Fundamenta Informaticae 131 (2014) 1–16
DOI 10.3233/FI-2014-980
IOS Press

Natural Language Understanding and Prediction: from Formal Grammars to Large Scale Machine Learning

Nicolae Duta
New England Research and Development Center
Microsoft, Cambridge, USA
niduta@microsoft.com

Abstract. Scientists have long dreamed of creating machines humans could interact with by voice. Although one no longer believes Turing's prophecy that machines will be able to converse like humans in the near future, real progress has been made in the voice and text-based human-machine interaction. This paper is a light introduction and survey of some deployed natural language systems and technologies and their historical evolution. We review two fundamental problems involving natural language: the language prediction problem and the language understanding problem. While describing in detail all these technologies is beyond our scope, we do comment on some aspects less discussed in the literature such as language prediction using huge models and semantic labeling using Marcus contextual grammars.

Keywords: Natural language understanding, language modeling, language prediction

1. Introduction

Scientists have long dreamed of creating machines humans could interact with by voice. In his most cited paper published in 1950, *Computing machinery and intelligence* Turing predicted that "at the end of the century the use of words and general educated opinion will have altered so much that one will be able to speak of machines thinking without expecting to be contradicted" [46]. "Thinking machines" involve multiple capabilities: recognizing the words which are said, understanding their meaning and being able to produce a meaningful reaction (e.g, answer a question which may imply reasoning in addition to simply querying a fact database, perform an action/transaction, etc).

Although, after several decades of research, one no longer believes Turing's prophecy that machines will be able to converse like humans in the near future, real progress has been made in the voice and

Address for correspondence: Microsoft NERD Center, One Memorial Drive, Cambridge, MA, 02142, USA text-based human-machine interaction. From a theoretical viewpoint, modern computational linguistics started in the late 1950s when Noam Chomsky introduced the theory of generative grammars which aimed at producing a set of rules that correctly predict which combinations of words form grammatical sentences [15]. The first practical attempts at natural language understanding by a computer were at MIT and Stanford in the 1960s: Daniel Bobrow's STUDENT system which used natural language input to solve algebra word problems and Joseph Weizenbaum's ELIZA system that could carry a dialog on many topics (although it did not have a real understanding of the language)[1].

However, the early systems were only using written text input; it would take two more decades of research until Automatic Speech Recognition (ASR) could allow for voice input. Throughout 1990-2000s, the Defense Advanced Research Projects Agency (DARPA) in the United States conducted several programs to advance the state of the art in ASR, spoken dialog and information extraction from automatically recognized text (ATIS 1990-1994, Hub4 1995-1999, Communicator 1999-2002, Ears 2002-2005, Gale 2005-2010) [52]. These scientific advances have also been sustained by the introduction and exponential growth of the *World Wide Web* and by the huge increase in computing power and miniaturization that led to the today's proliferation of smartphones.

This paper is a light introduction and survey of some of the deployed natural language systems and technologies and their historical evolution. We review two fundamental problems involving natural language: the language prediction problem and the language understanding problem. While describing in detail all these technologies is beyond our scope, we do comment on some aspects less discussed in the literature such as language prediction using huge models and semantic labeling using Marcus contextual grammars.

2. Natural Language Prediction

Language prediction is defined as the ability to predict which words naturally follow a given word sequence. It is generally assumed that natural languages are governed by a probability distribution on word sequences and the language prediction (actually called *Statistical Language Modeling*) models are trying to derive a good estimate of this distribution [4].

Language modeling/prediction has started as a part of the Automated Speech Recognition research effort and is now extensively used in most systems which convert some form of signal into text using a Bayesian approach:

- Automated Speech Recognition (ASR) for acoustic to text mappings [25]
- Optical Character Recognition (OCR) and Handwriting recognition which map document images into text [32][38]
- Automated Machine Translation (AMT) which maps text written in one language into text written in a different language [28]
- Spelling correction systems which map incorrectly spelled text into the correct form [6]
- Word completion and prediction systems (predict following letters in a word or words in a sms/email message considering context and previous user behavior) [9]

[1]A detailed historical perspective can be found in [45]

These systems are all trying to find the text sentence which maximizes the posterior probability $P(Sentence|Signal)$ which according to Bayes rule can be written as

$$P(Sentence|Signal) = P(Signal|Sentence) \times P(Sentence)/P(Signal) \tag{1}$$

$P(Signal|Sentence)$ is the underlying signal model based on acoustic, visual, translational, etc cues while $P(Sentence)$ describes the likelihood of a given sentence in a language.

Since a natural language like English has a lexicon of the order of 10^6 words, it is not possible to directly estimate $P(Sentence)$ for all sentences. Early ASR systems have restricted their language models to the set of sentences appearing in a training set. The drawback was that the system could only output one of the sentences it had seen in training no matter what the user said. Though the system had the ability to reject an input (not produce a text output if $P(Sentence|Signal)$ was too low), that was not very helpful in a practical system that had to deal with unconstrained speech.

Several techniques have been proposed for estimating $P(S)$ for every sentence $S = \{w_1, w_2...w_m\}$ (w_i are the sentence words in the order they are spoken) [4][40]; currently the most widely used models are based on decomposing $P(w_1, w_2...w_m)$ into a product of conditional probabilities

$$P(w_1, w_2...w_m) = P(w_m|w_{m-1}...w_1) \times P(w_{m-1}|w_{m-2}...w_1) \times P(w_1). \tag{2}$$

Since it was still impractical to directly estimate $P(w_1, w_2, ..., w_m)$ for long word histories, one assumed that words far away in the history of a target word do not have a large influence. That is, the word sequence $w_1, w_2, ..., w_m$ behaves like a Markov chain of some order n. Therefore one only needs to estimate the statistical distributions of n consecutive word sequences called *n-grams*. According to these models, the probability of a sentence can be decomposed into a product of conditional *n*-gram probabilities. Although counterintuitive, *n*-gram models take no advantage of the syntactic or semantic structure of the sentences they model.

