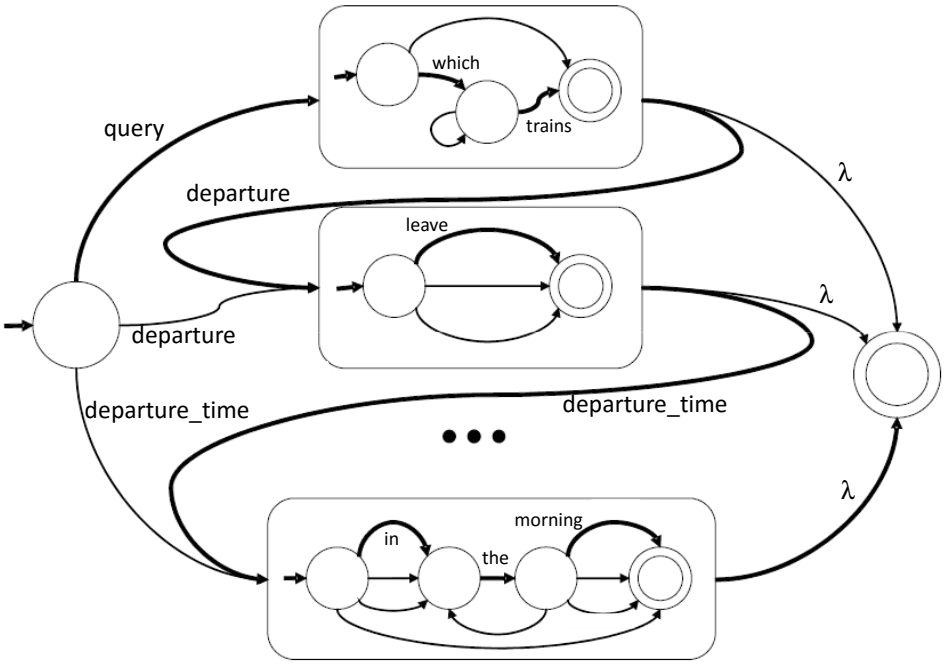

Figure 2.3: Example of a 2-level model for SLU.

Finite State Transducers (SFSTs). The final result is a SFST whose output alphabet are the concepts, even though some of its transitions may emit the empty string λ. In (Hahn et al. 2011), there is a nice formulation of a more general SFST-based approach, which also considers the ASR process as a transducer. Equation 2.8 shows the expression from which this approach is derived.

$$\hat{C} = \underset{C}{\operatorname{argmax}}\, p(C|A) = \underset{C}{\operatorname{argmax}} \max_{W} p(A|W) \cdot p(W|C) \cdot p(C) \tag{2.8}$$

Informally, this equation can be seen as a process that first translates an acoustic signal into words, then translates the words into concepts, and then combines the concepts. These translation processes can be modelled as an in-cascade composition of several SFST, as stated in Equation 2.9.

$$\lambda_{SLU} = \lambda_G[\circ\lambda_{gen}] \circ \lambda_{w2c} \circ \lambda_{SLM}[\circ\lambda_v] \tag{2.9}$$

The five SFSTs that are used in this equation to obtain λ_{SLU} have the following functions:

- The speech transcription process is represented by the λ_G transducer. From a SFSM point of view, a natural output of the ASR is a word lattice, which in this case can be seen as a SFSA that encodes the probability distribution $p(A|W)$ (i.e., the acoustic scores given each word in the lattice). This is a strength of this approach, since it allows to process lattices in a natural way, which is a feature that other SLU approaches do not have.

- The λ_{gen} SFST allows a lexical categorization of the words of the task. For example, it can convert the word *Paris* into a generic lexical category *city*, or the word *nine* into the category *number*. Hence, this transducer represents *a priori* lexical knowledge of the task. This transducer may not be statistical-based and it is optional.

- The phrases that represent concepts of the tasks are then translated into the corresponding concepts. This is performed via the λ_{w2c} transducer, which can be learnt either from training data or from human knowledge by means of a handwritten grammar. This transducer represents the probability distribution $p(W|C)$ and encodes a segmentation of the utterance in terms of concepts.

- A SFST that encodes a language model of sequences of concepts λ_{SLM} is used to compute the distribution $p(C)$. The transduction function for this transducer is the identity function, but the probabilities are important for computing the probability of the sequence of concepts.

- A fifth transducer λ_v can optionally be used to extract normalized attribute values from the sequences of words attached to the different concepts. Depending on whether or not this transducer is used or not, the transducer λ_{SLU} will have either the elements of a frame representation or the concepts of the task as its output alphabet.

Once the λ_{SLU} transducer has been obtained, a Viterbi search on it provides the best sequence of concepts $\hat{C}$ (or the best frame representation, depending on the output alphabet).

It is worth highlighting that the first transducer is input-dependent; in other words, it cannot be built before knowing the user's utterance. However, the distributive property of SFST composition can be used in this case, and all the other transducers can be composed

beforehand. Hence, when an utterance is recognized, the corresponding word lattice is composed with the pre-computed transducer, and then the Viterbi search can be performed.

An important drawback of this approach is that when the different transducers are fairly large, the size of the λ_{SLU} transducer increases, which means that the search space for the Viterbi algorithm also increases. This would lead to large space and time requirements, even if heuristics like beam-search are used to prune the search. Nevertheless, an advantage of this SFST-based approach is that it can be easily carried out using SFSM toolkits such as the AT&T FSM/GRM library (Mohri, Pereira, and Riley 2000; Allauzen, Mohri, and Roark 2005) or the OpenFst toolkit (Allauzen et al. 2007).

2.2.3 Linear chain conditional random fields

Linear chain conditional random fields (or simply Conditional Random Fields, CRF) are a discriminative formalism that has become very popular for SLU, mainly due to the good results that it achieves. In fact, in works in which several approaches are compared such as (Raymond and Riccardi 2007; Tur and De Mori 2011; Hahn et al. 2011), the best results are obtained using CRF.

CRF were introduced to the machine learning community by (Lafferty, McCallum, and Pereira 2001) as an extension of maximum-entropy hidden Markov models (McCallum, Freitag, and Pereira 2000). Actually, both of them are log-linear models and are based on a maximum entropy approach, and both have an underlying states-and-transitions topology. The main difference between them is the normalization factor Z that appears in their equations. Also, as in the previous state-based generative methods, each word in the input is attached to a concept label in the output. Hence, W and C are sequences of the same length, which will be denoted by K.

In a general way, log-linear models for a SLU context are described by Equation 2.10. They are expressed as the product of several functions that are succesively applied K times in order to find the probability of a sequence of concepts C of length K. Every time the functions are applied, they depend on the whole sequence of words W, the current concept

c_i, and the previous concept c_{i-1}. The final product is divided by a normalization factor Z which ensures that the log-linear model configures a probability distribution.

$$p(C|W) = \frac{1}{Z} \prod_{i=1}^{K} H(c_{i-1}, c_i, W) \tag{2.10}$$

Log-linear models for SLU are based on the combination of several feature functions $h(c_{i-1}, c_i, W)$ that take the current and the previous concepts and the whole sequence of words as their arguments. In this case, each feature function is a 0/1 function that describes whether or not one or more words of the input fulfil a certain condition, also taking into account the previous concept and the one to be predicted. Each function is weighted with a paremeter λ that says how important that function is in the prediction of the concept c_i. Let M be the amount of feature functions, and let $h_m(\cdot)$ be the m-th function of the set. Also let λ_m be the weight associated to h_m. Then, $H(c_{i-1}, c_i, W)$ is defined as:

$$H(c_{i-1}, c_i, W) = \exp(\sum_{m=1}^{M} \lambda_m \cdot h_m(c_{i-1}, c_i, W)) \tag{2.11}$$

However, in practice, instead of considering the whole sequence of words W as an argument for the feature functions, these feature fucntions are usually defined using a window of width Δ around the current word w_i. Thus, Equation 2.11 is re-written as:

$$H(c_{i-1}, c_i, W) = \exp(\sum_{m=1}^{M} \lambda_m \cdot h_m(c_{i-1}, c_i, W_{i-\Delta}^{i+\Delta})) \tag{2.12}$$

A crucial factor that, in fact, characterizes the different types of log-linear models is the normalization factor Z. For example, for CRF Z is a normalization factor that considers the full sequence of concepts, while in maximum-entropy HMM the normalization is performed at every single position. To define the normalization factor for CRF, let $\mathscr{C}^K$ be the set of all the sequences of concepts of length K. Then, Z is defined as the sum of the combination

of the different feature functions for each of the concepts that are contained in each the sequences in $\mathscr{C}^K$. Mathematically:

$$Z = \sum_{\tilde{C} \in \mathscr{C}^K} \prod_{i=1}^{K} H(\tilde{c}_{i-1}, \tilde{c}_i, W) \tag{2.13}$$

In the framework of log-linear models, CRF in this case, it is crucial to select a good set of features to achieve the best possible performance. In (Hahn et al. 2011), the feature functions are classified into the following categories: *lexical features*, which are about previous and following words; *concept bigram features*, also called *transition features* in (Dinarelli 2010), which capture the dependencies between pairs of consecutive concepts and which in the graphical model correspond to transitions between states that denote concepts; and *word part features*, which are based on sub-lexical units such as prefixes, suffixes, capital letters, etc. The feature functions can also use extra information apart from the input words, such as part-of-speech or syntactic tags, provided that they are also part of the input to the CRF. In (Hahn et al. 2011), the authors show that the set of feature functions and the width Δ of the analysis window that provide the best results vary depending on the corpus and the language being considered.

There are several publicly available toolkits such as CRF++ (Kudo 2005) that allow a CRF model to be trained and applied to SLU tasks. These toolkits are able to perform both the training stage, where the different weights of the model are estimated, and the test phase.

Modifications and optimizations on the standard model have also been proposed for SLU tasks. For example, in (Hahn et al. 2009), the authors propose both a mathematical modification of the original formulation in order to avoid numerical instability and a modification where ideas from large-margin methods (such as support vector machines) are brought into the CRF framework. Another variation of the model that is addressed to SLU tasks attempts to avoid the loss of information that is introduced into the model when an analysis window is used on the sequence of words, instead of being able to use the full sequence at any point in the analysis. To do this, (Jeong and Lee 2006; Jeong and Lee 2008) proposed using word triggers to model these long-term dependencies that fall outside the analysis window. The work in (Lehnen, Hahn, and Ney 2011) also follows this philosophy but with the aim of expanding the context of previous concepts that are taken into account. This

way, the graphical model underlying the CRF represents a Markov process of order higher than 1. One more variation that also modifies the linear chain CRF topology is obtained by taking into account that SLU systems are usually exploited within spoken dialogue systems. This way, if the information of the state of the dialogue manager is included in the SLU system, another formalism called triangular conditional random fields can be used (Jeong and Geunbae Lee 2008).

Both the original CRF model and these modifications are usually applied on a single transcription of the user's utterance since the whole sequence of words (or at least a window of a reasonable width) must be known in order to compute the feature functions. However, ASR errors may completely spoil the behaviour of the SLU module if only one transcription is used. Research on how to make use of CRF less sensitive to ASR errors, for example by working on more complex representations of the ASR output has recently been presented, with very encouraging results. An approach presented in (Deoras et al. 2013) is to enumerate the n-best hypotheses from the ASR (which can be obtained either directly or via a word lattice) and to perform the semantic decoding process on each one of the hypotheses separately. The final result is the one that maximizes the probability $p(C|A) = p(W|A) \cdot p(C|W)$, where W is each sequence of words contained in the n-best set, $p(W|A)$ is provided by the ASR, and $p(C|W)$ is obtained using the CRF model on the sequence of words W. However, it seems more reasonable to apply the understanding model directly on the word lattice, if possible.

Some recent research has been performed on how to apply CRF on graph representations, since it is not a straightforward matter. In (Deoras et al. 2012; Deoras et al. 2013), the authors propose a method for working with CRF on word lattices via the construction of an expanded lattice where the nodes represent the left and right context in which a word appears. This expanded lattice is much bigger than the original one, both in terms of number of nodes and arcs. This is due to the fact that, in usual word lattices, states do not store knowledge about the previous and future contexts, while, in this kind of expanded lattices, the nodes are defined using to this information. The left and right contexts that are represented in a node plus the current word define the analysis window that is used in the CRF feature functions. Consequently, the CRF model can be applied to the expanded lattice since the feature functions can be applied to every node locally, because all the information that is needed to evaluate the functions is contained in the node.

Word lattices can be converted into a more simple yet powerful representation: Word Confusion Networks (WCN). The main difference between the two types of graphs is the restrictions in their topology: word lattices can have any topology as long as there is at least one topological order in the graph; word confusion networks restrict the arcs to go from one node to the *following* one, without skips. Of course, every word confusion network is a word lattice, but not vice versa. There are algorithms such as the Pivot Algorithm (Hakkani-Tür and Riccardi 2003) that are able to convert word lattices into word confusion networks. In (Tur, Deoras, and Hakkani-Tur 2013), the authors take advantage of the topology of WCN to run a CRF model on them. They consider each set of arcs between two nodes to be a *bin*, and select the k words with the highest probability for each bin. The analysis window can then be expressed in terms of adjacent bins, and the feature functions are evaluated taking into account all the possible n-grams that appear within the window. In order to learn the feature functions, the authors also explore the possibility of considering all the neighbouring bins that result from recognizing the training utterances. This leads to an improvement over a system where the feature functions are learnt from single written sentences. This approach also has the advantage of not having to change the topology of the input graph of words. The sequence of concepts that is obtained this way has as many elements as there are bins in the WCN. However, it might be more difficult to recover the values associated to each concept since they must be searched for among all the possibilites encoded in the fragment of WCN that corresponds to the span of the concept.

2.3 Using multiple ASR outputs as the input to the SLU system

One important drawback of the SLU approaches that take just the 1-best ASR transcription as their input is that misrecognized words can dramatically spoil the result of the semantic decoding. This effect can be due to misrecognitions of named entities, which are usually very meaningful for SLU tasks. However, other words such as prepositions and numerals, which are necessary to identify relevant pieces of information within a sentence (e.g., the destination of a trip or the number of a hotel room), may also change the meaning of a sentence if they are not recognized properly. Furthermore, the Word Error Rate measure, which is widely used to assess the quality of a speech recognition system, does not take into account the semantic interest of a word. Errors made on semantically relevant words and semantically irrelevant words are counted equally. For this reason, it seems very convenient to convey multiple ASR outputs to the SLU system. This would make it possible to have a

better semantic interpretation for a sentence that is represented in this set but that is not the 1-best transcription. These multiple transcriptions are usually represented either as an n-best list of sentences or, more compactly, as a graph structure (a word lattice or a word confusion network). This interest in managing multiple transcriptions as the input to the SLU model has led researchers to develop extensions of well-known methods to deal with graph inputs, as described for the SFSMs and the CRF in the previous section. However, more approaches have been explored in this line. This section presents an overview of other methods that also follow this philosophy.

Some of the first works that used several ASR outputs in the SLU system were carried out by AT&T researchers whose goal was to classify calls in their customer-care service, *How may I help you?* (Gorin, Riccardi, and Wright 1997), by automatically analyzing the first utterance of the user. These works represented the multiple transcriptions as a graph of words (word lattice or word confusion network), which in many cases can be directly obtained from the ASR. In (Tur et al. 2002), the authors used the FST framework to perform this classification, which clearly outperformed a baseline that used only the 1-best transcription. This work was further extended in (Hakkani-Tür et al. 2006), where named entity detection that was based on handwritten and automatically learnt grammars was incorporated, and the call-type classifier was built on a combination of several simpler classifiers (boosting). More recent works have also addressed the problem of detecting named entities for a SLU module. For example, in (Svec, Ircing, and Smidl 2013), this problem is managed from the FST point of view by using factor automata.

In the Section 2.2.3, it was explained how the CRF formalism was extended to deal with uncertain data represented as graph inputs. However, instead of modifying a well-known approach, it is also possible to encode the information in the graph of words in such a way that this information can be used in a standard framework. This is the case of some approaches that have taken advantage of the information encoded in graphs of words provided by an ASR and have used it as the input to a SLU module based on Support Vector Machines (SVMs). One option in this case is to develop a kernel that is able to deal with word lattices and prepare the data to be used by a SVM. This was the idea developed in (Cortes, Haffner, and Mohri 2003) for call-type classification. However, another option is to build a feature vector from the information represented in the graph of words (e.g., n-grams, co-occurrences, etc.) and use it as the input to a SVM-based classifier. This idea, which is known as semantic tuple classifier (Mairesse et al. 2009), was first developed to

process single sentences. Its extension to graph structures is presented in (Henderson et al. 2012; Henderson 2015). Furthermore, works such as (Svec, Smidl, and Ircing 2013; Svec et al. 2015) have extended this idea by means of a multi-stage pipeline.

Another possibility for conveying multiple transcriptions to the SLU module is by means of an n-best list. This way, the SLU system can re-rank the elements in the list and provide the semantic interpretation of the most suitable sentence. This is done in (Bayer and Riccardi 2012), where the 100-best transcriptions from an ASR are fed to a SFST-based SLU module, which gives a semantic interpretation for each sentence in the set. A final re-scoring is then performed taking into account a joint language model of words and concepts. In (Khan et al. 2015), the authors give a different insight to n-best processing in the context of a multi-domain dialogue system. In their work, the n-best are processed separately by m SLU modules, where m is the number of domains of the dialogue system and the dialogue manager determines which is the most appropriate semantic interpretation for the user's utterance.

In summary, there are different possibilities when dealing with multiple outputs from an ASR as the input for a SLU module. If these multiple transcritpions are represented in a graph fashion, two options are available. The first one is to use a method that is able to naturally accept graphs as its input, either because the formalism is able to analyze them without any modification or because some extensions have been developed to deal with these structures. The second option is to convert the graph into another structure (e.g. a feature vector) that the chosen SLU approach can handle naturally. In contrast, when the input is presented using an n-best list, the most common option is to process all the transcriptions separately and have a module that decides which semantic decoding is the most suitable one for the user's utterance. In this thesis, a method that builds a graph structure from the elements of an n-best list will be described. This way, segments from different sentences can be combined in order to build a sentence that is more semantically appropriate than any of the individual transcriptions.

2.4 Multilingual spoken language understanding

Language portability of Natural Language Processing (NLP) systems, and especially those related to speech technologies, has been a very active research topic in recent years. This interest is motivated by the convenience of exploiting data or a system in one language to build a fully functional system in another language without having to develop it from scratch. For example, some NLP areas for which language portability has been explored are speech recognition (Schultz 2004; Stolcke et al. 2006; Lööf, Gollan, and Ney 2009), information retrieval (Nie 2010; Grefenstette 2012), question answering (Magnini et al. 2004; Neumann and Sacaleanu 2005), and plagiarism detection (Potthast et al. 2011; Barrón-Cedeño, Gupta, and Rosso 2013). Not only that, but also the very recent *Zero Resource Speech Challenge* presented at the INTERSPEECH 2015 conference aimed to start building speech processing systems from zero language-dependent resources, trying to mimic the learning process of human infants in their first years of life (Versteegh et al. 2015). If this technology is successfully developed, it would provide a homogeneous framework for developing speech processing systems from a certain amount of data, regardless of any language-dependent particularities.

The first efforts made to obtain multilingual SLU systems were performed in the 1990s. For example, the TINA SLU system was ported to several languages by developing grammars for each of these languages. These grammars were written by hand, which made the process human-dependent and error prone (Glass et al. 1995; Zue et al. 1996).

In the initial years of multilingual SLU, the two main schemes that have later been used for this task were outlined. In (Cettolo, Corazza, and De Mori 1998), the authors stated that there are two ways to port a SLU system to another language:

- to translate the training corpus from one language to another and then train a SLU system in this new language. Some years later this approach became known as of *train-on-target*.
- to automatically translate the test utterances into the SLU system's language and then process them using this system. This method later became known as *test-on-source*.

In 1998, the year in which that work was published, Machine Translation (MT) did not achieve the results that it offers today. This is why the authors thought that a human translator would be needed for train-on-target, or spoke about automatic partial translation for test-on-source. However, nowadays MT can be a very good tool for language portability of SLU systems, despite the unavoidable errors it makes (Suendermann et al. 2009). Actually, in the early 2000s, the interest in multilingual SLU decayed, because the necessary MT technology had not still been developed.

The modern implementations of these two approaches to multilingual SLU, which mainly rely on MT technology, are explained in detail below.

2.4.1 Train-on-target

The basic idea of train-on-target for porting a SLU system from an original language to a new one is to translate the training corpus and then train a SLU system with the translated corpus. However, a SLU corpus not only consists of sentences, but it also consists of semantic labels (concepts) that express the semantics of the sentence and usually segment it in terms of these concepts. Hence, when the corpus is translated into a new language, the concepts must also be conveniently assigned to the words that substantiate them. Table 2.2 shows an example of this problem. The column on the left shows a sentence in Spanish for a railway information system, the column on the right shows a possible translation of this sentence into English. It can be seen that the sequentiality of the concepts may not be the same in both languages due to word reordering issues. Also, it is often convenient not to do a word-by-word translation of the original sentence, but to change some verbs or structures that sound more natural in the other language. This is shown in this example where the Spanish word *hay* (there is/are) has been changed for the English verb *go*, which in addition is assigned to a different concept due to the change of the structure of the sentence. These are two of the issues that must be addressed when dealing with the train-on-target option for multilingual SLU.

In recent years, several approaches have been developed for translating the training corpus and its semantic labeling to another language. The main difference among them is the amount of human effort and supervision that is needed in the process. The most unsupervised approach consists of translating each of the semantic segments by means of several general-purpose translators and using this data to train a SLU system in the new language

Spanish (original)	English (translated)
Buenos días: courtesy	Good morning: courtesy
querría saber qué trenes: query	I'd like to know which trains: query
hay mañana: departure_date	go from Valencia: departure_city
de Valencia: departure_city	to Zaragoza: destination_city
a Zaragoza: destination_city	tomorrow: departure_date

Table 2.2: Example of the translation and re-segmentation of a training sentence.

(García et al. 2012). In this case, the portability of the segmentation is straightforward since each segment is translated separately. This strategy was used to translate the French MEDIA corpus into Spanish. The authors show that, despite the noise and the translation errors that these translators make, using several translators instead of just one increases the coverage of the translations and the performance of the multilingual SLU system.

The former approach provides a translation of the corpus at a very low cost, but it does not take into account any task-specific information. One way to take advantage of this information is to have an MT system trained with a parallel corpus that represents a domain that is close to the domain of the SLU system and to use this MT system to translate sentences from one language to the other (Misu et al. 2012). However, a parallel corpus that is related to the task is seldom available. Another option is that a small set of sentences from the training corpus of the SLU system is translated by humans and then use this parallel set to train a MT system (Jabaian, Besacier, and Lefèvre 2010; Lefèvre, Mairesse, and Young 2010; Jabaian, Besacier, and Lefèvre 2013). Several toolkits can be used for translation, but the most widely used one is the MOSES toolkit (Koehn et al. 2007). Then the rest of the training sentences that were not translated by humans can be translated by this system. This approach was used in (Jabaian, Besacier, and Lefèvre 2010; Jabaian, Besacier, and Lefèvre 2013) to train an SLU system for Italian from a small set of manually translated sentences from the French MEDIA corpus and it was also used in (Lefèvre, Mairesse, and Young 2010) to train an SLU system in French from an original corpus in English.

In order to port the segmentation from one language to another, several options are available. One option is to force MOSES to maintain the segmentations of the original sentences according to the semantic segments throughout the translation process. Another option is

to translate the original sentences without forcing any segmentation and then build an alignment between the original sentences and the translated sentences in order to project the assignment of words to concepts from one sentence to the other. Either of these methods provides a translated and segmented corpus for training a SLU system in the new language (Jabaian, Besacier, and Lefèvre 2010; Jabaian, Besacier, and Lefèvre 2013).

If it is possible to have human intervention during the translation process, then more alternatives are available. For example, in (Servan et al. 2010) a variation of the former option in presented. This is an iterative process that starts by splitting the training set in n blocks. The first block is translated and segmented by humans and is used to train an MT system. Then, the second block is automatically translated by this system. The portability of the semantic labels can be done using any of the above approaches. The automatically obtained translations are then checked and manually corrected by humans. These translations are added to the training set of the MT system, which is then re-estimated using the new training data. This procedure is performed until all the data is fully translated and checked. The authors show that a higher number of blocks (i.e., a greater amount of human work) leads to better performance of the SLU system.

By increasing the amount of human work, another possible choice is that the corpus is totally translated by humans, i.e. using a professional translation service. This method was used to translate the Italian LUNA corpus into Spanish, Greek, and Turkish (Stepanov, Riccardi, and Bayer 2014). In this work the authors also addressed the problem of translating the speech disfluencies from one language into another, which is a pragmatic aspect of the spoken language that is not usually taken into account when translating. Overall, this approach allows to obtain a high-quality parallel corpus that is valid for training a task-dependent machine translation system. Nevertheless, even though the authors do not provide information about the economic cost in their work, it is very likely that the professional translation effort involved a fairly high economic cost. Hence, the use of this approach allows high-quality task-dependent parallel corpus to be obtained, but at a certain monetary cost. In order to semantically label the LUNA corpus in the new languages, in (Chowdhury et al. 2014) the same authors propose a crowdsourcing strategy. In this strategy, several labelings are obtained for each sentence, which can be further combined to improve the overall quality of the semantic annotations (Chowdhury et al. 2015).

The most expensive and time-consuming option is to completely port the acquisition process of the corpus into the new language. In other words, to build a corpus in a new language using the same tools that were used to obtain the original corpus, by using native speakers in the acquisition process. This was the process that was used to obtain the Italian translation of the PORTMEDIA corpus (Lefèvre et al. 2012), which is based on the French MEDIA corpus. The original MEDIA corpus was acquired in French using a Wizard of Oz technique. To collect the Italian data, the scenarios and the system prompts were translated into Italian, so that the same procedure could be performed using native Italian speakers. It is clear that the aim of the authors was not to create a parallel corpus for machine translation, but to acquire brand new data using a strategy that had already worked for another language. Nonetheless, it is worth noting that from the point of view of building a parallel corpus for training a machine translation system this strategy does not seem very convenient, since when obtaining the data in the new language the speakers have the freedom to express their goal in the words they wish, without any knowledge of the original corpus. Thus, the sentence uttered by the user does not have to be a translation of the original sentence. However, by proceeding this way, the system can be completely ported to another language with the same guarantees and features than the original language, which is the actual goal of this strategy.

