A Knowledge-Based Architecture for using Semantics in Automatic Speech Recognition

By Benjamin E. Lambert

Ph.D. Thesis Proposal

Committee:
Scott E. Fahlman (chair)
Bhiksha Raj
Roni Rosenfeld
Candy Sidner (Worcester Polytechnic Institute)

Abstract

Human languages allow us to express infinitely many ideas and expressions, in phenomenally creative ways. The range and variety of expressivity in human language poses a significant challenge to automatic speech recognition. To automatically recognize speech, it is necessary to build computational models of speech, to model both the acoustics and the language. These are typically modeled separately—the acoustics are modeled using signal processing and pattern recognition techniques, whereas the language is typically modeled with a statistical language model. However, current statistical language models lack the ability to incorporate semantic and pragmatic knowledge. I propose to develop and demonstrate a framework for incorporating semantics of spoken language, as represented in a symbolic knowledge base, into automatic speech recognition (ASR) systems.

List of abbreviations

AMT	–	Amazon Mechanical Turk
ASR	–	Automatic speech recognition
HMM	–	Hidden Markov model
KB	–	Knowledge base
KR	–	Knowledge representation
LM	–	Language model
MEM	–	Maximum entropy model
MTurk	–	Amazon Mechanical Turk
NLU	–	Natural language understanding
SLU	–	Spoken language understanding
WER	–	Word error rate

List of figures

List of tables

1 Introduction

Recognizing speech can be a challenge even for people. For instance, understanding speech in a noisy room can be very difficult. Yet, people can recognize speech even in adverse environments remarkably well, perhaps using other clues, such as context, body language, and our own expectation of the speaker's intention and meaning. Automatic speech recognition (ASR) systems typically do not have access to any of these additional clues. I propose to develop and explore methods for modeling one of these directly: the meaning of the language. In addition to exploring how to effectively model meaning in spoken language, I will also explore how such a model can successfully inform an ASR system.

ASR systems typically maintain a model of the language they are designed to recognize. When the language is very restricted, as in some dialog systems, it is relatively straightforward to model everything that can be recognized by the system. However, human languages allow speakers to be very creative, such that it is simply not possible to *completely* and *accurately* model an entire human language. To make language modeling tractable, simplifying assumption about the language must be made.

Perhaps the most common and successful simplifying assumption is the assumption made by n-gram statistical language models. These models assume that the probably of a word occurring depends only on the identity of the few immediately preceding words. The simplest (and perhaps most successful of these) model only word occurrence and co-occurrence statistics, without any notion of syntax or semantics. Despite their simplicity, these models perform remarkably well, and it has proved to be very challenging to improve upon these models.

When ASR is performed as part of an application, more specific and focused simplifying assumptions can be made, often with a task-specific grammar. This is common in voice-based command and control, information access applications, and dialog systems. At any given point in the dialog, the speaker may be much more likely to say some things rather than others. And, since the speech is in the context of a particular application, often the recognizer can use knowledge of the system's capabilities to constrain the possibilities much further. However, knowledge of the system's capabilities and how those are expressed in language cannot be readily transferred to a new domain or application.

Thus, at one end of the spectrum, we have open-domain large vocabulary language modeling, which uses word statistics and little, if any, syntax or semantics. At the other end of the spectrum, we have semantic grammars that fully specify the syntax and semantics of the language they recognize. There is not much middle ground, although there has been some work incorporating open-domain syntax and semantics into general ASR.

Most of this work, which tries to bring open-domain syntax or semantics into ASR, has been focused on using syntax, since syntax is inherently domain-independent. However, the gains achieved by adding syntax have been modest. There has been little work using general semantics for domain-independent ASR. I propose to develop and explore exactly this.

I propose to bring meaning into ASR through use of a general common-sense knowledge base (KB), in conjunction with linguistic constructions to map from language to the KB. Using these two components, we will design and architecture and develop a system that prefers speech recognition hypotheses that are more meaningful and less nonsensical—that is that are consistent with the knowledge base and plausible. This will not replace standard

speech recognition techniques, but it will augment them. Our system, when it lacks relevant KB coverage, or when evidence from the acoustic and language models are sufficiently confident, will step aside altogether.

1.1 Problem statement

Acoustic information alone is insufficient for automatic speech recognition. A model of the language is necessary to restrict the acoustic recognizer from outputting complete gibberish. Both *n*-gram statistical language models (LMs) and semantic grammars are language models. What I am proposing is different from each of these. Rather than model the language directly, we will model the *meaning* within the language directly. By doing this we place additional constraints on the language *semantics*, not the word order or syntax.

We will not model *all* the meaning in the speech; rather we will only model as much of the meaning as our system can reasonably interpret. Thus, in the end, we only model only snippets or bits of meaning, those that we can model with confidence. This frees us from having to perform complete language understanding, and frees the user from having to obtain a comprehensive knowledge base. But, it adds a challenge for us: how to use this incomplete knowledge effectively.

The goal here is perhaps best understood looking at the basic process. The ASR process I propose to develop is as follows: 1) the ASR system produces several candidate recognition hypotheses; 2) our NLU—as much as possible—extracts semantic representations for each of these; 3) check whether the semantic representations for some of the ASR hypotheses seem to be more plausible than others—using a variety of techniques; 4) if these techniques provide evidence that some of the speech recognition hypotheses are more or less plausible than others, incorporate that evidence into that hypothesis' score (along with acoustic and LM scores). Steps 2-4 are the focus of this proposal and this research.

Each step of this process is great challenge in itself. However, we believe that by assuming the KB and NLU are imperfect and/or incomplete, and deferring to the statistical models whenever necessary, that we may be able to make some progress and open some new doors. This is a proposal to explore and understand if and how and in what ways this may be accomplished.

2 A brief history of semantics in ASR

Using semantics in automatic speech recognition (ASR) is not a new idea. In fact, some of the earliest speech recognition systems had a semantic component integrated deep, right alongside the acoustic component. The HEARSAY-I system was one such system (Reddy, 1973).

HEARSAY-I was integrated into a computer chess game, such that one could play "voice chess" against the computer. Each time it was the player's "turn" HEARSAY would compute all the player's valid moves and each move's likelihood. Using a grammar of 18 rules (Figure 1), HEARSAY determined *all* ways of speaking those valid moves. Although language in the chess domain is relatively simple, the system used a complete semantic model, which was critical to its success at recognizing speech.

```
1.  <move>          ::= <move1> <check-word> | <move1>

2.  <move1>         ::= <regular-move> | <capture> | <castle>

3.  <castle>        ::= <castle-word> ON <uniroyal> SIDE
                      | <castle-word> <uniroyal> SIDE
                      | <castle-word>

4.  <regular-move>  ::= <man-loc> <move-word> <square>

5.  <capture>       ::= <man-loc> <capture-word> PAWN EN-PASSENT
                      | <man-loc> <capture-word> <man-loc>

6.  <castle-word>   ::= CASTLE | CASTLES
```

Figure 1 - The first six rules of HEARSAY-I's 18-rule semantic grammar (Reddy, 1973).

Like HEARSAY, many academic research systems in the 1970's and 1980's, were used in small enough domains that it was possible to encode everything that could be recognized in a semantic grammar. A semantic grammar is a context-free grammar where the non-terminals are semantic classes (Burton, 1976). For most of these systems, *the grammar is the language model*. In addition to providing a language model, the semantic grammar is a vehicle for extracting an *interpretation*.

All of the systems in the 1971-1976 ARPA Speech Understanding Research project used semantic grammars (summarized by Klatt, 1977). The speech in this project, to query for academic articles and plane fares, had simple enough sentence structure that it was feasible to use manually write a semantic grammar. Table 1 shows examples of the speech recognized by these systems.

System(s)	Example speech
CMU (Hearsay-II and HARPY)	"How many articles on psychology are there?"
BBN "Hear What I Mean" (HWIM)	"What is the plane fare to Ottawa?"
SDC	"How fast is the Theodore Roosevelt?"

Table 1 - Example speech from ARPA SUR project (Klatt, 1977)

At the same time, Jelinek, Mercer, Bahl, and colleagues at IBM were developing statistical n-gram language models for ASR (Jelinek, 1976; Bahl, 1983). IBM's goal was to build a speech transcriber (then called a "voice typewriter"). Since this needed to recognize general English speech, semantic grammars were not an option. Jim Baker's early work on the DRAGON system used similar word transition probabilities (Baker, 1976). These techniques did not become usable until the late-80s after better methods of smoothing unseen n-gram probabilities were developed, also at IBM (Nadas 1984; Katz, 1987).

These statistical language models became usable at just the right time, because in the late 80's and early 90's DARPA released some datasets for which it was not feasible to write an entire semantic grammar. Sentences in the Resource Management dataset (Price, 1988) and the ATIS speech datasets (Hemphill, 1990) were too complex to build a grammar for—the grammars would have to be very big, complex, and difficult to maintain. Table 2 shows example sentences from these two datasets.

Resource Management: *Show locations and C-ratings for all deployed subs that were in their home ports April 5.* ATIS: *I need fares from Milwaukee to Long Beach Airport.*

Table 2 - Example speech from the new DARPA datasets: Resource Management (1988) and ATIS (1990).

The impracticality of using a semantic grammar on such datasets, led to the development of a two-step process that is still common today. The semantic grammar had been serving two purposes: 1) it *was* the LM, and 2) it helped to determine an *interpretation* of the speech. The LM functionality was replaced by statistical n-gram LMs. Finding an interpretation of the speech was performed by a separate natural language understanding (NLU) module. First speech recognition was performed with the aid of the statistical LM, and then NLU would extract an interpretation from the output of the ASR.

This two-step process, ASR then NLU, works very well when the ASR module is correct. The problem with splitting this process into two steps is what happens when the ASR module is incorrect. When the ASR is incorrect, the NLU unit might derive an incorrect interpretation, or it might not determine any interpretation at all. By splitting the process into two steps, we no longer have any guarantee that the output of the recognizer will be even close to interpretable, that is it may be very far from "making sense."

It did not take long to develop some methods to cope with this discrepancy. For instance, if the ASR unit produces a list of *n* best hypotheses, the NLU unit can work down the list until it finds a hypothesis that is interpretable.

However, methods for bringing task and domain semantics into the ASR process have not been entirely satisfactory. Perhaps there are a few reasons for this. In many applications—where the language can be very restricted—it makes sense to just use a semantic grammar, and not bother with these two steps. In other applications, the statistical LM works well enough. Finally, when task/domain semantics are used they are often not portable to another application or domain. It appears that for these reasons and perhaps others, this is not a well explored research area. Thus, this discrete two-step process remains common today (as shown in Figure 2 from a popular SLU and ASR text book).

However, in the last 20 years, voice-based systems have been becoming increasingly sophisticated (e.g. tutoring systems and collaborative interfaces) and open-ended (e.g. voice-based information retrieval). The more these systems grow in sophistication and domain independence, the less they will be able to rely on conventional NLU (i.e. form-filling) (Kawahara, 2009). I believe the time is right to explore new ways of using semantics in ASR. How to do this is the topic of this thesis.

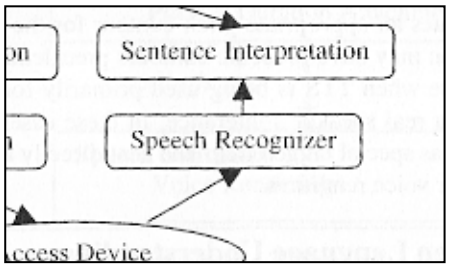


Figure 2 - The two step process: recognition then understanding (Huang, 2001).