However, if we are using the Maximum Likelihood (ML) estimate for

$$P_{ML}(w_n|w_{n-1}, ..., w_1) = Count(w_n, ..., w_1)/Count(w_{n-1}, ..., w_1) \tag{3}$$

we are facing the issue of assigning a null probability to those *n*-grams not seen in the training data. Even with a training corpus in excess of a few billion words (that's about the size of all newspaper text published in the US in the 1990s) there are still 10-20% valid 3-grams which have not been seen before (last row of Table 1). To properly handle them, one applies a technique called interpolated discounting (also called *smoothing*): set aside a part of the probability mass to account for unseen events and recursively interpolate longer history probabilities with shorter history probabilities:

$$P(w_i|w_{i-1}, w_{i-2}) = P_{ML}(w_i|w_{i-1}, w_{i-2}) \times \alpha(w_i, w_{i-1}, w_{i-2}) + \beta(w_{i-1}, w_{i-2}) \times P(w_i|w_{i-1}) \tag{4}$$

where α and β are called *smoothing functions* and model the amount of the probability mass that is left aside for unseen *n*-grams. To maintain a probability model we need it to sum to 1 over w_i:

$$\beta(w_{i-1}, w_{i-2}) = 1 - \sum_{w_i} P_{ML}(w_i|w_{i-1}, w_{i-2}) \times \alpha(w_i, w_{i-1}, w_{i-2}) \tag{5}$$

A large body of language modeling research in the 1990s has focused on finding suitable values for α and β. Two popular choices are called Witten-Bell [51] and Kneser-Ney [27] discounting:

$$Witten - Bell\ discounting \qquad Kneser - Ney\ discounting \tag{6}$$

$$\alpha = \quad \frac{Count(.|w_{i-1}, w_{i-2})}{Uniq(.|w_{i-1}, w_{i-2}) + Count(.|w_{i-1}, w_{i-2})} \qquad 1 - \frac{D(Count(w_i|w_{i-1}, w_{i-2}))}{Count(w_i|w_{i-1}, w_{i-2})} \tag{7}$$

$$\beta = \quad \frac{Uniq(.|w_{i-1}, w_{i-2})}{Uniq(.|w_{i-1}, w_{i-2}) + Count(.|w_{i-1}, w_{i-2})} \qquad \frac{\sum\limits_{w_i} D(Count(w_i|w_{i-1}, w_{i-2}))}{Count(w_i|w_{i-1}, w_{i-2})} \tag{8}$$

Starting in the early 2000s, the proliferation of documents posted on the internet generated a potentially huge LM training set. However, internet scraped text could not be directly used for LM training since: (i) Almost all of it was out of domain for the systems built at the time[2]. (ii) The computational resources (memory and computing speed) were not sufficient to accommodate the huge number of resulting *n*-grams (a 5 billion-word newspaper corpus generates about 0.8B unique 3-grams and 1.5B unique 4-grams).

Multiple directions of research started to address these issues. One of them was LM pruning: some *n*-grams considered not too informative were discarded (although after being used in computing global statistics of the data). The simplest pruning technique is to discard the least frequent, higher-order *n*-grams which one may assume are not statistically significant. A more sophisticated technique is entropy pruning which considers the relative entropy between the original and the pruned model [44]. However, there appears to be a complex interaction between the pruning method/parameters and the type of discounting used in training the model and that can impact the speech recognition accuracy by as much as 10% [12].

A second research direction was to redesign the LM estimation toolkits and speech recognition pipelines to accommodate all *n*-grams seen in the data. It is important to know the difference between *n*-grams that are unobserved because they are rare and those that are impossible[3]. As shown in Table 1, keeping one billion 3-4 grams in the LM reduces the Word Error Rate (WER) by about 6% in the Broadcast News recognition domain [20]. Although received with skepticism in the academia [11], this direction (along with a distributed data processing framework like Map-Reduce [16]) largely contributed to the recent success of the Google ASR system [13].

For many speech recognition applications (e.g. conversational speech) sufficient in-domain language data has not always been available and a solution was found to be the use additional out-of-domain data (especially internet scraped). Unfortunately, a simple mix of two (different in nature) corpora does not usually result in a better LM and a successful mixing strategy is often regarded as an art. Therefore, a third area of research has focused on combining in-domain with out-of-domain data or even bootstrapping an in-domain LM only using out-of-domain data [8][19].

While this is still an active research area we would like to point out two interesting phenomena. The first is that the colloquial forms of some languages like Arabic and their literary counterparts (e.g. the

[2]*n-gram* models are very sensitive to changes in the style, topic or genre of the text on which they are trained (called in-domain data) [40].

[3]An analysis of the text currently used in sms messages and twitter postings shows that almost everything is now possible due to word mispelling, abbreviation and lack of syntactic structure

Table 1. The effects of LM prunning on the English broadcast news task [20]

.

$LMOrder$	$LMsize$ [4-grams,3-grams]	$HitRates$ [4-grams,3-grams]	WER
3	[0, 36M]	[0, 76%]	12.6%
3	[0, 305M]	[0, 84%]	12.1%
4	[40M, 36M]	[49%, 76%]	12.1%
4	[710M, 305M]	[61%, 84%]	11.8%

Modern Standard Arabic-MSA used in newspaper articles and TV broadcasts) although have the same word lexicon, share very few of the higher order *n*-grams (see Table 2). That means that published texts and TV transcripts are not effective for training a conversational LM [26].

Table 2. Vocabulary coverage and 3-gram hit rates for LMs based on the Arabic Conversational (150K words), Broadcast News (300M words) and Conversational + BN data

LM training data	Vocabulary coverage	3-gram Hit Rate
Conversational alone	90.6%	20%
Broadcast News	89.5%	4%
Conversational + News	96.6%	21%

The second phenomenon is that even though there may still be a significant accuracy gap between speech recognition using a fully in-domain LM and that using a bootstrapped LM, the semantics of the recognized sentence may be far less impacted. That is, one can still figure out the semantic intent of a sentence even when some of the words are misrecognized [19][47].