2.4.2 Test-on-source

The test-on-source approach to multilingual SLU consists of translating the user's utterance into a language for which a SLU system has already been trained, and then to perform the semantic decoding of the translation using this SLU system. Therefore, the machine translation process comes between the speech transcription module (ASR) and the SLU module.

Similarly to the train-on-target case, a wide range of approaches can be found depending on the amount of the human effort that is required in order to obtain the MT system. The strategy that requires the least human effort is to use a pre-existing translator to bring the user's utterances into the system's language. For example, in (He et al. 2013) the Microsoft Bing translator is used for this purpose. This approach provides translations of the user's utterances at a very small economic and time cost. Nevertheless, even though there is a task-dependent corpus available in the system's language, it does not exploit any in-domain data during the MT process.

One way to make use of task-dependent data during the MT process while keeping the human effort low is to employ a pre-existing MT system (e.g., a general-purpose web translator) to obtain a translation of the training corpus in the user's language and then to train a MT system (e.g., MOSES) from the user's language to the system's language using this automatically collected parallel corpus. The quality of the translations obtained this way is not as high as if the corpus were translated by professional human translators, particularly because of the multiple errors that the MT systems make. These errors provide a lower quality parallel corpus, which would lead to multiple errors when translating the user's utterances. Several MT systems can be used to mitigate these errors, providing two main advantages. First, if t translators are used, then the size of the resulting parallel corpus is t times the size of the original one. This increase in the size of the corpus is convenient for MT purposes, since MT systems often need a large amount of data to be conveniently trained. Second, since there are t possible translations in the corpus for each original sentence, a statistical machine translation system can learn which structures appear more often, even in different contexts, and provide good translations despite the noisy training corpus. This approach was used to translate the training part of the Spanish DIHANA corpus into French and English (García et al. 2014).

Some human intervention is needed to avoid completely relying on general-purpose MT systems. The same approaches that were used to translate the training corpus for a train-on-target approach can be also used for a test-on-source approach. One example is to use a small number of translations obtained by humans to bootstrap a MT system that is used to translate the rest of the training corpus (Jabaian, Besacier, and Lefèvre 2010; Lefèvre, Mairesse, and Young 2010; Jabaian, Besacier, and Lefèvre 2013). Another example is to have the whole corpus translated by a professional translation service (Stepanov et al. 2013). Of course, in both cases, once the parallel training corpus has been obtained, it is necessary to train an MT system that translates the corpus from the user's language to the system's language.

Regardless of whether an automatic approach or human effort is used to obtain a task-dependent parallel corpus, this parallel corpus is usually made up of only a few thousand sentences. However, many more sentences are often needed to be able to train a high-performance MT system. One possible option is to combine the task-dependent dependent corpus with an out-of-domain corpus. However, it would be desirable for this new data to be close to the domain of the SLU system. A solution in this direction is provided in

(Stepanov et al. 2013). It consists of using a search engine to retrieve documents that are close to the domain of the task, and then the sentences of these documents that are the most representative for the domain are selected. These close-to-domain sentences can then be added to the task-dependent corpus to enrich it, without taking most of the data from outside of the domain.

In this test-on-source setting, a problem that also appears in a monolingual system that combines an ASR and a SLU module in-cascade is accentuated. This problem is the mismatch between the errors that can appear at the input of the SLU system (regardless of whether it comes from an ASR or an MT system) and the data that was used to train the SLU system, which is usually orthographically and syntactically correct. This means that recognition and translation errors can dramatically spoil the performance of the SLU system when the input is noisy with respect to the training data.

Two main solutions have been presented for this problem. One solution consists of an automatic post-editing of the output of the MT system to correct the errors that may exist (Jabaian, Besacier, and Lefèvre 2011; Jabaian, Besacier, and Lefèvre 2013; Stepanov et al. 2013). The other solution is to re-train the SLU system using the original training corpus plus some sentences in which translation errors have been added (Jabaian, Besacier, and Lefèvre 2011; Jabaian, Besacier, and Lefèvre 2013; He et al. 2013). The goal is for these translation errors to match the translation errors that the test sentences have and then to provide a good semantic decoding even for these noisy inputs. One simple way to obtain sentences with translation errors is to translate the training sentences into another language and then translate them back into the original language. Then, the semantic labeling of these new sentences can be performed using any of the techniques that are used for labeling a corpus in the train-on-target approach.

Another option that has been explored in the last few years is to integrate the MT and SLU processes more closely (Jabaian, Lefèvre, and Besacier 2014), instead of having them as two modules in a pipeline that are connected by the transmission of a single sentence. This is also one of the goals of the SLU method that has been developed for this thesis.

Finally, it is important to highlight that some works that have performed a comparison between train-on-target and test-on-source methods suggest that, in practice, the test-on-source approach achieves better results (Jabaian, Besacier, and Lefèvre 2010; Lefèvre, Mairesse, and Young 2010; Jabaian, Besacier, and Lefèvre 2013).

Chapter 3

Description of the corpora

In this chapter a description of the corpora that has been used for the empirical assesment of the experiments reported in this thesis is provided. First, details on the acquisition and semantic labelling of the Spanish DIHANA corpus are presented. This corpus has been used for most of the monolingual SLU experiments. Then, the extension of this corpus to a multilingual environment is explained. This extension constitutes the multilingual DIHANA corpus, and all the multilingual SLU experiments in this thesis have been performed using this corpus. The multilingual DIHANA corpus was developed and acquired in the context of this thesis (García et al. 2014). Finally, some details on the French MEDIA corpus are provided. The MEDIA corpus was only used for one monolingual experiment, in order to compare the SLU approach developed in this thesis with other state-of-the-art SLU methods using a widely spread corpus.

3.1 The Spanish DIHANA corpus

One of the goals of the Spanish DIHANA project was to research new methodologies for the development of spoken dialogue systems. The chosen task to carry out this study was an information system for nationwide Spanish trains. This information system should be able to handle spoken dialogues in order to provide information about train schedules and prices by means of natural language. Also, this information system may be accessed by a telephone call.

For the purposes of this research project, the DIHANA corpus (Benedí et al. 2006) was collected. During the acquisition of the corpus, it was important that it could accurately model the task from all the lexical, syntactic, and semantic points of view. In other words, it should reflect the way in which a *naïve* user would interact with the SDS. Hence, these two points should stand:

- From the lexical and syntactic points of view, no restriction is applied. The user is expected to interact with the system using spontaneous speech, and so, grammatical errors, on-line corrections of the sentence, pauses, doubts, etc. may appear.
- From the semantic point of view, it has a clearly restricted domain, which is the topic of the SDS.

In order to adequately collect this corpus, a *Wizard of Oz* methodology was used (Fraser and Gilbert 1991; Griol 2007). In this methodology, a person simulates the behaviour of the Dialogue Manager, and receives all the information that is being processed (even the untranscribed acoustic signal). During the acquisition, the Wizard must follow a pre-stablished strategy, and the user must not know that a person is taking the place of the automatic Dialogue Manager.

With the aim of fulfilling the semantic restrictions, several scenarios were defined. This way, the users should try to achieve the goal specified in their scenario by holding a dialogue with the SDS (secretly managed by the Wizard), and providing it the needed information to achieve the goal. Each scenario was described by a goal, it is, the information that the user must obtain from the system; a motivational situation (for example, the user has a meeting in some city and wants to arrive there before the beginning of the meeting); and some restrictions and preferences (e.g. desire of a non-smokers carriage, arrival time restrictions, and similar). A total of 300 scenarios were defined. The three types of scenarios that were defined are:

A) The goal is to obtain the schedules of one-way train trips. For this purpose, the destination and the date of the trip must be specified. The origin, a time interval for the trip and the type of train are optional. Eventually, the user can also ask for the type of the train.

B) The goal is to obtain the price and, optionally, the schedules for one-way train trips. The destination and the date of the trip are mandatory, while the origin, a time interval for the trip and the type of train are optional.

C) These scenarios are analogous to type B) but for round trips.

These scenarios were presented to a total of 225 *naïve* users, and their interactions with the system (user turns) were used to build the corpus. This acquisition was performed in three different cities – Bilbao, Valencia, and Zaragoza – and the resulting dataset was composed of 900 dialogues uttered by 153 males and 72 females. The recordings have a total length of 10.8 hours of speech, and the total number of user turns amounts to 6,278. All the acquisition was performed using telephone communication between the user and the Wizard. Thus, the different recordings have the acoustic conditions of this channel.

The recordings of the user turns were carefully transcribed into text, reproducing as well spontaneous speech phenomena such as disfluencies. These transcriptions were semantically labelled using a set of 30 concepts. These concepts can be divided into two categories: task-dependent concepts, such as *origin_city* and *departure_time*; and task-independent concepts, like *query*, *negation* and *null*. Furthermore, the set of task-dependent concepts are divided in two subsets: the concepts that express that a value has been specified by the user, for example, in the sentence *I want to leave at two o'clock* the concept *departure_time* can be identified; and those that ask the system for a value for that concept, for example, in the sentence *What time does the earliest train depart at?* it is clear that the object of the query is a departure time. In this case, the concept is enclosed by angle brackets (e.g. <*departure_time*>). The semantic labeling of a sentence using a sequence of concepts constitutes a segmentation of the sentence in terms of the sequence of concepts. Table 3.1 shows some figures about the semantically labelled corpus.

Using this semantic segmentation, a frame representation is also defined for each sentence of the corpus. This frame representation aims at summarizing all the relevant information contained in the sentence by means of the use of pairs of attributes and values. Furthermore, if the user is asking for the value of any of the attributes, it is represented between round brackets. In order to standarize the content of a frame, a canonical representation is obtained by sorting its attributes following a pre-stablished order, and the numerical values that appear in the sentence are instantiated. However, sometimes a point of reference is

Number of user turns	6,229
Total number of words	47,222
Vocabulary size	811
Number of concepts	30
Average number of words per user turn	7.6
Total number of semantic segments	18,588
Average number of words per semantic segment	2.5
Average number of segments per user turn	3.0
Average number of samples per concept	599.6

Table 3.1: Characteristics of the semantically labelled Spanish DIHANA corpus.

needed in order to convert a value from a lexicalized form to a numerical representation. For example, the specific date corresponding to the word *tomorrow* will depend of the current date. For this purpose, some reference values are also defined. Table 3.2 shows an example of both the semantic segmentation of a sentence and its frame representation.

Sentence	*Quiero obtener el horario para el viaje de ida de Vitoria a Madrid el catorce de mayo para llegar antes de mediodía* (I want to know the schedule for a one-way trip from Vitoria to Madrid on May 14th arriving before noon)
Semantic segmentation	*quiero obtener*: query *el horario*: <time> *para el viaje de ida*: type_of_trip *de Vitoria*: origin_city *a Madrid*: destination_city *el catorce de mayo*: date *para llegar*: arrival *antes del mediodía*: time
Frame representation	TYPE-OF-TRIP = one-way (TIME) ORIGIN-CITY = Vitoria DESTINATION-CITY = Madrid ARRIVAL-TIME <= 12:00 DATE = 14/05/2015

Table 3.2: Example of the different types of semantic labellings that are contained in the DIHANA corpus.

3.2 The multilingual DIHANA corpus

The multilingual DIHANA corpus (García et al. 2014) was acquired with the goal of enabling the research on multilingual spoken language understanding grounded on the Spanish DIHANA corpus. As it was already discussed in Section 2.4, the creation of a multilingual SLU corpus is a great effort in which several important aspects must be taken into account, such as the amount of human effort that is going to be used, its economic cost, and the desired quality of the corpus, both in terms of the quality of the sentences that constitute the corpus and the closeness to the task domain of these sentences (i.e. they can range from completely out-of-domain sentences to translations of the original sentences of the corpus). Actually, these points are correlated, since a higher human intervention usually leads to a better quality of the corpus, but at a greater economic cost. However, an important factor for current language portability of Natural Language Processing systems is that it can be effectively achieved using a rapid and low-cost approach. This philosophy was adopted for the construction of the multilingual DIHANA corpus.

As it has already been discussed, the use of current machine translation technology can be very helpful to obtain multilingual SLU corpora. Nevertheless, the problems that arise when working with a test-on-source strategy or with a train-on-target approach are not the same. The selection of a train-on-target methodology implies that, for most SLU methods, the semantic segmentation of the original sentences must be ported to the new language in order to train a new SLU system. On the contrary, for test-on-source, any translation system can be used in order to translate the user's utterance into the system's language. One option is to build a task-dependent translator using sentences from the original corpus, but their translation must be previously obtained in order to build a parallel corpus. The multilingual DIHANA corpus fulfills this requirement, since it is a semi-automatically collected corpus that enables the development of a task-dependent translator for test-on-source multilingual SLU.

In order to build this corpus, the Spanish DIHANA corpus was first split in two sets: $4,887$ sentences for the *training* set, and 1340 for the *development and test* set. This distinction is important because each of these two sets is processed independently and in a different manner.

Spanish	*Quisiera horarios para el próximo sábado a Barcelona por la mañana*
French	*Serait planifier pour le samedi matin à Barcelona* *J'aimerais planifier pour samedi prochain à Barcelona le matin* *Il voulût horaires pour le prochain samedi à Barcelona par le matin* *Il voudrait des horaires pour samedi prochain à Barcelona le matin*
English	*I would like schedule for Saturday morning to Barcelona* *I would like to schedule for next Saturday to Barcelona in the morning* *It wanted schedules for next Saturday to Barcelona in the morning* *Would want schedules for next Saturday to Barcelona in the morning*

Table 3.3: Example of the translations that were collected in English and French for a single training sentence in Spanish using the four selected general-purpose web translators.

The translation of the training set was performed using several general-purpose web translators, namely Bing Translator[1], Google Translator[2], Lucy[3], and Opentrad[4]. This way, four translations are obtained for each training sentence. The advantages of using several translators are twofold. One one side, the coverage on the vocabulary in the user's language is increased. On the other, errors of single translators on keywords that are strongly related to the meaning of the whole sentence are mitigated. This is important because if these errors were transmited to the final translation, the performance of the SLU system would be severely damaged.

Table 3.3 shows an example extracted from the corpus of the different translations that the four general-purpose web translators provided in English and French for a original training sentence in Spanish. This example shows that each individual translation may have many errors corresponding to words and structures that a native speaker would not say, or that are simply translation errors. However, the combination of all the four sentences in a translation model could help to obtain translations of a reasonable quality, despite the errors they have, since their structures are mixed and generalized. Unfortunately, the existence of these translation errors makes this tool not effective enough in order to model a native user that speaks to the SLU system in a real multilingual environment.

[1] `www.bing.com/translator`
[2] `translate.google.com`
[3] `www.lucysoftware.com`
[4] `www.opentrad.com`

The goal that is pursued in the translation of the development and test set is that it can adequately model a situation in which a person that speaks English or French natively interacts in a natural way with the system in which the multilingual SLU module is integrated. Thus, in order to reflect this situation, this set of sentences was manually translated into English and French by native speakers. A subset of these sentences was also uttered by native speakers. The development and test set in French is composed of $1,277$ sentences, 500 of which were uttered by 4 native speakers. In the English case, this set is composed of $1,336$ sentences, and all of them were uttered by a total of 6 native speakers. The human translators had in some cases information of the context of the dialogue in which the sentences appeared. This way, they could incorporate some contextual information in the translations if they thought that it was more natural for a native speaker of that language to say it in that way. This is shown in the French translation shown in Table 3.4, where the phrase *il arrive trop tard* explicitly refers to the arrival time of a train (the pronoun *il* refers to the word train), while in the other languages the user just says "that's too late" without an explicit mention to the fact that it is an arrival time of a train.

Spanish	*Es demasiado tarde, querría uno que llegara un poco más pronto*
French	*Il arrive trop tard, je voudrais en prendre un qui arrive un peu plus tôt*
English	*That's too late, I wanted one that arrives a bit earlier*

Table 3.4: Example of human translations of a test sentence into English and French.

3.3 The French MEDIA corpus

The French MEDIA/EVALDA project (Devillers et al. 2004) had as one of its goals the development of an evaluation protocol which allowed to assess the performance of SLU in the context of SDSs (Bonneau-Maynard et al. 2006). For this purpose, the French MEDIA corpus (Bonneau-Maynard et al. 2005) was created. The selected task for the corpus was a tourist information system that allowed the user to book hotel rooms by keeping a spoken dialogue by telephone with the system.

For the acquisition of this corpus, a Wizard of Oz system was used in order to simulate the dialogue manager. In order to acquire a varied but in-domain dataset, several scenarios were generated from a set of templates, obtaining a range of eight levels of difficulty. Furthermore, the Wizard could be less or more cooperative with the user at his discretion. At the end

of the acquisition, 1,257 dialogues were collected using 250 speakers, amounting to a total length of around 70 hours of speech.

The text transcriptions of the recordings were semantically annotated using a hyerarchical semantic representation which is based on (Bonneau-Maynard and Rosset 2003) and constitutes an ontology for the domain of the task. For the purpose of the experimentation in this thesis, 73 concepts from this ontology were used, plus a *null* concept that was added in order to identify all the unlabelled sequences of words. Table 3.5 shows more features of the semantically annotated corpus. In addition, Table 3.6 shows an example of semantic labelling of a sentence.

Number of user turns	16,279
Total number of words	114,969
Vocabulary size	2,357
Number of concepts	74
Average number of words per user turn	7.1
Total number of semantic segments	53,942
Average number of words per semantic segment	2.1
Average number of segments per user turn	3.3
Average number of samples per concept	709.8

Table 3.5: Characteristics of the semantically labelled French MEDIA corpus.

Sentence	*je souhaiterais réserver pour la deuxième semaine de janvier à Paris à côté de l'Arc de Triomphe* (I would like to book for the second week of January next to the Arc de Triomphe)
Semantic segmentation	*je souhaiterais réserver*: command-tache *pour la deuxième*: rang-temps *semaine*: temps-unite *de janvier*: temps-mois *à Paris*: localisation-ville *à côté de*: localisation-distanceRelative *l'Arc de Triomphe*: localisation-lieuRelatif

Table 3.6: Example of the semantic labelling of a sentence of the MEDIA corpus.

Chapter 4

A graph-based approach to spoken language understanding

In this chapter, a graph representation of the SLU problem and a graph-based approach to SLU are presented. Previous works that show development of this strategy are (Calvo et al. 2012a; Calvo et al. 2012c; Calvo et al. 2013a; García et al. 2015b; Calvo et al. 2015).

This chapter begins with an explanation of the statistical ground of this graph-based SLU method, which is derived from the generative approach to the SLU problem. A strong point of this method is that it can naturally handle a graph representation of the combination of multiple recognition hypotheses generated by one or more ASRs. The SLU process is divided into several modules. These modules are introduced and analyzed along the chapter, and at the same time the graph representation of the SLU problem is described. Finally, some experimental results will be presented and discussed.

4.1 System overview

One of the approaches to SLU consists on finding the sequence of concepts $\hat{C}$ that best fits the semantics contained in an utterance A. Considering a stochastic SLU system, it can be stated as:

$$\hat{C} = \underset{C}{\operatorname{argmax}}\, p(C|A) \tag{4.1}$$

where C stands for any sequence of concepts made of elements of the set of concepts $\mathscr{C}$ defined for the task.

If we also take into account the sequence of words W - unknown beforehand - which underlies A, Equation 4.1 can be re-written as follows:

$$\hat{C} = \underset{C}{\operatorname{argmax}} \max_{W} p(W, C|A) \tag{4.2}$$

Or, equivalently, applying the Bayes' Rule:

$$\hat{C} = \underset{C}{\operatorname{argmax}} \max_{W} \frac{p(A|W, C) \cdot p(W, C)}{p(A)} \tag{4.3}$$

Finally, it can be assumed that the utterance A and the sequence of concepts C are independent variables. Also, the denominator can be removed, since it is not part of the maximization. Hence, it can be equivalently expressed as:

$$\hat{C} = \underset{C}{\operatorname{argmax}} \max_{W} p(A|W) \cdot p(W|C) \cdot p(C) \tag{4.4}$$

Many SLU systems approximate this equation by decoupling the ASR and SLU processes. This way, they take as the input to the SLU module the best transcription generated by the ASR, and they take it for granted all along the semantic decoding process, which converts Equation 4.4 into:

$$\hat{C} = \underset{C}{\operatorname{argmax}}\, p(W|C) \cdot p(C) = \underset{C}{\operatorname{argmax}}\, p(W, C) \tag{4.5}$$

where W stands here for the sequence of words provided by the ASR.

However, feeding the SLU module with just the 1-best hypothesis from the ASR implies two substantial limitations. First, it is extremely hard for the forthcoming modules (if not impossible) to correct the errors made in the recognition stage. Second, there is many information generated by the ASR, such as n-best lists, lattices, acoustic posteriors, etc. that is no longer considered. Despite most of the state-of-the-art approaches to SLU are designed to take a single sentence as their input (which can be seen in the formulae that support each of the approaches), there has been a wide interest in the last years in trying to overcome these limitations derived from using just one hypothesis. Some works like (Hakkani-Tür et al. 2006; Henderson et al. 2012; Tur, Deoras, and Hakkani-Tur 2013; Svec, Ircing, and Smidl 2013) aimed at exploiting the information contained in the ASR lattices. Another option is to convert the ASR n-best list into a single sentence, by merging all the hypotheses into one by means of a voting algorithm (Fiscus 1997). Then, this sentence can be processed by any of the available SLU methods.

The idea underlying the approach developed here is to overcome the two limitations stated above by taking multiple transcriptions provided by one or more ASRs, and keeping as much information as possible in each of the steps of the process. This will be achieved by building graphs, which, in each case, will represent different linguistic information. An advantage of using graphs is that they provide an efficient and compact way to convey multiple hypotheses among the different modules without any loss of information. Even more, extra information may be inferred when building the graphs, if a grammatical inference algorithm is used for this purpose. However, the most widely used state-of-the-art SLU methods do not accept graphs as their input in a native way. Hence, a very important feature of this SLU system is that it is completely based on algorithms on graphs. Figure 4.1 shows a scheme of the decoupled modular architecture which implements this approach, which is based on Equation 4.4.

The most important modules will be explained in detail in the following sections, but in order to give the reader a general overview, we will provide now a brief description of them.

1. A first module dedicated to Automatic Speech Recognition, which will be used as a black box. As we would like to take advantage from multiple hypotheses generated by this module, its output will be a set of sentences provided either by a single ASR

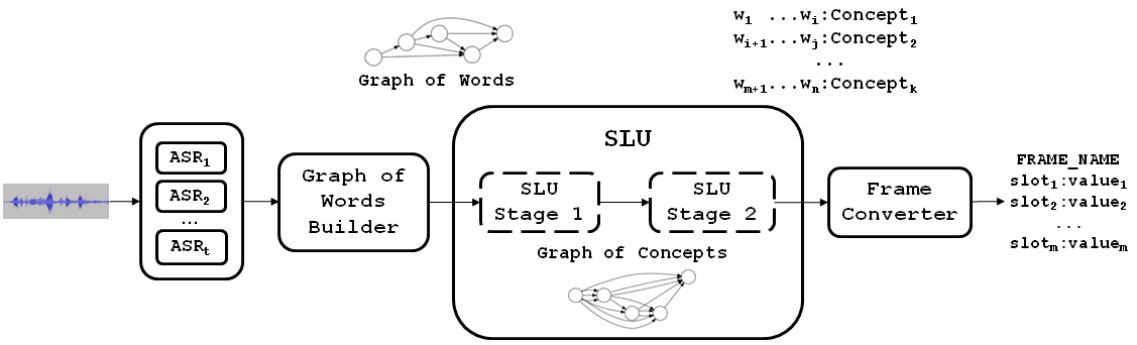

Figure 4.1: Scheme of the architecture of the complete SLU system.

(n-best list) or by several ASRs working in parallel (set of 1-best or n-best decodings). It is also possible that the output of the ASR module is a word lattice. In this case, as a lattice is a graph of words itself, the next step can be skipped.

2. The graph of words builder module takes a set of sentences provided by the ASR module and outputs a graph. This graph represents a set of sentences that not only contains the input ones, but also a generalization of them according to their structures (roughly, the new sentences in the set are made of pieces of the original sentences). This way, the coverage increases, as the SLU module is provided with new inputs that were not in the original set. Also, a graph is a more concise and efficient representation than an enumeration of hypotheses. This graph also provides a re-estimation of the probability distribution $p(A|W)$, but restricted to the sequences of words that are represented in the graph. It is also worth noting that a word lattice generated by an ASR is a graph of words itself. Hence, another option is to take a word lattice from the ASR module and skip this step. In both cases, in this system the words will be represented on the arcs of the graph, along with a score showing the contribution of that arc to $p(A|W)$.

3. Then, the semantic decoding is carried out by means of a SLU module that is able to deal with graphs of words. For this purpose, a semantic decoding methodology that computes the remaining terms of Equation 4.4 in two steps has been developed. This methodology is entirely based on algorithms on graph structures. In particular, all of them are Dynamic Programming algorithms, as this strategy allows to efficiently optimize a function in a large search space. The first stage takes as input a graph

of words and a set of graph structures which model the lexical realizations of each concept. It makes a joint analysis of them and builds a graph of concepts. A graph of concepts represents matchings between sequences of words and concepts, and this information is represented on its arcs. Then, the graph of concepts is analyzed in a second stage, which also takes a graph representation of the possible concatenations of concepts in the scope of the task. The output of this module is the best sequence of concepts, as well as the underlying sequence of words and its segmentation in terms of the concepts. Thus, this module not only provides a semantic representation of the user's utterance, but also a transcription of what the user said, taking into account lexical, syntactic and semantic knowledge.

4. Finally, the semantically relevant information is extracted, normalized, and converted into a frame representation. This is performed by a combination of both rules and stochastic models.