3 Modeling semantics

3.1 Human knowledge

Ultimately, our goal is: given a speech recognition hypothesis, is it something a human is likely to have spoken? Or, given a few recognition hypotheses, which is *more* likely to have been spoken? But, rather than determine this with word statistics, we will use the human notion of sense and knowledge. That is given ASR hypotheses:

"The Pope will be used to produce newsprint," and
"The pulp will be used to produce newsprint"

Which would the average person believe is more likely, using their own knowledge and common sense?

Presumably, most people's knowledge would tell them that "the pulp will be used to produce newsprint" is more likely. In this research, we attempt to *approximate* how people use their knowledge to make this decision.

However, human knowledge and common sense is not well-defined. Even ignoring the differences in each person's knowledge, people do not reason or use knowledge consistently or deterministically. So, on the one hand, people are somewhat unpredictable in this regard, but on the other hand, most people are not completely random either.

Just as human common sense may be imperfect, we will emulate this and do not attempt to be perfect in this regard. But this begs some interesting questions about how reliable, practical, and usable sense (human or artificial) is for something like ASR. Although it is not the focus of this thesis, we will explore this question to some extent. That is, when and how often is human knowledge useful in ASR? We begin to ask some of these questions in the results section of this proposal. Specifically, we consider what happens when give people the exact task that we are attempting to automate.

3.2 A computational model of semantics

We seek to model, approximately, the human notion of sense, so the very first thing we need is a computational model of semantics. We will use a symbolic knowledge base (KB) for this. This formalism for knowledge representation is not intended to be a model of how people actually represent and reason about knowledge. It is intended to represent the *kinds* of knowledge that may mimic some of the human reasoning that could be used to help understand speech.

After formalizing this representation, we will look at how to use this model to represent meaning in spoken language, and how to automatically identify and extract that representation. After that, we will consider *how* this could aid ASR.

The knowledge representation system we use is Scone (Fahlman, 2006). There are many other formal knowledge representation systems available, and many of those could similarly be used. We use Scone, because, among other reasons it is specifically designed to represent meaning in language, and because it is designed to mimic human intelligence (not super human intelligence).

For this proposal, we will not go into detail about other knowledge representation (KR) systems. However, I will very briefly mention the major ways in which these systems differ. The primary differences include:

- Reasoning capability (from none to full theorem proving)
- Expressivity of the representation (e.g. propositional logic, first-order logic, higher-order logics)
- Speed of operation and ease-of-use

Scone performs reasoning, but does not attempt to be logically complete, nor does it perform theorem proving. This means that Scone is very fast. We believe this will be an important feature for a KR system in ASR, since speech is often very rapid! Additionally, since Scone is specifically designed to represent meaning in language, Scone's expressivity should be well suited to our needs.

A few other popular knowledge representation systems include:

- OWL (Web Ontology Language) a family of "description logics" (W3C, 2009) standardized by the World Wide Web consortium.
- Cyc, a large-scale commercial knowledge representation system (Cyc).
- First-order logic.
- Frame-based KR, such as FrameNet.

3.2.1 Scone knowledge representation

A knowledge representation system is like a dictionary in many ways, but with two major differences. The first is that a dictionary is designed to describe and define *words*, whereas a knowledge base is designed to describe and define *concepts*. The second major difference is that a dictionary is designed to be read by people, whereas a knowledge base defines concepts formally and is more easily read by other computer systems.

Concepts and words are not equivalent. The concept of the country of the *United States* can be referred to in language directly as in "U.S.," "U.S.A.," and "The United States of America." This concept can also be referred to indirectly with phrases like "my country" or "that country." Thus, there are many different ways one can refer to the same concept, using very different words or phrases. In order to model semantics in spoken or written language, we need to maintain a model of this mapping, from words to concepts, where many words may map to the same concept.

Conversely, individual words can refer to multiple concepts. For instance, "bank" might refer to a *river bank* or a *financial institution*.

Thus, there is a many-to-many mapping from words to concepts (as illustrated in Figure 3), and it is important to distinguish between two. We represent each concept with a specific formal identifier. This identifier serves as a handle or symbol for the real-world concept. And, it allows us to represent statements and facts about the concept, irrespective of the words used to refer to it. Very often, going in either direction (from words to concepts or from concepts to words) is non-trivial. In this proposal, I will follow the convention of writing concepts in *italics*, and words in "quotation marks."

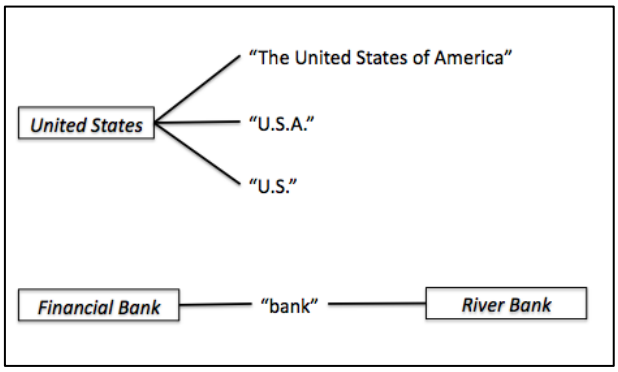


Figure 3 – Many-to-many concept-word mapping

Central to the Scone knowledge representation system is a hierarchy of types and individuals. The concept of a *country* is a type, an abstract concept. This type may also have sub-types; for instance, a *European country* is a sub-type of *country*. *France* is a specific individual country, which cannot have its own sub-types. The hierarchy of types is like a taxonomy (similar to biological taxonomies for classifying species). Hierarchies like this are also known as ontologies. (See Figure 4).

In Scone, the type hierarchy is represented by *is-a* links. In the Scone formalism, the interpretation of *is-a* links is that the child node inherits everything that is known about the parent concept. Thus, if *elephant* is a sub-type of *mammal*, then *elephant* inherits everything that is represented about *mammal*. An individual elephant, say *Clyde*, inherits everything that is known about the general *elephant* type. This type hierarchy can also be viewed as a tree of unary predicates.

Thus, if mammals have hair, then all elephants have hair, including Clyde. Scone also allows exceptions, so we can specify that Clyde is a hairless elephant, or that some particular species does not have hair.

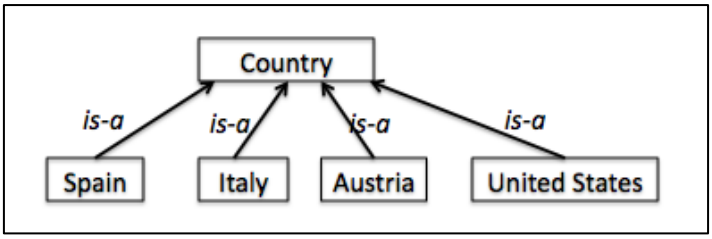


Figure 4 – A few instances of type country

The distinction between types and individuals goes beyond the fact that individuals cannot have sub-types. They are usually expressed differently in language. To make a general statement about the type elephant, one might say "elephants are grey," but individual elephants would be referred to as: "this elephant," "the elephant," "an elephant," "Clyde." This is especially important because in language, anonymous individuals are created "on-the-fly" frequently. The use of an indefinite article (e.g. "an elephant") is an indication that the speaker is referring to a specific individual, unidentifiable element.

Inheritance in the Scone hierarchy occurs by default, but Scone also allows exceptions. We may represent that all birds fly (a universal quantification), but specify the exception that penguins do not fly (Figure 5). This kind of representation and reasoning is called default reasoning with exceptions, or non-monotonic reasoning.

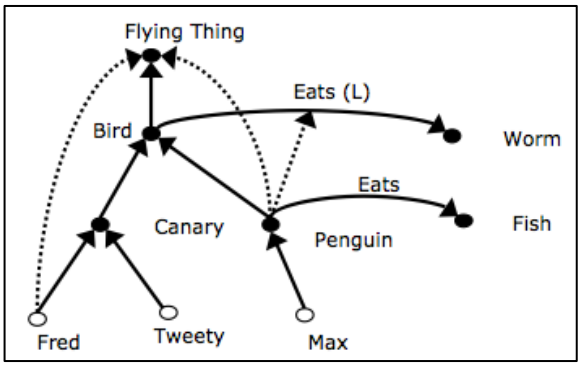


Figure 5 - Cancellation: penguins eat fish, not worms (Fahlman, 2006)

In addition to unary predicates, binary predicates can specify relations between two concepts. For instance *greater than*, *taller than*, and *friends with* are binary relations. The concepts that can be related by a particular relation are often type restricted. For instance, *greater than* applies to quantities, *taller than* applies to physical objects, and *friends with* applies to people (and perhaps animals or robots).

Relations can be specified to be symmetric or transitive, which is reflected by the reasoning machinery. In Scone, a particular instance of a relation is called a statement. Thus, *Fred* is *taller than Jim* is a statement about *Fred* and *Jim* that is of the type *taller than*. Scone can also represent higher-order relations, so it can represent statements about statements, as in: Joan believes that Fred is taller than Jim. Many other knowledge representations systems cannot represent these higher order relations, yet they are common in natural language.

Another way of specifying relations among concepts is to use *roles*. These are like the slots in a frame representation. Roles may have *fillers* and may be type restricted. Roles are particularly useful for representing the participants in an event. For instance, the event of baking something for someone has three participants: the baker, the recipient, and the thing that is baked. These would likely be type-restricted as a person, another person, and some kind of food. The event of *rowing* is typically performed by a person, on a boat, as represented in Figure 6.

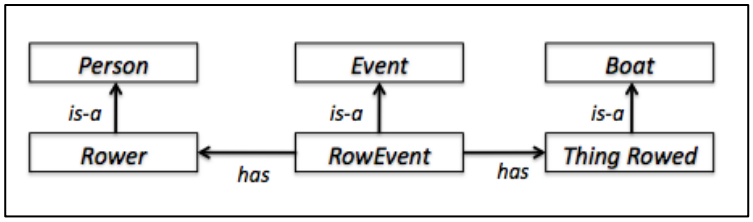


Figure 6 – A graphical representation of the event type for *rowing*

Just as a relation is instantiated as a statement, an event type is instantiated as an individual event. Figure 7 shows an instantiated event, in this case for "we row a placement bureau." Note that *placement bureau* violates the type constraint of *boat*, so this type restriction would need to be explicitly *overridden* with an exception to instantiate this event.

Figure 7 - An instance of a *row* event for "we row a placement bureau"

The last feature of Scone that I will mention here is the machinery for representing "possible worlds"—these are called *contexts* in Scone. These contexts inherit from one another, so a more specific world representation can be created without re-writing the entire KB (e.g. the Harry Potter World has a lot in common with the real world with a few differences). Scone contexts are particularly suitable for representing belief states (perhaps to be used in a collaborative dialog system (Lochbaum, 1998)), and could be used to model dialog contexts. For a more complete description of Scone, see Fahlman (2006).

3.2.2 How knowledge is obtained

Knowledge bases are often constructed manually. Recently many techniques have been developed to build knowledge bases automatically by mining instances, types, statements, and relations from the Web (Wang and Cohen, 2009). In addition, a number of recent standardized formalisms (such as OWL), and the growth of the Web have led to the availability of knowledge bases in many formats. Often the knowledge bases that use different formal representations can be at least imperfectly translated to another formalization (e.g. we could convert an OWL KB to a Scone KB).

Scone contains a small core knowledge base that represents high-level concepts, such as physical objects, abstract ideas, animals, people, times, and so forth. For this research, we use both small hand-crafted knowledge bases, to test our techniques, and larger imported KBs to test coverage.