Finally, we would like to mention the latest trends in Language Modeling. *Discriminative language models* (DLMs) [14] aim at directly optimizing word error rate by rewarding features that appear in low error hypotheses and penalizing features in misrecognized hypotheses. Since the estimation of discriminative LMs is computationally more intensive than regular *n*-gram LM one has to use distributed learning algorithms and supporting parallel computing infrastructure [16]. *Neural network language models* embed words in a continuous space in which probability estimation is performed using neural networks (feed-forward or recurrent, very recent work is based on multiple hidden layer networks called *deep networks* [2]). The expectation is that, with proper training of the word embedding, words that are semantically or gramatically related will be mapped to similar locations in the continuous space. Because the probability estimates are smooth functions of the continuous word representations, a small change in the features results in a small change in the probability estimation and NNLM may achieve better generalization for unseen *n*-grams.

3. Natural Language Understanding

3.1. Brief history

During the last couple of decades there has been a tremendous growth of deployed voice driven language understanding systems; however mostly designed for limited domains. At first, these systems were able to recognize and interpret (through fixed grammars) some predetermined phrases and named entities like locations or business names. Most popular were the Directory Assistance (DA) systems built by TellMe, Phonetic Systems/Nuance, BBN, Jingle, Google, etc.

Later on, the ASR technology started to support constrained digit sequences (dates, phone numbers, credit card and bank account numbers) and form filling directed dialog systems were designed for tasks like flight reservation. In such systems, users are asked to provide answers to what the system has asked for, which often consists of a single piece of semantic information. Directed dialog systems evolved into mixed-initiative systems where both users and the system can control the dialog flow and which allowed users to provide more semantic information in a single utterance and in any sequence they choose. The language understanding task became higher resolution with more semantic entities in need to be identified, segmented and normalized.

The DARPA Airline Travel Information System (ATIS) project [3] was initiated in the 1990s for the flight information domain. Users provide some flight attributes like departure and destination cities, dates, etc. However there were no constraints on how the information could be expressed. That is, users could say either “I need a flight reservation from Boston to Miami leaving tomorrow and returning in two weeks” or “Please show me the flight to Miami departing Boston tomorrow”. One can notice that beyond this freedom of expression there is a clear semantic structure with crisp and unambiguous semantic entities like Departure/Arrival Cities/Date/Times. These entities, known as “semantic slots” or “frame elements” are considered to be part of a set of templates (semantic frames) which represent the structure of the semantic space. The language understanding component in a frame-based system has to choose the correct semantic frame for an utterance and to segment and normalize the associated semantic slots. For example, the “Departure Date” slot expressed as the word “tomorrow” has to be normalized to something like “03/11/2013” in order to be useful for searching a flight database. Most ATIS systems employed either a statistical classification approach (those coming from the speech processing community) such as AT&T’s CHRONUS [37] and BBN’s hidden understanding models [30] or a knowledge-based approach (mostly from the computational linguistics community) such as the MIT’s TINA [42], CMU’s Phoenix [50], and SRI’s Gemini [17].

TINA [42] is basically a context-free grammar converted to a probabilistic network and implements a seamless interface between syntax and semantics. The initially bootstrapped context-free grammar is built from a set of training sentences where each sentence is translated by hand into a list of the rules invoked to parse it. The rule set is converted to a form that merges common elements on the right-hand side (RHS) of all rules sharing the same left-hand side (LHS). Elements on the LHS become parent nodes in a family tree. Through example sentences, they acquire knowledge of who their children are and how they can interconnect. The injection of domain-dependent semantics is done by replacing the low-level syntactic non-terminals with semantic non-terminals. For example, the syntactic rule-based derivation

SUBJECT => NOUN_PHRASE => ARTICLE NOUN => the Hyatt

is replaced by the semantic derivation [48]

SUBJECT => ARTICLE PLACE => ARTICLE HOTEL => the Hyatt

A main limitation of the knowledge-based systems is that the grammar design process is tedious, slow and requires a lot of expertise. The semantic space partition into semantic frames may be subjective and the set of slots for a frame are imposed in a top-down fashion rather than extracted from data. Therefore some natural language sentences may not be well modeled in this framework.

At the other end of the semantic spectrum are systems which only need to extract the sentence intent without other semantic entities. An example of such systems are the *Call Routers* whose goal is to automatically route a telephone query from a customer to the appropriate set of agents based on a brief spoken description of the problem. Call routers are nowadays deployed in most of the large call centers because they reduce queue time and call duration, thus saving money and improving customer satisfaction by promptly connecting the customer to the right service representative. These systems remove the constraints on what a user can say but at the expense of limiting the target semantic space. That is, call routers are specifically built for business verticals (e.g. telecommunication, government, utility companies) and are only designed to detect the kinds of semantic intents specific to that vertical (e.g. a telecommunication provider may allow a customer to perform one of several actions: canceling some service, resolving a billing issue, paying a bill, adding a new telephone line, etc).

Well known call routing systems are the AT&T How may I help you? (HMIHY) [22] and the BBN Call director [33]. The users are greeted by an open-ended prompt like How May I Help You?, which encourages them to speak naturally. To find the meaning of a human utterance in a call routing system, the caller's speech is first translated into a text string by an ASR system and the text is then fed into a NLU component called Router. The NLU task is modeled as a statistical classification problem: the text corresponding to an utterance is assigned to one or more of a set of predefined user intents (routes).

3.2. Current NLU architecture

The explosion of mobile computing power that came with the smartphones allowed the development of more sophisticated NLU systems that could handle combinations of many user intents along with the associated named entity extraction. There is now a proliferation of more complex, dialog-based voice search systems and mobile personal assistants that are configured to understand and perform several tasks [18]. Each task may have different sets of semantic entities that can be formulated and uttered differently by different users. The NLU goal in such systems is to also identify which task the user would like to perform (usually called *user intent*).