4.2 The graph of words builder

One of the key ideas of the SLU system presented here is that the different modules of the system share as much information as possible, in order to avoid critical errors made by one of the modules that can not be amended afterwards. For example, it is very important not to make recognition mistakes which could spoil the semantic decoding process. However, it is also convenient that the SLU system is not ASR-dependent, that is, that one can use any ASR as the front-end for the SLU system. Then, the ASR engines should be treated as "black boxes". In consequence, there are two different options in order to keep an opportune amount of information for the following modules:

- The ASR supplies a word lattice generated by the search algorithm. This lattice is usually weighted using the probabilities $p(A_{t_1}^{t_2}|w)$ for each word w that appears in the lattice, where t_1, t_2 are the start and end times estimated for that appearance of w, and $A_{t_1}^{t_2}$ is the fragment of audio that spans from t_1 to t_2. Hence, for each sequence of words W that can be built from a path on the lattice, the combination of the different weights associated to the words of W provides the probability $p(A|W)$.

- The output of the ASR module is a set of sentences, which may be a n-best list from one or more ASRs.

Inspired by the idea that underlies word lattices, if the output of the ASR module is a set of sentences, it can also be converted into a graph representation. One possible way to do this is to build an automaton that accepts the language constituted by that set of sentences, which can be efficiently performed using, for example, the algorithm presented in (Daciuk et al. 2000). However, it may also be convenient to have a closer look to the set of sentences, and analyze them in order to extract new information according to their structure. This can be performed via a grammatical inference algorithm.

4.2.1 Grammatical inference

According to (Fu and Booth 1975), grammatical inference, is the "problem of learning a grammar based on a set of sample sentences". In other words, the aim of grammatical inference is to identify the language $L(G)$ from which a set $S \subseteq L(G)$ of sentences (or strings) were drawn, by building a grammar G that generates $L(G)$. However, this can only be achieved when all the information about $L(G)$ is known, which implies that $S = L(G)$. As it is generally not the case, the learning process will find a grammar G' such that $S \subseteq L(G')$. This learning is usually performed by considering and generalizing the structural features that appear in the samples and using them to build G'. Given a set S, the obtained grammar G' will depend on the construction algorithm that is used for building the grammar.

However, it would be useful for many practical problems that the inferred grammar also provides the probability of generating each of the strings of the language it represents. These grammars that provide probabilities along with the structure associated to the language are called *Stochastic Grammars*. In the case of automata, they are called *Stochastic Finite State Automata* (SFSA). If they are consistent, both of them represent a probability distribution over the strings of the language, since for every inferred grammar G' Equation 4.6 must stand.

$$\sum_{x \in L(G')} p(x) = 1 \tag{4.6}$$

Let $D(x)$ be the set of leftmost derivations of x using G'. If the grammar is unambiguous, or the automaton is deterministic, the cardinality of $D(x)$ is always 1, since there is a

unique way of deriving each string that belongs to the language. Let d be the derivation that provides x, and let $R(d)$ be the set of rules used in d. Then, the probability of x to be generated by G', $p(x)$, is defined as:

$$p(x) = \prod_{r \in R(d)} p(r) \tag{4.7}$$

Furthermore, stochastic grammars and automata can also be obtained via grammatical inference. The techniques to estimate the probabilities attached to an inferred stochastic grammar or automaton are based on the frequency counts of the structual features of the elements in S.

Provided the usefulness of inferring grammars or automata from samples in the fields of syntatic and structural pattern recognition and computational linguistics (among others), many algorithms have been proposed in order to address this subject, depending also on the features of the samples and on the specific task in which it is going to be used. For example, (De La Higuera 2005; D'Ulizia, Ferri, and Grifoni 2011; Stevenson and Cordy 2014) present a nice review of grammatical inference algorithms and techniques, and also discuss their practical applications. In this thesis, grammatical inference will be used for several purposes. In the following lines, one of them will be described: the generalization of multiple outputs provided by the ASR module in order to be analyzed by the SLU system developed in this work.

4.2.2 Building a graph of words from several ASR hypotheses

A graph of words GW is a graph structure in which lexical and syntactic information are represented. One of the possible ways to represent the lexical information is to attach words to the arcs of the graph, as well as weights which represent the relevance of each word. The structural information that is provided by the arcs represent syntactic information on how the words can follow each other, i.e., a path in the graph represents a possible concatenation of words. If the graph has an initial node I and a final node F, all the paths that start from I and reach F shape the set of sentences L that are represented in the graph. The combination of the weights of the arcs used while building a sentence give a score that represents the relevance of that sentence in L. This definition of a graph of words makes it

equivalent to a weighted finite state automaton, where the language it represents is $L(GW)$. Moreover, if the weights of the graph represent probabilities, their combination is performed by means of a multiplication, and the sum of the weights for all the sentences in $L(GW)$ is 1 (i.e. it represents a probability distribution, see Equation 4.6), then a graph of words is equivalent to a SFSA. In this work, a graph of words is built from multiple ASR outputs in such a way that it has SFSA properties. This is performed by means of a grammatical inference algorithm.

The algorithm that is used in this thesis for building a graph of words from multiple ASR hypotheses is based on the idea of searching for the segments of words that represent an agreement among all the ASR hypotheses, and produce an alignment matrix according to these agreements. Then, the graph of words is constructed from the information contained in this matrix. This idea was first presented in (Bangalore, Bordel, and Riccardi 2001), where the authors applied it for finding a consensus between several translations provided by a machine translation system.

In order to analyze the set of transcriptions of the utterance, to look for their similarities, and to build a matrix with this information, a Multiple Sequence Alignment (MSA) algorithm is used. The Multiple Sequence Alignment problem is a generalization to K sentences ($K > 2$) of the well-known pairwise Edit Distance problem, which is particularly useful for the alignment of sequences that represent biological information. Similarly to the Edit Distance problem, the MSA problem can be solved exactly via Dynamic Programming. However, it has a computational complexity of $\mathcal{O}(N^K \cdot 2^K)$, where N represents the length of the sentences to be aligned, and K is the number of sentences. Actually, it was proved in (Wang and Jiang 1994) that this problem is NP-Hard. Due to this high complexity, several heuristics have been proposed. One of them is implemented by the ClustalW (Larkin et al. 2007) Multiple Sequence Alignment software.

This software was initially conceived for aligning biosequences. Hence, it was necessary to convert the sequences of words into symbols that represent elements in a biological sequence. In addition to these symbols, a special symbol is also used for Multiple Sequence Alignment. This symbol is usually represented as '-' and denotes a *gap* in the alignment. In other words, it represents an insertion of an "empty symbol", meaning that no word in this sentence was aligned with any word of any other sentence for this position (non-alignment

point). Regarding weights, all substitutions of different symbols and gap insertions have the same cost.

The result of the Multiple Sequence Alignment process is an alignment matrix. Each row in this matrix represents a different sentence, and each column represents the alignment of each symbol. The total number of columns is usually greater than the length of the longest sequence due to the existence of gaps. Figure 4.2 shows an example of an alignment matrix built from several ASR outputs.

Multiple ASR Outputs:
me puede decir horarios de trenes Alicante
puede decir horas de trenes Alicante
me puede decir hola trenes a Alicante

MSA Matrix:

	me	puede	decir	horarios	de	trenes	–	Alicante
	–	puede	decir	horas	de	trenes	–	Alicante
	me	puede	decir	hola	–	trenes	a	Alicante

Figure 4.2: Alignment of several ASR outputs.

Once the MSA alignment matrix is built, the information contained in it is used to construct a weighted directed acyclic graph of words. In a graph of words the nodes denote *temporal* information, in the sense that they inform about which words can follow other words; and the arcs provide information about the words themselves and their weights. Let us define a graph of words $GW = (V_{GW}, A_{GW})$ as a tuple, where V_{GW} is the set of vertices or nodes of the graph and A_{GW} is the set of directed arcs in the graph. Each arc can be represented as the 4-tuple (*src, dest, word, weight*), where each element represents respectively the source node of the arc, the destination node of the arc, the word attached to the arc, and its weight. For notation purposes, if $a \in A_{WG}$, then src(a), dest(a), word(a) and wgt(a) will provide access to the corresponding attributes of the arc.

The graph construction algorithm starts creating as many nodes as columns in the alignment matrix, plus one for the starting node. Both the nodes and the columns are numbered correlatively: the nodes are numbered starting in 0, where node 0 is the initial node, and the columns are numbered starting in 1. This way, a correspondence between the nodes of the graph and the columns of the matrix is established, except for the initial node of the graph. Then, for each cell (k, j) in the matrix that contains an aligned word w, the information it contains is translated into the graph representation. Let i be the previous cell of the same row not containing a gap ('-'), or 0 if this cell does not exist. If there was

not already an arc from i to j labelled with w in the graph of words, then it is built, and a counter initialized to 1 is attached to it. However, if it was already created, its counter is incremented by 1. The node that represents the last column of the matrix (i.e. the one with the highest index) is the final node of the graph.

When all the cells representing an aligned word have been processed, all the counters of the arcs are normalized in a range between 0 and 1. These weights are computed in the following way. For each node i in the graph of words, let $A_{GW}(i)$ be the set of arcs that depart from i, and let $S(i)$ be the sum of the counters of the elements in $A_{GW}(i)$. Then, the final weight for each arc in $A_{GW}(i)$ is computed by dividing its counter by $S(i)$. This procedure guarantees that $\sum_{a \in A_{GW}(i)} \text{wgt}(a) = 1$. This fact makes the weights represent probabilities, which are estimated according to a Maximum Likelihood criterion. The whole method to build a graph of words from a MSA matrix is shown in Algorithm 1. Figure 4.3 shows an example of the full process that creates a graph of words from multiple ASR outputs.

Both the time and space complexity of Algorithm 1 are $\mathcal{O}(K \cdot N)$, where K and N are the dimensions of the MSA matrix (number of sentences provided by the ASR module and number of columns of the matrix, respectively).

Every path from the initial node to the final node in a graph of words which is built using Algorithm 1 represents a recognition alternative for the input utterance. The probability of a sentence contained in a graph is the product of the probabilities attached to the arcs in the path for that sentence. This way of defining the probability of a sentence, and the fact that for each node the sum of the probabilities of the arcs that depart from it is 1, imply that the graph of words yields a probability distribution on the different sentences that are contained in the graph. In other words, if $L(GW)$ represents the set of sentences represented in a graph of words GW, then it stands that $\sum_{x \in L(GW)} p(x) = 1$.

Another property of the graphs built using Algorithm 1 is that taking their nodes in ascending index order represents a topological sorting of these nodes. It stands because all the arcs in the graph span from a node of lower index (which represents an earlier column of the MSA matrix) to a node of higher index (which represents a later column).

Algorithm 1 Method for building a graph of words from a MSA matrix.

Input: M = MSA matrix $[1..K][1..N]$
Output: Graph of words $GW = (V_{GW}, A_{GW})$

1: V_{GW} = sequence of $N+1$ nodes (labelled from 0 to N)
2: $A_{GW} = \emptyset$
3: $degrees[0..N-1] = \{0, 0, \ldots, 0\}$ // Output degree for each node
4: **for** $k = 1$ **to** K **do**
5: $i = 0$ // Last processed word
6: **for** $j = 1$ **to** N **do**
7: **if** $M[k][j]$ has a word w **then**
8: a = arc from i to j labelled with w
9: **if** $a \in A_{GW}$ **then**
10: counter(a) = counter(a) + 1
11: **else**
12: add a to A_{GW}
13: counter(a) = 1
14: **end if**
15: $degrees[i] = degrees[i] + 1$
16: $i = j$
17: **end if**
18: **end for**
19: // We must ensure that do not have several final states.
20: // If so, we add a λ-transition with the probability of that state of being final.
21: **if** $i \neq N$ **then**
22: a = arc from i to j labelled with λ
23: **if** $a \in A_{GW}$ **then**
24: counter(a) = counter(a) + 1
25: **else**
26: add a to A_{GW}
27: counter(a) = 1
28: **end if**
29: $degrees[i] = degrees[i] + 1$
30: **end if**
31: **end for**
32: // Convert counters into probabilities
33: **for** $i = 0$ **to** $N-1$ **do**
34: **for all** arc $a \in A_{GW}$ departing from node i **do**
35: wgt(a) = counter(a) / $degrees[i]$
36: **end for**
37: **end for**
38: **return** $GW = (V_{GW}, A_{GW})$

Moreover, this method for building a graph of words from multiple ASR hypotheses can be seen as grammatical inference process. Let GW be a graph of words that was built using Algorithm 1, and let $L(GW)$ be the set of sentences represented in GW. If there is some disagreement among the different ASR outputs that are used to build the graph (i.e. they are not all the same), then L will not only contain these sentences, but it will also contain new sentences formed by pieces of them. Also, these new sentences present similar structural features to the original sentences. This happens because the MSA algorithm searches for segments that are common to all the original transcriptions, and aligns them by placing the common segments in the same columns, and locating the rest of the words in the other cells of the alignment matrix, making use of gaps if necessary. Then, Algorithm 1 combines the multiple transcriptions by keeping the certainty in the zones where there is an agreement among them, but providing several options where there are disagreements. Please note that if the output of the ASR module is a single sentence or all the ASR outputs are equal, then the graph built this way would only contain one path and all the probabilities in the arcs would be 1.0.

Figure 4.3 shows an example of the whole process for building a graph of words from multiple ASR hypotheses. As shown, this graph represents not only the input sentences, but also a set of sentences with similar structual features. For example, the correct sentence *me puede decir horarios de trenes a Alicante* (could you tell me train timetables to Alicante) was not among the transcriptions provided, but it can be built using the information that is in the graph (red path on the graph). It can be also seen that this is not the path of maximum probability on the graph, since if the blue arc labelled with the word "Alicante" was taken to go from node 6 to node 8, the probability of this path would be higher than the probability of the path that contains the sequence of words that constitutes the best recognition for the utterance. However, this information will be combined in the next module with semantic knowledge, in order to find the best semantic interpretation of the content of the graph.

As an additional element, λ-transitions are added between every pair of adjacent nodes, with a probability very close to zero. These transitions will allow to skip unknown words when they are analyzed by the semantic decoding algorithm.

Bringing all together, this way of building a graph of words from multiple ASR outputs provides a re-estimation of the probability distribution $p(A|W)$ by restricting the set of possible sentences to a generalization of the individual transcriptions provided by the ASR

Correct utterance: *me puede decir horarios de trenes a Alicante*
(could you tell me train timetables to Alicante)

Multiple ASR Outputs:
me puede decir horarios de trenes Alicante
puede decir horas de trenes Alicante
me puede decir hola trenes a Alicante

MSA Matrix:

1	2	3	4	5	6	7	8
me	puede	decir	horarios	de	trenes	–	Alicante
–	puede	decir	horas	de	trenes	–	Alicante
me	puede	decir	hola	–	trenes	a	Alicante

Graph of words:

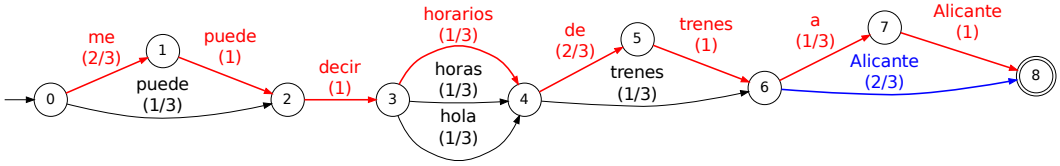

Figure 4.3: Method to build a graph of words from multiple ASR outputs.

module, and weights them according to a Maximum Likelihood criterion. This re-estimation of $p(A|W)$ is represented in an efficient way by means of a graph of words which is equivalent to a SFSA. Moreover, regarding the structure of the graphs of words that are built using this method, they can be considered word lattices. This implies that the same mechanisms that have been developed for analyzing these graphs also work if their input are word lattices supplied by an ASR.

4.3 The semantic decoding algorithm

The approach to SLU that is described in this thesis is based on finding the sequence of concepts $\hat{C}$ that maximizes Equation 4.4, which is reproduced here for clarity. In this equation, W stands for any sequence of words (perhaps within a restricted vocabulary) and C stands for any sequence of concepts built from the set of concepts of the task.

$$\hat{C} = \underset{C}{\operatorname{argmax}} \max_{W} p(A|W) \cdot p(W|C) \cdot p(C)$$

In this system, the probabilities $p(A|W)$ can be provided either in the form of a word lattice generated by the ASR search algorithm, or as a graph of words built from several ASR outputs. Since both of them are graph structures, it is necessary to build a semantic decoding algorithm that is able to deal with graphs of words[1]. In this case, a generative approach is used.

The graph-based semantic decoding algorithm works in two stages. First, it builds a graph of concepts from a graph of words. The method for building the graph of concepts is based on seeking those paths in the graph of words which underlying segment of words represent any of the concepts of the task. The arcs of the graph of concepts are built according to the segments found and the concepts they represent. In order to find the segments of words that represent concepts, statistical semantic models that provide the probabilities $p(W|c)$ of the different lexical realizations W that can express each of the concepts of the task c are trained. These semantic models are learnt from semantically segmented and labelled data.

The second stage finds the best path in the graph of concepts. For this purpose a statistical semantic model is used to represent the probability $p(C)$ of each possible sequence C of concepts of the task. This model is also learnt from a semantically segmented and labelled training corpus.

4.3.1 Stage 1: Building a graph of concepts

The first stage of the semantic decoding process builds a graph of concepts from a graph of words. In a graph of concepts, its arcs represent the different segments of words that correspond to paths in the graph of words, and can be assigned to any of the concepts of the task. In other words, the sequences of words that are attached to the arcs represent lexicalizations of the concepts of the task they are associated to. The arcs in the graph of concepts are weighted with a combination of the score of the corresponding path in the graph of words and the probability of that segment of words according to the concept.

Let $GC = (V_{GC}, A_{GC})$ be a graph of concepts that is built from a graph of words $GW = (V_{GW}, A_{GW})$, where in each case V represents the set of vertices or nodes of the graph and A the set of arcs. According to the definition of a graph of concepts, that is based

[1] In the rest of this chapter both the word lattices generated by an ASR and the graphs built using the algorithm explained in Section 4.2.2 will be referred to as graphs of words.

on scanning the graph of words looking for sequences of words that are relevant to any of the concepts of the task, the semantically relevant segments can span between any pair of nodes $i, j \in V_{GW} : i < j$. Hence, the nodes of the graph of concepts must be the same as the nodes of the graph of words, it is, $V_{GC} = V_{GW}$.

Regarding arcs, let us define an arc of the graph of concepts as the 5-tuple (*src, dest, words, conc, weight*), where each element represents respectively the source node of the arc, the destination node, a sequence of words, the concept represented by the sequence of words, and the weight associated to the arc. Let also $M = \{M(c_1) \dots M(c_u) \dots M(c_{|\mathscr{C}|})\}$ be a set of semantic models such that, for each concept $c_u \in \mathscr{C}$, $M(c_u)$ provides the probability of a sequence of words W given the concept c_u, it is, $p(W|c_u)$. Then, for each concept c_u, a path P_i^j in the graph of words WG that spans from node i to node j and induces the sequence of words W_i^j is a candidate to generate an arc from i to j in the graph of concepts GC if $p(W_i^j|c_u) \neq 0$, according to $M(c_u)$. Given a candidate, its weight wgt is defined as a weighted combination of $p(W_i^j|c_u)$ and the weight of P_i^j. It is possible that there are several candidates that, while their sequences of words are different, they represent the same concept and start and finish at the same pair of nodes. Since the second step of the semantic decoding method will search the best path (maximum weight path) in the graph of concepts taking also into account how the concepts are chained, it is not necessary to keep all the "candidates" that represent the same concept between the same pair of nodes, but only the one of maximum weight. Hence, the set of arcs of a graph of concepts A_{GC} is defined as the set of tuples (i, j, W_i^j, c_u, wgt) induced by paths P_i^j in the graph of words such there is no other path Q_i^j such for the same concept c_u would provide a candidate with a higher weight than wgt. In other words, given a pair of nodes i, j and a concept c_u the algorithm that builds the graph of words seeks the sequence of words that spans from i to j in the graph of words and provides the maximum combined score of the score of the path and the probability of the semantic model corresponding to concept c_u, and converts this information into an arc.

As it has been stated, the set of semantic models M that provide the probability of a sequence of words W given a concept $p(W|c_u) \forall c_u \in \mathscr{C}$ plays a very important role in the process of building a graph of concepts. Although these models can be trained in a variety of ways, the one that has been used in this work is training a n-gram language model for each concept of the task. In order to train these models, it is needed that the training corpus is segmented and labelled in terms of concepts.

Let the training corpus T be a set of sentences, where each sentence $Z = w_1 \dots w_n$ is represented as a sequence of m pairs (s_i, c_u), where:

- each s_i is a segment of words made of consecutive words of W,
- s_1 starts with w_1,
- s_m finishes with w_n,
- $\forall w_k \in Z \; \exists s_i : z \in s_i$,
- $\forall i, j \in [1, m] \; s_i \cap s_j = \emptyset$,
- and $c_u \in \mathscr{C}$.

Let us call this sequence of pairs a *segmentation* of the sentence Z in terms of the set of concepts of the task $\mathscr{C}$.

Then, it is possible to collect for each $c_u \in \mathscr{C}$ all the segments $s_i \in T$ that are attached to c_u and build a n-gram language model using just these segments as the training samples. Thus, these language models provide the probability $p(W|c_u)$ of a sequence of words W given a concept, as they are trained using only positive samples for that concept. These language models are part of the semantic models that will be used to compute the semantics of the input utterance, and they will be referred to as *concept-dependent semantic models*. These models can be represented as Stochastic Finite State Automata. Figure 4.4 shows an overview of the knowledge sources and the data structures that constitute the inputs and the output for this first SLU stage.

With all these definitions, the problem of building a graph of concepts can be stated as the problem of searching, for each pair of nodes of the graph of words and each concept-dependent semantic model, the sequence of words that maximizes their combined probability. Algorithm 2 shows a method that exhaustively checks the feasibility of all the possible subpaths in the graph of words given the set of concept-dependent semantic models. This is not the algorithm that is ultimately used for building the graph of concepts, but the objective of including it here is twofold: not only it is very useful in order to better clarify the goal that is being pursued, but it also constitutes the bottom line from which the definitive Dynamic Programming algorithm will be built.

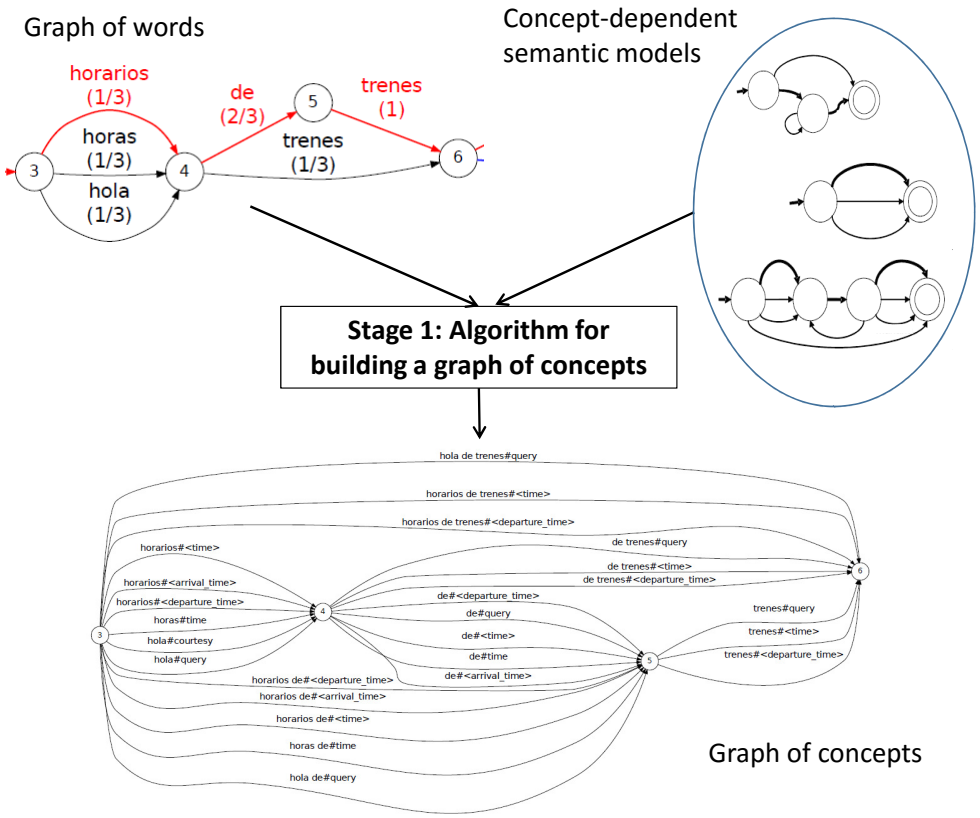

Figure 4.4: Overview of the Stage 1 of the graph-based semantic decoding algorithm.

In Algorithm 2 two heuristic factors that are usual in speech applications (such as speech recogintion) are introduced. These factors are tuned empirically, and they are:

- a *grammar scale factor* α that aims to compensate the magnitude differences between the graph of words and the concept-dependent semantic models,
- and a *word insertion constant* β that is added for each word that is appended to a segment.

Algorithm 2 starts creating a graph of concepts with as many nodes as the graph of words has, and with the same numeric labels, and an empty set of arcs. Then, for every possible path P_i^j in the graph of words and each concept c_u the combination of the probabilities provided both by the graph of words and the semantic model corresponding to the concept

Algorithm 2 Unoptimized method for building a graph of concepts from a graph of words and a set of concept-dependent semantic models.