The focus of this research is not in writing and importing knowledge bases. The goal will be to show that some knowledge can increase speech recognition performance, and that more knowledge will further increase performance. Thus, when we do not have a suitable knowledge base, we will attempt to answer questions like: what knowledge *would* be needed? If we had the knowledge how much would it help? If we can demonstrate techniques that perform better with more knowledge, then further improving performance becomes a matter of adding more knowledge.

In addition to Scone's core KB, we have imported the WordNet hierarchy of concepts into Scone. Over the years, many other KBs have been imported into Scone. Some additional work will likely be needed over the course of this research to import other knowledge sources (e.g. FrameNet).

As mentioned in the previous section, individual elements and statements are often created "on the fly" in natural language. These can be referred to in the same sentence, dialog, or context. Thus, in a way, this becomes background knowledge in itself. Maintaining and using this contextual knowledge is also not the focus of this research, but having this antecedent knowledge in the knowledge base may be very helpful for some speech recognition. We should be able to explore this question at least to some degree. That is: how much antecedent knowledge to retain, and how to use it?

A related issue that I will discuss later on is how to populate a KB with instance data from a dataset, and then use that instance data to judge the plausibility of other instance data. For example, we could extract all the instances of binary relations in a speech dataset, count how often certain relations occur between certain types of individuals, and use those frequencies to determine the plausibility of new ASR hypotheses.

4 Identifying linguistic representations of semantics

Our goal is to identify the meaning of hypothesized spoken language, so that we may compare it with our semantic world model to determine if the meaning appears to "make sense." In order to do this, we need to first perform the very non-trivial task of identifying and extracting the meaning from the language.

In a very specific voice application, the precise meaning of a complete utterance may not be needed for the application to function properly. For instance, on ATIS-like data, the system needs only to identify that the speaker is looking for flight times and prices, then the origin and destination locations, and preferred times. The application can potentially ignore the meaning of all the other words if it is able to pick up these key bits of information.

We approach the problem of extracting meaning from the words in a similar, but more general, fashion. Rather than attempt to produce a *complete* understanding of the language, and rather than looking for very *specific* bits of information, we attempt to identify *as many small bits of general meaning as we can*. Thus, rather than build a machine to identify destination locations, we build little machines to identify bit of information like adjective-modified nouns, for which we have sufficient knowledge to interpret.

Thus, our information extractors are more general than in a dialog system. However, there is a more important distinction. Rather than use the information extractor specifically to interpret the meaning of the speech, we check the meaning against the knowledge base. If the extracted meaning appears to be either reasonable or nonsense, we feed that information back to the recognizer. The goal of the meaning extraction is to do some verification, and then to feed the results back to the recognizer so that the recognizer will attempt to produce recognition hypotheses that are more likely to be meaningful.

4.1 From words to meaning

To get from the words in a speech recognition hypothesis to a representation of their meaning, we use a formalism inspired by the linguistic theory of construction grammars. The linguistic theory of construction grammar has evolved gradually since around 1987. Goldberg (2006) is a recent and accessible book on the linguistic theory. Only recently, have rigorous grammar formalisms and implementations of this theory become available (Bryant, 2008; Steels and De Beule, 2006; Kay, 2002), each with its own focus and goals.

However, the techniques described here are not intended to be an implementation of construction grammar theory. Rather, this theory seems to be inline with our goals. We will borrow aspects of the theory when it seems appropriate, but we will mostly be guided by data and the task of speech recognition.

The formalism I present here is neither complete (e.g. it lacks facilities for modeling subject-verb agreement) nor formally rigorous. Rather than locking ourselves into one formalism, we will adapt the following proposed formalism as necessary to achieve our goals in speech recognition.

We will first look at several examples then briefly discuss the implementation.

4.1.1 Constructions

In this theory and in this proposal, each construction is a *direct mapping* from a surface form to a meaning representation. This is, as opposed to theories and implementations that perform a syntactic parse first, followed by transformation of the syntax tree into a semantic representation.

As a form to meaning mapping, each construction consists of two parts: form and meaning. The form is typically a pattern of words, lexical items, or typed variables. The

form is used to match words in a speech recognition hypothesis. The meaning is a recipe for creating KB structure that represents the meaning of the matched words.

Ultimately, the goal of this research is to determine whether a sequence of words makes sense or not. We accomplish this by using a parser to match the form-side of our constructions to words in a speech recognition hypothesis. When the form matches, we instantiate the meaning side of the representation; this instantiation returns a formal representation of the meaning of the matched words.

Often, the form side of a construction is a list of words, typed variables, or both. We will often see words in the form side of idiom constructions, as in "<person> kicked the bucket."[1] The typed variables in the form might have either syntactic or semantic type. In the case of grammatical types, the types are likely to be parts of speech (such as determiner, noun, verb, gerund) or grammatical types (noun phrase, verb phrase, etc.). Semantically typed variables are likely to be types in the KB type hierarchy (e.g. *person*, *agent*, *tangible object*, *food*). We could also combine these, for instance specifying a NP that must refer to an agentive action.

We can also place more complex semantic constraints on the form side of a construction. For instance, we could constrain the type of one NP to be something that is *loved by* another NP in the construction. However, it may be easier to put such complex restrictions in the meaning side of the construction.

The following table illustrates some form specifications, and example strings that match:

Construction form specification	**Matching strings**
Det	The, a, an
Noun	Book, table, shoe, glass
Verb	Run, ran, type, book
Adj	Green, colorless
Det Adj Noun	The green idea
NP	Maya's great new idea
VP	Went to the store
Food	Cake, the cake, apple, my apple, mother's apple pie
Agentive action	Give, put, kick
Physical object	Truck, building, rock, John's pet rock
Person "kicked the bucket"	Fred kicked the bucket Dear old granddad kicked the bucket

Table 3 - Examples of construction "forms" and strings that match those forms.

The meaning side of a construction specifies how to build the corresponding KB structure when the construction is matched. That is, when a construction matches the words "a green apple," what structure do we create in the KB?

The indefinite article, specifies that we the speaker is talking about an anonymous individual instance. This is as opposed to "the green apple" where it is assumed that the listener has some way knowing or determining the specific green apple that being referred to. So, the indefinite article says: create a new individual. The words "green apple" tell us what the type of this individual will be. In particular, this says that the individual is an apple, and that this particular apple is green. However, before we can do this, we need to

[1] Note that the words are not exact string matches; in "kick the bucket" the verb still has tense and number agreement.

find the concepts associated with these words. That is "green" and "apple" are words, so first we need to identify the concepts associated with them. In this case, the meaning is obvious, "green" refers to the color *green,* and apple refers to the fruit *apple*. The word "green" has other meanings, as in the "newbie" meaning.

To get from the word to the concept is often very non-trivial. This is a research area – word sense disambiguation (WSD)—for which hundreds of papers have been written, and yet it is far from a solved problem. For this proposal, we are largely going to gloss over this issue. Many words do not have more than one word sense, or if they do, the most common sense is far more common than the other senses. One technique for dealing with WSD is to always assume a word refers to its most common sense, and use that concept. This will be imperfect, and will certainly limit the system's accuracy. However, it may be the case that one could improve ASR performance even when hindered by word sense ambiguity.

I will discuss the problem of WSD briefly later on, but for now we will just assume we have a perfect function called *conceptOf()*. This function takes a word (or words), and determines the concept to which that word (or words) refers.

Here is an example construction, with form and meaning. The other functions that are used in the meaning-side are described below.

FORM:	Determiner Adjective Noun
MEANING:	*newIsA*((*instanceOf*(*conceptOf*(Noun)), *conceptOf*(Adjective))

- *conceptOf*("word(s)")—given a word or several words, this function returns the *concept* referred to by the word or words. The actual implementation of this function is far from trivial due to word sense ambiguity. We will discuss the issue of word sense later-on, but for now, for simplicity, we will just assume we have a way to write this function.

  Additionally, since some concepts can be referred to by multiple words, there is more ambiguity present in the implementation of this function, when we encounter a phrase like "the white house." These two words may refer to the individual concept of the U.S. presidential mansion, or to a different house that is white. Using the more specific interpretation may be a good heuristic, but will be often incorrect
- *instanceOf*(*concept*)—Given a concept, for instance *boat*, create a new instance of that concept. This function does not give a name to this instance, but does give it a unique identifier (e.g. *boat-77*).
- *newIsA*(*concept1*, *concept2*)—create an *is-a* link from concept1 to concept2. Concept1 and concept2 may be either types or instances.
- *meaningOf*("words")—this is similar to conceptOf, but this identifies more complete semantic representations than just concepts. Thus, meaningOf("colorless green ideas") is an instance of the concept *ideas*, which has been connected by is-a links to *colorless* and *green*. This function is effectively the function we are defining in this section, so one could think of it as a recursive call.

4.1.2 A noun phrase construction

Here is a simple construction that will match sequences of three words. We have specified syntactic constraints on each; they must be a determiner, an adjective, and a noun.

FORM:	Determiner Adjective Noun
MEANING:	*newIsA*((*instanceOf*(*conceptOf*(Noun)), *conceptOf*(Adjective))

Thus, if the words "a red dog" are matched by this construction, the recipe will generate a semantic representation of this form:

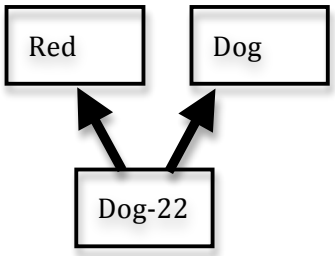


Figure 8 - The KB representation of "a red dog"

This particular construction does not use the determiner at all in the meaning-side of the definition; it is only used as part of the matching process. Thus, we obtain the exact same semantic representation for "the red dog" and "a red dog." However, this framework does support modeling the distinction if we choose to.[2]

For instance if the KB specifies that a dog cannot be red, then the instantiated meaning of "the red dog" would be inconsistent with the knowledge base. A better example comes from Chomsky's phrase "colorless green ideas." Ideas are represented as intangible concepts that cannot have a color; only tangible objects may have color. Furthermore, green things cannot be colorless. The marker-passing algorithm in Scone that identifies these conflicts quickly is described in Fahlman (2006).

Very briefly, to illustrate the steps that are taken when the construction above it matched, we perform four steps:

1. Find the concept referred to by the noun—we'll call this *concept1*
2. Find the concept referred to by the adjective—we'll call this *concept2*
3. Create a new anonymous instance of *concept1*—let's call this *instance1*
4. Create a new *is-a* link in the KB from *instance1* to *concept2*

4.1.3 A phrase construction—verb-argument constructions

In the previous section, we saw how we can extract a (partial) meaning representation of a noun-phrase, and could use the representation to detect some semantic conflicts, such as in "colorless green ideas." Next, we will look at a construction that matches a verb and its arguments. Verb-argument constructions are described at length in Goldberg (1995). I will briefly describe a few here.

The transitive verb construction is defined roughly as follows:

---

[2] ASR recognition errors on determiners and functions words are very common. However, the differences in meaning are actually relatively subtle. So, while it may be easy to model the difference in meaning, actually using that semantic representation to identify an error could be very difficult. For "a" vs. "the", the reasoning might come down to: is there an identifiable referent or not? This sort of reasoning is hard enough in itself. Thus, in this work we will focus on errors in contents words, such as "pope" vs. "pulp."