A modern client-server voice-based transactional system including dialog is depicted in Fig. 1 (see also [49], [23]). A user opens a client application on his phone and utters a sentence (e.g. query or command). The client sends the acoustic signal to the server system where it is first converted into text by the ASR module. Next, a NLU module extracts the semantic information from this text. A popular approach is top-down hierarchical meaning extraction. A semantic domain classifier can be used to determine which part of the semantic space a query belongs to. For example, the query "I need a table for two at the closest Bertucci's restaurant for tomorrow" belongs to the "Restaurant" domain. Then, using domain dependent models, a second classifier finds the query intent (what the user asks for). In our example, the intent is "Restaurant reservation". Finally using domain and intent dependent models, one segments the semantic slots (basic semantic entities) associated with the given domain and intent which have been specified by the user. In our case, the following slots appear and can be extracted from the sentence: (i) Restaurant name = "Bertucci's" (ii) Reservation date = "tomorrow", (iii) Party size = "two" and (iv) Restaurant location = "closest". After that, a normalizer translates each slot value into a form that

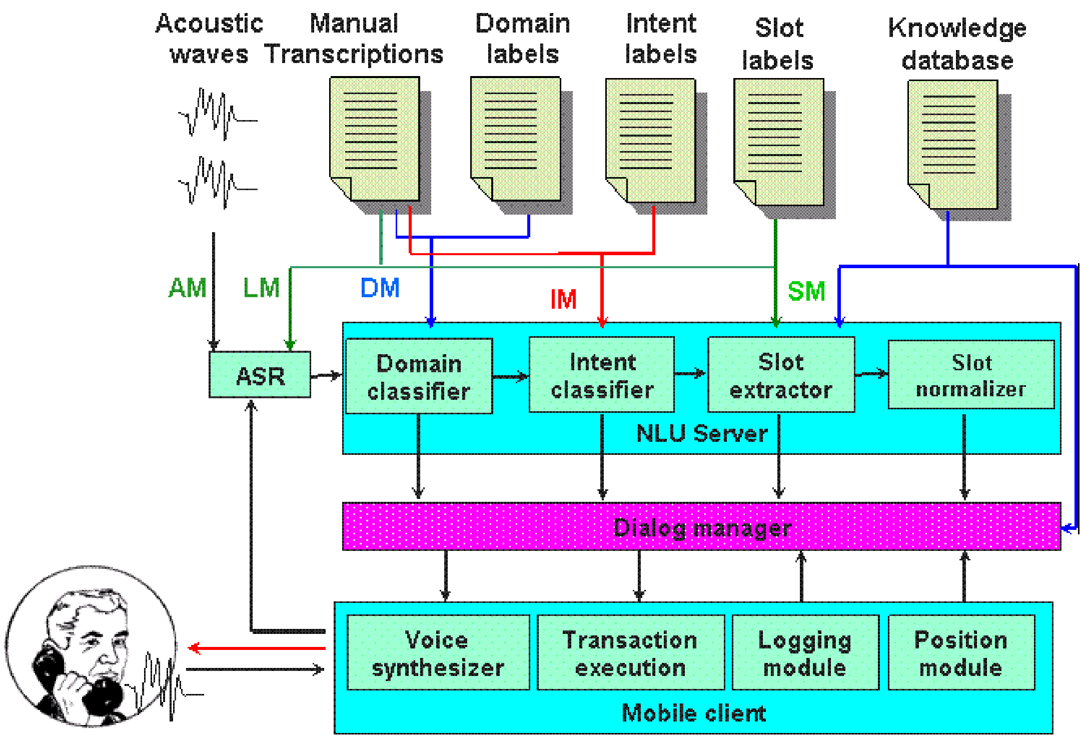

Figure 1. The architecture of a client-server voice-based transactional system including dialog.

can be used to query a knowledge-database. For our query, we could get the following slot normalized values: (i) Restaurant name = "Bertucci's pizzeria" (ii) Reservation date = "03/15/2013" and (iii) Party size = "2". The query domain, intent and normalized slot values are further sent out to a *dialog manager* which has detailed information about the domain and determines the next system action. In our case, the dialog manager asks the client application to provide the current user location[4], then it interrogates a business database to find the closest Bertucci's restaurant and finally detects that a restaurant reservation also requires time information in order to be fulfilled. Therefore, it will issue back to the user a question regarding the reservation time. The dialog manager produces the question as text, but that is fed into a speech synthesizer and turned into an audio signal which is played back to the user. Let's assume the user answers "Hum let's say six in the evening". The NLU system now detects a single semantic slot Time = "six in the evening" which is normalized as Time = "6 pm" and sent to the dialog manager along with "Unknown" domain and intent. The dialog manager, which also keeps track of the dialog states (possible using a stack), knows that this is the missing piece of information from a previous query and it can now take action on the query. Some systems send back to the user the parsed information asking for confirmation: "Ok, I'll make a reservation for two at Bertucci's on Main street for March 15th 2013 at 6pm. Is that correct?" If the user agrees, the Execution unit sends all the information to a restaurant reservation service/web site which performs the actual reservation.

One can easily notice that the dialog system architecture in Fig. 1 generalizes all systems built in the past. The DA systems had no client application (at the time users were using landlines), dialog manager

[4]either from the internal GPS or from the wireless provider signal triangulation

(there was a single-shot query which was automatically routed to a human agent if the system returned a low confidence), no domain or intent classifiers (the system's goal was only to return the phone number of a certain business or individual). They only had a primitive slot extractor (either business name or location, though often they were asked for separately) and normalizer. The directed dialog systems added a dialog manager and a small number of fixed intents often specified as a single piece of information[5]. On the other hand, the call routers added an intent classifier (with a number of intents ranging from a few tens to a few hundreds) and a very small number of slots.

From a linguistic viewpoint, these systems could be characterized by the following four criteria (see Table 3 and [49]): the naturalness and the size of the space of input sentences, the resolution of the target semantic representation and the size of the target semantic space. The systems have evolved from a low naturalness, input space, semantic resolution and space size (directed dialog) to medium-high naturalness, large input space, high semantic resolution and space size (today's voice transaction systems).