Input: Graph of words $GW = (V_{GW}, A_{GW})$,
set of semantic models $M = \{M(c_1) \dots M(c_{|\mathscr{C}|})\}$,
grammar scale factor α,
word insertion constant β
Output: Graph of concepts $GC = (V_{GC}, A_{GC})$

$V_{GC} = V_{GW}$
$A_{GC} = \emptyset$
for $i = 0$ **to** $|V_{GC}| - 1$ **do**
  for $j = i + 1$ **to** $|V_{GC}|$ **do**
    for all path P_i^j that spans from node i to node j in GW **do**
      $wgtGW = \prod_{a \in P_i^j} \text{wgt}(a)$
      W_i^j = sequence of words underlying P_i^j
      for $u = 1$ **to** $|\mathscr{C}|$ **do**
        $probLM = p(W_i^j | c_u)$ calculated using $M(c_u)$
        if $probLM \neq 0$ **then**
          $wgt = probLM^{\alpha} \cdot wgtGW \cdot \beta^{j-i}$
          // The symbol $*$ represents a wild card that matches any value
          if $\exists e = (i, j, *, c_u, *) : e \in A_{GC}$ **then**
            if $wgt > \text{wgt}(e)$ **then**
              $A_{GC} = A_{GC} - \{e\}$
              $newarc = (i, j, W_i^j, c_u, wgt)$
              $A_{GC} = A_{GC} \cup \{newarc\}$
            end if
          else
            $newarc = (i, j, W_i^j, c_u, wgt)$
            $A_{GC} = A_{GC} \cup \{newarc\}$
          end if
        end if
      end for
    end for
  end for
end for
return $GC = (V_{GC}, A_{GC})$

c_u is computed, taking into account the factors α and β. If there was no arc in the graph of concepts from node i to node j labelled with concept c_u, then the information in P_i^j is converted into an arc in the graph of concepts, which is also labelled with c_u and weighted

with the computed combined probability. However, if there was already an arc from node i to node j labelled with concept c_u, only the hypothesis with the maximum probability is kept. Please note that the notation of the loops in lines 3 and 4 works because the sequence of node labels $\{0, 1, \ldots, |V_{GW}| - 1\}$ provides a topological sorting of the nodes of the graph of words.

The time complexity of Algorithm 2 is $\mathcal{O}(K^{|V_{GW}|} \cdot |V_{GW}|^3 \cdot |\mathscr{C}|)$, where K is the number of hypotheses the ASR module provided, and therefore the maximum output degree a node of the graph of words can have. The exponential factor comes from the fact that every possible path in the graph of words has to be computed, in order to see whether it will lead to an arc in the graph of concepts or not. It can be assumed that the operations on the graph of concepts take $\mathcal{O}(1)$, if efficient data structures are used to represent the graph.

The space complexity of Algorithm 2 is $\mathcal{O}(|V_{GW}|^2 \cdot |\mathscr{C}|)$, because it is an upper bound of the number of arcs that the graph of concepts can have (one for each concept and each pair of nodes in topological order).

Despite Algorithm 2 is a good way to understand how to build a graph of concepts, it is not useful in practice due to its excessive time complexity. However, a closer look to the algorithm reveals that many calculations are computed multiple times, and the use of Dynamic Programming would speed up the algorithm. For example, for every path from i to j of length $l > 1$, all the calculations corresponding to preffixes of length $l-1$ were already performed, and could be used to work out the values when adding a new element. On one side, given a preffix x of length $l-1$, the result of adding the arc a to this preffix is $\text{wgt}(x) \cdot \text{wgt}(a)$. On the other side, the concept-dependent semantic models will be represented as SFSA. Hence, let M_{c_u} be the SFSA representing the concept-dependent semantic model for concept c_u, and let $Q(M_{c_u})$ be the set of states of this SFSA. Let $q \in Q(M_{c_u})$ be the state in which the analysis of the string corresponding to x ends, and $p(x|M_{c_u})$ the probability that M_{c_u} gives to x. Then, the probability of the concept-dependent semantic model when the word attached to the arc a is appended to x is $p(x|M_{c_u}) \cdot p(\text{word}(a)|q)$, where the latter probability is the probability attached to the transition in M_{c_u} that departs from q and is labelled with word(a). This incremental procedure leads to the Dynamic Programming (DP) algorithm that is stated in Algorithm 3.

Algorithm 3 has three parts. The first one creates as many nodes as the graph of words has, and assigns them the same labels. The second part is described by Equation 4.8. In this

Algorithm 3 Dynamic Programming method for building a graph of concepts from a graph of words and a set of concept-dependent semantic models.

Input: Graph of words $GW = (V_{GW}, A_{GW})$,
set of concept-dependent semantic models $M = \{M(c_1) \dots M(c_{|\mathscr{C}|})\}$,
grammar scale factor α,
word insertion constant β

Output: Graph of concepts $GC = (V_{GC}, A_{GC})$

1: $V_{GC} = V_{GW}$
2: **for** $u = 1$ **to** $|\mathscr{C}|$ **do**
3: // T is the DP matrix. Each node of the graphs and each state of M_{c_u} are mapped to an integer in order to index the matrix.
4: $T[0..|V_{GC}| - 1][0..|V_{GC}| - 1][0..|Q(M_{c_u})| - 1] = \{0, 0, ..., 0\}$
5:

$$T(i,j,q) = \begin{cases} 1 & \text{if } i = j \wedge q = q_0(M_{c_u}) \\ 0 & \text{if } i = j \wedge q \neq q_0(M_{c_u}) \\ 0 & \text{if } j < i \\ \max\limits_{\substack{\forall a \in A_{GW} : \text{dest}(a) = j \\ \forall q' \in Q(M_{c_u})}} T(i, \text{src}(a), q') \cdot p(q|q', \text{word}(a))^{\alpha} \cdot \text{wgt}(a) \cdot \beta & \\ & \text{otherwise} \end{cases} \tag{4.8}$$

6:

$$\begin{aligned} &\forall i, j \in V_{GC} : i < j, \\ &(i, j, W_i^j, c_u, x) \in A_{GC} \text{ if } x = \max_{q \in M_{c_u}} T(i,j,q) \\ &\text{where } W_i^j \text{ is the sequence of words attached to this path} \end{aligned} \tag{4.9}$$

7: **end for**
8: **return** $GC = (V_{GC}, A_{GC})$

equation $T(i, j, q)$ represents the cell of the DP matrix that stores the best score that can be achieved by a sequence of words which starts at the node i of the graph of words, ends at the node j of the same graph, and finishes its analysis by the concept-dependent semantic model M_{c_u} at the state q. Also $p(q|q', \text{word}(a))$ represents the probability of moving from state q' to state q using a transition labelled with word(a), and $q_0(M_{c_u})$ is the initial state of the SFSA corresponding to M_{c_u}.

The base cases of this DP equation are pretty straightforward. First, each 0-length path will have a score of 1 if it is paired with the initial state of the concept-dependent semantic

model, and 0 otherwise. Also, the final node of the path must appear later than the initial node; thus, if it is not the case, the assigned score will be 0.

In order to fill the matrix for the general case, the is and js are considered in ascending order, with $i < j$. Furthermore, the maximization is performed, for each pair of nodes, at the language model state level. This way, the optimum is always kept, along with other hypotheses that arrive to other states of the language model which could overtake the maximum until this step when analyzing further words. Moreover, the case of λ-transitions in the graph of words are handled as any other transition, but no score from the concept-dependent model is considered since no word is added to the current segment.

According to the definition of the graph of concepts, for every pair of nodes $i, j \in V_{GC} : i < j$, and for each concept $c_u \in \mathscr{C}$, only the sequence of words W_i^j represented by a path in the graph of words from i to j that yields the maximum combined score will lead to an arc in the graph of concepts. Using the information provided by Equation 4.8, it is easy to find the hypothesis that has the maximum score by iterating over the different states q (keeping fixed i and j), and then build the new arc using this information. This procedure is stated in Equation 4.9, which is the third part of the algorithm. This equation implies that the sequence of words that provides the maximum score for each i, j, q is known when building the arcs for the graph of concepts. Therefore, the values in the DP matrix of Equation 4.8 must be enriched in order to store this information.

It is also worth noting that it is not necessary to completely fill the DP matrix to apply Equation 4.9. On the contrary, it is sufficient to have it filled for all the cells corresponding to a pair of nodes (i, j) to be able to run Equation 4.9 for this pair, which it is sure to be true when Equation 4.8 starts to fill the cells where the origin node is j. Therefore, Equations 4.8 and 4.9 can be interleaved, fixing each time the pair of nodes i, j that are to be processed.

Figure 4.5 shows an example of how a graph of concepts is built. In this example, for concept c_1, the hypotheses that depart from node i of the graphs of words go to different states on the automaton M_{c_1}. Let us suppose that the combined probability correspoding to w_2 is greater than the one for w_1. This makes that an arc in the graph of concepts labelled with the pair (w_2, c_1) is built. However, when w_3 is considered, both segments collapse in the same state q of M_{c_1}, and a maximization is performed. In this case, the hypothesis corresponding to the sequence $w_1 w_3$ is given a higher probability, and is the one

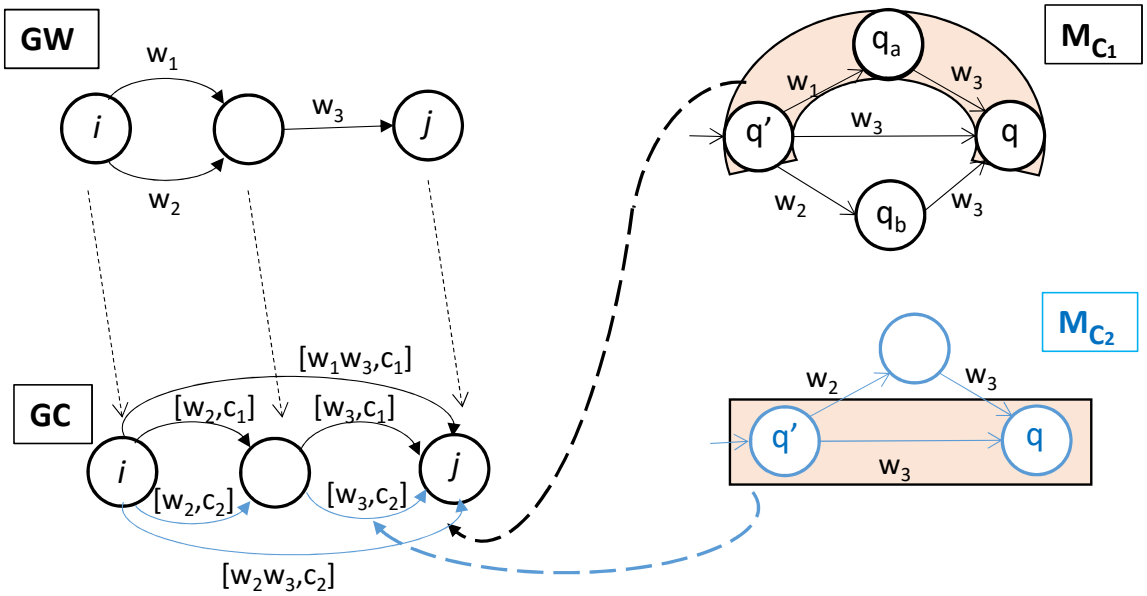

Figure 4.5: Example of the construction of a graph of concepts where a small graph of words and two concepts are considered.

that is converted into an arc from node i to node j in the graph of concepts. Also, it can be seen that the same word can be attached to many concepts (w_2 is associated to both c_1 and c_2), but for every pair of nodes there must not exist two arcs labelled with the same concept.

The space complexity of Algorithm 3 is $\mathcal{O}(|V_{GW}|^3 \cdot \max_{c_u \in \mathscr{C}} |Q(M_{c_u})|)$. The data structure that leads to this cost is the Dynamic Programming matrix, which has a size of $|V_{GW}| \times |V_{GW}| \times \max_{c_u \in \mathscr{C}} |Q(M_{c_u})|$ cells. For each cell, the algorithm must keep track of the words that led to that score, in case they are later needed to build the corresponding arc for the graph of concepts. In the worst case, the number of words that could be associated with a cell is $|V_{GW}| - 1$.

However, this matrix can be stored in a more efficient way without worsening its temporal cost using the following guidelines:

1. For a given graph of words and a concept-dependent semantic model M_{c_u}, the most frequent case is that, for a given pair of nodes i, j, the analysis of all the subpaths

from i to j end at a small set S of different states of M_{c_u}. Actually, the upper bound for the cardinality of S is $\min(K^{\min(n-1,j-i)}, |Q(M_{c_u})|)$, where K is the number of sentences the ASR module provided (and thus the maximum output degree of a node in the graph of words), and n is the maximum length of the n-grams is M_{c_u}. In practice, it often stands that $S << |Q(M_{c_u})|$. This makes the DP matrix very sparse. Hence, storing only the elements of the DP matrix that are not null is a good idea.

2. Each time a new starting node i is being considered, all the information in the cells (i', j, q) with $i' < i$ can be deleted, as they will not be used for the calculations of the hypotheses starting at i.

In consequence, if bigrams are used for the concept-dependent language models (which will be the case in most of the experiments in this thesis), the space complexity can be more tightly expressed as $\mathcal{O}(|V_{GW}|^3 \cdot K)$. Furthermore, the lengths of the sequences of words that represent concepts are in practice much lower than the number of nodes of the graph. Hence, in practice the space cost is $\mathcal{O}(|V_{GW}|^2 \cdot \bar{X} \cdot K)$, where $\bar{X}$ is the average length of a sequence of words representing a concept.

The time complexity of the algorithm built by interleaving Equations 4.8 and 4.9 is $\mathcal{O}(|V_{GW}|^2 \cdot \max_{c_u \in \mathscr{C}} |Q(M_{c_u})|^2 \cdot K \cdot |\mathscr{C}|)$. A $\mathcal{O}(|V_{GW}|^2 \cdot \max_{c \in \mathscr{C}} |Q(M_{c_u})|)$ factor comes from the loops that iterate for each cell of the 3-dimensional matrix. Then, for each cell, there is a maximum with two arguments. The first argument iterates for each arc that arrives to a given node j. Because of the way the graph of words was built, the input degree of a node of this graph can be at most K, so this part has a cost of $\mathcal{O}(K)$. The second argument is the set of states of the language model. Thus, the cost that this argument provides to the full process is $\mathcal{O}(\max_{c \in \mathscr{C}} |Q(M_{c_u})|)$. This process is repeated for each concept, hence a multiplying factor of $\mathcal{O}(|\mathscr{C}|)$ must also be added. The algorithm of Equation 4.9 does not contribute to the total cost, since when interleaved with Equation 4.8 it would just contribute with an addend that is smaller than the rest of factors of the cost.

Nevertheless, if the improvements on space mentioned before are applied, then the time complexity decreases to $\mathcal{O}(|V_{GW}|^2 \cdot K^2 \cdot \max_{c \in \mathscr{C}} |Q(M_{c_u})| \cdot |\mathscr{C}|)$. Furthermore, a small change in the algorithm can be performed in order to improve the temporal efficiency. This modification is related to the maximization that iterates for every possible predecessor in

the concept-dependent language model. Instead of doing the dynamic programming in a backwards way, by looking the possible predecessors of each hypothesis, it is possible to do it in a forward manner, by expanding the possible children that each hypothesis can have. Then, for a given word, it is not necessary to search all the possible destinations, as there is only one possibility. Thus, if an appropiate data structure is used for this step, its cost becomes $\mathcal{O}(1)$. Then, the time complexity for this algorithm becomes $\mathcal{O}(|V_{GW}|^2 \cdot K^2 \cdot |\mathscr{C}|)$.

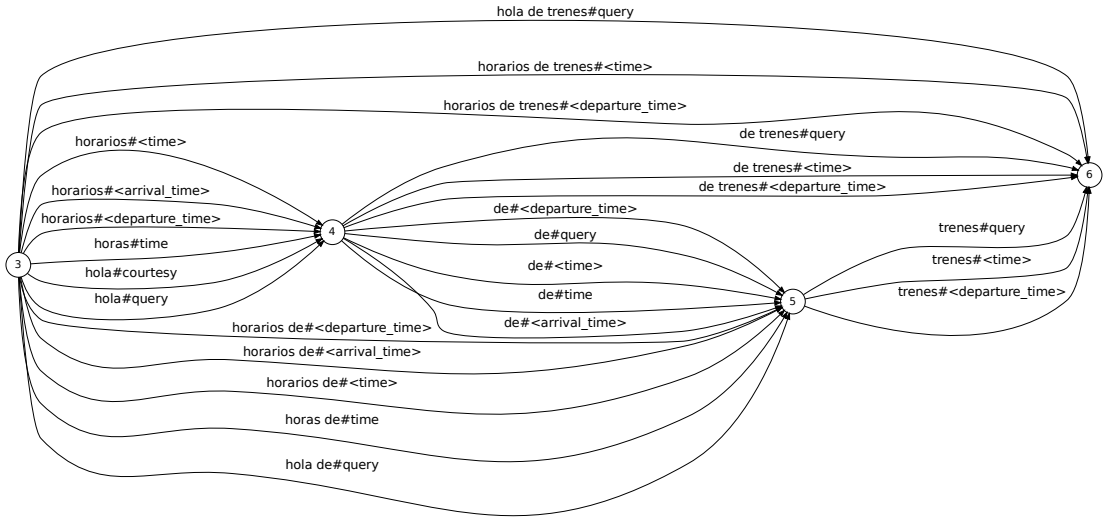

Figure 4.6: Fragment of a graph of concepts built from the subgraph induced by the set of nodes $\{3, 4, 5, 6\}$ of the graph of words of Figure 4.3. For the sake of clarity, only a set of six concepts from the DIHANA task are considered, and weights are omitted.

Figure 4.6 shows a fragment of a graph of concepts built from the graph of words of Figure 4.3. This fragment corresponds to the subgraph induced by the set of nodes $\{3, 4, 5, 6\}$, and it is built taking into account only six of the concepts from the DIHANA task. As it is shown, all segments of any length that can be attached to any concept are found, but for every pair of nodes and each concept, only one segment is kept (the one of maximum combined score). For example, *horas* could also be attached to the concept *<time>*, but if the probability that *horarios* represents this concept is higher, then the latter is turned into an arc, but not the first. Another aspect is that the segments can actually have a correct syntactic structure or not (due to speech recognition errors and the generalization process used to build the graph of words), but if a concept-dependent semantic model gives

the segment a not null probability, then the segment has chances of being converted into an arc. For example, the segment *hola de trenes* (hello of trains) is syntactically incorrect, but the model corresponding to the concept query gives it a not null probability. However, other segments will actually have a correct syntactic structure, like *horarios de trenes* (train timetables), which in this example can be attached either to the concept <time> or to the concept <departure_time>.

Considering together how the graph of concepts are built and Equation 4.3, a question may arise: how can the probability of a sequence of words given a sequence of concepts $p(W|C)$ be computed, if the concept-dependent semantic models only provide the probability of a sequence of words given a concept? The answer to this question is based in the notion of segmentation that was introduced before. If it is possible to assume that each word of the input utterance must be attached to exactly one concept (which it is usually the case, as non-meaningful words can be assigned to some kind of *null* or *void* concept), then the output of the SLU system is a segmented sentence, according to the concepts of the task. Hence, for each segment W_i^j in this segmentation that spans from word i to word j and is attached to concept c_u, the probability of that segment given the whole sequence of concepts is equal to the probability of the sequence given just the concept it is attached to, that is, $p(W_i^j|C) = p(W_i^j|c_u)$. Thus, if S is a sequence of pairs (W_i^j, c_u) that represents a segmentation of W and the concatenation of all the c_u is equal to C, then $p(W|C) = \prod_{(W_i^j,c_u)\in S} p(W_i^j|c_u)$. In the terms of a graph of concepts, S is a full path in the graph (from node 0 to node $|V_{GC}|-1$), and each element of S is an arc with both the pairs (segment of words, concept) and the conditional probabilities, although these are combined with acoustic probabilities.

In addition to individual arcs, paths made of more than one arc are possible, like the one that concatenates the arc where *hola* (hello) is attached to courtesy and the one where *de trenes* (of trains) represents the concept query. Actually these paths are the fundament of the Stage 2 of the semantic decoding algorithm.

4.3.2 Stage 2: Finding the best sequence of concepts

The second stage of the semantic decoding completes the search process by finding the best sequence of concepts. For this purpose, a Dynamic Programming algorithm is applied on the graph of concepts and a stochastic model which represents the probability distribution $p(C)$. This way, the scores of the graph of concepts are combined with the probabilities from a stochastic model of sequences of concepts, in order to fulfill Equation 4.4. For this purpose, this Dynamic Programming algorithm searches for the full path in the graph of concepts which maximizes its score combined with the probability that the stochastic model gives to the underlying sequence of concepts. Furthermore, this method not only provides the best sequence of concepts, but also more information that is also contained in the graph of concepts. Hence, the output of the algorithm consists of several items:

- The sequence of concepts $\hat{C}$ of maximum combined probability, which is obtained taking into account acoustic, lexical, syntactic and semantic information. This is the sequence of concepts underlying the path in the graph of concepts which is found by the algorithm.
- The sequence of words $\tilde{W}$ which is attached to the path in the graph of concepts that provides $\hat{C}$. This sequence of words represents the best recognition of the input utterance by incorporating semantic knowledge to the speech recognition process.
- A segmentation $S(\tilde{W}, \hat{C})$ of $\tilde{W}$ according to $\hat{C}$. This is obtained by taking separately each arc in the path on the graph of concepts that leads to $\hat{C}$.

Figure 4.7 shows an overview of this second stage of the SLU algorithm.

As it has been mentioned, it is necessary to train a model that provides the probability of a sequence of concepts, it is, $p(C)$. To train this model, it is possible to take advantage of the segmentation of the training sentences. This way, since a segmentation is a sequence of pairs (segment of words, concept) that fully splits a training sentence, it is possible to train a stochastic semantic model from the sequences obtained by concatenating these concepts. One of the possible stochastic semantic models that can be trained is an n-gram language model of sequences of concepts, which will be the one used in this work. Also, this model will be represented as a Stochastic Finite State Automaton.

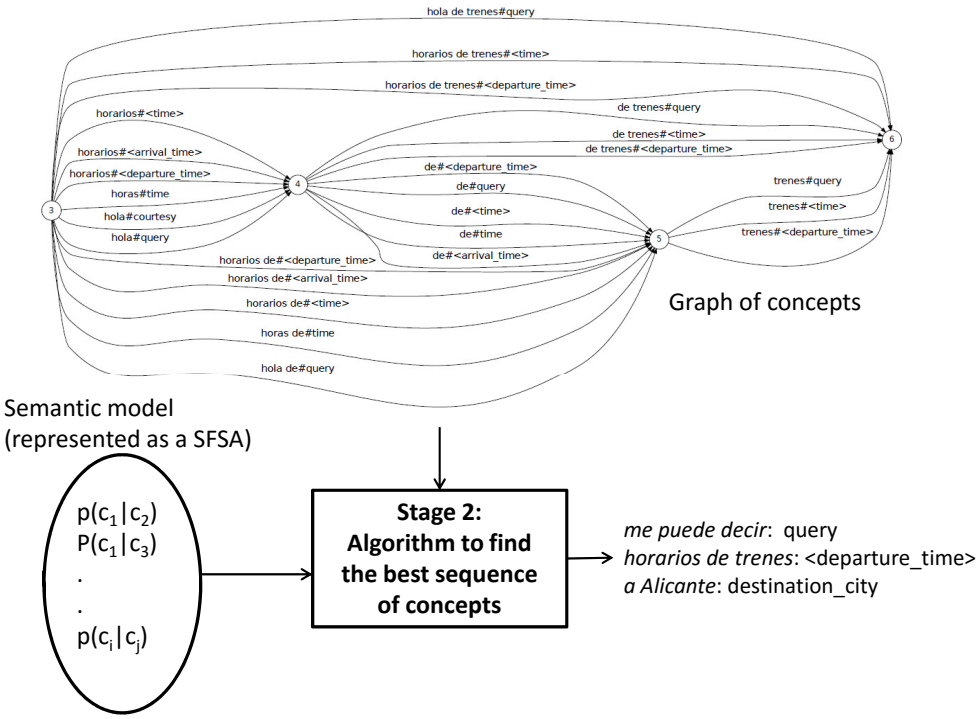

Figure 4.7: Overview of the Stage 2 of the graph-based semantic decoding algorithm.

Algorithm 4 describes a Dynamic Programming method to find the full path in the graph of words that maximizes the combined score. Let i be a node in the graph of concepts, and let q be a state of M_C, it is, $q \in M_C$. Let T be a Dynamic Programming table where $T(i, q)$ represents the maximum score achieved by a path that arrives to node i and the analysis of the underlying sequence of concepts arrives to the state q. The method for calculating this DP matrix is stated in Equation 4.10. It is also worth mentioning that, as it happened in Algorithm 3, there are a grammar scale factor γ and an insertion constant (in this case a concept insertion constant) μ in order to adequately weight the different scores that take part in the algorithm.

Furthermore, when the analysis of the graph of concepts reaches its final node, a transition is forced from every active state in M_C to the final state of the automaton $q_F(M_C)$, using for this purpose the special symbol "</s>" (Equation 4.11). The hypothesis that achieves the

Algorithm 4 Algorithm that finds the best sequence of concepts and its underlying information.

Input: Graph of concepts $GC = (V_{GC}, A_{GC})$,
semantic model M_C,
grammar scale factor γ,
concept insertion constant μ

Output: Best sequence of concepts $\hat{C}$,
underlying sequence of words $\tilde{W}$,
segmentation of $\tilde{W}$ according to $\hat{C}$ $S(\tilde{W}, \hat{C})$

1: $T[0..|V_{GC}| - 1][0..|Q(M_c)| - 1] = \{0, 0, ..., 0\}$

2:

$$T(i,q) = \begin{cases} 1 & \text{if } i = 0 \wedge q = q_0(M_C) \\ 0 & \text{if } i = 0 \wedge q \neq q_0(M_C) \\ \max\limits_{\substack{\forall a \in A_{GC} : \text{dest}(a) = i \\ \forall q' \in Q(M_C)}} T(\text{src}(a), q') \cdot p(q|q', \text{conc}(a))^{\gamma} \cdot \text{wgt}(a) \cdot \mu & \\ \text{otherwise} & \end{cases} \tag{4.10}$$

3:

$$best = \underset{q \in Q(M_C)}{\text{argmax}}\, T(|V_{GC}| - 1, q) \cdot p(q_F(M_C)|q, </\text{s}>)^{\gamma} \tag{4.11}$$

4: $pbest$ = path in the graph of concepts associated to $best$
5: $\hat{C}$ = sequence of concepts of $pbest$
6: $\tilde{W}$ = concatenation of the segments of words in $pbest$
7: $S(\tilde{W}, \hat{C})$ = sequence of pairs (segment of words, concept) in $pbest$
8: **return** $\hat{C}$, $\tilde{W}$, $S(\tilde{W}, \hat{C})$

maximum score is kept, and the path that led to this hypothesis is recovered, for example using backpointers.