FORM:	$NP_1\ V_{transitive}\ NP_2$
MEANING:	*newEvent* = conceptOf($V_{transitive}$) agentOf(*newEvent*, meaningOf(NP_1)) patientOf(*newEvent*, meaningOf(NP_2))

This construction matches phrases like:

"Fred rowed a boat"

"John threw a ball"

The form-side of this construction specifies that it will only match a noun phrase, followed by a transitive verb[3], followed by another noun phrase. These are all syntactic constraints, at the word-level (transitive verb) and the phrase-level (noun-phrase).

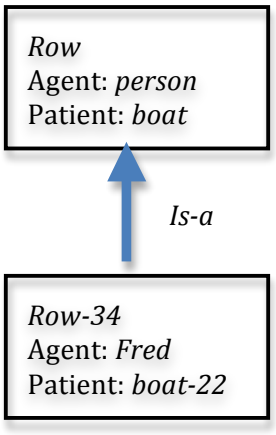
The meaning-side of this construction additionally requires that the event referred to by the verb have both *agent* and *patient* roles.

The event or action referred to by the verb may additionally put its own semantic constraints on the agent and patient. For instance, the agent of both *row* and *throw* is likely to be a *person*, or at least *animate*. The patient of *row* is likely to be a *boat* and the patient of *throw* is likely to be a small physical object.

If any of the constraints are not met, then we determine that the sequence of words is not consistent with the knowledge base. Thus, "we row a placement bureau" would be determined inconsistent with the KB because placement bureaus are too big and not buoyant enough to row.

The KB may not have enough information to determine if the constraints are met or not. For instance, if "we row that thing" is spoken, it is not known what "that thing" is, so it is not known if it can be rowed.

We represent events like *row* and *throw* with as events with agent and patient *roles*. When we encounter a transitive verb construction, we first assume that the two noun phrases have been instantiated appropriately by *meaningOf* (as *individuals*). We instantiate the verb to create a specific individual event to represent this phrase. Pictorially, this looks something like:



[3] The notion of transitivity is handled differently in construction grammar theories than other linguistic theories—it may be more reasonable to enforce transitivity in the KB, rather than in the construction form—but we will not concern ourselves with these issues presently.

Among the benefits of specifically instantiating each event is that we can save that representation for later. The instantiated event could be used at a later point in a dialog. Or it could be counted, going back to a frequency-based method, and perhaps even be used to estimate some statistics, such as the expected value that the object of a *row* event is a *boat*, or that the expected value that the agent of a *row* event is a *person*.

4.1.4 Additional example constructions

Here are a few more constructions. For these I will not be as precise defining the semantics. For additional examples of construction, see Croft (2001), Fillmore et al. (1988), Goldberg (1995), Goldberg (2006), and Langacker (2008).

Idioms

The constructions we have looked at so far have been fairly general and compositional. Next, we will look at a very specific construction, an idiom. If one encountered the words "dear old granddad kicked the bucket" in text or speech, one would not likely interpret this literally with the transitive construction. More likely, they would interpret it with the more specific idiom, meaning that he died. Note that we do not even need to represent that "the bucket" is a noun phrase; these words are tied directly into the construction's definition.

FORM:	*NP* "kicked the bucket"
MEANING:	*NP* died

Note that having this construction and the transitive construction in our inventory of constructions introduces an ambiguity. "Fred kicked the bucket" could be interpreted literally or idiomatically. One might assume that the more specific interpretation, the idiom, would be preferred in the absence of evidence to the contrary. Evidence to the contrary might include actual physical buckets being in the context (physical context or discourse context). Additionally, the subject not being a person is might prefer the literal interpretation, as in the "the horse kicked the bucket." In this case, there may not be a black-and-white reasoning process to determine one interpretation over the other (e.g. people may personify their horses).

There are many idiomatic expressions in a language. Here are just a couple others in English:

- "X got out of line"
- "X got taken to the cleaners"
- "Across the board"

Nominalized verbs

Verbs and their arguments may also be referring to in a nominalized form, for instance: "his admiration for Tony Provenzano." Semantically, this is the same as the transitive verb construction, but has a different form.

FORM:	NP1's nom-verb of NP2
MEANING:	verb, agentOf(verb, NP1), patientOf(verb, NP2)

Non-canonical argument structure

One somewhat odd construction may help to illustrate the power of a construction grammar formalism. The "caused-motion construction," described by Goldberg (2006), can be used with many verbs, not just transitive/intransitive/etc. For example, the verb sneeze is typically described as intransitive, but one can say:

"Joe sneezed the foam off the cappuccino."

In this construction, it is perhaps more the construction than the verb that specifies the meaning. The definition of this construction given by Goldberg is the following:

FORM:	X verb Y PP
MEANING:	X caused Y to move along path designated by PP

The construction itself requires that the subject be animate, that the object is something movable, and that the prepositional phrase specifies a path along which the object travels. The verb itself, *sneeze* in this case, only specifies that the subject is animate.

We believe that this kind of flexibility will be needed when used in speech recognition because this usage is perhaps especially common in spoken language.

4.2 Parsing constructions

We have now defined a few constructions, and briefly touched on how they might be used in a speech recognizer. We have not yet described how to implement or match them.

Thus far, we have simply considered the process involved here as: 1) matching the form, and 2) instantiating the meaning. I briefly mentioned that a match could be determined using either syntactic or semantic constraints. I also briefly mentioned that constructions can embed, as we saw two NP's embedded in the transitive verb construction—I didn't explain explicitly how we do the embedding, just that we called a meaningOf() function, which is like a recursive call.

The process we would like to implement is a matching these rules, allowing them to possibly embed. This is almost precisely what a context-free grammar (CFG) parser does, with a few specializations and differences. Our implementation is based on a modified version of the Earley parser (Earley, 1970).

The specializations and modifications we require:

- It should be bottom-up. This is because:
    - We want to identify the meaning of *portions* of the input, not necessarily *all* of it. Thus we want to be able to parse "we certainly don't run a placement bureau" in "'We certainly don't run a placement bureau,' says Charles Heck North American director." without having to produce a complete parse.
    - We will not be given sentence boundaries—real speech does not denote "sentences."
- The ability to skip words, possibly arbitrarily. This is important in speech where repeats ("I- I- have to …") and disfluencies (e.g. "um", "uh") are common. Lavie (1996) developed techniques for this.
- A specialized "matching" procedure that can match both syntactic and semantic constraints to determine when a match is found.

Since the form constraints can be either syntactic or semantic, each partial parse (e.g. the subject noun phrase before the verb and the rest have been parsed) will contain both syntax and semantics.

We have implemented most of this. The implementation is not perfect and does not have some of features that more sophisticated parsers have, but it has the speech-specific features we need and will be adapted as we go. Figure 9 roughly depicts the overall process.

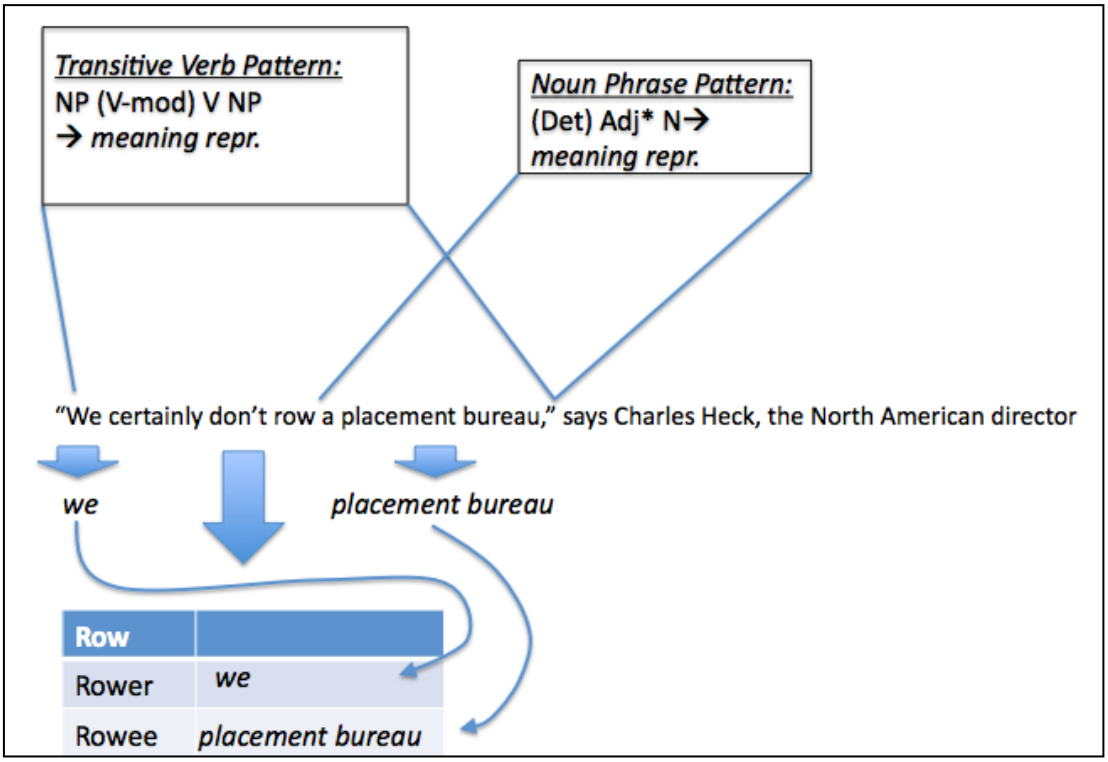


Figure 9 - An example showing how a couple construction patterns together derive a *partial* semantic representation for *part* of a sentence.

4.3 Challenges and complications

Many details could impede these efforts. For instance, by looking only for snippets of meaning, it is possible that we will find meaningful bits that include nonsensical portions, and nonsensical phrases that include perfectly valid language. Many of these challenges will be addressed along the way. Additionally, we will need to determine as we go the extent to which we will model the semantics of the language. For instance, do we want to represent negations and modal verbs?

The following is a partial list of the challenges we will face. Some of these we may be able to address and others we will not. Those that we cannot or do not address will limit our ability to help, or in some cases cause our techniques to do the wrong thing and be detrimental to performance. With this in mind, the key for us to improve performance is *to help more than we hurt*.

- **Subtle semantic distinctions** - The difference between "a" and "the" is actually subtle. Typically, "a" is used to refer to an anonymous red dog that the listener cannot identify specifically, and "the" is used when the specific red dog can be identified by the listener. To actually use this difference to improve the speech recognizer, would involve searching the dialog context, or the world knowledge, to determine if there is a specifically identifiable referent. Speech recognition mistakes on words like "a" and "the" are actually pretty common (especially when "the" is pronounced "THUH" rather

than "THEE"), but we will not focus on correcting this kind of mistake due to the significant amount of overhead in reasoning to make the decision. In this research, we will focus on using more local reasoning, especially within an utterance, rather than within a dialog. In some applications, recognizing the correct determiner could be critical. For instance, there may be a big difference between *this target* and *that target*.

For the moment, we will put aside discourse phenomena and representing the meaning of the determiner. Instead, we will use this construction to ensure that the concepts referred to by the adjective and noun are semantically compatible, that we can connect them with an *is-a* link.