Table 3. Comparison of several NLU systems with respect to the characteristics of the input utterance space and the output semantic space (adapted from [49])

NLU system	User input	utterances	Target semantic	representation
	Naturalness	Input space	Resolution	Semantic space
Directed dialog	Low	Small	Low	Small
DA	Low	Large	Low	Small
Mixed initiative	Low-medium	Small	High	Small
Call routing	High	Medium	Low	Small
Voice search & personal assistant	Medium-high	Large	High	Large

One important phenomenon is that the text which modern systems are attempting to understand obeys less and less the syntax rules of the language. Spoken language often contains dysfluencies, restarts and recognition errors while written text may be highly abbreviated and/or truncated. For example, an SMS line may be "he like u" while a speech recognized sentence may be "by movie uh kings speech" (the spoken query was "buy movie king's speech").

3.3. Semantic data annotation

As previously mentioned, modern NLU systems often consist of sets of text classifiers which extract various types of semantic information: query domain, query intent, semantic slots and/or other attributes of the domain (facets) which sometimes may only be mentioned implicitly (e.g. the fragment "which make me laugh" in the query "find movies which make me laugh" should be interpreted as a movie genre and one may need some sort of logic reasoning for extracting these mappings [10]).

These classifiers need to be trained on large amounts of data in which the semantic entities of interest are manually annotated. As shown on top of Fig. 1 several sets of manual annotations are necessary: (i)

[5]The system could have asked: "What transaction would you like to perform: Flight information, reservation, cancellation, other?"

Speech transcriptions (a textual form of the user spoken utterances) (ii) Semantic domain and/or intent annotations and (iii) Semantic slot annotations. Although one tries to carefully annotate the data, the references produced by different human annotators are not identical. Sometimes that is due to annotator fatigue but most of the time there is a subjective component especially for the semantic annotations. The inter-annotator disagreement may be 6% for speech transcription [21] but it can get much higher when semantics is involved.

Therefore one may try to automate parts of the data annotation process. Semantic slot annotation could be done using Marcus contextual grammars [29][36] which have been theoretically studied for a long time (a brief introduction is given in the Appendix). We will show here an example of how to construct and use such a grammar. Let's assume the vocabulary V is the set of English words and the starting language A over V is the set of sentences a human might use for interacting with an NLU system as described in Section 3.2. The set of selectors correspond to the semantic entities we would like to label and the set of contexts contain the English word we would like to label them with. Let's say S_1 is the set of restaurant names, S_2 is the set of location names and C_1, C_2 are their corresponding semantic labels:

$$S_1 = \{McDonald's, Boston\ \ Market, ...\}, \quad C_1 = \{(< Restaurant >, < /Restaurant >)\},$$
$$S_2 = \{Boston, Cambridge, ...\}, \quad C_2 = \{(< Location >, < /Location >)\}$$

and so on. A possible derivation in this grammar is

Find me a McDonald's in Boston =>
Find me a <*Restaurant*>McDonald's</*Restaurant*> in Boston =>
Find me a <*Restaurant*>McDonald's</*Restaurant*> in <*Location*>Boston</*Location*>

In order to generate correct annotations, we require the derivations to be in maximal global mode. That is, at each derivation step, the word selector x is maximal with respect to all selectors $S_1, ..., S_n$. That is enforced if we always label first the longest semantic entity that could be labeled. The resulting annotated sentence obeys Occam's razor[6] (annotates as many words as possible with as few labels as possible) and is most of the time correct. A simple example is

Find me a Boston Market in Cambridge =>
Find me a <*Restaurant*>Boston Market</*Restaurant*> in Cambridge =>
Find me a <*Restaurant*>Boston Market</*Restaurant*> in <*Location*>Cambridge</*Location*>

If the derivation is not in the maximum global mode one could get:

Find me a Boston Market in Cambridge =>
Find me a <*Location*>Boston</*Location*> Market in Cambridge =>
Find me a <*Location*>Boston</*Location*> Market in <*Location*>Cambridge</*Location*>

which is obviously incorrect.

In [29], the finite and regular families of selectors are investigated. Although in practical systems the vocabulary, starting axioms, selectors and contexts are all finite, the case where the selectors are generated by a context sensitive mechanism is of high interest. That is because one name entity may belong to multiple semantic classes. For example the word "Eagles" belongs to MusicBand, SportsTeam and Bird classes. For such ambiguous cases, the set of selectors must also contain some contextual words to disambiguate the semantic class. As such, the word "Eagles" should appear in fragments like

[6]For a mathematically formalized version of Occam's razor see Ray Solomonoff's theory of universal inductive inference [43]

"Eagles songs" in MusicBand, "Eagles scores" in SportsTeam and "Eagle food" in Bird. The problem becomes even harder when the sets of context sensitive selectors have to be automatically extracted from un-annotated data.

An easy way to implement semantic annotation with a Marcus contextual grammar is by using Finite State Transducer technology [3] [31]. In the Xerox FST toolkit language [3], the grammar shown above can be written as:

```
define Location [{Boston}|{Cambridge}] EndTag(Location); # Selector Location
define Restaurant [{McDonald's}|{Boston Market}] EndTag(Restaurant); # Selector Restaurant
regex Location | Restaurant; # Grammar definition with implicit vocabulary
```

and the annotated output produced by the toolkit is:

```
fst[1]: pmatch Find me a Boston Market in Cambridge
Find me a <Restaurant>Boston Market</Restaurant> in <Location>Cambridge</Location>
```

The main advantage of parsing with FSTs is that the models are very compact (the FST network built using a location list of 320K items is about 25MB in size) and the amount of processing time is very low (a few ms per sentence).

3.4. Semantic classification

Semantic classification is the task of mapping relevant pieces of information from a sentence into semantic labels (classes). It mostly relies on constructing features to represent the sentence and building a classification model. As shown in Fig. 1, one can perform several types of semantic classification. The semantic domain and intent classification assign to each sentence a single class while the semantic slot extraction identifies and labels parts of the sentence[7]. There are mainly two types of statistical classification approaches [41]:

- Generative (also known as Informative) methods that directly model each of the class densities separately. Classification is done by examining the likelihood of each class producing the features ($P(Class|Features) \sim P(Features|Class) \times P(Class)$) and assigning to the most likely class. Although not difficult to train, these methods often lag in accuracy. Some examples are Fisher Discriminant Analysis, Hidden Markov Models and Naive Bayes and they were used in the BBN Call Director [33] and the AT&T HMIHY [22].