Regarding costs, the space complexity of Algorithm 4 is $\mathcal{O}(|V_{GC}| \cdot |Q(M_c)|)$, and it is due to the size of the Dynamic Programming table that is built in Equation 4.10. Backpointers can be stored in order to recover the best path at the end of the algorithm (line 4). This way, the space complexity is not asymptotically increased, but a linear traversal of the backpointers must be performed in order to recover this path.

The time complexity of this algorithm is $\mathcal{O}(|V_{GC}|^2 \cdot |Q(M_c)| \cdot |\mathscr{C}|)$, and it is determined by the maximization performed in Equation 4.10. The computation of this cost is divided into two parts. First, each element of the DP matrix has to be processed, which leads to

a factor of $\mathcal{O}(|V_{GC}| \cdot |Q(M_c)|)$. Second, for each element of the matrix a maximization is performed, which takes into account each arc that arrives to the corresponding node of the graph of concepts. The worst case can be found at the final node of the graph, where, due to the own construction of the graph of words, the possible number of arriving arcs can be up to $(|V_{GC}|-1) \cdot |\mathscr{C}|$, it is, an arc for each concept starting from each of the other nodes in the graph. For the second element of the maximization, which iterates for each predecessor in the semantic model, its cost can be reduced to $\mathcal{O}(1)$ if the DP algorithm is implemented in a forward manner, it is, using successors instead of predecessors.

Table 4.1 shows an example of the output of the second stage of the semantic decoding algorithm, and thus the output of the SLU system.

Sequence of concepts	query <departure_time> destination_city
Sequence of words	*me puede decir horarios de trenes a Alicante* *(could you tell me train timetables to Alicante)*
Semantic segmentation	*me puede decir* : query *horarios de trenes* : <departure_time> *a Alicante* : destination_city

Table 4.1: Example of the output of the full SLU process. This example corresponds to the one introduced in Figure 4.3.

4.4 The frame converter

SLU usually takes place in the context of a spoken dialogue system, and its output is often fed into a module called dialogue manager. For convenience on this purpose, the output of the SLU module can be taken through a post-process that filters out the irrelevant concepts, sorts the remaning concepts according to a canonical order, and standarizes the values that appear associated to the concepts. The structure that results from this process is called *frame*. Each frame is composed by a frame name and a set of attributes, which are called slots, that have values associated to them. As an example, Table 4.2 shows the frame representation that is obtained from the semantic segmentation shown in Table 4.1. In this case, the frame name is (DEPARTURE-TIME), which indicates that the user asked for the departure time of some train, and there is a slot called DESTINATION-CITY that has the value Alicante showing the place the user wants to travel to.

(DEPARTURE-TIME)
DESTINATION-CITY = Alicante

Table 4.2: Frame representation corresponding to the semantic segmentation obtained in Table 4.1.

In order to obtain this representation, the frame converter module uses a strategy which is very similar to the one used for the dialogue system developed for the DIHANA project (Griol et al. 2006). The translation from the semantic segmentation into the frame structure is mainly performed by means of rules. However, for some values such as times and dates, stochastic automata learnt from data are used. Furthermore, some structures such as *tomorrow* or *in the morning* must be translated into numerical values, which, in addition, are in some cases relative to some implicit information (for example, the current date or the current time). For example, assuming that the current date is the 4th of October 2015, the sentence *"I want to know the trains that arrive to Alicante tomorrow morning"* can be represented by the frame representation shown in Table 4.3.

(QUERY)
DESTINATION-CITY = Alicante
ARRIVAL-DATE = 05/10/2015
ARRIVAL-TIME = [07:00–12:00]

Table 4.3: Frame representation for the sentence *I want to know the trains that arrive to Alicante tomorrow morning.*

In order to standarize the representation of the semantic information by means of a frame, the attributes are arranged following a pre-stablished order, and the values are converted into a canonical representation. This allows for a smooth mechanism of communication with the forthcoming dialogue manager (in the case of a full spoken dialogue system), as well as to easily compute a measure of how well the semantic information is being caputred by the SLU module by using the edit distance algorithm.

4.5 Experimental evaluation

In order to evaluate this approach, an exhaustive experimentation has been performed using two different corpora: the DIHANA corpus in Spanish, and the MEDIA corpus in French. First, a set of experiments using the DIHANA corpus will be presented, in order to show the behaviour of this approach when combining multiple outputs from several ASR engines. Next, this approach will be compared for both corpora with other two state-of-the-art approaches: CRF, and the two-level approach.

The following measures were used to assess the performance obtained in the different experiments:

- The Concept Error Rate (CER), which quantifies the percentage of errors in the output of the SLU module. In other words, this measure determines how well the concepts of the sentence are found, according to a reference.
- The Frame-Slot Error Rate (FSER), which evaluates the percentage of errors in the slots of the frames in the final output of the system. That is, it checks how many of the relevant concepts of the sentence were found, along with their corresponding value.
- The Word Error Rate (WER), which measures the percentage of errors in the final transcriptions provided by the system. These transcriptions are obtained as the sequence of words underlying the best sequence of concepts. Hence, when a combination of multiple transcriptions in a graph of words is used, the WER provides a measure of the contribution of the semantics to the recognition process.

All these measures are calculated using the edit distance between the output of the system and the reference, either at concept, frame component or word label. The error rates are calculated in all cases according to Equation 4.12, where the numerator represents the amount of the usual edit operations (substitutions, deletions and insertions) that are obtained from the calculation of the edit distance, and the denominator is the number of items of the reference.

$$\text{Error Rate} = \frac{S + D + I}{L} \tag{4.12}$$

Moreover, it is also worth noting that the FSER is likely to be lower than the CER because all the concepts that are not relevant for the frame representation are not transmitted to it. For example, the concept *courtesy* in the DIHANA corpus is not conveyed into the frame, thus errors in this kind of concepts do not affect the FSER. Also, sometimes two different concepts are translated into a single item in the frame representation. For example, if the user expressed their will to depart from some city and then mentioned a time, it is likely that this time will be a departure time. Unfortunately, to evaluate the FSER for a corpus it is required that a frame representation exists for that corpus. It actually exists for DIHANA, but not for MEDIA. Hence, it has only been possible to evaluate the FSER for the DIHANA corpus.

4.5.1 SLU on the combination of multiple outputs from the ASR module

A first experimental evaluation is focused on checking how a combination of multiple ASR outputs lead to a best understanding result. This experimentation was performed using the DIHANA corpus. For this purpose, $4,887$ utterances were used for training, and $1,109$ for test. The $4,887$ sentences of the training set were used to build the semantic models needed for the SLU method. All the semantic models were estimated as bigram models of the corresponding units, it is, words in the case of the concept-dependent semantic models, and concepts for the second stage of the SLU process. Also, when using speech input, some acoustic and language models of some ASRs were trained using the utterances from this part of the corpus.

The ASRs used for this experimentation were HTK (Young et al. 1997), Loquendo and the Google ASR. For HTK, both the language and the acoustic Models of the ASR were trained with the training set of the corpus. For Loquendo, only the language model was trained with information from the corpus – the acoustic model used was the one it incorporates by default. No information of the task was provided to the Google ASR. The WERs obtained for the test set considering the 1-best output of each of the ASRs individually were: 17.9 for HTK, 17.9 for Loquendo and 29.5 for Google. It is worth mentioning that the DIHANA corpus has very hard acoustic conditions, in addition to being telephonic spontaneous speech. Results point out that both the HTK acoustic models learnt from the corpus and the built-in Loquendo acoustic models are adequate for this task, since their WER is the same in similar language modelling conditions, and their language models were estimated in both cases from training data from the task. Of course, the fact that HTK and

ASR	Acoustic models trained from the corpus	Language models trained from the corpus	WER
HTK	yes	yes	17.9
Loquendo	no	yes	17.9
Google	no	no	29.5

Table 4.4: Corpus-dependent training information provided to each of the ASRs, and WERs obtained from each of them, measured on the test set.

Loquendo have the same WER does not mean that they make the same errors (actually, they are very different). The WER obtained for the Google ASR is higher than the rest because it is a general purpose ASR, without any knowledge of the task, while the others have some information about it. This information is summarized in Table 4.4.

Several experiments were driven using different inputs built from the outputs of the ASRs:

1. Taking the 1-best of each ASR separately, in order to know the performance that can be achieved without any combination. These experiments constitute the baselines.

2. Using as the input for the SLU module the word lattices generated by HTK. In this case, the graph builder module is skipped, as the output of the ASR is itself a graph of words that can be processed by the graph-based SLU module.

3. Transferring to the SLU module a graph of words that is built as a combination of the 3, 5 and 20-best hypotheses provided by the Google ASR.

4. Taking the 1-best recognitions provided by all the three ASRs, and combining them into a graph of words.

5. Additionally, an experiment using as the input to the SLU module the correct textual transcriptions of the user's utterances was performed. This gives an idea of the best performance that can be achieved by the speech understanding system.

Table 4.5 shows the results obtained for each of these experiments. The first row indicates the results obtained with the written transcriptions of the utterances, which simulates a *perfect* ASR. This establishes an upper bound for the results using speech input. The following three lines show the baseline results obtained using the 1-best outputs from the

Input graphs of words	CER	FSER	WER
Text reference	5.4	1.4	–
HTK 1-best	17.7	13.0	17.9
Loquendo 1-best	18.3	11.9	17.9
Google 1-best	25.8	23.4	29.5
HTK word lattice	14.2	11.2	17.9
Google 3-best	20.0	18.7	25.2
Google 5-best	18.9	17.7	24.5
Google 20-best	18.4	17.3	23.3
HTK + Google + Loquendo 1-bests	12.8	8.9	13.5

Table 4.5: Results obtained using the different compositions of ASR outputs, as well as the textual transcriptions and the individual 1-bests.

three ASRs separately. As expected from their WER, HTK and Loquendo achieve better results than Google. Furthermore, the different nature of the errors made by the HTK and Loquendo ASRs is also reflected on this Table, since the HTK output allows to better identify the sequence of concepts, but the transcription provided by Loquendo identifies better the relevant concepts, leading to a lower FSER.

Then, the results obtained by performing a semantic decoding of the HTK word lattices are shown. These results should be compared to the ones obtained with the HTK 1-best. It outperforms the CER and FSER obtained by the 1-best, which means that the semantic decoding algorithm is able to accurately decide among the different options that are represented in the word lattice which one leads to a better semantic interpretation. However, it is interesting to note that the WER obtained this way (it is, the WER calculated from the sentence underlying the best sequence of concepts) is the same than the 1-best. A closer analysis of the recognition errors made for each of the inputs reveals that the sentences underlying the best sequences of concepts contain more semantically relevant words, even when they constitute errors, and some non meaningful words, such as stopwords, are sometimes deleted. This indicates that the SLU algorithm searches for sequences of words that contain semantically relevant words in order to build the graph of concepts, and thus reinforce the probability of the arcs containing these concepts, even if some grammatically necessary but non meaningful words are skipped. However, when the input is the 1-best from the ASR, the errors are mainly due to acoustic confusion that leads to an erroneous analysis by the language model. Of course, the acoustic confusion is also a problem when

the word lattice from the ASR is used, but since two layers of semantic information are applied during the semantic analysis, the semantic information turns to be more relevant for the search of the best path than the acoustic information.

The next section of the Table shows the evolution when several sentences are taken from the Google ASR and are combined in the graph builder module. The baseline in this case are the results obtained for the Google 1-best. The greatest improvement is observed when the 3-best sentences are taken to perform the semantic decoding, with respect to the results obtained with just the 1-best transcription. Taking the 3-best improves the numbers of the 1-best in 5.8 absolute points of CER, 4.7 points of FSER and 4.3 points of WER. This improvement indicates that it is actually convenient to build a consensus graph using the different transcriptions, and then use a SLU algorithm that is able to process it and search for the most convenient semantic interpretation. When more hypotheses are added to the n-best set the results improve steadily, but at a slower pace.

Finally, the last row shows the figures obtained using as the input for the SLU module a combination of the 1-best outputs from the HTK, Loquendo, and Google ASRs. This combination provides the best results, which indicates that combining information from several sources in an adequate way leads to an improvement on the performance of the system. This way, the different kinds of errors made by each of the ASRs can be counterbalanced by the other ASRs, leading to a better semantic interpretation as well as to a better transcription of the user's utterance.

In conclusion, the SLU method developed for this thesis is able to obtain a semantic interpretation of single sentences and graphs of words (including word lattices from an ASR) in an homogeneous way. Results show that an adequate combination of several transcriptions outperform the results obtained with just the 1-best. This indicates that the SLU algorithm is able to choose among the different options in order to build the segments of words that represent concepts, and then obtain the best sequence of concepts.

4.5.2 Comparison with other state-of-the-art approaches

In order to assess the performance of the graph-based SLU method developed for this thesis, an empirical comparison with two other state-of-the-art methods has been performed. One of them is a discriminative method that has been proven to be very effective for SLU tasks: conditional random fields. Many of the best results obtained for several SLU tasks, such as the French MEDIA, have been achieved using this approach (Raymond and Riccardi 2007; Hahn et al. 2011). The other method that has been chosen for this comparison is the two-level approach: a generative method that performs a Viterbi search on a composition of several automata. Since these two models usually work on a single sentence, this experimental comparison was carried out using the correct textual transcriptions of the user's utterance for both corpora, and the 1-best output from the HTK ASR for the DIHANA corpus.

To train the models needed for the two-level approach, several bigram language models were learnt: one for the lexical realizations of each concept plus another for the sequences of concepts. Despite several ways of learning these models can be used, the reason for choosing bigram language models is that they are the ones most used in the literature for this approach. Hence, the learning process for these models is very similar to the one used for the graph-based SLU system.

The experimentation with CRF was driven using the toolkit CRF++ (Kudo 2005). The CRF were trained using an IOB labelling of the training sentences (Ramshaw and Marcus 1999), in order to better represent the dependences of the words within a concept. The set of features used is based on the lexical features within a window of ± 2 words (the two precedent and the two following words), considering both unigrams and bigrams.

To calculate an upper bound of the performance that can be obtained using these three systems, an *Oracle* has been defined. This Oracle takes the outputs from the three SLU systems (graph-based, two-level, and CRF), compares them to the reference at a concept level, and keeps the sequence of concepts that is closer to the reference (i.e. takes the semantic decoding that provides the lowest CER).

Table 4.6 shows the results obtained when the input is the correct text transcriptions of the user's utterances. For this input, the graph-based system outperforms the other two in both CER and FSER. The improvement respect to the CRF system is in line with the

System	CER	FSER
Graphs	5.4	1.4
Two-level	11.0	3.9
CRF	8.8	4.1
Oracle	3.0	0.7

Table 4.6: Results for the DIHANA corpus using as input for the SLU module the correct text transcriptions of the utterances.

results reported in (Raymond and Riccardi 2007), where the generative method (Finite State Transducers in their work) outperformed the discriminative CRF when the training corpus was not very large. This can be explained by the fact that the CRF model must estimate parameters for many feature functions, and thus, needs a large amount of data in order to estimate these values properly. In fact, in that work it was reported that for the MEDIA corpus the CRF outperform the generative model when the training corpus reaches a size of $7,000$ sentences. In this case, the training corpus of the DIHANA task is composed by less than $5,000$ sentences, and despite the fact that the number of concepts and the size of the vocabulary are smaller than in MEDIA, this set of sentences turns to be still inadequate for a proper learning of the parameters of the CRF. On the contrary, this set of sentences permits to adequately learn the concept-dependent semantic structures and the sequences of concepts represented as n-gram models to be used in generative models such as the graph-based or the two-level. However, it turns that the graph-based method behaves much better than the two-level, despite they share the semantic modelling for the task. This indicates that a two-step processing allows for a better identification of the concepts than an integrated model (two-level). Also, the different models can be better weighted using a two-step procedure. Furthermore, the results obtained with the Oracle show that depending on the input sentence, the best semantic decoding is obtained with a different SLU method.

Table 4.7 shows the results when the input to the SLU systems is the 1-best transcription from the HTK ASR. For this setting, the CRF obtains better results than the other two methods. However, the FSER obtained by the CRF and the graphs are very similar, which means that many errors that the graphs make are on non-meaningful concepts, while the slots and values corresponding to relevant concepts are well identified.

System	CER	FSER
Graphs	17.7	13.0
Two-level	20.5	14.8
CRF	15.9	12.7
Oracle	12.8	11.4

Table 4.7: Results for the DIHANA corpus using as input for the SLU module the 1-best output of the HTK ASR.

In order to know how the graph-based SLU system behaves in another language and another task, also in comparison to the two-level and CRF approaches, a set of experiments with the textual references of the French MEDIA corpus were also conducted. The 16,279 user utterances of the corpus were split into three disjoint sets: 12,000 sentences for training the SLU models, 1,279 for development, and 3,000 turns for test. In this case, since the training set is larger than in the DIHANA task, the semantic models were trained as trigram language models. Table 4.8 shows the results obtained in this set of experiments. As a frame representation of the sentences is not available for this corpus, the FSER cannot be measured. However, in order to approximate this criterion, a slightly different one has been determined: the CER considering only the relevant concepts (i.e. those different to *null* and similar). It is shown in Table 4.8 as CER*.

System	CER	CER*
Graphs	17.1	15.9
Two-level	20.4	19.4
CRF	13.4	12.7
Oracle	9.2	9.0

Table 4.8: Results for the MEDIA corpus using as input for the SLU module the correct text transcriptions of the utterances. The column labelled as CER* refers to the CER obtained without considering the non-meaningful concepts, such as *null*.

These results confirm the aforementioned behaviour: a larger training corpus (12,000 sentences in this case) favour the performance of the CRF model. The two generative models are still well trained and show a reasonable performance, specially the graph-based method, but they are beaten by the CRF. Furthermore, while in the written DIHANA corpus the training sentences provide a good knowledge of the structures that appear in the test set,

in the case of MEDIA the training and the test set present a lower correlation. This setting is to some extent comparable to the case of speech input, where the errors in the transcriptions de-correlate the training and the test sets. As in that setting, here the CRF are more robust to unseen structures.

Another feature of the MEDIA corpus plays an important role in support of the CRF method. There are some concepts such as *nombre_ chambre* (room number), *nombre_ reservation* (number of booking), and *nombre_ hotel* (number of hotels) that are very similar in their lexical realizations (numbers in these cases). Hence, they are only distinguishable by taking into account their semantic context and some long-term dependencies in some cases. CRF have good built-in mechanisms to model long-term dependencies within their analysis window. On the contrary, many generative models are only able to model short-term dependencies – for example, bigram models only reflect the dependencies with respect to the last analyzed item. However, the graph-based method can model to some extent long-term relations. For any two adjacent concepts in the graph of concepts, first their underlying sequences of words are analyzed separately, and then the concepts are considered together taking into account the concept-dependent probabilites of their sequences of words. Hence, when combining the probabilities of two arcs, a dependency between two adjacent sequences of words conditioned to their concepts is also considered. This ability for considering these longer-term dependencies help the graph-based approach outperform the two-level method, despite that their underlying semantic models are very similar.

In conclusion, when just one sentence is taken as the input for the SLU algorithm, if the training set is not very large and the test set is quite correlated with the training sentences, then the graph-based method outperforms the other two. However, when the test set is de-correlated with respect to the training sentences, or the number of training samples is increased, then the CRF reaches the highest performance among the three methods considered. Nevertheless, the main strength of the graph-based approach is to be able to analyze in a natural and efficient way uncertain input represented as a graph of words. Actually, it has been shown in this Chapter that an adequate combination of multiple outputs from one or more ASRs leads to better results both from the point of view of the semantic decoding and from the point of view of the quality of the transcription.

Chapter 5

A test-on-source approach to multilingual spoken language understanding

In this Chapter, the approach presented in Chapter 4 is extended in order to deal with the problem of multilingual SLU, by means of a test-on-source strategy (Calvo et al. 2012b; Calvo et al. 2013b; Laguna et al. 2014). Two systems are presented, depending on how the speech translation stage is performed. However, both systems share the same statistical background. This approach combines a speech translation module, in which a generalization of multiple translations is performed by means of graphs, and the graph-based SLU that was previously presented. Finally, an exhaustive experimentation using the multilingual DIHANA corpus is shown and discussed.

5.1 System overview

As it has already been mentioned, the multilingual SLU approaches that exist in the literature can be grouped in two large sets depending on which data is translated into another language: *train-on-target*, where the training corpus must be translated into the user's language (and often more postprocesses are also needed such as resegmenting and relabeling the training sentences) and a new SLU system must be trained; and *test-on-source*, where there is a monolingual SLU system built for some language and the user's utterance is translated into it. The method described in this chapter follows the test-on-source approach.

Like in the monolingual case, the multilingual SLU problem can be stated as the problem of finding the best sequence of concepts $\hat{C}$ given an utterance A, it is:

$$\hat{C} = \underset{C}{\operatorname{argmax}}\, p(C|A) \tag{5.1}$$

Let u be the user's language, and let s be the system in which the monolingual SLU system was trained. Of course, u and s are different, otherwise it would be the monolingual case. Let W_s be a translation into the language s of the sequence of words underlying the user's utterance A. Introducing this as a hidden variable into Equation 5.1, it can be rewritten as:

$$\hat{C} = \underset{C}{\operatorname{argmax}} \max_{W_s} p(W_s, C|A) \tag{5.2}$$

Which, applying the Bayes' Rule, can be transformed into:

$$\hat{C} = \underset{C}{\operatorname{argmax}} \max_{W_s} \frac{p(A|W_s, C) \cdot p(W_s, C)}{p(A)} \tag{5.3}$$

On one side, it must be noted that the denominator in Equation 5.3 is independent of the maximization, so it can be considered a constant and thus be removed. On the other, it is reasonable to assume that A and C are independent variables (the acoustic utterance does not depend on the sequence of concepts to be expressed, and the sequence of concepts does not depend on the acoustic signal). Thus, the latter equation can be stated as:

$$\hat{C} = \underset{C}{\operatorname{argmax}} \max_{W_s} p(A|W_s) \cdot p(W_s, C) \tag{5.4}$$

From the point of view of the test-on-source approach to multilingual SLU, Equation 5.4 can be split into two parts. First, the $p(A|W_s)$ can be seen as a speech translation process which finds the best translation W_s for the input utterance A. Then, the term $p(W_s, C)$ represents a monolingual SLU process which finds the best sequence of concepts for the input W_s. However, following the same reasoning of the monolingual case, taking only one translation from the speech translation module as the input to the SLU system may be very

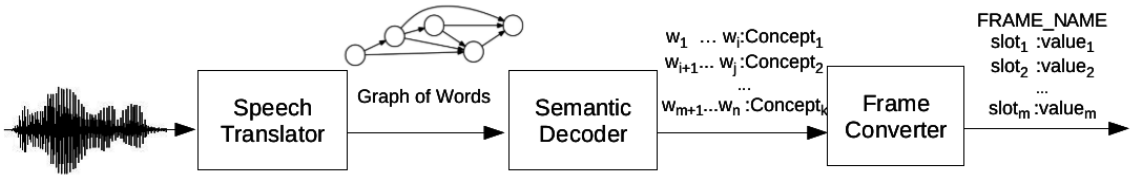

Figure 5.1: General architecture of a test-on-source multilingual SLU system using graphs as the output of the speech translator module.

error prone, since translation errors can strongly damage the overall quality of the system. For this reason, the output of the speech translation module developed for this multilingual SLU system is a graph of words that represents multiple alternative translations for the input utterance, where each of them is weighted with a probability representing $p(A|W_s)$. Then, the same monolingual SLU system presented in Chapter 4 can be used to perform the semantic decoding of this graph of translations. Figure 5.1 shows a rough diagram corresponding to this approach.

As mentioned, the semantic decoder and the frame converter modules can be the same than for monolingual SLU. However, a proper speech translation strategy must be defined in order to supply the monolingual SLU module with suitable translations to be semantically decoded. In the next Section, two approaches to speech translation will be presented, in the framework of test-on-source multilingual SLU.

5.2 Two approaches to speech translation for test-on-source multilingual SLU

Speech translation can be defined as the problem of translating a speech signal into text or voice in another language. The test-on-source multilingual SLU approach needs to use a speech translation system that provides the inputs to the semantic decoder, since the user's utterance must be translated into the system's language. However, errors produced by the speech translation system can severely damage the performance of the SLU system. As a solution for this problem, it is proposed in this work to obtain several translation candidates for the input utterance, and then combine them into a graph of words. This graph not only represents the original translations, but also some new ones that are built

from segments of the original ones. Hence, following this apprach, the components of the speech translation stage are:

1. An ASR working in the user's language that provides a transcription of the user's utterance.

2. One or more translators that supply several translation alternatives for the user's utterance.

3. A graph of words builder module which combines the multiple translations into a single graph of words by means of a grammatical inference process. In this system, it is the same module that joined several recognitions into a graph of words in the monolingual SLU system.

The final output of these three steps is a graph of words that represents a set of alternative translations. Similarly to the monolingual case, each of these translations is weighted with a probability, representing in this case the probability distribution $p(A|W_s)$, but restricted to the set of translations contained in the graph.

In the framework of this thesis, two approaches have been developed to perform the speech translation process. The first of them is based on the use of several general-purpose web translators. The second approach tries to take advantage of the domain-dependent information that is present in a SLU task, by building a task-dependent Statistical Machine Translation (SMT) system.