- **Person/number agreement** - One kind of grammatical error that this construction will not identify, is "a red dogs." "A" needs a singular noun. Other grammar formalisms enforce this as ungrammatical by using feature structures such as 'number=singular' and 'number=plural', and when a conflict is discovered, the parse fails. It is not yet clear to me how important agreement mistakes like this will be in speech recognition, and where agreement could be represented in the parser (e.g. as a feature in the form-side, or as a semantic conflict during instantiation?).
- **Ambiguity**—All parsers face a big challenge in ambiguity: phrase-structure ambiguity, lexical ambiguity, part-of-speech ambiguity. Ambiguity is even bigger challenge for us. In addition to typical kinds of ambiguity we have:
  - Word meaning ambiguity (i.e. word-sense disambiguation). Since we actually instantiate the meaning, we must confront this somehow. We might either:
    - Disambiguate
    - Maintain the ambiguities throughout the parsing process
    - Represent the meaning only approximately (e.g. as a vacuous *thing* concept, the lowest common ancestor of the concepts, or by creating a new union-class of all the concepts)
  - Concept segmentation ambiguities (e.g. "white house" → "*white*" "*house*" or "*white house*")
  - Partial-parse ambiguities (is it "we certainly don't row a placement bureau" or is it just "we certainly don't row")
  - Sentence boundary ambiguities (does this noun-phrase go with the verb before it or the verb after it?)
  - Disfluency skipping ambiguities (what does "I want, I don't want, a side salad" mean? Do we "skip" the "I want" or the "I don't want"?)

  Some of these ambiguities will be disambiguated by and during the parsing process. For instance, information in that latter part of the sentence might disambiguate the meaning of a word (or provide a preference for one interpretation over another), as in "the bank was covered with turtles" vs. "the bank was full of accountants."

  However, we can never expect all of the ambiguities to just "go away" as they often seem to magically for human language understanding. Perhaps the best we can hope for are some good heuristics, well-written constructions, and relatively well-behaved speech to minimize the ambiguity.
- **Context**—A significant part of language understanding requires context. We do not address this directly, but we do address it indirectly. That is the context may be at least partially specified and represented in the knowledge base.
- **Word sense disambiguation**—Again we do not address this directly, but we get some WSD free. That is, for example, several sense of a single verb may have different valences and slot restrictions. Finding a match for that verb will give us some information about which verb was used.

- **Metaphor and figurative language**—these fall apart under all of the techniques mentioned above. We simply do not have good computational models for these. All we can do is hope we do not encounter them, or avoid applications and domains where this language is common. (E.g. ASR of poetry is not something we will attempt to do).

4.4 Extensions

In addition to what is described above, there are a number of extensions to this that could be useful, especially in as ASR application.

Assembling a semantic representation with incomplete syntax—There are a number of other features that may be desirable for this work. For example, in speech, sometimes the syntax is so broken that it cannot be parsed. In these cases, it may be possible to piece together the meaning anyway, for example by checking if *any* of the noun-phrases in the vicinity of a verb *might* be a good subject or object. This is similar to what natural language understanding components of dialog systems do. That is, if you can determine the person want flight information, and you can determine the origin and the destination, then it doesn't matter what ordered they were spoken in. "To Boston ticket from Pittsburgh" is no different than, "I would like to buy a ticket the leaves from Pittsburgh and goes to Boston." However, performing a similar task when not restricted to a single domain may prove to be very computationally demanding.

Coping with unknown words—There is one very important and useful variation on the process I have described above, that I will only mentioned briefly here in this proposal. That is: what do we do if one of the words in not in the KB? If the *conceptOf*() function returns false? Should we try to represent anything if we see/hear "a asdknasdn dog" or "the white asddsdf"? One option would be to just "give up" and abort the matching process. Another would be to represent as much as we can. Thus, the representation of "a asdknasdn dog" would be the same as if we had just seen "a dog," and "the white asddsdf" would be represented as just "the white thing."

Even, having only the type dog, or only the property yellow, may give us critical information that we can use later (e.g. "the white asdknasdn is red" might have been a misrecognition of "the right asdknasdn is red"). I believe that this kind of partial representation, will be especially useful for us, when recognizing speech. It means that the KB does not need to contain every word that is spoken, and it gives us a way to use these techniques even when one or more words are masked by noise or are not recognizable acoustically.

5 Using semantics in ASR

In the previous two sections, we saw how to formally represent semantics—concepts, facts, and relations in the real-world—then we saw how identify and construct semantic representations from actual sequences of words. Next, we look at a few ways that we can *use* the extracted semantic representation in *speech recognition*.

The guiding assumption we make here is that people speaking will attempt to make sense, but incorrect speech recognition hypotheses will often *not* make sense. Neither of these assumptions is always correct: people sometimes do speak nonsensically, and sometimes incorrect ASR hypotheses do not make sense. However, we expect that these will be true more often than not.

Using this assumption, we will attempt to prefer speech recognition hypotheses that appear to *make sense* more so than those that *do not make sense*. In particular, we will

attempt to derive both positive and negative evidence from the semantic representation, for example:

- Utterance is meaningful and consistent with the KB (positive)
- Utterance in meaningful but inconsistent with the KB (negative)

Additionally, wherever possible, we will attempt to quantify this KB evidence.

In many cases, there will be no evidence that a snippet of text is either consistent or inconsistent with the knowledge base. In such cases, our technique defers completely to the acoustics and the statistical language model.

At this point, we will limit the discussion to deriving positive or negative evidence for particular (and complete) speech recognition hypotheses, not how to use that evidence. In the section following this, we will see how to use this evidence relative to other hypotheses, and in conjunction with acoustic scores.[4]

5.1 Topic affinity

Here is a simple test for the reader. Which of the following lists "goes together" better, or has higher *topic affinity*?

1. Spain, Italy, Austria, France, United States.
2. Spain, Bennett Lee, Austria, France, United States.

Most people would probably choose #1 because each of the concepts listed is a country. In #2, *Bennett Lee* does seem to fit because it is not a country.

We will use the idea of topic affinity in a similar way for ASR. That is, even without considering the grammatical and semantic relations among the concepts mentioned in the sentence, sometimes they just fit together better than others. For instance, we could use the same rationale as above to choose the 44th ranked hypothesis in the following example from the WSJ speech dataset.

1. Germans **big** Spain **Bennett Lee** Austria France and the U.S. (051c0313)
…
44. Germans pick Spain then Italy, Austria, France, and the U.S.

In this case, defining topic affinity seems trivial; they are all instances of the same type. However, examples like this are not common in language. Even if they are not instances of the same type, some other concepts have very high affinity even out of any linguistic context. *Jockey* and *horse race* are one example of this; *Garden of Eden* and *serpent* are another example of this.

In other cases, it may not be as straightforward. For instance, which of the concepts *hockey* and *jockey* has higher affinity with the concept of *sport*? Perhaps it depends who you ask. Thus, in general, topic affinity can be difficult to define either in human terms or computationally.

---

[4] The examples shown in this section are written and in some cases punctuated. In an actual ASR system there is no such punctuation or capitalization. Some punctuation may be represented acoustically (Selting, 2007), but this is very different than written punctuation and is not readily available anyway.

It is common to approximate some measure of affinity by computing word co-occurrence statistics, or by counting the number of logical links through a knowledge base to get from one concept to another. We believe these approximations are insufficient, and propose to define an alternative. We believe rather than counting links, a more effective measure of topic affinity will use links in the KB much more selectively.

One possible alternative to counting links is to only consider those entities that exist in a single frame, event, or description to have high affinity. For instance, we might say that the main participants in the Garden of Eden story all have high affinity (Adam, Eve, God, the serpent, and the apple), but that more minor participants (e.g. the fig tree) do not. Another possibility would be to try to learn the truly useful/relevant affinities from data. In this research, we will explore using these more focused affinities.

5.2 Semantic slot constraints

As discussed in section four, constructions and specific event representations, may have type constraints on their participants. Here we use those constraints to derive evidence for or against a speech recognition hypothesis. Consider to the n-best list for the spoken sentence:

> *"We certainly don't run a placement bureau says Charles Heck the North American director."*

1. We certainly don't **row** placement bureau says Charles Heck the north American director.
2. We certainly don't **roh** placement bureau says Charles Heck the north American director.
3. We certainly don't **roe** placement bureau says Charles Heck the north American director.
4. We certainly don't **rowe** placement bureau says Charles Heck the north American director.

…

22. We certainly don't run a placement bureau says Charles Heck the north American director.

In this example, the verb run (in the transitive *to operate* sense) is acoustically mistaken for the verb *row* (as well as some non-verbs). However, the grammatical object here, "placement bureau," is a clear indication that *row* is incorrect. The verb *row* is very specific and usually only takes an object that is some kind of boat.

A *placement bureau* is not something that can be *rowed*, but it is something that can be *run*. Thus, we have negative semantic evidence for hypothesis #1, and positive evidence for hypothesis #22. We likely do not have any semantic evidence for or against hypotheses #2 through #21 because they do not parse. We may or may not want to use the fact that they do not parse as negative evidence (e.g. "roh" might be a verb that's not in our KB/lexicon). In either case, positive evidence for #22 may boost it past hypotheses 2 through 21.

Notice that we are able to derive this evidence using only a partial parse of each hypothesis. That is the parse only covers the first seven words of this utterance. We need not be able to parse the entire sentence to derive this positive and negative evidence. As a consequence, we will derive the same evidence even if the latter part of the sentence is completely misrecognized (which may be very common when recognizing speech).

Also, notice that, in this example we do not need a complete interpretation of the phrase to prefer the correct hypothesis. That is, even if we do not model the semantics of "don't" and "certainly," we are able to derive the evidence we need. We believe this may be a very useful fact, since it can be very challenging to model some language, such as the semantics of modals, or the scope of negations.

5.3 Proposition plausibility

By proposition plausibility, we mean: given a semantic representation for some language, that is *consistent* with the knowledge base, is it plausible? What does the knowledge base tell us about how plausible this particular proposition is? There are many ways we could quantify this plausibility. I will sketch one of them here. But regardless of how this is defined, it may be somewhat heuristic anyway. Perhaps using several measures of plausibility using some heuristics to combine or prioritize them would be most effective.

A proposition could take several forms, for instance a "green idea" IsA(*idea-22*, *green*), or a binary (or trinary, or *n*-ary) event relationship such as: in "we run a placement bureau." The goal is: given a proposition (an instance of a relation or event) that has been identified in the speech, is it plausible/likely or not?

The first step in determining this is to look for other similar instances. An important and unclear question is how to define "similar." If the proposition we are given is "John loves Mary," do we look for other instances of John loving Mary? Or, do we look for other instances of one *person* loving another *person*? Or a mammal loving another mammal? Or, for other instances of John *showing affection for* Mary? Perhaps these questions are best answered empirically.

Once we have decided what it mean to be similar, what do we do after we find all the similar instances? If there are none, does that mean this proposition is implausible? Perhaps we just have not seen it before? If there are just a few does that mean it is rare, or OK? Perhaps we need to compare the frequency that we have seen this with whatever the alternatives are? If we have seen similar events many times before, does that mean this is definitely plausible? Perhaps not?

Finally, since our KB allows exceptions, what if the proposition in question is forbidden by the semantic constraints in the KB, but we have seen one or more exceptions? Similarly, what if we have encountered similar instances in a very different context? (E.g., it has never been seen before in the real world, but has been seen in the Harry Potter world). Again, it is not clear what the most effective answers might be; these are more questions that we need to answer empirically.

5.4 Additional methods

In addition to these methods, we can try anything that we can formulate computationally with constructions, and with the KB. For instance, using Scone we can easily consider the validity/plausibility of speech in different context (e.g. Harry Potter world, or specific dialog contexts).

Additionally, we could construct very specialized methods for specific domains. If the speech we're recognizing contains a lot of arithmetic (and it is presumed to be correctly stated), we could write constructions for speech like "X times Y is Z," for which the reasoning involves checking if the arithmetic is correct.