- Discriminative methods that model the class boundaries or class membership directly rather than the class feature distributions. Because these models take into account all classes simultaneously, they are harder to train, often involve iterative algorithms and might not scale well. Examples include Neural Networks, Support Vector Machines, AdaBoost (AT&T SLU system [23]), Conditional Random Fields (Microsoft NLU [10]).

In early systems, the feature set used to represent a sentence was mostly a *bag of words* or *n*-grams. Since many words express no semantics, this was later refined to consist of *salient phrases* computed based on mutual information. For example, the fragment "cents a minute" strongly suggests a calling plan [22]. However, sometimes sentence fragments may have a completely different meaning than any

[7]Slot extraction also performs sentence segmentation and is a more difficult classification task

of their constituent words (e.g. “flying spaghetti monster” is a religous sect). The matter is even more complicated if the semantic segmentation is unknown. If “George Washington” is segmented as a single semantic entity then it can be interpreted as a person (US president) name. But if contains two semantic entities then it should be interpreted as a “Town State” entity[8]. In order to address these issues, newer systems include semantic parsing based features [10]. Given a semantic dictionary list (also known as a *gazeteer*), the entity types of the sentence fragments found in the dictionary can be used as features instead of the bare words. Other types of features are Language Model scores, syntactic parsing labels or even semantic class information from another (possibly noisy) source.

3.5. Parsing-based semantics

There has been a large amount of recent work (especially from the *Information Extraction* community) dealing with extracting semantics from queries people submit to search engines. These queries can be either spoken or typed and have been mentioned in Section 3.1 as “voice search” data. One can roughly divide them into [7]:

(i) *Navigational*: reaching a website explicitly requested (e.g. “go to facebook”) or a certain state in the dialog flow (e.g. “go back” or ”cancel”),
(ii) *Informational*: finding information on the web (e.g. “capital grille restaurant reviews”) and
(iii) *Transactional*: conducting a transaction on a website (e.g. “make a reservation at capital grille”).

These queries are relatively short, contain many named entities and are often formulated as a concatenation of keywords rather than in a natural language. This particular structure makes it easy to generate a compact representation called *query templates*. A template is a sequence of terms that are either text tokens or variables that can be substituted from some dictionary or taxonomy [5]. For example, if the named entities are replaced by their type in the annotated sentence in Section 3.3 we obtain the template

Find me a $<$*Restaurant*$>$ in $<$*Location*$>$

It has been reported that a large number of queries follow a small number of structured patterns / templates: 90% for real estate and hotel related queries and 80% for automobile and car rental queries [1]. The template extraction process is based on abstracting the semantic entities/slots and sometimes needs a context sensitive mechanism for disambiguation (see the *“Eagles”* example in Section 3.3). There are also queries which are inherently ambiguous (have multiple meanings). For example *“jobs at apple”* may refer to either employment with Apple or to the former Apple executive Steve Jobs [1].

Each parse can receive a score indicative of its quality. While several scoring functions are analyzed in [34], a simple heuristic can be Occam’s razor: models which are shorter (contain a smaller number of slots) and more complete (abstract as much as possible of the query) are to be prefered. Notice that this heuristic is in itself an optimization process and it is applied to each query at runtime. This contrasts to the statistical classification methods presented in Section 3.4 where some optimization is performed on a training set and one hopes that it will generalize to unseen samples.

Parsing-based semantics employs a set of query templates and several fact databases to extract the query intent and semantic slots. The semantic domain and intent classes can be associated to each template rather than to individual queries (a template represents an equivalence class of queries in the

[8]There exists indeed a town named George in the Washington state and a person could say “George Washington” with the same meaning as “Seattle Washington”. However, people usually avoid this kind of ambiguities in their communication.

semantic space). If *click-through* data (search instances that led to clicks on some of the returned links) is available, this assignment can be done automatically [5], otherwise manual assignment can be performed starting with the templates that have the highest recall (cover the largest query classes). The names of the parsed semantic slots can also be used as features for statistical classification complementing those described in Section 3.4.

There are several advantages of representing queries by template models:

1. Templates generalize the set of target queries and model queries that follow the same patterns but did not appear in the training data [5]. That is especially useful when bootstrapping an NLU system with very little usage data available.
2. Templates models do not require retraining as new entities emerge. If a new restaurant opens, its name can be added to the Restaurant list and all requests applying to other restaurants will generalize to the new one [5].
3. Since they are derived from real data, templates are more comprehensive than hand-crafted rules and far more compact than non-generalizing whitelists (lists of cached queries) [1][5].
4. Template models allow for quick query parsing and matching using FST technology (see Section 3.3 and [34]).
5. Template models do not require an apriori domain schema that specifies the semantic slots and their values. Instead, it learns the most frequent slots automatically while identifying the most relevant templates [1].

Finally we would like to mention that automatically extracted templates have been successfully used for semantic reasoning and relation extraction [35]. A small set of manually identified seed facts that are in a "hidden relation" (e.g. *(Vincenzo Bellini, 1801)*) was used to extract patterns from a large amount of web documents. An example template is "**LHS** *BE_BORN MONTH* **RHS**" (**LHS** and **RHS** denote the Left Hand Side and Right Hand Side of the seed facts respectively). These templates were in turn used to infer the same relationship for many other instances of the two semantic entities (*Person* and *BirthYear*).