5.2.1 An approach based on the use of several general-purpose web translators

Current state-of-the-art machine translation systems need large parallel corpora to be trained. A parallel corpus is a collection of sentences in two languages, such that for each sentence in one language, there is a sentence in the other language that corresponds to its translation. There are many large vocabulary parallel corpora available, which offer training data for a variety of languages. For example, one of them is the Europarl parallel corpus (Koehn 2005), which is composed by parallel text from proceedings of the European Parliament. There are also many machine translation systems, such as MOSES (Koehn et al. 2007), that can be trained using these corpora. However, SLU tasks usually work within

a restricted domain, with usually a not so large vocabulary. Unfortunately, in the general case there are not enough parallel data available to train task-specific machine translation systems for SLU. Hence, using general-purpose translators for this function is a reasonable approach. Nevertheless, due to the differences in their vocabularies, it may happen than when using general-purpose data to translate restricted domain sentences, many errors appear in the translated sentences, with respect to the reference sentences. These errors fall into these two categories:

- *Errors related to semantics and pragmatics*: due to the characteristics of the translation system, it often happens that some words are translated by synonyms of the word that appears in the reference sentence. Then, the meaning of the sentence does not change, but these words are usually considered errors by the metric that evaluates the performance of the machine translation system. However, from the point of view of pragmatic linguistics, despite two words are synonyms for some meanings, sometimes one of them is preferred to the other in some specific contexts, which may be the case of the domain of a multilingual SLU system. Hence, if the translation system is trained with data from other domain, it may result in systematic errors due to words that synonyms but one is preferred to the other for the context of the SLU system. Two other aspects of pragmatic linguistics that can also play an important role in this translation process are the language style and the register. For example, if the translation system was trained using formal language and the user is expected to use spontaneous informal language (as it is usually the case of SLU systems). This difference between styles and registers may also lead to a large amount of translation errors. In fact, some recent works like (Stepanov et al. 2013) have investigated how to perform style adaptation in order to improve the translation step of test-on-source multilingual SLU systems.

- *Other translation errors*: as it happens in other machine learning systems, errors in machine translation systems are unavoidable, which can be due to a large variety of factors: structures that do not appear very frequently, confusion between structures, reordering issues, etc.

Both of these types of problems depend on the features of the translation system that is being used: whether it is statistical-based or rule-based, how the parameters were tuned, which training data was provided to the system, etc. Hence, a possible solution is to mix

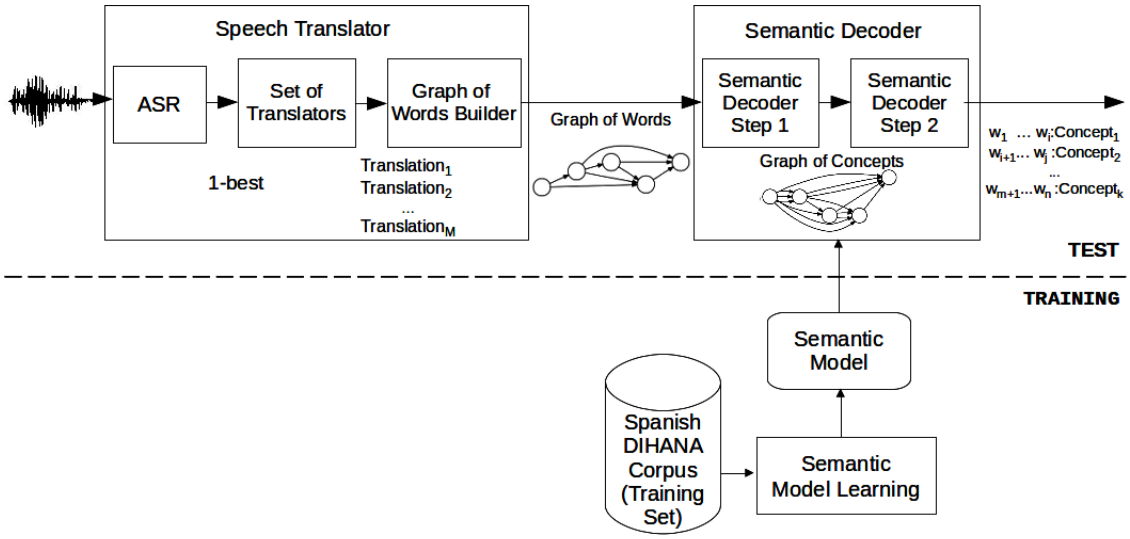

Figure 5.2: Scheme of the architecture based on the combination of the outputs of several general-purpose web translators.

several outputs from different machine translation systems by means of a method that finds a consensus between them. This way, the overall coverage of the translation system is increased and the impact of errors of single translators is minimized. In this thesis, the graph of words builder algorithm presented in Section 4.2.2 has been used in order to find a consensus between translators.

One way to obtain several translations for a sentence from a variety of translators is to use the existing resources that are available in the web. There are many general-purpose web translators that are freely available, and whose outputs can be combined. However, for many of these translators it is impossible to know how they were built and trained, but it is reasonable to think that at least each one used different data during the training step. Other previous works, such as (García et al. 2012) also explored this possibility. In it, the authors used several general-purpose web translators to build a SLU system for Spanish from the French MEDIA corpus using a train-on-target approach, for which they used translations of the sequences of words that represent each isolated concept.

Figure 5.2 shows another system based on the combination of several general-purpose web translators, following in this case the test-on-source philosophy, which has been developed

for this thesis. In this system, the user's utterance is first transcribed using a single ASR in the user's language. Only the 1-best transcription is taken, in order not to generate more variability when mixing several translations. Then, the transcription is translated into the system's language by several general-purpose web translators. Finally, a graph of words is built from the translations, using the graph builder algorithm presented before.

Figure 5.3 shows an example of the complete speech translation process using six different translators. In this example, the user speaks in French and the system's language is Spanish. Also, for the sake of this example, it is assumed that the ASR in French makes no errors for this sentence. A natural translation in Spanish for the input utterance is *puede repetir a qué hora sale el primero*.

In this example, there is one translation among the six provided by the translators that is very similar to the natural sentence in Spanish. Specifically, the translation provided by translator number 2 only changes the word *sale* by the word *marcha*. This is an example of a pragmatic error. The words *sale* (form of the verb *salir*) and *marcha* (form of the verb *marchar*) are synonyms in this context, since both of them mean *to depart*. However, a native Spanish speaker is much more likely to say *sale* rather than *marcha* in this context. Despite being synonyms, this translation error could severly damage the results of the semantic decoder, since it is very possible that the word *marcha* was not present in the sentences used to train the SLU system because it does not usually appear in the context of the SLU task.

Regarding the graph of words, the one in this example can be divided in three parts. In the first part, that spans from node 0 to node 3, there is much confusion because there is little agreement between the translators. The second part goes from node 3 to node 6, and it only has one arc between each pair of adjacent nodes because all the translations agree in this segment. Finally, there is again some disagreement between the different translations from node 6 until the end. It can be verified that this process generalizes the sentences provided by the translators, since the sentence *puede repetir a qué hora sale el primero* was not among the original ones, but there is a path in the graph that allows to build this sentence. The path corresponding to this good translation of the utterance is highlighted in red in the Figure. It is worth noting that this path, despite respresenting a good translation, it is not the one of maximum probability, since if the arc coloured in blue was taken to go from node 0 to node 3 was taken, then the total probability of the path would be higher.

source sentence: *pouvez-vous répéter à quelle heure part le premier*
(could you repeat what time the first one departs at)

(1) **multiple translations**:

poded a qué hora sale el premier
puede repetir a qué hora marcha el primero
poded a qué hora sale el premier
puede repetirse a qué hora parte el primero
pueda repetirse a qué hora parte el primero
puede me repetir a qué hora marcha el primero

(2) **alignment matrix**:

-	-	poded	a	qué	hora	sale	el	premier
puede	-	repetir	a	qué	hora	marcha	el	primero
-	-	poded	a	qué	hora	sale	el	premier
puede	-	repetirse	a	qué	hora	parte	el	primero
pueda	-	repetirse	a	qué	hora	parte	el	primero
puede	me	repetir	a	qué	hora	marcha	el	primero

(3)

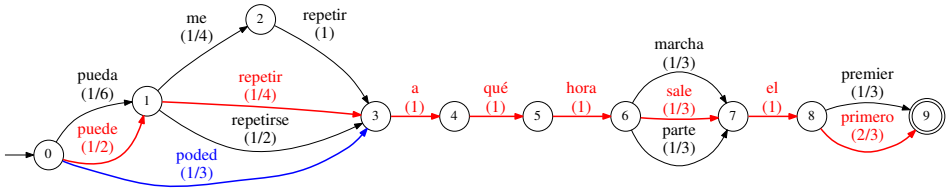

Figure 5.3: Steps for obtaining the graph of words from the original sentence *"pouvez-vous répéter à quelle heure part le premier"*, (*"could you repeat what time the first one departs at"*). A good translation for this sentence into Spanish is *"puede repetir a qué hora sale el primero"*.

However, building a graph of words provides the SLU module with more sentences in order to build a better semantic interpretation of the input utterance. Furthermore, since the SLU module will finally choose a sentence that is represented by a path in this graph, as a subproduct of the semantic decoding process a translation of the utterance will be built, by incorporating semantic knowledge for its construction.

A key feature of this approach is that, despite the graph of words built this way may have any words in the system's language since the translators are general-purpose, the SLU module is able to restrict the words provided by the different translators to the system's

task-dependent domain. For example, if many translators provide a word that is not in the domain of the system, and just one outputs an in-domain word, then the paths that are chosen to build the graph of concepts will contain this in-domain word, and the out-of-vocabulary words will be avoided. Thus, in this approach the conversion from a general domain to a restricted one is performed by the SLU module via its restricted task-dependent vocabulary.

5.2.2 An approach based on a task-dependent statistical machine translation system

One drawback of general-purpose machine translators applied to restricted-domain tasks, such as multilingual SLU, is that they do not take advantage of domain-specific knowledge. This is even more important in the case of multilingual SLU, since in language understanding tasks there are usually some keywords that determine a large part of the meaning of a sentence, and if these keywords are lost during the translation process, sometimes due to synonyms and other pragmatic errors, then the understanding result would be very poor. This problem could be solved using a task-dependent (or domain-specific) corpus to train the machine translation system. In the case of the multilingual DIHANA corpus, a task-dependent parallel corpus was obtained by translating the training data by four general-purpose web translators. This parallel corpus is then used to train a task-dependent MOSES translator.

Figure 5.4 shows a diagram of the resulting architecture. In it, the training corpus (Spanish DIHANA in this case) has a double function. On one side, it is used to train the semantic models for the SLU module that works in Spanish. On the other, it is the ground from which the statistical machine translation system for translating the user's utterances will be built. All the sentences from this training corpus are translated by several general-purpose web translators, building this way a parallel task-dependent corpus. Then, this parallel corpus is used to train a statistical machine translation system, which in this system is the MOSES software. In this training phase lexical models, phrase-based models and reordering models are built. It is worth highlighting the alignment and phrase extraction steps. In the alignment step each pair of training sentences (it is, each sentence and its translation) are aligned according to some metric (Och and Ney 2003), building this way an alignment table. Then, in the phrase extraction step this matrix is processed to determine which chunks of

the sentence in one language are translated by which segments of the sentence in the other language. The phrases determined here will be later used to translate the input sentence: when a sentence is to be translated, the phrases that appear in it are identified and used in the translation algorithm. Hence, in the case of the multilingual DIHANA corpus, this means that the different translations that were obtained using several general-purpose web translators can be mixed, if the translation algorithm finds that it will lead to a translation with a higher score. It also means that the projection to in-domain words in the system's language is performed via this machine translation process, since the translations will be based on the original training sentences, and they are expected to be adequate for the SLU task.

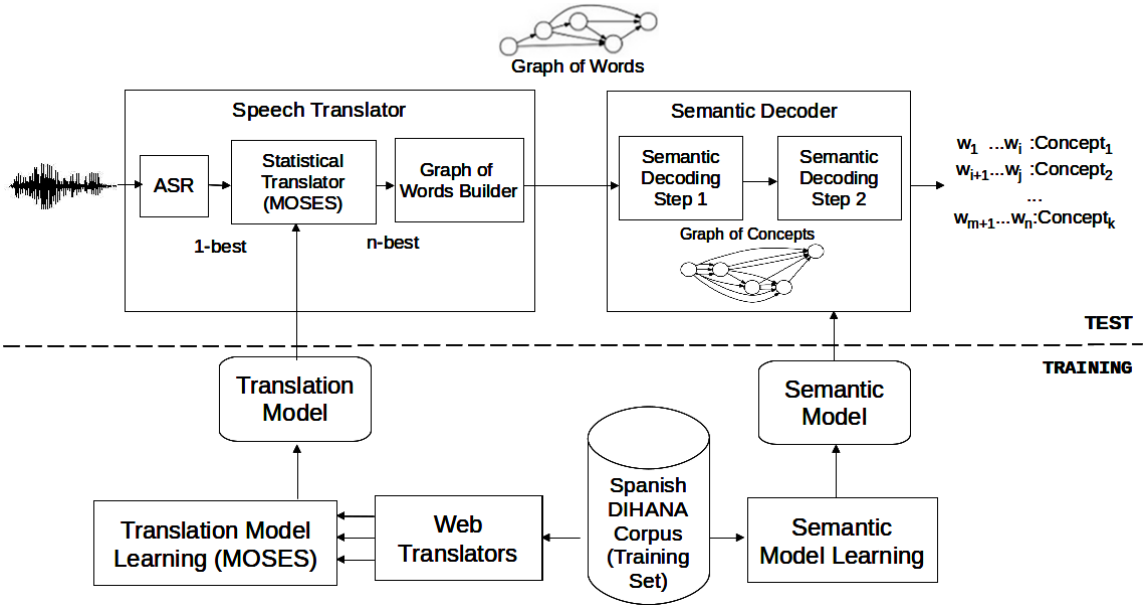

Figure 5.4: Scheme of the architecture based on a task-dependent machine translation system using MOSES.

In order to tune the parameters of the machine translation system, for example using the MERT algorithm (Och 2003), a small set of development sentences translated by humans may be used, which must be different to the training ones.

Finally, the n-best translations provided by the statistical machine translation system are used to build a graph of words in the same way that it worked for the other system. This

allows to amend the errors that the 1-best translation may have, possibly due to the presence of errors in the training corpus, if the correct word is in the other sentences of the n-best set.

5.3 Graph-based SLU

Following the test-on-source approach, once the input utterance has been translated into the system's language, then any monolingual SLU engine may be used to extract the semantic information contained in it. In this case, the output of the speech translation module is a graph of words which represent a set of possible translations of the input utterance, each of them weighted with a probability of the distribution $p(A|W_s)$ restricted to the sentences represented in the graph. This case is very similar to when in the monolingual case several recognitions of an utterance were joined into a graph of words. Hence, the same approach to graph-based SLU can be used here (see Section 4.3 for more detail on the algorithms used).

First, the graph of words is converted into a graph of concepts, by searching those segments of words that are lexical realizations of a concept. These segments of words are converted into arcs in the graph of concepts, along with the concept they represent. The weight of this arc is a combination of the product of the arcs of the graph of words that lead to this path and the probability that the concept gives to the segment of words. More formally, this weight is $p(A|W_{si}^j) \cdot p(W_{si}^j|c)$, where i and j are the beginning and ending nodes of the path in the graph of words and c is the concept represented by the segment W_{si}^j. The models that provide the probability of a sequence of words given a concept, it is, $p(W_{si}^j|c)$, are called concept-dependent semantic models, and can be learnt as n-gram language models of the sequences of words that lexicalize the concepts of the task.

The graph of concepts is then processed in order to find the best sequence of concepts $\hat{C}$. This is performed by computing the path in the graph of concepts that maximizes the combination of its probability and the probability $p(C)$ that a semantic model gives to the underlying sequence of concepts. This semantic model can be estimated as an n-gram language model of sequences of concepts.

This way of finding the best sequence of concepts fulfills what it was stated in Equation 5.4. It is also worth noting that in both stages of the graph-based SLU process there are

a few tuning parameters that must be adjusted. A development dataset may be used to determine an empirical value for them.

The search of the best path in the graph of concepts not only provides the best sequence of concepts $\hat{C}$, but also some additional information. Since each arc of the graph of concepts has a segment of words associated to it, joining the segments corresponding to the best path provides a translation $\tilde{W}_s$ of the input utterance. This translation is expected to be more semantically consistent than a sentence provided by a single translation system (although in some cases they might be the same sentence). The output of the SLU process also provides a segmentation of the translation $\tilde{W}_s$ in terms of the concepts of $\hat{C}$.

As an example, Table 5.1 shows a possible output of the SLU module when the input is the graph of words of Figure 5.3. The segmentation can be further processed to obtain a frame representation of the semantics.

Sequence of concepts	query <departure_time> relative_number
Sequence of words	*puede repetir a qué hora sale el primero* *(could you repeat what time the first one departs at)*
Semantic segmentation	*puede repetir* : query *a qué hora sale* : <departure_time> *el primero* : relative_number

Table 5.1: Example of a possible output for the graph of words of Figure 5.3.

5.4 Experimental evaluation

In order to evaluate the two presented systems that implement a test-on-source approach to multilingual SLU, an exhaustive experimentation using the multilingual DIHANA corpus (García et al. 2014) has been driven. For this experimentation, the partition of the corpus that has been used is analogous to the one used in the monolingual experimentation: $4,889$ sentences for training, 340 for development, and $1,000$ for test. The training set was translated using four general-purpose web translators, which amounts to $4,889 \cdot 4 = 19,556$ training sentences. The development and test sets were translated and uttered by native speakers of the different languages.

The experiments were conducted using English and French as the user's languages, being Spanish in all cases the system's language. Also, both textual and spoken inputs were considered. The ASR used for spoken inputs was the Google speech recognizer. The Word Error Rates obtained during speech recognition were 20.0 for English and 19.5 for French. It must be noted that, since the French dataset has just 500 spoken utterances, the experiments with French speech input were performed using 500 items, instead of the 1,000 used in the English dataset. Also, for the spoken datasets, all the modules of the system were tuned using correct written inputs.

For the system that performs a combination of several outputs from general-purpose web translators, four translators were used to obtain the translations of the test utterances that are to be combined. These translators are the same that were used to obtain the training part of the multilingual DIHANA corpus: Bing Translator, Google Translator, Lucy, and Opentrad.

For the system based on a task-dependent machine translation system, the MOSES toolkit was used. The translation models were learnt using the training part of the multilingual DIHANA corpus, and the system was tuned by optimizing the BLEU score on the written sentences of the development set. Up to the 5-best translations were taken as the output of this module.

Similarly to the monolingual case, all the semantic models were trained as bigram models of linguistic units: words in the case of the concept-dependent semantic models and concepts for the other semantic model.

To compare the results obtained by these two systems, a experimentation in the same conditions was performed with two other systems:

- A system that takes a set of translations and performs the semantic decoding using the graph-based algorithm but taking each translation separately. Then, the output to which the SLU module assigned the maximum score is the one provided by the system. This system is equivalent to removing the graph of words builder module, then having several SLU systems for just one sentence working in parallel, and finally taking the one with maximum score as the final output. This allows to analyze the consequences of the inference process which builds the graph of words that contains several new translations. The inputs to this system have been the set of translations

provided by the general-purpose web translators, and the 5-best translations from MOSES.

- A system that takes a single translation and uses a CRF model to obtain the best sequence of concepts. The inputs to this system were the individual outputs from the general-purpose web translators and the 1-best translation from the MOSES system. This allows to compare the graph-based multilingual SLU system to another in-cascade multilingual SLU setting composed by a machine translation system plugged to a state-of-the-art CRF-based Monolingual SLU engine. For the CRF, the CRF++ toolkit was used. With the objective of improving the representation of the relation between words within a concept in order to achieve better SLU results (and thus to obtain a harder baseline) the training set was labelled using IOB tags. Lexical features based on unigrams and bigrams were used to train the CRF tagger.

Several measures have been used to assess the quality of the different systems. To evaluate the quality of the semantic decodings provided by the SLU module, the Concept Error Rate (CER) and the Frame-Slot Error Rate (FSER) have been computed (see Section 4.5). However, the SLU module also builds a translation of the input utterance which takes into account the translations provided by one or more systems and the semantic knowledge of the task. To judge the quality of these translations, the Word Error Rate (WER) and the BLEU score have been determined. BLEU stands for Bilingual Evaluation Understudy and it is nowadays the most used translation score. It was presented in (Papineni et al. 2002) and it is based on a modification of the n-gram precision of a translation according to a reference. A higher BLEU score indicates a better translation.

For reference purposes, the monolingual results for Spanish are shown in Table 5.2, where the Google ASR was used to transcribe the speech input. In this table, the WER refers to the one achieved by the speech recognizer.

Input	WER	CER	FSER
Text	–	4.5	1.4
Speech	17.6	14.2	12.8

Table 5.2: Results achieved by the monolingual system in Spanish.

Translators	CER	FSER	WER	BLEU
1000	24.1	15.2	45.6	35.2
0100	24.5	17.5	47.0	37.0
0010	32.3	19.6	52.3	25.6
0001	31.9	26.5	54.2	20.6
1100	20.7	11.9	43.2	40.5
1010	21.7	12.9	43.5	38.4
1001	22.1	14.4	43.5	36.8
0110	20.8	11.8	44.1	38.8
0101	24.0	18.1	45.3	37.9
0011	28.1	19.6	48.2	29.0
1110	**19.2**	**10.6**	**42.5**	**41.1**
1101	21.0	12.8	**42.5**	40.6
1011	21.5	13.9	43.5	38.2
0111	19.9	12.2	42.8	39.6
1111	20.8	12.9	43.6	39.2

Table 5.3: Results for English text using general-purpose web translators.

5.4.1 Results with the system based on general-purpose web translators

For each of the languages and types of input (text or speech), all the combinations of the four general-purpose web translators were experimentally explored. This amounts to 15 possible combinations of translators for each language and type of input. In order to give a concise representation of the translators used in each experiment, they have been represented as a binary code, where 1 indicates that the translator was used, and 0 denotes its absence. In this binary representation, from left to right, the different numbers will represent the presence or absence of: Bing, Google, Lucy, and Opentrad. For example, the code 0110 means that the translations from Google and Lucy were combined, discarding those from Bing and Opentrad.

Table 5.3 shows the results obtained when the input are correct written English sentences. It can be observed that the performance in terms of the semantic measures showed by each of the translators alone is very poor when compared to their aggregation. For example, the combination 1100 outperforms the best single translator by 3.4 points of CER and 3.3 points of FSER. In this case, among all the possible combinations, the one that provides

the best results is not the union of all the translators, but the combination of three of them (1110). This outperforms the best single translator in 4.9 points of CER and 4.6 points of FSER. Also this combination provides the translations of best quality, achieving a WER of 42.5 and a BLEU of 41.1, while the best results achieved by any single translator are a WER of 45.6 and a BLEU of 37.0.

It is interesting to note that the combination 1101 achieves the same WER than 1110, but leads to worse BLEU, CER and FSER. In the case of BLEU, it is explained because the WER measure is defined in terms of words and BLEU is determined by common n-grams. Then, it can happen that two sequences that lead to the same WER, have different number of common n-grams with respect to a reference. Regarding CER and FSER, this indicates that the combination 1101 makes more translation errors that are semantically relevant, it is, words which failure lead to errors in the identification of the semantic segments (concepts). On the contrary, the combination 1110 makes the same amount of errors (in percentage) but more semantically relevant words are preserved.

Table 5.4 shows the results when the input to the system are spoken utterances in English. In this case, the best combination for all the understanding measures is also 1110, which outperforms in 6 absolute points of CER and 3.8 absolute points of FSER the best result obtained by a single translator. Also this is the combination that achieves the best BLEU score, 31.9, which amounts to 3.3 points more than the best single translator. However, the combination that provides the best WER is 0111. This can be explained similarly to the previous case: the sequence of words that provides the best Word Error Rate does not have to lead to the best results on CER, FSER, and BLEU.

Both Table 5.3 and Table 5.4 show that an adequate combination of the information provided by several translators leads to both an improvement of the quality of the semantic interpretation and an increase of the quality of the resulting translation. This means that when pieces of information form several translations are combined by means of a graph, the SLU algorithm *jumps* from one to other translation in order to select in each case the segments of words that lead to the best semantic intepretation, and additionally, to a translation of the user's utterance that also incorporates semantic knowledge. This shows the suitability of this strategy for multilingual SLU. Furthermore, the improvement in the quality of the translations suggests that the incorporation of semantic knowledge to ma-

Translators	CER	FSER	WER	BLEU
1000	35.9	28.8	55.0	25.2
0100	39.7	33.0	56.5	28.6
0010	40.8	33.1	58.4	21.7
0001	40.4	38.3	59.8	18.3
1100	32.2	25.9	52.4	30.9
1010	32.1	27.1	51.9	28.9
1001	32.7	27.6	52.7	27.3
0110	33.5	27.2	52.6	30.8
0101	34.4	30.1	53.7	30.0
0011	37.0	30.6	55.7	24.3
1110	**29.9**	**25.0**	51.6	**31.9**
1101	32.6	26.5	51.6	31.4
1011	31.4	26.2	51.8	30.0
0111	30.7	25.2	**51.2**	31.4
1111	31.8	26.3	53.1	30.9

Table 5.4: Results for English speech using general-purpose web translators.

chine translation systems could lead to a decrease in the number of errors made by the system, specially in those related to semantics and pragmatics.

It is interesting to point out that for both modalities of English input the best results are achieved when not all the translators are used. This means that, provided a graph of words built from some set of translations, adding one translation to the set may not induce a graph of words that is more appropriate for the semantic decoding process. When adding a new translation to a set, the resulting graph of words may differ in two ways from the former graph: (i) in the probabilities represented in the arcs, and (ii) in their topology. These new probabilities and the new topology may lead to a completely different graph of concepts whose probabilities mislead the semantic decoding process through a path that contains more errors. In other words, it sometimes happens that adding more translations to a set either increases the entropy of this set or reinforces a wrong path, resulting in a less adequate graph of words that generates a not so adequate graph of concepts and thus to errors in the semantic interpretation.

Translators	CER	FSER	WER	BLEU
1000	30.7	23.1	52.4	30.1
0100	33.0	25.4	54.6	27.8
0010	27.5	18.5	50.6	30.1
0001	30.6	20.0	54.3	24.5
1100	26.3	18.9	50.1	34.0
1010	22.7	15.4	46.5	38.3
1001	23.9	16.6	49.3	33.8
0110	24.9	18.0	47.9	36.5
0101	23.3	16.7	48.9	34.4
0011	25.0	16.5	48.7	33.5
1110	21.6	14.4	45.9	39.7
1101	21.7	14.8	47.4	37.3
1011	21.2	13.7	45.8	38.7
0111	21.2	14.4	**45.7**	39.2
1111	**20.2**	**13.2**	**45.7**	**39.9**

Table 5.5: Results for French text using general-purpose web translators.