For this research, we will focus on the three methods described above: topic affinity, semantic slot restrictions, and proposition plausibility. We will explore other methods as needed, but we believe these three would be able to identify a significant portion of the identifiable *semantic* errors in ASR.

The specific methods chosen and how each is used, will affect performance in different ways, and each will bias the system in different ways. That is, each method that provides negative evidence will bias the system against recognizing utterances of that form, and each method that provides positive evidence will bias the system toward recognizing utterances of that form.

The precise behavior that one desires from the system depends on the application. Thus, the specific kinds of knowledge, constructions, and evidence that are used depend on the domain and typical speech patterns used in the application. The bias in effect allows one to tune the system such that it "tries" to hear some meaningful patterns more than others do.

5.5 Prevalence and an upper-bound of these

We do not currently have statistics of the prevalence of the errors these methods identify. From my own experience pouring through WSJ n-best lists, I would guess that for the verb-argument techniques (slot restriction and proposition plausibility), with a reasonably easy to define KB, we might be able to identify the incorrect hypothesis and the correct one in the top-n (for some small n) maybe once every 100 or so sentences.

What appears much more common, is being able to identify the correct or the incorrect one, but with the correct one being "too far" on the n-best list, or not on the n-best list at all. For these, we likely would not be able to improve WER. However, we might be able to be move the correct hypotheses (or *more* correct hypotheses) up the n-best list. In the WSJ data, these might occur one for every 10 or 20 sentences. I have not yet looked at the prevalence of these in other datasets.

Thus, the upper-bound on the improvement that these techniques may be able to provide may be small in terms of absolute word error rate reduction. An early experiment to perform in this research would be to gauge some of these numbers. We will be able to do this both with our own data analysis, as well as experimentation with services like Amazon Mechanical Turk (as described in the results section of this proposal).

6 Integrating semantics into ASR

In the previous section, we saw how to derive positive and negative evidence for or against a speech recognition hypothesis. Now we look at how to integrate this evidence into the speech recognition system.

We can formulate the overall goal of the speech recognizer as follows. The goal is to find the word sequence, that:

1. Acoustically matches the audio signal, and
2. Fits some notion of what a person might have spoken

Everything I have described up to this point falls under point #2: we want to determine if the word sequence is something that a person is likely to have spoken. In particular, we have considered addressing #2 by identifying what *makes sense* and what does not.

Next, we will look at how to combine #2, using the semantic notions we have looked at, as well as a statistical LM, with #1. We begin with a little background on speech recognition.

6.1 Automatic speech recognition (ASR) overview

Ultimately, the goal of ASR is given some audio to determine what was spoken. Typically, the problem is framed as: what *words* were spoken?[5] The problem is usually formulated

[5] Depending on the application or context the ideal answer to what was spoken may not be a sequence of words. For instance in a dialog (automatic or human-to-human), we may not need to hear every single word to understand the information and meaning being conveyed.

probabilistically and becomes: *given some audio, what is the **most likely** sequence of words that was spoken*.

The mathematical notation for this is as follows. Here the w_1, w_2, w_3... are words, and "Audio" is the audio:

$$w_1 w_2 w_3 ... w_n = \arg\max_{all\ \ word\ \ seqences}(P(w_i w_j w_k ... w_m \mid Audio)$$

That is, given some audio, find the sequence of words that is most probable.

Usually, the conditional probability above, the probability of a sequence of words given some audio, is split into several pieces using Bayes' rule. Using Bayes' rule, it becomes:

$$P(w_1 w_2 w_3 ... w_n \mid Audio) = \frac{P(Audio \mid w_1 w_2 w_3 ... w_n) P(w_1 w_2 w_3 ... w_n)}{P(Audio)}$$

In the maximization above, *P(Audio)* never changes, so this simplifies to:

$$P(w_1 w_2 w_3 ... w_n \mid Audio) \approx P(Audio \mid w_1 w_2 w_3 ... w_n) P(w_1 w_2 w_3 ... w_n)$$

The right-hand side of this has two parts. The first, $P(Audio \mid w_1 w_2 w_3 ... w_n)$, is the probability of the audio given the sequence of words. This is the acoustic model, a generative acoustic model. The second, $P(w_1 w_2 w_3 ... w_n)$, is the probability of the sequence of words. This is the language model. This determines, how likely the sequence of words is, in the language.

The first step in building many ASR systems is to combine the two models, the acoustic model and the language model, into one very large graphical model. Typically, this is a hidden Markov model (HMM). The diagram below, Figure 10, is an extremely simple language HMM for the language that consists of two sentences: "Rock star" and "dog star."

The probabilities in yellow (e.g. *P(Dog)* and *P(Star|Dog)*) come from the language model. The three little circles in each of the three boxes below are small hidden Markov models (HMMs) of the individual words. Each of these little HMMs models the acoustics of a single word. (The top green box represents the acoustic model for the word "dog"). That HMM, given an actual audio signal (or features computed from one), outputs the probability that the given audio is someone speaking "dog."

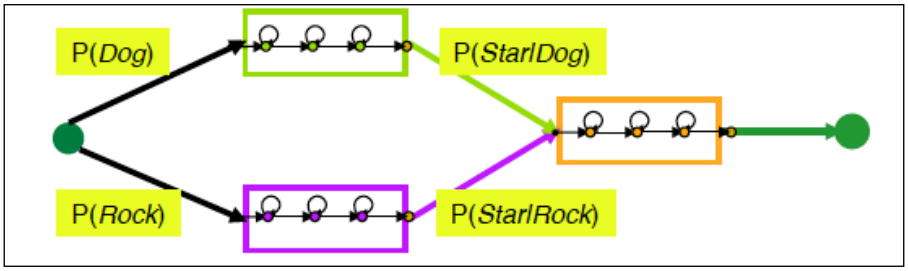


Figure 10 - A language hidden Markov model for a very simple language (Raj, 2009).

Most ASR systems contain within them a data structure that resembles the graph above but very much larger. Each path through the graph represents a sequence of words. Given some audio, an ASR system tries to find the path through the graph that has the highest probability of matching the audio. But rather than explicitly look at every possible path through this graph in place, the graph is expanded into a larger graph called a "trellis" which is more efficient to search.

The trellis is a rectangular graph where the vertical nodes correspond to each node in the original HMM graph (like the one above), and the horizontal nodes represent each time point in the given audio signal (Figure 11). Thus, each node in this graph represents being in a particular state of the above HMM graph after having "consumed" a portion of the audio.

Speech recognition is performed by finding the shortest (or lowest cost) path from the lower-left node to the upper-right node. That path represents the most likely word sequence for the given audio, according to the acoustic model and the language model. This path can be computed efficiently with a dynamic programming algorithm, because the best (shortest) path to any node in the trellis only needs to be computed once.

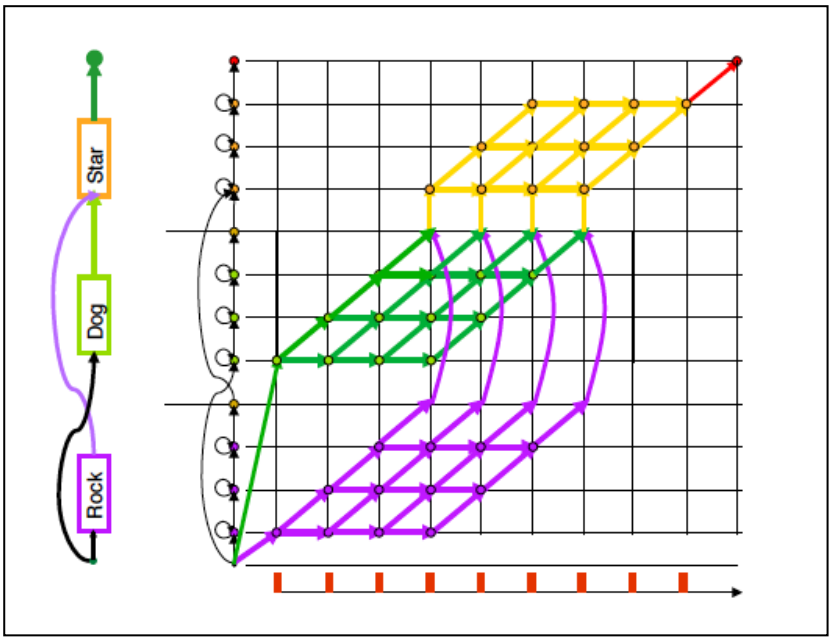


Figure 11 - A "trellis" search graph (Raj, 2009).

During the search of the trellis, the algorithm computes a score for each node which; this score is the shortest distance to that node from the lower-rightmost node. Additionally, the algorithm maintains a data structure that represents all of the shortest paths arriving at that node. Thus, at any given node, the algorithm has references to all the shortest paths to that node. At any given node in the graph, the probability of the path leading up to that node is:

$$Score = P(Audio \mid w_1 w_2 w_3 ... w_j) P(w_1 w_2 w_3 ... w_j)$$

To simplify the computation, we usually take the logarithm to end up an equivalent equation. (The lambda coefficient here is a scaling factor, between the acoustics and the language model):

$$Score = \log(P(Audio \mid w_1 w_2 w_3 ... w_n)) + \lambda \log(P(w_1 w_2 w_3 ... w_n))$$

The process of computing this shortest path is called *decoding*, and the decoder constructs a set of partial hypotheses as it searches the trellis.

Typically, the probability of the word sequence, $P(w_1 w_2 w_3 ... w_j)$, is simplified in a few ways. First, rather than compute the probability of an entire sequence of words, the probability of the subsequent word is computed as a conditional probability, given the all of the words that occur before it, P (w_n | w_{n-1}, w_{n-2},… w_1). This is further simplified using a Markov assumption that the probability of each word depends only on those immediately before it. In an n-gram language model, each word's probability depends on the previous n-1 words. Thus, the probability above is approximated for a k-gram LM as:

$$P(w_n \mid w_{n-1}, w_{n-2}, \ldots w_1) \approx P(w_n \mid w_{n-1}, w_{n-2}, \ldots w_{n-k})$$

This, of course, has nothing close to syntax or semantics in it, and this is what we hope to improve upon by adding semantics to the ASR system.

However, at this point in the computation, this n-gram LM is approximating the probability of a word given all of the words before it, and the algorithm maintains a structure containing all of the paths leading up to that point in the trellis. Thus, it is possible to use information from all of the words leading up to each node in the trellis during the process of decoding.

The set of partial hypotheses is represented as a tree of words. The decoder's job is to construct the tree. Figure 12 is an example of a partially-constructed decode tree. Each edge (or node) also has an associated probability (not shown).

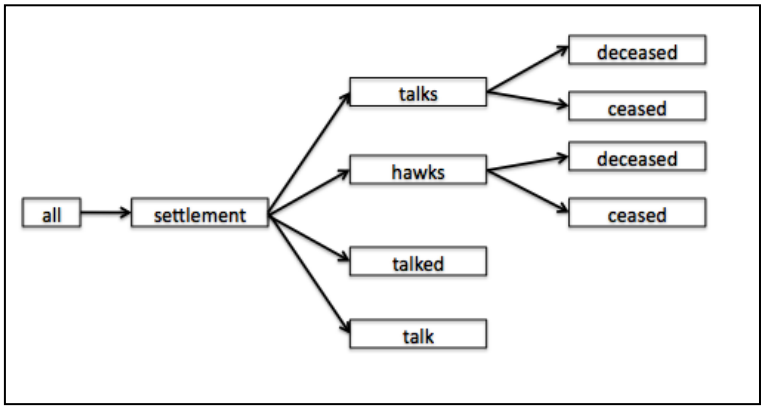


Figure 12 - A partially constructed hypothesis tree, during construction by the decoder.