4. Conclusions and future developments

After five decades of research, natural language understanding and prediction technology has become an essential part of many human-machine interaction systems (and even human-to-human; see automated translation). We believe that the tipping point for the large scale deployment of this technology has been attained with the introduction of smartphones in the late 2000s. There are now voice-based personal assistants, search and transactional systems for most smartphone platforms [18]. The technology is pushed even further by the search engines (Google, Bing and Yahoo!) which have evolved from simple keyword search to semantic search [24]. They can now provide direct answers to a wide range of questions (e.g. "What's the weather tonight in Boston" or "What are the latest Bruins scores") rather than links to web documents.

5. Appendix

In this section we provide definitions for some acronyms and measures used throughout the text:

WER: Word Error Rate measures the quality of the output produced by a speech recognizer and has typically been measured against a human-made ground truth reference of the audio input. WER is computed as the sum of the errors in each of three classes (word substitutions, insertions and deletions) and is normalized by the number of reference words.

***N*-gram hit rates** express the percentage of *n*-grams in a corpus which are retained (explicitly modeled) by a Language Model.

Marcus contextual grammar is a construct $G = (V, A, (S_1, C_1), ..., (S_n, C_n))$, $n \geq 1$ where V is a vocabulary, A is a finite language over V, S_1,...,S_n are languages over V and C_1,...,C_n are finite subsets of $V^* \times V^*$ (V^* is the set of all words/strings over V, including the empty one). The elements of A are called axioms (starting words), the sets S_i are called selectors, and the elements of sets C_i, written in the form (u, v), are called contexts.

The direct derivation relation on V^* is defined as $x => y$ *iff* $x = x_1x_2x_3$, $y = x_1ux_2vx_3$, where $x_2 \in S_i$, $(u, v) \in C_i$ for some i, $1 \leq i \leq n$. A derivation is called in **maximum global mode** *if* there are no x'_1,x'_2,$x'_3 \in V^*$ such that $x = x'_1x'_2x'_3$, $x'_2 \in S_j$ for some $1 \leq j \leq n$ and $|x'_1| \leq |x_1|$, $|x'_3| \leq |x_3|$, $|x'_2| > |x_2|$.

Semantic template coverage is the ratio of the number of queries that are instances of the template and the total number of queries.

References

[1] Agarwal, G., Kabra, G., Chang, K. C. C.: Towards rich query interpretation: walking back and forth for mining query templates, *in Proc of the 19th international conference on World Wide Web*, 2010, 1-10.

[2] Arisoy E., Sainath T. N., Kingsbury B., Ramabhadran B.: Deep neural network language models. *In Proceedings of the NAACL-HLT 2012 Workshop: Will We Ever Really Replace the N-gram Model? On the Future of Language Modeling for HLT*, 2012, 20-28.

[3] Beesley K. R., Karttunen L.: *Finite state morphology*, Center for the Study of Language and Information Publication, 2003. FST toolkit can be downloaded at http://www.stanford.edu/ laurik/fsmbook/home.html

[4] Bellegarda J. R.: Statistical language model adaptation: review and perspectives, *Speech Communications*, **42**, 2004, 93-108.

[5] Bortnikov E., Donmez P., Kagian A., Lempel R.: Modeling transactional queries via templates, *in Proc. 34th European conference on Advances in Information Retrieval ECIR 2012*, 2012, 13-24.

[6] Brill E., Moore R. C.: An improved error model for noisy channel spelling correction, *in Proc. of the 38th Annual Meeting on Association for Computational Linguistics*, 2000, 286-293.

[7] Broder, A.: A taxonomy of web search, *In ACM Sigir forum*, **36**(2), 2002, 3-10.

[8] Bulyko I., Ostendorf M., Siu M., Ng T., Stolcke A., Cetin O.: Web resources for language modeling in conversational speech recognition, *ACM Trans. on Speech and Language Processing*, **5**(1), 2007.

[9] Burns M.: Nuance supercharges Swype, adds new keyboard options, XT9 predictive text and Dragon-powered voice input, *TechCrunch.com*, 2012, Available at http://techcrunch.com/2012/06/20/ nuance-supercharges-swype-adds- new-keyboard-options- xt9-predictive-text-and- dragon-powered- voice-input/

[10] Celikyilmaz A., Hakkani-Tur D., Tur G.: Statistical semantic interpretation modeling for spoken language understanding with enriched semantic features, *in Proc of IEEE Workshop on Spoken Language Technologies*, 2012, 216-221.

[11] Charniak E., Gales M.: Personal communication, *EARS RT-04 workshop*, Yorktown Heights, NY, 2004.

[12] Chelba C., Brants T., Neveitt W., Xu P.: Study on interaction between entropy pruning and Kneser-Ney smoothing, *in Proc of Interspeech*, 2242-2245.

[13] Chelba C., Bikel D. M., Shugrina M., Nguyen P., Kumar S.: *Large scale language modeling in automatic speech recognition*, Google technical report, 2012, Available at: http://static.googleusercontent.com/ external_content/untrusted_dlcp/ research.google.com/en/us/ pubs/archive/40491.pdf

[14] Chelba C., Xu P., Pereira F., Richardson T.: Distributed acoustic modeling with back-off n-grams, *in Proc of ICASSP 2012*, 2012, 4129-4132.

[15] Chomsky N.: Three models for the description of language, *IRE Transactions on Information Theory*, **2**, 1956, 113-124.

[16] Dean J., Ghemawat S.: MapReduce: simplified data processing on large clusters, *OSDI'04: Sixth Symposium on Operating System Design and Implementation*, San Francisco, CA, December, 2004.

[17] Dowding J., Gawron J. M., Appelt D., Bear J., Cherny L., Moore R., Moran D.: Gemini: A natural language system for spoken language understanding, *in Proc. ARPA Workshop on Human Language Technology*, Princeton, NJ, Mar. 1993.

[18] Duffy J.: Apple's Siri versus Dragon Go! and Vlingo, *PC Magazine*, 10/6/2011, Available at http://www.pcmag.com/article2/0,2817,2394267,00.asp

[19] Duta N.: Transcription-less call routing using unsupervised language model adaptation, *In Proc. Interspeech 2008*, Brisbane, Australia, September 22-26, 2008.