Tables 5.5 and 5.6 show the results obtained for both text and spoken French input. In both cases the best result is now obtained when a combination of all four translators is taken. This means that, overall, all the translators contribute to an increase of the quality of the resulting graph of words. Also in both modalities significant improvements are achieved in all the measures with respect to the best single translator. This confirms the hypothesis that the SLU algorithm is able to adequately combine chunks from several translations in order to find the best semantic interpretation and the most semantically appropriate translation, which, in turn, also improves the measures related to the quality of the translation.

Translators	CER	FSER	WER	BLEU
1000	35.3	30.7	58.7	23.1
0100	37.3	33.5	61.2	22.7
0010	30.4	26.1	58.4	21.1
0001	32.4	27.3	61.5	17.7
1100	31.1	26.9	55.9	28.2
1010	27.7	24.4	53.8	28.9
1001	28.5	24.9	56.2	27.0
0110	29.7	26.3	55.8	28.5
0101	28.8	24.8	55.9	27.1
0011	28.7	23.9	56.8	24.4
1110	26.2	21.7	53.1	32.3
1101	26.5	22.6	53.9	30.7
1011	26.3	22.0	53.3	30.7
0111	26.7	23.0	53.3	31.0
1111	**24.8**	**20.8**	**52.3**	**33.1**

Table 5.6: Results for French speech using general-purpose web translators.

In order to further analyze the improvements that offer the inferred translations represented in the graphs of words, another experiment without this combination has been performed. In it, the different translations provided by the general-purpose web translators are analyzed by the graph-based SLU module independently, and the semantic interpretation that obtains the best score is kept. Table 5.7 shows the results obtained in this experiment, compared to those achieved taking only the sentences provided by the single translator that provides the minimum CER, and to the best results that were reached by means of a combination of hypotheses.

			CER	FSER	WER	BLEU
English	Text	Best single translator (1000)	24.1	15.2	45.6	35.2
		Maximum score separately	21.9	12.1	44.1	29.4
		Graph combination (1110)	**19.2**	**10.6**	**42.5**	**41.1**
	Speech	Best single translator (1000)	35.9	28.8	55.0	25.2
		Maximum score separately	33.7	26.1	54.1	22.9
		Graph combination (1110)	**29.9**	**25.0**	**51.6**	**31.9**
French	Text	Best single translator (0010)	27.5	18.5	50.6	30.1
		Maximum score separately	27.3	18.9	49.8	25.8
		Graph combination (1111)	**20.2**	**13.2**	**45.7**	**39.9**
	Speech	Best single translator (0010)	30.4	26.1	58.4	21.1
		Maximum score separately	31.6	26.4	57.4	20.8
		Graph combination (1111)	**24.8**	**20.8**	**52.3**	**33.1**

Table 5.7: Results for text and speech in English and French selecting for each utterance the translation from the general-purpose web translators that achieves the maximum score in the SLU module. These results are compared to the best results obtained by a single translator and to the best results obtained using a combination of the different translations by means of a graph of words.

The numbers show that the best results are obtained in all cases using a combination of hypotheses by means of a graph of words. For example, in the case of French text input, the graph combination improves the separate processing of the translations by 7.1 absolute points of CER and 5.7 absolute points of FSER. These results clearly show that, in fact, the SLU algorithm selects at each step the word in the graph that is more suitable to build the sequences of words of the graph of concepts, regardless of whether two consecutive words in the segment came from the same original translation or not. This leads to a better quality of the segments built, and then to a higher quality semantic interpretation. A better

translation is obtained this way as well, thanks to the semantic knowledge that the SLU process adds to the machine translation engines.

It is also worth noting that in the English case the separate combination of translations outperforms the results of the best single translator in the semantic measures, but the BLEU it achieves is worse than the single translator. This indicates that just selecting a translation among a set according to semantic criteria does not lead to an improvement of the quality of the translations, while an accurate mix of the translations actually improves this quality. However, in the French case, the CER and FSER obtained by the separate combination of translations are slightly worse than the numbers for the best single translator.

In consequence, this naïve selection method actually allows to prove that: (i) given two sentences and a set of translators, the translator that provides the best results for the first sentence may not be the best for the second, since each sentence may have different structures and each translator was learnt using different data, and (ii) mixing subsequences of words from several translations by means of an adequate mechanism (a graph of words in this case) provides better results than processing each sentence separately and then deciding which semantic interpretation should be kept.

Finally, Table 5.8 shows a comparison between the performances of the graph-based multilingual SLU system and a configuration in which the output of each translator is fed into a CRF-based Monolingual SLU system. Two different kinds of contrasts are shown in the Table. First, for each single general-purpose web translator, the SLU results obtained with the CRF system are shown, as well as the variation (in absolute points) obtained with respect to the graph-based system when just that translator is used. A negative variation indicates that the CRF behaves better than the graph-based system, while a positive variation shows that the results obtained with the graph decoding are better. Second, the row labelled as Graphs shows the best results obtained by using a combination of translators on the graph-based system.

A comparison of the results obtained by each translator, language and type of input separately does not clarify much this discussion. Sometimes the CRF behaves better (it improves the numbers of the graph-based system by 5.0 points of CER and 3.3 points of FSER in one of the settings that use English speech), but other times the graph-based semantic decoder outperforms the CRF (for example, in one of the French speech settings it offers an improvement of 2.4 points of CER and 5.3 points of FSER over the CRF). In general,

		Text		Speech	
	Translators	CER	FSER	CER	FSER
English	1000	24.5 (+0.4)	17.4 (+2.2)	33.7 (-2.2)	29.3 (+0.5)
	0100	23.9 (-0.6)	15.8 (-1.7)	34.7 (-5.0)	29.7 (-3.3)
	0010	31.2 (-1.1)	22.2 (+2.6)	37.3 (-3.5)	31.0 (-2.1)
	0001	33.1 (+1.2)	26.6 (+0.1)	40.2 (-0.2)	35.8 (-2.5)
	Graphs	**19.2**	**10.6**	**29.9**	**25.0**
French	1000	30.1 (-0.6)	24.3 (+1.2)	33.6 (-1.7)	31.3 (+0.6)
	0100	32.7 (-0.3)	27.4 (+2.0)	37.5 (+0.2)	34.5 (+1.0)
	0010	26.2 (-1.3)	22.2 (+3.7)	30.4 (0)	31.1 (+5.0)
	0001	30.8 (+0.2)	22.6 (+2.6)	34.8 (+2.4)	32.6 (+5.3)
	Graphs	**20.2**	**13.2**	**24.8**	**20.8**

Table 5.8: Results for text and speech using CRF in English and French taking as input the translations generated by the general-purpose web translators separately. The numbers in parentheses are the variations with respect to the graph-based approach when the same translator is used. A negative variation indicates that the CRF obtains better results than the graph-based approach for that translator, and a positive variation indicates a better behaviour of the graph-based strategy. The row labelled as Graphs shows the best result obtained by a combination of translators and the graph-based approach.

it seems that the CRF obtain better results for the CER measure, but the graph-based approach achieves a better performance for FSER. This indicates that CRF provides a more accurate sequence of concepts, but in the underlying segmentation some meaningful words are assigned to the wrong concept, which leads to more mistakes when a more concise representation (such as a frame) is used. However, it is clear that the combination of translations by means of graphs of words certainly outperforms the numbers obtained with the CRF. This means that an adequate combination of translations that is analyzed by a SLU algorithm that is able to process and exploit them in a natural and efficient way is more convenient than the use of a state-of-the-art SLU system that only utilizes a single translation.

5.4.2 Results with the system based on a task-dependent MOSES translator

For the experiments with the system based on a task-dependent MOSES translator, up to the 5-best translations were taken for each of the languages and input modalities (text or speech). The set of translations used to build the different graphs of words was built incrementally, it is, the 2-best translation was added to the 1-best, the 3-best was added to the 1 and 2-best, and so on. Hence, in the following tables, a row labelled as 3 means that the 3 first hypotheses provided by MOSES were used to build the graph of words. The MOSES systems were tuned to optimize the BLEU score on the correctly written development set. This means that two different translators were built: one for English and another for French, and for each language each translator processed both the correctly written and the speech transcription inputs. It should be highlighted that the sizes of the training and the development sets used in these experiments (19,556 and 340 respectively) are much smaller than the sizes of usual machine translation corpora, which usually have several hundred thousand sentences for training and several thousand for development. This may lead to problems related to this data scarcity, such as a weak estimation of the different probabilities of the translation models, as well to a suboptimum tuning of the different parameters of the translator.

Table 5.9 shows the results obtained for the English correctly written input. The CER obtained for the 2 and 4 best translations are very similar, but the lowest is obtained using the 2-best. However, it does not lead to the best FSER, which is obtained just using the 1-best. This small difference may be due to slight differences in the segmentations obtained by each system, and specifically in the non-relevant concepts (such as *courtesy*), which are better identified by the system that uses the 2-best hypotheses. It is also worth noting that when the 3-best translation is added to the set (and the same happens with the 5-best) the quality of the graph of words decays considerably. This can be explained similarly to the case of the general-purpose translators: if a bad translation is added to the set and it did not already have a strong probability mass assigned to arcs that provide a good assignment of words to concepts, then the quality of the graph of concepts that is subsequently built is also deteriorated, which worsens the quality of the final semantic decoding.

However, this explanation arises a new question regarding the quality of the translations: why the 3-best translation is a *bad* translation, while the 4-best is a *good* translation since it leads to an improvement of the results? Actually, this could happen in any machine

n-best	CER	FSER	WER	BLEU
1	21.9	**12.8**	44.1	38.5
2	**21.2**	13.3	43.5	39.0
3	22.7	14.8	43.5	38.8
4	21.4	13.4	43.5	38.8
5	22.5	14.7	**43.2**	**39.3**

Table 5.9: Results for English text using MOSES.

translation system (there are re-ranking strategies in order to find the most accurate translation, which may not be the 1-best), but in this case it is related to the training data scarcity. In the multilingual DIHANA corpus, each Spanish sentence has four translations obtained by means of general-purpose web translators. These translations have also errors since they are unavoidable in machine translation. Hence, different structures (probably with errors) in English and French are alligned to phrases of the same correctly written Spanish sentence. The integration of all these structures leads in the translation models to a large variability in the different assignments of phrases between languages (which are called alignments). When translating correct English and French sentences into Spanish, the translator searches for structures that it saw during the training phase, and the phrases in the other language that are aligned with them. In the case of the multilingual DIHANA corpus, two effects can be observed during the translation process. First, due to the errors present in the training corpus, the structures of the written sentences that coincide with structures of the training sentences may be very short. Second, due to the data scarcity, the alignments that were obtained during the training phase may not be of a high quality, resulting in a translation by a sequence of words that is not appropriate. However, it may happen that a translation that is below in the n-best list (i.e. has received a lower score by the translator) is actually more adequate. Hence, when this translation is added to the set, an improvement of the results is observed.

Table 5.10 shows the results for the English speech input. The tendency of the results obtained with this system is similar to the one that was presented by the textual input.

Another effect that is observed for both modalities of English input is that the best quality of the translations is obtained when all the 5-best translations are combined. MOSES itself tries to obtain translations that optimize the BLEU measure, but when several translations

n-best	CER	FSER	WER	BLEU
1	31.0	**24.2**	53.1	27.9
2	**30.6**	25.0	52.7	28.1
3	31.5	26.0	52.9	27.9
4	30.9	25.4	52.8	28.1
5	31.5	25.7	**52.4**	**28.3**

Table 5.10: Results for English speech using MOSES.

are combined using task-dependent semantic knowledge, then the quality of the translations is improved.

Tables 5.11 and 5.12 show the results obtained when the input language is French. In the case of text input, the best results according to the semantic measures are obtained using the 3-best translations, while the translation quality measures achieve their best results when using all the 5-best translations. However, in the case of speech input, the best results for all the measures are obtained when the 5 translations are used, with a steady improvement of the figures as the number of translations increases. Hence, the tendencies observed for French are similar to those presented by the experiments using English: (i) a higher number of translations may not lead to a better semantic decoding, and (ii) the incorporation of semantic knowledge to the combination of translations allows to improve the results provided by the translation models of MOSES.

Another important fact is observed in all the four tables obtained in this experimentation until now: all the numbers for all the measures are very close, much more than in the case of the general-purpose web translators. This is explained by the different natures of the translators used in each of the cases. On one side, in the case of general-purpose web translators, the 1-best outputs from several translators, each one probably built from different data, were mixed. Furthermore, the output vocabulary in Spanish was not limited to the scope of the SLU task, and it was the SLU algorithm who decided which words to take to represent each concept, avoiding out-of-vocabulary words when building the graph of concepts. Then, the combination of several translators makes it more possible that more in-vocabulary words appear in the graph of words, which leads to a better quality graph of concepts and then to a significant improvement of the results. On the other side, when using a MOSES-based translation system, several sentences from a single

n-best	CER	FSER	WER	BLEU
1	21.9	15.2	45.8	37.4
2	21.8	15.1	45.8	37.5
3	**21.6**	**14.7**	45.8	37.6
4	21.6	15.0	45.6	37.8
5	21.6	14.8	**45.4**	**38.1**

Table 5.11: Results for French text using MOSES.

n-best	CER	FSER	WER	BLEU
1	27.4	23.1	51.5	31.0
2	26.8	22.1	51.4	31.4
3	26.3	21.8	51.2	31.4
4	26.2	21.7	51.1	31.6
5	**26.0**	**21.7**	**51.0**	**31.7**

Table 5.12: Results for French speech using MOSES.

translator are used, which means less variability than using several translator. In addition, the output vocabulary is restricted to the task, which means that the difference between the translations of the 5-best set will not be very large in most of the cases. Hence, the limited changes that can be found among the options provided by the MOSES translator also lead to tighter variations in the results obtained for each experiment.

The described effect can also be observed in Table 5.13, where the results obtained with the 1-best decoding and the graph combination that optimizes the CER are compared to the strategy of processing each sentence of the 5-best separately and then taking the one that is given the best score by the SLU module.

In this Table it is shown that selecting a sentence among the ones in the 5-best set always outperforms the results obtained for the semantic measures when just the 1-best is taken. This means that there are other sentences in the set that are more semantically adequate than the 1-best (despite the BLEU obtained is lower). However, the separate processing and the graph combination achieve in most cases very similar results. It is due to the way MOSES works, in which performs an inference process with the training sentences in which their different structures may be mixed, but limited to the task-dependent output

			CER	FSER	WER	BLEU
English	Text	1-best	21.9	12.8	44.1	38.5
		Maximum score separately	21.3	**12.3**	**43.5**	30.9
		Graph combination (2-best)	**21.2**	13.3	**43.5**	**39.0**
	Speech	1-best	31.0	24.2	53.1	27.9
		Maximum score separately	**29.9**	**23.2**	**52.7**	23.5
		Graph combination (2-best)	30.6	25.0	**52.7**	**28.1**
French	Text	1-best	21.9	15.2	45.8	37.4
		Maximum score separately	**21.4**	14.8	**45.4**	30.0
		Graph combination (3-best)	21.6	**14.7**	45.8	**37.6**
	Speech	1-best	27.4	23.1	51.5	31.0
		Maximum score separately	26.4	22.0	51.3	26.0
		Graph combination (5-best)	**26.0**	**21.7**	**51.0**	**31.7**

Table 5.13: Results for text and speech in English and French selecting for each utterance the translation from the 5-best provided by MOSES that achieves the maximum score in the SLU module. These results are compared to the ones obtained using the 1-best translation and to the best results obtained using a combination of several translations by means of a graph of words.

vocabulary. Adding a combination mechanism of several translations by means of graphs provides a systematic and effective way to mix them and incorporate semantic knowledge in order to build the semantic structures that can be inferred from them.

Finally, Table 5.14 shows a comparison of the graph-approach to a CRF semantic decoder which analyzes the 1-best output from MOSES. The row labelled as 1-best+CRF shows the results of this system, and the numbers in parentheses show the difference to analyzing the same translation with the graph-based system. In this comparison, the CRF always outperforms in CER the results obtained by the graph-based system processing the 1-best. However, regarding FSER, in almost all the graphs surpass the CRF system. This implies that the CRF system does not identify the meaningful concepts as accurately as the graph-based system.

If the CRF are compared now to the best results obtained with the graph-based system (second row for each language), some disparity is observed. CRF work better for English, while in general the graphs work better for French. It can be then concluded that both approaches are comparable and that the graph-based approach outperforms the state-of-the-art CRF formalism for some of the settings.

		Text		Speech	
	SLU method	CER	FSER	CER	FSER
English	1-best+CRF	**19.3 (-2.6)**	**13.3 (+0.5)**	**26.8 (-4.2)**	**23.1 (-1.1)**
	Graphs	21.2	**13.3**	30.6	25.0
French	1-best+CRF	21.8 (-0.1)	16.1 (+0.9)	**24.7 (-2.7)**	23.5 (+0.4)
	Graphs	**21.6**	**14.7**	26.0	**21.7**

Table 5.14: Comparison of the results obtained using a CRF system on the 1-best output of MOSES and the best results obtained by the graph-based approach using MOSES. The numbers in parentheses are the variation with respect to the graph-based approach using just the 1-best translation. A negative variation indicates that the CRF approach obtains better results than the graph-based approach, and a positive variation indicates a better behaviour of the graph-based strategy.

5.4.3 Comparison between systems

In order to compare the two approaches to multilingual SLU that have been presented in this chapter, Tables 5.15 to 5.18 show the best results obtained by each of these systems. In both cases, the results refer to systems with the full working pipeline, it is, when the graph builder algorithm is used and the graph-based method is applied at the SLU stage. In all cases, the best results are obtained by the system that uses general-purpose web translators. However, as aforementioned, it is important to highlight that the training and development corpora for the MOSES system were very small compared to usual corpora for machine translation, and probably this had a negative effect on the results for this approach. As future work, a larger task-dependent corpora may be collected, but a more interesting idea would be to use some domain adaptation techniques for machine translation, such as interpolating a general-purpose corpus with a task-dependent one, with the goal of increasing the amount of data used, its quality, and its coverage for a SLU task.

Furthermore, in the case of the results using the general-purpose web translators and the graph-based method, they are in many cases better than the results obtained with the CRF approach and the 1-best output from MOSES (this is the best setting for the CRF method, see Table 5.8).

In conclusion, results show the usefulness of combining several translations from one or more translators by means of a graph that generalizes these translations, and the use of a SLU system that is able to exploit this combination by identifying semantic structures that

can be made of parts of different original sentences. This combination is more convenient if general-purpose translators are used, since it allows to better fuse in-vocabulary segments of sentences that may contain several out-of-vocabulary words, which generally are not useful for SLU purposes.

System	CER	FSER	WER	BLEU
general-purpose web translators (1110)	**19.2**	**10.6**	**42.5**	**41.1**
task-dependent MOSES translator (2-best)	21.2	13.3	43.5	39.0

Table 5.15: Best results obtained for English text for each of the approaches.

System	CER	FSER	WER	BLEU
general-purpose web translators (1110)	**29.9**	**25.0**	**51.6**	**31.9**
task-dependent MOSES translator (2-best)	30.6	**25.0**	52.7	28.1

Table 5.16: Best results obtained for English speech for each of the approaches.

System	CER	FSER	WER	BLEU
general-purpose web translators (1111)	**20.2**	**13.2**	**45.7**	**39.9**
task-dependent MOSES translator (3-best)	21.6	14.7	45.8	37.6

Table 5.17: Best results obtained for French text for each of the approaches.

System	CER	FSER	WER	BLEU
general-purpose web translators (1111)	**24.8**	**20.8**	52.3	**33.1**
task-dependent MOSES translator (5-best)	26.0	21.7	**51.0**	31.7

Table 5.18: Best results obtained for French speech for each of the approaches.

Chapter 6

Conclusions

This thesis has addressed the challenges of spoken language understanding. For this purpose, a graph representation of this problem has been developed and a complete SLU system based on graphs of linguistic units and statistical models has been built.

There are two main modules in the developed graph-based SLU system. The first module is a graph of words builder module that uses a grammatical inference algorithm to construct a graph structure from a set of sentences. The resulting graph generalizes the input set of sentences based on their syntactic structures. The second module is a graph-based semantic decoding module that uses stochastic models and works in two steps. The first step builds a graph of concepts by searching for the sub-paths in the graph of words that correspond to lexical realizations of the concepts of the task. To do this, a dynamic programming graph search algorithm that combines the information of the graph of words and the stochastic semantic models has been developed. The second step finds the best path in the graph of concepts by means of another dynamic programming algorithm by also using the information of another stochastic semantic model. Besides the graph of words built with the presented algorithm, the input to the SLU module can also be a word lattice generated by an ASR. The output of the semantic decoding module is the best sequence of concepts, the sequence of words contained in the graph of words that underlies the best sequence of concepts, and a segmentation of the sequence of words in terms of the concepts. The experimental evaluation of this SLU system has shown that the results achieved are competitive with other state-of-the-art approaches when a single sentence is provided as its

input. Furthermore, the performance of the system improves when several sentences are provided by means of a graph of words, also obtaining an enhancement in the quality of the speech transcriptions when semantic information is used.

In order to perform semantic decoding experiments in a multilingual environment, a brand new corpus was built for this work. This corpus is the multilingual DIHANA corpus, which is an extension of the Spanish DIHANA corpus to English and French. A low-cost semi-supervised approach was developed to obtain the training part of the corpus in the new languages by means of several general-purpose web translators. The development and test sets were manually translated by native speakers in order to better model real users. Also, a subset of the test sentences were uttered by native speakers.

The proposed graph-based approach to monolingual SLU has also been extended to a multilingual setting using a test-on-source approach. For this purpose, two speech translation techniques were developed. One technique uses several general-purpose web translators, and the other technique uses a task-dependent parallel corpus. In both cases, the set of translations obtained is used to build a graph of words in the same way as for the monolingual case. These graphs of words are then analyzed by the graph-based SLU method.

The experimental evaluation of this multilingual SLU approach was carried out using the multilingual DIHANA corpus in both English and French with both text and speech inputs. For the sake of comparison, two other systems were used, but taking single transcriptions from the translators instead of a graph of words. All the experiments showed the convenience of using several translations as the input of the SLU system. Both the measures that assess the quality of the semantic interpretation and those that determine the quality of the translations were improved when several translations were combined and the semantic information was used in order the most semantically convenient translation. This improvement was particularly important in the case of the system based on general-purpose web translators, where the best results even outperformed those achieved with the CRF system. In the case of the MOSES-based speech translation system, the results are comparable to those obtained using the 1-best translation and the CRF system. Overall, the best results were obtained when several translations from different translators were considered since the knowledge provided by each of them was combined, filtered, and analyzed by the graph-based SLU algorithm. Thus, it has been shown that the approach presented in this thesis is appropriate for monolingual and multilingual SLU.

Finally, there are several lines of future work that can be followed to further extend the work presented in this thesis. These include:

- the development and evaluation of other grammatical inference algorithms to determine if the performance of the SLU method can be further improved by exploring other combinations of translations and transcriptions.
- further research on developing discriminative models for SLU using a graph representation of the SLU problem.
- the evaluation of the test-on-source multilingual SLU method with other corpora and other less related pairs of languages, such as Greek - Italian and Turkish - Italian, for which the multilingual LUNA corpus can be used.
- exploring the possibility of using graphs for combining and improving the translations provided by several MT systems in order to obtain a training corpus for a train-on-target SLU system.

Bibliography

Allauzen, Cyril, Mehryar Mohri, and Brian Roark (2005). "A general weighted grammar library". In: *Implementation and Application of Automata*. Springer, pp. 23–34.

Allauzen, Cyril et al. (2007). "OpenFst: A general and efficient weighted finite-state transducer library". In: *Implementation and Application of Automata*. Springer, pp. 11–23.

Atrey, Pradeep K et al. (2010). "Multimodal fusion for multimedia analysis: a survey". In: *Multimedia systems* 16.6, pp. 345–379.

Bangalore, Srinivas, German Bordel, and Giuseppe Riccardi (2001). "Computing Consensus Translation from Multiple Machine Translation Systems". In: *In Proceedings of IEEE Automatic Speech Recognition and Understanding Workshop (ASRU-2001)*, pp. 351–354.

Barrón-Cedeño, Alberto, Parth Gupta, and Paolo Rosso (2013). "Methods for cross-language plagiarism detection". In: *Knowledge-Based Systems* 50, pp. 211–217.

Bayer, Ali Orkan and Giuseppe Riccardi (2012). "Joint language models for automatic speech recognition and understanding". In: *Spoken Language Technology Workshop (SLT), 2012 IEEE*. IEEE, pp. 199–203.

Bender, Oliver et al. (2003). "Comparison of alignment templates and maximum entropy models for natural language understanding". In: *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics-Volume 1*. Association for Computational Linguistics, pp. 11–18.

Benedí, José-Miguel et al. (2006). "Design and acquisition of a telephone spontaneous speech dialogue corpus in Spanish: DIHANA". In: *Proceedings of LREC 2006*. Genoa (Italy), pp. 1636–1639.

Bertoldi, Nicola and Marcello Federico (2005). "A new decoder for spoken language translation based on confusion networks". In: *Automatic Speech Recognition and Understanding, 2005 IEEE Workshop on*. IEEE, pp. 86–91.

Bertoldi, Nicola, Richard Zens, and Marcello Federico (2007). "Speech translation by confusion network decoding". In: *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*. Vol. 4. IEEE, pp. 1297–1300.

Bonneau-Maynard, H and S Rosset (2003). "A Semantic representation for spoken dialogs". In: *Proceedings of Eurospeech*, pp. 253–258.

Bonneau-Maynard, H. et al. (2005). "Semantic annotation of the French MEDIA dialog corpus". In: *Proc. of InterSpeech 2005*. Portugal, pp. 3457–3460.

Bonneau-Maynard, Hélene and Fabrice Lefèvre (2005). "A 2+1-level stochastic understanding model". In: *Automatic Speech Recognition and Understanding, 2005 IEEE Workshop on*. IEEE, pp. 256–261.

Bonneau-Maynard, Hélene, Matthieu Quignard, and Alexandre Denis (2009). "MEDIA: a semantically annotated corpus of task oriented dialogs in French". In: *Language Resources and Evaluation* 43.4, pp. 329–354.

Bonneau-Maynard, Hélène et al. (2006). "Results of the French Evalda-Media evaluation campaign for literal understanding". In: *5th International Conference on Language Resources and Evaluation (LREC2006)*.