When the decoder is finished, we end up with a *complete* tree, with every leaf node connected to one terminal node "<\s>" (as in Figure 13):

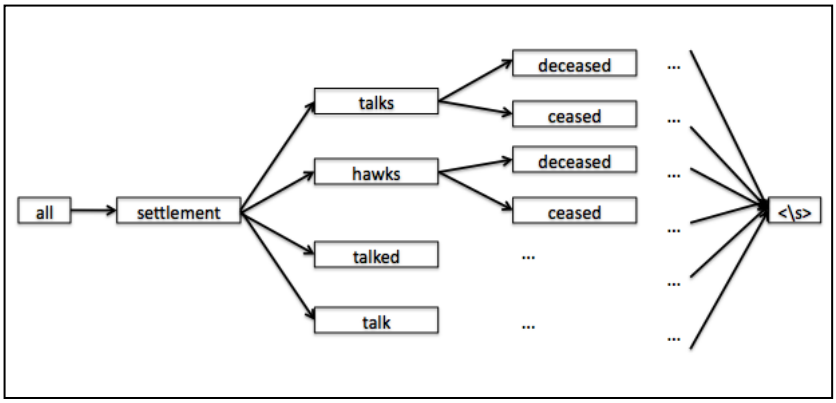


Figure 13 - A complete hypothesis tree as constructed by the decoder.

At this point, when the decoder reaches the end of the utterance, it could stop and return the most probable path. However, sometimes we can find a better sequence of word by performing post-processing on the most likely word sequences.

There may be redundancy in this tree that can be condensed. We condense, and possibly prune, the tree and to end up with a *graph*, a directed acyclic graph. This graph represents all hypothesized word sequences. This graph is called a word lattice (Figure 14) (node/edge probabilities not shown).

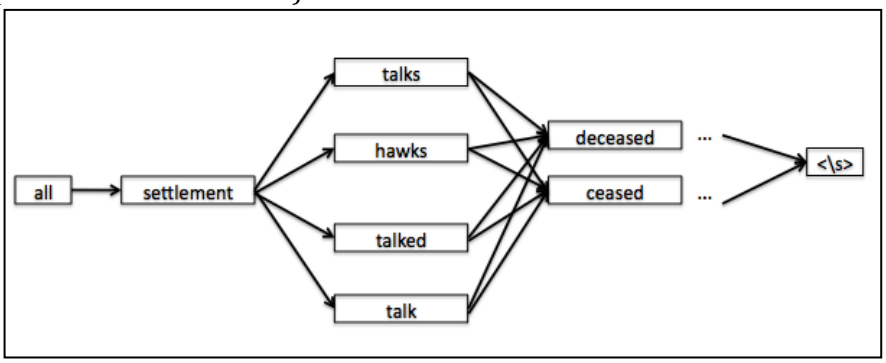


Figure 14 - A word lattice graph that represents all ASR hypotheses.

It is also common to use produce a list of the top-*n* paths through the graph. This list of top-*n* paths is called the *n*-best list. Each hypothesis on an *n*-best list has an associated score, the estimated probability that it was spoken. Figure 15 shows the 10-best list for the example above.

N-best lists are easy to work with because each hypothesis is complete. This makes it easy to perform whole sentence post-processing, such as running a parser, on each of the *n*-best.

#	Hypothesis	Score
1	ALL SETTLEMENT TALKS DECEASED AFTER THE FILING PERIOD	-17639530
2	ALL SETTLEMENT HAWKS DECEASED AFTER THE FILING PERIOD	-17722405
3	ALL SETTLEMENT TALKED DECEASED AFTER THE FILING PERIOD	-17856712
4	ALL SETTLEMENTS HAWKS DECEASED AFTER THE FILING PERIOD	-17948226
5	ALL SETTLEMENT TALKS THE DECEASED AFTER THE FILING PERIOD	-17982869

6	ALL SETTLEMENT TALKS THE DECEASED AFTER THE FILING PERIOD	-17983709
7	ALL SETTLEMENT TALKS A DECEASED AFTER THE FILING PERIOD	-18049775
8	ALL SETTLEMENT HAWKS THE DECEASED AFTER THE FILING PERIOD	-18065744
9	ALL SETTLEMENT TALK THE DECEASED AFTER THE FILING PERIOD	-18077603
10	ALL SETTLEMENT TALKS CEASED AFTER THE FILING PERIOD	-18112330

Figure 15 - An *N*-best list

6.2 Using semantic evidence in an ASR system

In the last section, I described the basic function of a speech recognizer. The next question we will consider is how to incorporate the methods described in section 5, such as topic affinity, and proposition plausibility into the ASR system. There are two primary considerations here:

1. *Where* to incorporate semantics, and
2. *How* to incorporate semantics

In this section, I will propose three methods to answer each of these questions of where and how. For each of the six locations and methods, we will consider the pros and cons.

The three locations *where* will consider incorporating semantics are:

1. Directly in decoding (while searching the trellis)
2. Re-scoring the word lattice, or
3. Re-scoring and re-ranking *n*-best lists.

And, the three methods for *how* to incorporate semantic evidence are:

1. Semantic evidence as hard-evidence—e.g. prune hypotheses for which there is negative evidence
2. Linear score interpolation—quantify the semantic evidence and combine that with the ASR score in a linear combination.
3. Maximum entropy model—this also combines semantic evidence with ASR scores, but explicitly recognizes that the knowledge ands semantics may be incomplete.

In general, all of the methods for incorporating semantics will either explicitly accept/reject hypotheses, or will adjust the acoustic and LM scores.

6.3 *Where* to incorporate semantic into ASR

There are at least three locations in the ASR process in which the semantic evidence I have described may be used. The three steps of the process in which we will look at are:

- During decoding
- Post-processing word lattices, and
- Post-processing n-best lists

Figure 16depicts the locations in the overall process where semantics may be used. Post-processing of the word lattice performed in the lattice re-scorer; post-processing *n*-best lists is performed by the *n*-best re-ranker.

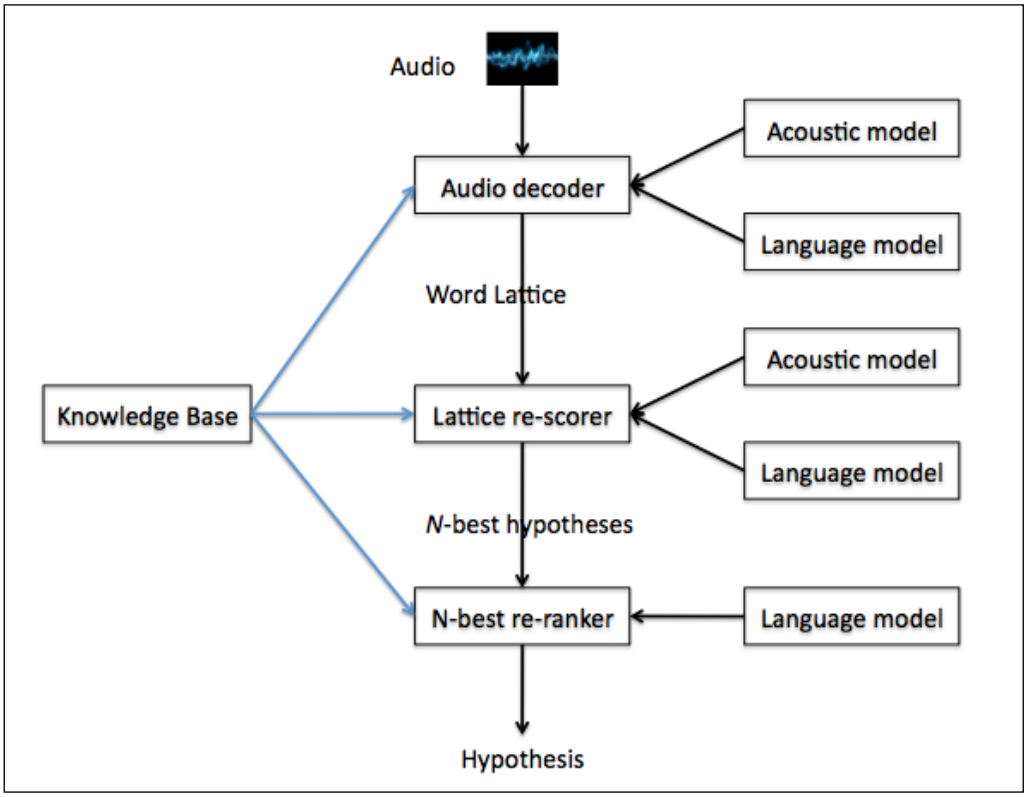


Figure 16 - Locations where semantics may be used in ASR

6.3.1 *N*-best list re-ranking

Recall that an *n*-best list is simply a ranked list of complete hypotheses, each with an associated score.

The methods for incorporating semantics that either accept or reject hypotheses are very simple to use in an *n*-best list. There is a subtle but important difference between using semantics to *accept* and to *reject* hypotheses. If the semantics were only used to accept hypotheses, then the process for using it in an *n*-best would be to simply scan from top to bottom until a hypothesis is accepted. When rejecting, we scan from top to bottom, until we reach a hypothesis that is not rejected.

Figure 17shows a 10-best list with each associated ASR score. KB evidence is shown as a check or and ex. The portion of each hypothesis that generated positive or negative semantic evidence is highlighted in bold. If we use positive evidence to accept, then hypothesis #10 would be chosen. If we use negative evidence to reject, then hypothesis #3 would be chosen.

The methods that combine quantified semantics with ASR scores are very simple also. For these, we first quantify the semantic evidence. We then combine that evidence with the score of each of the n best hypotheses, using one of the methods described in the upcoming section 6.4. Then we re-rank the *n*-best list by the combined score, and simply choose whichever hypothesis is at rank #1. The top-ranked hypothesis might be the originally top-ranked hypothesis, or some other hypothesis could have bubbled up to the top of the list.

#	Hypothesis	ASR	KB
1	**ALL SETTLEMENT TALKS DECEASED** AFTER THE FILING PERIOD	-17639530	X
2	**ALL SETTLEMENT HAWKS DECEASED** AFTER THE FILING PERIOD	-17722405	X
3	ALL SETTLEMENT TALKED DECEASED AFTER THE FILING PERIOD	-17856712	
4	**ALL SETTLEMENTS HAWKS DECEASED** AFTER THE FILING PERIOD	-17948226	X
5	ALL SETTLEMENT TALKS THE DECEASED AFTER THE FILING PERIOD	-17982869	
6	ALL SETTLEMENT TALKS THE DECEASED AFTER THE FILING PERIOD	-17983709	
7	ALL SETTLEMENT TALKS A DECEASED AFTER THE FILING PERIOD	-18049775	
8	ALL SETTLEMENT HAWKS THE DECEASED AFTER THE FILING PERIOD	-18065744	
9	ALL SETTLEMENT TALK THE DECEASED AFTER THE FILING PERIOD	-18077603	
10	**ALL SETTLEMENT TALKS CEASED** AFTER THE FILING PERIOD	-18112330	√

Figure 17 - Positive and negative semantic evidence for hypotheses on an n-best list

Pros and cons—The n-best list is the latest point in the speech recognition process that we can perform any processing. One advantage of this is that we can easily use other post-processing techniques before performing any semantic processing. Thus, we could use Google n-grams or a syntactic parser *before* performing any semantic processing. Running other post-processing methods before this may yield hypotheses for which we can more easily/accurately extract a semantic representation.