[20] Duta, N., Schwartz R.: Using a large LM, *EARS technical workshop, Eurospeech 2003*, Martigny, Switzerland.

[21] Duta, N., Schwartz R., Makhoul J.: Analysis of the errors produced by the 2004 BBN speech recognition system in the DARPA EARS evaluations, *IEEE Trans. on Audio, Speech, and Language Processing*, **14**(5), 2006, 1745-1753.

[22] Gorin A. L., Riccardi G., Wright J. H.: How may I help you? *Speech Communication*, **23**(1-2), 1997, 113-127.

[23] Gupta N., Tur G., Hakkani-Tur D., Bangalore S., Riccardi G., Gilbert M.: The AT&T spoken language understanding system, *IEEE Trans. on Audio, Speech, and Language Processing*, **14**(1), 2006, 213 - 222.

[24] Imielinski, T., Signorini, A.: If you ask nicely, I will answer: semantic search and today's search engines, *In Proc. IEEE International Conference on Semantic Computing*, 2009, 184-191.

[25] Jelinek F.: *Statistical methods for speech recognition*, MIT Press, 2001.

[26] Kirchhoff K., Bilmes J., Da, S., Duta N., Egan M., Ji G., He F., Henderson J., Liu D., Noamany M., Schone P., Schwartz R., Vergyri D.: Novel approaches to Arabic speech recognition: Report from the 2002 John-Hopkins summer workshop, *In Proc. ICASSP 2003*, I 344-347.

[27] Kneser R., Ney H.: Improved backing-off for m-gram language modeling, *In Proc. ICASSP 1995*, 181-184.

[28] Lopez, A.: Statistical machine translation, *ACM Computing Surveys*, **40**(3), 2008, 1-49.

[29] Marcus S., Paun G., Martin-Vide C.: Contextual grammars as generative models of natural languages *Computational Linguistics*, **24**(2), 1998, 245-274.

[30] Miller S., Bobrow R., Ingria R., Schwartz R.: Hidden understanding models of natural language, *in Proc. Annual Meeting Association for Computational Linguistics*, Las Cruces, NM, Jun. 1994.

[31] Mohri M., Pereira F. C. N., and Riley M.: The design principles of a weighted finite-state transducer library, *Theoretical Computer Science*, **231**, 2000, 17-32.

[32] Mori S., Nishida H., Yamada H.: *Optical character recognition,* John Wiley and Sons, 1999.

[33] Natarajan P., Prasad R., Suhm B., McCarthy D.: Speech enabled natural language call routing: BBN call director, *in Proc. Int. Conf. Spoken Language Processing,* Denver, CO, Sep. 2002.

[34] Parameswaran, A., Kaushik, R., Arasu, A.: *Efficient parsing-based keyword search over databases*, Technical Report, Stanford University, 2012.

[35] Pasca M., Lin D., Bigham J., Lifchits A., Jain A.: Organizing and searching the World Wide Web of facts - Step one: the one-million fact extraction challenge, *in Proc. of the 21st National Conference on Artificial Intelligence (AAAI-06),* Boston, Massachusetts, 2006, 1400-1405.

[36] Paun Gh.: *Marcus contextual grammars,* Kluwer Academic Publishers, Dordrecht, Boston, London, 1997.

[37] Pieraccini R., Tzoukermann E., Gorelov Z., Levin E., Lee C., Gauvain JL.: Progress report on the Chronus system: ATIS benchmark results, *In Proc. of the workshop on Speech and Natural Language*, 1992, 67-71.

[38] Plamondon R., Srihari S. N.: On-line and off-line handwriting recognition: a comprehensive survey, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **22**(1), 2000, 63-84.

[39] Price P. J.: Evaluation of spoken language systems: The ATIS domain, *in Proc. DARPA Workshop on Speech and Natural Language*, Hidden Valley, PA, Jun. 1990.

[40] Rosenfeld, R.: Two decades of statistical language modeling: Where do we go from here? , *Proceedings of the IEEE*, **88**(8), 2000.

[41] Rubinstein, Y. D., Hastie, T.: Discriminative vs. informative learning, *In Proc. Third Int. Conf. on Knowledge Discovery and Data Mining*, 1997, 49-53.

[42] Seneff S.: TINA: A natural language system for spoken language applications, *Computational linguist*, **18**(1), 1992, 61-86.

[43] Solomonoff R.: A formal theory of inductive inference, *Information and Control*, Part I: **7**(1), 1964, 1-22.

[44] Stolcke A.: Entropy-based pruning of backoff language models, *in Proc. DARPA Broadcast News Transcription and Understanding Workshop*, Lansdowne, VA, 1998, 270-274.

[45] Tur G., De Mori R. (eds): *Spoken language understanding - Systems for extracting semantic information from speech,* John Wiley and Sons, 2011.

[46] Turing A. M.: Computing machinery and intelligence, *Mind*, **49**, 1950, 433-460.

[47] Wang Y. Y., Acero A., Chelba C.: Is word error rate a good indicator for spoken language understanding accuracy?, *inProc. ARPA HLT Workshop,* St. Thomas, USA, pp. 577-582, 2003.

[48] Wang Y. Y., Deng L, Acero A.: *Semantic frame-based spoken language understanding*, in Tur G., DeMori R. Eds. Spoken Language Understanding, John Wiley and Sons, 2011.

[49] Wang Y. Y., Yu D., Ju Y.C, Acero A.: *Voice search*, in Tur G., DeMori R. Eds. Spoken Language Understanding, John Wiley and Sons, 2011.

[50] Ward W., Issar S.: Recent improvements in the CMU spoken language understanding system, *in Proc. DARPA Broadcast News Transcription and Understanding Workshop,* Lansdowne, VA, 1998, 270-274.

[51] Witten, I., Bell, T.: The zero-frequency problem: Estimating the probabilities of novel events in adaptive text compression, *IEEE Trans. on Inform. Theory,* **37**(4), 1991, 1085-1094.

[52] The history of automatic speech recognition evaluations at NIST, *Available at http://www.itl.nist.gov/iad/mig/publications/ASRhistory*