Brill, Eric and Jun Wu (1998). "Classifier combination for improved lexical disambiguation". In: *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics-Volume 1*. Association for Computational Linguistics, pp. 191–195.

Calvo, Marcos et al. (2012a). "A methodology for obtaining concept graphs from word graphs". In: *10th International Workshop on Finite State Methods and Natural Language Processing*, pp. 75–79.

Calvo, Marcos et al. (2012b). "A multilingual SLU system based on semantic decoding of graphs of words". In: *Advances in Speech and Language Technologies for Iberian Languages*. Springer Berlin Heidelberg, pp. 158–167.

Calvo, Marcos et al. (2012c). "Un algoritmo para la comprensión automática del habla sobre grafos de palabras". In: *Procesamiento del lenguaje natural* 48, pp. 105–112.

Calvo, Marcos et al. (2013a). "Exploiting Multiple ASR Outputs for a Spoken Language Understanding Task". In: *Speech and Computer*. Springer International Publishing, pp. 138–145.

Calvo, Marcos et al. (2013b). "Exploiting multiple hypotheses for Multilingual Spoken Language Understanding". In: *CoNLL-2013*, pp. 193–201.

Calvo, Marcos et al. (2015). "Combining Several ASR Outputs in a Graph-Based SLU System". In: *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*. Vol. 9423, pp. 551–558.

Cettolo, Mauro, Anna Corazza, and Renato De Mori (1998). "Language portability of a speech understanding system". In: *Computer Speech & Language* 12.1, pp. 1–21.

Chen, Yun-Nung, William Yang Wang, and Alexander I Rudnicky (2015). "Learning Semantic Hierarchy with Distributed Representations for Unsupervised Spoken Language Understanding". In: *Sixteenth Annual Conference of the International Speech Communication Association*.

Chowdhury, Shammur Absar et al. (2014). "Cross-Language Transfer of Semantic Annotation via Targeted Crowdsourcing". In: *INTERSPEECH*, pp. 2108–2112.

Chowdhury, Shammur Absar et al. (2015). "Selection and Aggregation Techniques for Crowdsourced Semantic Annotation Task". In: *Sixteenth Annual Conference of the International Speech Communication Association*.

Cortes, Corinna, Patrick Haffner, and Mehryar Mohri (2003). "Lattice kernels for spoken-dialog classification". In: *Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03). 2003 IEEE International Conference on*. Vol. 1. IEEE, pp. I–628.

Daciuk, Jan et al. (2000). "Incremental construction of minimal acyclic finite-state automata". In: *Computational linguistics* 26.1, pp. 3–16.

De La Higuera, Colin (2005). "A bibliographical study of grammatical inference". In: *Pattern recognition* 38.9, pp. 1332–1348.

De Mori, Renato et al. (2007). "Spoken language understanding: a survey." In: *ASRU*, pp. 365–376.

Den Os, Els et al. (1999). "Overview of the ARISE project." In: *EUROSPEECH*. Citeseer.

Deoras, Anoop et al. (2012). "Joint Decoding for Speech Recognition and Semantic Tagging." In: *INTERSPEECH*.

Deoras, Anoop et al. (2013). "Joint Discriminative Decoding of Words and Semantic Tags for Spoken Language Understanding". In: *Audio, Speech, and Language Processing, IEEE Transactions on* 21.8, pp. 1612–1621.

Devillers, Laurence et al. (2004). "The French MEDIA/EVALDA Project: the Evaluation of the Understanding Capability of Spoken Language Dialogue Systems." In: *LREC*. Citeseer.

Dinarelli, Marco (2010). "Spoken Language Understanding: From Spoken Utterances to Semantic Structures". PhD thesis. University of Trento.

Dinarelli, Marco et al. (2009). "Annotating spoken dialogs: from speech segments to dialog acts and frame semantics". In: *Proceedings of the 2nd Workshop on Semantic Representation of Spoken Language*. Association for Computational Linguistics, pp. 34–41.

Dyer, Christopher, Smaranda Muresan, and Philip Resnik (2008). "Generalizing Word Lattice Translation". In: *ACL-08: HLT*, pp. 1012–1020.

D'Ulizia, Arianna, Fernando Ferri, and Patrizia Grifoni (2011). "A survey of grammatical inference methods for natural language learning". In: *Artificial Intelligence Review* 36.1, pp. 1–27.

Eskevich, Maria et al. (2015). "SAVA at MediaEval 2015: Search and anchoring in video archives". In: *Working notes of the MediaEval 2015 Workshop, Wurzen, Germany*.

Fiscus, Jonathan G (1997). "A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (ROVER)". In: *Automatic Speech Recognition and Understanding, 1997. Proceedings., 1997 IEEE Workshop on*. IEEE, pp. 347–354.

Florian, Radu et al. (2003). "Named entity recognition through classifier combination". In: *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*. Association for Computational Linguistics, pp. 168–171.

Fraser, Norman M and G Nigel Gilbert (1991). "Simulating speech systems". In: *Computer Speech & Language* 5.1, pp. 81–99.

Freitag, Markus, Matthias Huck, and Hermann Ney (2014). "Jane: Open Source Machine Translation System Combination". In: *Proc. of the Conf. of the European Chapter of the Assoc. for Computational Linguistics (EACL), Gothenburg, Sweden*, pp. 29–32.

Fu, King-Sun and Taylor L Booth (1975). "Grammatical inference: Introduction and survey-Part I". In: *IEEE Transactions on Systems, Man, and Cybernetics* 1.SMC-5, pp. 95–111.

García, Fernando et al. (2011). "An active learning approach for statistical spoken language understanding". In: *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*. Springer, pp. 565–572.

García, Fernando et al. (2012). "Combining multiple translation systems for spoken language understanding portability". In: *Spoken Language Technology Workshop (SLT), 2012 IEEE*. IEEE, pp. 194–198.

García, Fernando et al. (2014). "Obtaining parallel corpora for Multilingual Spoken Language Understanding tasks". In: *In Proceedings of the Iberspeech*. Las Palmas de Gran Canaria, pp. 208–215.

García, Fernando et al. (2015a). "Adaptive Training for Robust Spoken Language Understanding". In: *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*. Springer, pp. 519–526.

García, Fernando et al. (2015b). "Diseño y comparación de varias aproximaciones estadísticas a la Comprensión del Habla en dos tareas e idiomas distintos". In: *Sociedad Española para el Procesamiento del Lenguaje Natural* 55, pp. 31–38.

Glass, James et al. (1995). "Multilingual spoken-language understanding in the MIT Voyager system". In: *Speech communication* 17.1, pp. 1–18.

Gorin, Allen L, Giuseppe Riccardi, and Jeremy H Wright (1997). "How may I help you?" In: *Speech communication* 23.1, pp. 113–127.

Grefenstette, Gregory (2012). *Cross-language information retrieval*. Vol. 2. Springer Science & Business Media.

Griol, David (2007). "Desarrollo y Evaluación de Diferentes Metodologías para la Gestión Automática del Diálogo". PhD thesis. Universitat Politècnica de València.

Griol, David et al. (2006). "Development and evaluation of the DIHANA project dialog system". In: *Proc. of Interspeech-06 Satellite Workshop Dialogue on Dialogues. Multidisciplinary Evaluation of Advanced Speech-based Interactive Systems*, pp. 9–12.

Gupta, Narendra et al. (2006). "The AT&T spoken language understanding system". In: *Audio, Speech, and Language Processing, IEEE Transactions on* 14.1, pp. 213–222.

Haffner, Patrick (2006). "Scaling large margin classifiers for spoken language understanding". In: *Speech Communication* 48.3, pp. 239–261.

Hahn, Stefan et al. (2008). "A Comparison of Various Methods for Concept Tagging for Spoken Language Understanding." In: *LREC*, pp. 2947–2949.

Hahn, Stefan et al. (2009). "Optimizing CRFs for SLU tasks in various languages using modified training criteria." In: *INTERSPEECH*, pp. 2727–2730.

Hahn, Stefan et al. (2011). "Comparing stochastic approaches to spoken language understanding in multiple languages". In: *Audio, Speech, and Language Processing, IEEE Transactions on* 19.6, pp. 1569–1583.

Hakkani-Tür, D. et al. (2006). "Beyond ASR 1-best: Using word confusion networks in spoken language understanding". In: *Computer Speech & Language* 20.4, pp. 495–514.

Hakkani-Tür, Dilek and Giuseppe Riccardi (2003). "A general algorithm for word graph matrix decomposition". In: *Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03). 2003 IEEE International Conference on.* Vol. 1. IEEE, pp. 596–599.

Hazen, Timothy J, Stephanie Seneff, and Joseph Polifroni (2002). "Recognition confidence scoring and its use in speech understanding systems". In: *Computer Speech & Language* 16.1, pp. 49–67.

He, Xiaodong et al. (2013). "Multi-style adaptive training for robust cross-lingual spoken language understanding". In: *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on.* IEEE, pp. 8342–8346.

He, Yulan and Steve Young (2003). "Hidden vector state model for hierarchical semantic parsing". In: *Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03). 2003 IEEE International Conference on.* Vol. 1. IEEE, pp. 264 –268.

He, Yulan and Steve Young (2005). "Semantic processing using the hidden vector state model". In: *Computer speech & language* 19.1, pp. 85–106.

Hemphill, Charles T, John J Godfrey, and George R Doddington (1990). "The ATIS spoken language systems pilot corpus". In: *Proceedings of the DARPA speech and natural language workshop*, pp. 96–101.

Henderson, Matthew (2015). "Discriminative Methods for Statistical Spoken Dialogue Systems". PhD thesis. University of Cambridge.

Henderson, Matthew et al. (2012). "Discriminative spoken language understanding using word confusion networks". In: *Spoken Language Technology Workshop (SLT), 2012 IEEE*. IEEE, pp. 176–181.

Hinton, Geoffrey et al. (2012). "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups". In: *Signal Processing Magazine, IEEE* 29.6, pp. 82–97.

Issar, Sunil and Wayne Ward (1993). "CMU's robust spoken language understanding system". In: *Proceedings of Eurospeech*. Vol. 93.

Jabaian, Bassam, Laurent Besacier, and Fabrice Lefèvre (2010). "Investigating multiple approaches for SLU portability to a new language." In: *INTERSPEECH*, pp. 2502–2505.

— (2011). "Combination of stochastic understanding and machine translation systems for language portability of dialogue systems". In: *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*. IEEE, pp. 5612–5615.

Jabaian, Bassam, Laurent Besacier, and Fabrice Lefèvre (2013). "Comparison and combination of lightly supervised approaches for language portability of a spoken language understanding system". In: *Audio, Speech, and Language Processing, IEEE Transactions on* 21.3, pp. 636–648.

Jabaian, Bassam, Fabrice Lefèvre, and Laurent Besacier (2014). "A unified framework for translation and understanding allowing discriminative joint decoding for multilingual speech semantic interpretation". In: *Computer Speech & Language*.

Jeong, Minwoo and G Geunbae Lee (2008). "Triangular-chain conditional random fields". In: *Audio, Speech, and Language Processing, IEEE Transactions on* 16.7, pp. 1287–1302.

Jeong, Minwoo and Gary Geunbae Lee (2006). "Exploiting non-local features for spoken language understanding". In: *Proceedings of the COLING/ACL on Main conference poster sessions*. Association for Computational Linguistics, pp. 412–419.

— (2008). "Practical use of non-local features for statistical spoken language understanding". In: *Computer Speech & Language* 22.2, pp. 148–170.

Johnsen, Magne Hallstein et al. (2000). "Stochastic modeling of semantic content for use IN a spoken dialogue system." In: *INTERSPEECH*, pp. 218–221.

Khan, Omar Zia et al. (2015). "Hypotheses Ranking and State Tracking for a Multi-Domain Dialog System Using Multiple ASR Alternates". In: *Sixteenth Annual Conference of the International Speech Communication Association*.

Kittler, Josef et al. (1998). "On combining classifiers". In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 20.3, pp. 226–239.

Koehn, Philipp (2005). "Europarl: A parallel corpus for statistical machine translation". In: *MT summit*. Vol. 5. Citeseer, pp. 79–86.

Koehn, Philipp et al. (2007). "Moses: Open source toolkit for statistical machine translation". In: *Proceedings of the 45th annual meeting of the ACL on interactive poster and demonstration sessions*. Association for Computational Linguistics, pp. 177–180.

Kudo, Taku (2005). "CRF++: Yet another CRF toolkit". In: *Software available at http://taku910.github.io/crfpp/*.

Lafferty, J., A. McCallum, and F. Pereira (2001). "Conditional random fields: Probabilistic models for segmenting and labeling sequence data". In: *International Conference on Machine Learning*. Citeseer, pp. 282–289.

Laguna, Sergio et al. (2014). "A Multilingual Spoken Language Understanding System". In: *Proceedings of the Iberspeech*. Las Palmas de Gran Canaria, pp. 348–353.

Larkin, M. A. et al. (2007). "ClustalW and ClustalX version 2.0". In: *Bioinformatics* 23.21, pp. 2947–2948. ISSN: 1460-2059. DOI: 10.1093/bioinformatics/btm404.

Lefèvre, Fabrice (2006). "A DBN-based multi-level stochastic spoken language understanding system". In: *Spoken Language Technology Workshop, 2006. IEEE*. IEEE, pp. 78–81.

— (2007). "Dynamic bayesian networks and discriminative classifiers for multi-stage semantic interpretation". In: *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*. Vol. 4. IEEE, pp. 13–16.

Lefèvre, Fabrice, François Mairesse, and Steve Young (2010). "Cross-lingual spoken language understanding from unaligned data using discriminative classification models and machine translation." In: *INTERSPEECH*. Citeseer, pp. 78–81.

Lefèvre, Fabrice et al. (2012). "Leveraging study of robustness and portability of spoken language understanding systems across languages and domains: the PORTMEDIA corpora". In: *The International Conference on Language Resources and Evaluation*.

Lehnen, Patrick, Stefan Hahn, and Hermann Ney (2011). "N-Grams for Conditional Random Fields or a Failure-Transition (ϕ) Posterior for Acyclic FSTs". In: *Twelfth Annual Conference of the International Speech Communication Association*, pp. 1437–1440.

Lööf, Jonas, Christian Gollan, and Hermann Ney (2009). "Cross-language bootstrapping for unsupervised acoustic model training: rapid development of a Polish speech recognition system." In: *Interspeech*, pp. 88–91.

López-Cózar, Ramón, Zoraida Callejas, and Germán Montoro (2006). "DS-UCAT: A new multimodal dialogue system for an academic application". In: *Proc. of Interspeech'06-Satellite Workshop Dialogue on Dialogues, Multidisciplinary Evaluation of Advanced Speech-Based Interactive Systems, Pittsburgh, Pennsylvania, US*, pp. 47–50.

Macherey, Klaus, Oliver Bender, and Hermann Ney (2009). "Applications of statistical machine translation approaches to spoken language understanding". In: *Audio, Speech, and Language Processing, IEEE Transactions on* 17.4, pp. 803–818.

Magnini, Bernardo et al. (2004). "The multiple language question answering track at clef 2003". In: *Comparative Evaluation of Multilingual Information Access Systems*. Springer, pp. 471–486.

Mairesse, François et al. (2009). "Spoken language understanding from unaligned data using discriminative classification models". In: *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*. IEEE, pp. 4749–4752.

Marasek, Krzysztof and Ryszard Gubrynowicz (2008). "Design and Data Collection for Spoken Polish Dialogs Database." In: *LREC*, pp. 185–189.

McCallum, Andrew, Dayne Freitag, and Fernando Pereira (2000). "Maximum Entropy Markov Models for Information Extraction and Segmentation." In: *ICML*. Vol. 17, pp. 591–598.

Mesnil, Grégoire et al. (2013). "Investigation of recurrent-neural-network architectures and learning methods for spoken language understanding." In: *INTERSPEECH*, pp. 3771–3775.

Mihalcea, Rada and Dragomir Radev (2011). *Graph-based natural language processing and information retrieval*. Cambridge University Press.

Miller, Scott et al. (1996). "A fully statistical approach to natural language interfaces". In: *Proceedings of the 34th annual meeting on Association for Computational Linguistics*. Association for Computational Linguistics, pp. 55–61.

Misu, Teruhisa et al. (2012). "A bootstrapping approach for SLU portability to a new language by inducting unannotated user queries". In: *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*. IEEE, pp. 4961–4964.

Mohri, Mehryar, Fernando Pereira, and Michael Riley (2000). "The design principles of a weighted finite-state transducer library". In: *Theoretical Computer Science* 231.1, pp. 17–32.

Neumann, Günter and Bogdan Sacaleanu (2005). "Experiments on robust NL question interpretation and multi-layered document annotation for a cross–language question/answering system". In: *Multilingual Information Access for Text, Speech and Images*. Springer, pp. 411–422.

Nie, Jian-Yun (2010). "Cross-language information retrieval". In: *Synthesis Lectures on Human Language Technologies* 3.1, pp. 1–125.

Och, Franz Josef (2003). "Minimum error rate training in statistical machine translation". In: *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*. Association for Computational Linguistics, pp. 160–167.

Och, Franz Josef and Hermann Ney (2003). "A Systematic Comparison of Various Statistical Alignment Models". In: *Computational Linguistics* 29.1, pp. 19–51.

Ortega, Lucía et al. (2010). "A statistical segment-based approach for spoken language understanding." In: *INTERSPEECH*, pp. 1836–1839.

Papineni, Kishore et al. (2002). "BLEU: a method for automatic evaluation of machine translation". In: *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, pp. 311–318.

Pieraccini, Roberto and Esther Levin (1995). "A spontaneous-speech understanding system for database query applications". In: *Spoken Dialogue Systems-Theories and Applications*, pp. 85–88.

Pieraccini, Roberto, Esther Levin, and Chin-Hui Lee (1991). "Stochastic Representation of Conceptual Structure in the ATIS Task." In: *HLT*. 121–124.

Pieraccini, Roberto et al. (1992). "A speech understanding system based on statistical representation of semantics". In: *Acoustics, Speech, and Signal Processing, 1992. ICASSP-92., 1992 IEEE International Conference on*. Vol. 1. IEEE, pp. 193–196.

Pla, Ferran et al. (2001). "Language understanding using two-level stochastic models with POS and semantic units". In: *Text, Speech and Dialogue*. Springer, pp. 403–409.

Planells, Joaquin et al. (2012). "An Online Generated Transducer to Increase Dialog Manager Coverage." In: *INTERSPEECH*.

Planells, Joaquin et al. (2013). "A multi-domain dialog system to integrate heterogeneous spoken dialog systems." In: *INTERSPEECH*, pp. 1891–1895.

Poignant, Johann, Hervé Bredin, and Claude Barras (2015). "Multimodal person discovery in broadcast TV at MediaEval 2015". In: *Working notes of the MediaEval 2015 Workshop, Wurzen, Germany*.

Potthast, Martin et al. (2011). "Cross-language plagiarism detection". In: *Language Resources and Evaluation* 45.1, pp. 45–62.

Ramshaw, Lance A and Mitchell P Marcus (1999). "Text chunking using transformation-based learning". In: *Natural language processing using very large corpora*. Springer, pp. 157–176.

Raux, Antoine et al. (2005). "Let's go public! taking a spoken dialog system to the real world". In: *in Proc. of Interspeech 2005*.

Raymond, Christian and Giuseppe Riccardi (2007). "Generative and discriminative algorithms for spoken language understanding." In: *INTERSPEECH*, pp. 1605–1608.

Raymond, Christian et al. (2006). "On the use of finite state transducers for semantic interpretation". In: *Speech Communication* 48.3, pp. 288–304.

Riccardi, Giuseppe, Roberto Pieraccini, and Enrico Bocchieri (1996). "Stochastic automata for language modeling". In: *Computer Speech & Language* 10.4, pp. 265–293.

Rokach, Lior (2010). "Ensemble-based classifiers". In: *Artificial Intelligence Review* 33.1-2, pp. 1–39.

Ross, Arun and Anil Jain (2003). "Information fusion in biometrics". In: *Pattern recognition letters* 24.13, pp. 2115–2125.

Rudnicky, Alexander I et al. (1999). "Creating natural dialogs in the carnegie mellon communicator system." In: *Eurospeech*.

Ruta, Dymitr and Bogdan Gabrys (2000). "An overview of classifier fusion methods". In: *Computing and Information systems* 7.1, pp. 1–10.

Sanchis, Emilio et al. (2000). "Modelización de la comprensión mediante técnicas de aprendizaje automático". In: *Primeras Jornadas de Tecnologia del Habla (1JTH), Sevilla*.

Schultz, Tanja (2004). "Towards rapid language portability of speech processing systems". In: *Conference on Speech and Language Systems for Human Communication, Delhi, India*.

Schwartz, Richard et al. (1996). "Language understanding using hidden understanding models". In: *Spoken Language, 1996. ICSLP 96. Proceedings., Fourth International Conference on*. Vol. 2. IEEE, pp. 997–1000.

Schwenk, Holger and Jean-Luc Gauvain (2000). "Combining multiple speech recognizers using voting and language model information." In: *INTERSPEECH*, pp. 915–918.

Segarra, Encarna et al. (2002). "Extracting semantic information through automatic learning techniques". In: *International Journal of Pattern Recognition and Artificial Intelligence* 16.03, pp. 301–307.

Seneff, Stephanie (1989). "TINA: A probabilistic syntactic parser for speech understanding systems". In: *Proceedings of the workshop on Speech and Natural Language*. Association for Computational Linguistics, pp. 168–178.

— (1992). "TINA: A natural language system for spoken language applications". In: *Computational linguistics* 18.1, pp. 61–86.

Servan, Christophe et al. (2010). "On the use of machine translation for spoken language understanding portability". In: *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*. IEEE, pp. 5330–5333.

Stepanov, Evgeny et al. (2013). "Language style and domain adaptation for cross-language SLU porting". In: *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*. IEEE, pp. 144–149.

Stepanov, Evgeny A, Giuseppe Riccardi, and Ali Orkan Bayer (2014). "The Development of the Multilingual LUNA Corpus for Spoken Language System Porting". In: *The 9th edition of the Language Resources and Evaluation Conference (LREC 2014)*, pp. 2675–2678.

Stevenson, Andrew and James R Cordy (2014). "A survey of grammatical inference in software engineering". In: *Science of Computer Programming* 96, pp. 444–459.

Stolcke, Andreas et al. (2006). "Cross-domain and cross-language portability of acoustic features estimated by multilayer perceptrons". In: *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*. Vol. 1. IEEE, pp. I–I.

Suendermann, David et al. (2009). "Localization of speech recognition in spoken dialog systems: how machine translation can make our lives easier." In: *INTERSPEECH*, pp. 1475–1478.

Svec, Jan, Pavel Ircing, and Lubos Smidl (2013). "Semantic entity detection from multiple ASR hypotheses within the WFST framework". In: *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*. IEEE, pp. 84–89.

Svec, Jan, Lubos Smidl, and Pavel Ircing (2013). "Hierarchical discriminative model for spoken language understanding". In: *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, pp. 8322–8326.

Svec, Jan et al. (2015). "Word-semantic lattices for spoken language understanding". In: *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*. IEEE, pp. 5266–5270.

Torres, Inés and Amparo Varona (2001). "k-TSS language models in speech recognition systems". In: *Computer Speech & Language* 15.2, pp. 127–149.

Tur, Gokhan and Renato De Mori (2011). *Spoken language understanding: Systems for extracting semantic information from speech*. John Wiley & Sons.

Tur, Gokhan, Anoop Deoras, and Dilek Hakkani-Tur (2013). "Semantic Parsing Using Word Confusion Networks With Conditional Random Fields". In: *Proc. of the INTERSPEECH*.

Tur, Gokhan et al. (2002). "Improving spoken language understanding using word confusion networks." In: *Interspeech*. Citeseer.

Tur, Gokhan et al. (2012). "Exploiting the Semantic Web for Unsupervised Natural Language Semantic Parsing." In: *INTERSPEECH*.

Versteegh, Maarten et al. (2015). "The zero resource speech challenge 2015". In: *Proc. of Interspeech*.

Vukotic, Vedran, Christian Raymond, and Guillaume Gravier (2015). "Is it time to switch to Word Embedding and Recurrent Neural Networks for Spoken Language Understanding?" In: *To appear at Proceedings of INTERSPEECH 2015*.

Walker, Marilyn A, Rebecca Passonneau, and Julie E Boland (2001). "Quantitative and qualitative evaluation of DARPA Communicator spoken dialogue systems". In: *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, pp. 515–522.

Wang, Lusheng and Tao Jiang (1994). "On the complexity of multiple sequence alignment". In: *Journal of computational biology* 1.4, pp. 337–348.

Wang, Ye-Yi and Alex Acero (2006). "Discriminative models for spoken language understanding." In: *INTERSPEECH*, pp. 1766–1769.

Xu, Lei, Adam Krzyżak, and Ching Y Suen (1992). "Methods of combining multiple classifiers and their applications to handwriting recognition". In: *Systems, man and cybernetics, IEEE transactions on* 22.3, pp. 418–435.

Yao, Kaisheng et al. (2013). "Recurrent neural networks for language understanding." In: *INTERSPEECH*, pp. 2524–2528.

Yao, Kaisheng et al. (2014). "Spoken language understanding using long short-term memory neural networks". In: *Spoken Language Technology Workshop (SLT), 2014 IEEE*. IEEE, pp. 189–194.

Young, Steve et al. (1997). *The HTK book*. Vol. 2. Entropic Cambridge Research Laboratory Cambridge.

Zamora-Martínez, Francisco et al. (2012). "Cache neural network language models based on long-distance dependencies for a spoken dialog system". In: *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*. IEEE, pp. 4993–4996.

Zue, Victor et al. (1992). "The VOYAGER speech understanding system: a progress report". In: *Proc. DARPA Speech and Natural Language Workshop*. Springer, pp. 179–189.

Zue, Victor et al. (1996). "Multilingual human-computer interactions: From information access to language learning". In: *Spoken Language, 1996. ICSLP 96. Proceedings., Fourth International Conference on*. Vol. 4. IEEE, pp. 2207–2210.

Zue, Victor et al. (2000). "JUPITER: a telephone-based conversational interface for weather information". In: *Speech and Audio Processing, IEEE Transactions on* 8.1, pp. 85–96.