Another advantage to performing semantic processing at this stage is the ease of integration. Each hypothesis is simply a string of words to process; we simply give each string to the construction parser and/or the topic affinity processor, and combine the score from those with the ASR score and re-rank. We do not have to consider re-scoring paths in a graph.

However, having the "entire" hypotheses string is not all good. An "entire" hypothesis might look very different depending on how segmentation is performed. Our system does not require that hypotheses be complete sentences; in fact, it is specifically designed to be able to handle sentence fragments and incomplete speech. But this means things may be complicated when a hypothesis is very long. Then we will encounter issues with conflicting evidence for or against some hypotheses. Additionally, very long hypotheses are likely to contain more than one "sentence," and a lack of sentence boundaries can be a source of errors for us. Finally, running a parser on a very long string can be very computationally intensive.

However, there are also a number of disadvantages to using n-best lists. Each hypothesis must be processed separately, which may be very time-consuming. (This is especially exaggerated when the hypotheses are long or the *n*-best list is very long). When there are numerous ambiguous portions of the speech, these combinatorially explode the length of the *n*-best list. Not only does this cause us to perform a lot of redundant computation, but it may also cause the "correct" or at least the "good" ones to end up at an incredibly high rank on the list, such as rank 100,000. And since it is most likely not feasible to process so many hypotheses, there is a good chance the correct and/or good hypotheses, were pruned from consideration long ago (e.g. also during decoding).

6.3.2 Word lattice re-scoring

Since our semantic evidence often will only apply to part of a speech recognition hypothesis, re-scoring a lattice gives us a good opportunity to re-score good or bad portions of the hypotheses without the computational redundancy that may occur in n-best re-ranking. We simply identify the portions of the lattice for which we have positive or negative semantic evidence and modify those accordingly. For example, the lattice shown in figure 18, contains a segment for which we have negative semantic evidence. We can easily use this evidence by lowering the score through that path. It is of course non-trivial to identify the portions for which have evidence, especially when the lattice consists of hundreds of thousands of nodes.

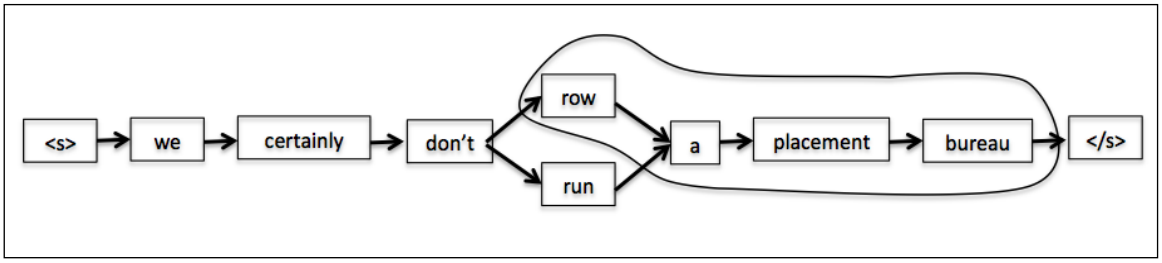


Figure 18 - Negative semantic evidence in this lattice, for "row a placement bureau."

However, in many cases, we may need to modify the graph so that changing the score of one partial path does not affect the scores of any other path that does not contain that path. For instance, in figure 18, we have identified negative evidence for the snippet that is circled. To prune or re-score this portion of the lattice we must separate it from the other path through the lattice ("run a placement bureau"), as shown in figure 19, so we do not affect its score.

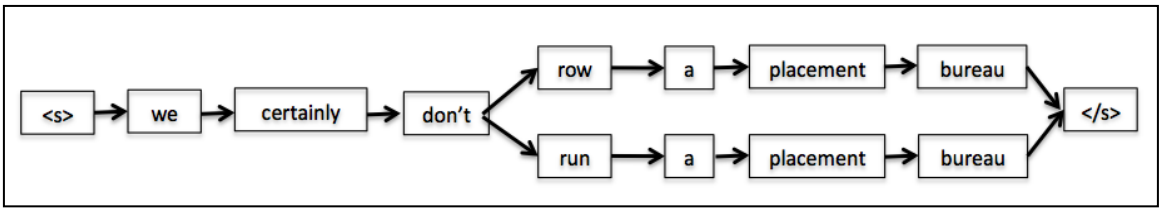


Figure 19 - The lattice, after splitting "row a placement bureau" from "run a placement bureau."

If the semantic model is used to accept/reject hypotheses, then we use it to *prune* the word lattice. When *rejecting* hypotheses, we prune all the partial hypotheses that violate the semantic constraints. When *accepting* hypotheses, we prune all but those paths that are accepted by the semantic model.

Each edge (or node) in a word lattice has an associated score, assigned to it by the decoder (derived from the acoustic and language models). Thus, each path through the lattice also has a score, the cumulative score of the nodes/edges along that path. After identifying positive or negative semantic evidence along some path in the word lattice, we can use that evidence to alter the score of the associated path

Pros and cons—Lattice rescoring is more challenging to implement than n-best re-ranking for a number of reasons. It is more difficult to identify evidence (e.g. parsing a word lattice is possible, but a non-trivial extension of text parsing), and it is more difficult to

incorporate that evidence (e.g. we may need to add/delete/change nodes in the lattice, and altering the scores too much or too little could negatively affect the lattice search algorithm).

However, since many of the redundancies in the n-best lists may be collapsed, we can do not need to repeat as much work as when re-ranking n-best lists. This means we can process more of the hypothesis space in less time. Additionally, the word lattice may represent many more hypotheses than a reasonably sized n-best list. Thus, there is a better chance that the correct/good hypotheses have not been eliminated from consideration from the lattice, than from *n*-best lists.

However, a lattice is still a pruned representation of everything that could have possibly been spoken in the audio. So there is still a chance that the correct/good hypotheses are not even present in the lattice—the only way we can avoid the possibility of eliminating those entirely would be to integrate into the decoder.

6.3.3 Decoding

As the trellis is searched, the algorithm constructs a tree of hypotheses, it computes the best path and highest probability node for each column of the trellis. The probabilities typically depend only on the score of the previous node in the path and the probability of the transition from that node to the current node. However, at this point in decoding the decoder has access to all paths to the node that is being processed.

The fact that during decoding we have access to the best paths up to a node, means that we can use this history to guide the search. That is, as the algorithm considers each node in the trellis, it can examine the words that led up to that node. By looking at these words, the algorithm can adjust the score of each node, using semantic evidence, "on the fly."

If this can be implemented cleanly, it gives us an opportunity to effectively prune semantically inconsistent paths through the lattice as soon as they occur. Thus, in figure 12, a partially constructed decode tree, as soon as we encounter "hawks deceased" we can stop searching paths that extend those that path. Or we can re-score those paths so that they may be considered again, only the alternatives are even worse.

Pros and cons—Integrating semantics into the decoder is the most challenging of all of these to implement. However, if it can be implemented well, it has the potential to achieve the greatest gains in performance. In theory, if we can catch semantic errors as soon as possible, we can ensure that only semantically valid hypotheses remain when decoding is complete. Of course, "garden path" sentences could mislead this technique, and cause the correct hypotheses to be eliminated from consideration.

But, although it is in decoding where the greatest potential lies, the adverse effects could be very severe. Any imperfections in our techniques will be magnified when used at this level, and may introduce errors that cannot be recovered from. Thus, it is in decoding that we may find the biggest gains, but also the largest risk.

Integrating any new technique for ASR into decoding is sufficiently challenging that it is often not attempted at all. That is, one can often learn a lot about new techniques, and whether they *might* work in a decoder simply by experimentation with n-best lists and/or lattices. I will attempt to actually implement this in the decoder because, among other reasons, I will be implementing a decoder in Lisp for class-work anyway. (Lisp is the programming language in which Scone and this work is implemented). The recognizer will be rudimentary, and so experimental results may not be very interesting, but it will give us a sense for some of the good and bad things that could occur by integrating into the decoder.

But, implementation details aside, an extremely interesting set of research questions is: how much can very early identification of semantic errors help? In principle, it could help a

lot, because when the KB coverage falls off, the statistical LM would take over. But if the semantics can give the LM the "correct" *n-1* words, then the LM can take over from there. As much as possible, we would like to get a sense for the bounds on performance change with using semantics in the decoder, experimentally.

6.4 *How* to incorporate semantics into ASR

We now consider *how* we can incorporate our proposed semantic processing into the ASR process.

6.4.1 Semantic evidence as hard evidence

If the knowledge base and grammatical constructions are known to be very accurate and consistent with the language, then they may be used as hard constraints.

For instance, if the semantic representation extracted from the speech is inconsistent with the knowledge base (i.e. violating some KB constraint)—since we know the KB is correct—then we could discard all hypotheses containing the inconsistent part (perhaps a "fast-fail" method). Likewise, if a construction match provides positive evidence we might in turn discard all hypotheses that do not contain that language ("accept immediately").

Such hard constraints might be most effective in domain-specific application or dialog system. In fact, in many ways, this may be identical to how some dialog systems work.

In a way, the fast-fail method could be likened to a parse failure of a semantic grammar. The "accept immediately on identification" approach could be likened to spotting bits of information that the system is requesting (e.g. like a case-frame parser). In either case, if the constraints are enforced in a carefully controlled way, looking at highest scoring hypotheses first, then we should be able to guarantee that the hypothesis accepted is the highest scoring hypothesis that is consistent with the knowledge-base.

Pros and cons—One of the benefits of this method is that we have the most control over the system. This means it may be the most appropriate for a system like a dialog system. In addition, it may be the best method if we have very specific constraints that we can impose with very high confidence. For instance, if this were used in a desktop dictation program, and the program was consistently making the same error, we can use this to permanently prevent that error from occurring again.

However, if we are wrong in what we say is allowed and not allowed, then we will be irrevocably hurting the performance of the system. People very often overstate their knowledge; that is, they might think they are encoding knowledge that will always be true, but are wrong, and just have not anticipated some of the alternatives. Thus, for many applications it would be preferable to use a more flexible manner of incorporating the semantic evidence.

6.4.2 Linear score interpolation

In the more likely case that the knowledge base and constructions are incomplete, we will need to combine the knowledge-base evidence with the acoustic and language model scores.

Some of the proposed methods of using the KB to derive evidence provided non-numeric evidence, such as a binary judgment—consistent/inconsistent with the KB. This kind of evidence needs to be quantified to be used here. The very simplest thing to do would be to map these to a 0/1 or -1/1 value. Other methods for quantifying these could be explored; the best ways of doing this may depend on the application or the data. For instance, some evidence might be more important than other evidence.

Perhaps the simplest way to combine this quantified evidence with the acoustic and statistical language model scores is a linear interpolation. This simply means adding them,

but also scaling the scores up or down appropriately, based on the importance and reliability of the evidence. The magnitude of the scaling is usually determined by optimizing the magnitudes in a training dataset. The optimal values can be determined using an expectation-maximization style algorithm.

Here we will only look at scoring an entire hypothesis or sentence, however the same equations can be applied to individual words or phrases in the hypothesis. The linear combination is defined simply, as follows. Here $P_i(S)$ is the *score* given by the *i*th source of evidence, and λ_i is the computed optimal weight for the ith source of evidence.

$$P_{COMBINED}(S) = \sum_{i=1}^{k} \lambda_i P_i(S)$$

If we were to use the three KB sources of evidence proposed in section five, this equation would become:

$$P_{COMBINED}(S) = \lambda_1 P_{Acoustic\&LM}(S) + \lambda_2 P_{TopicAffinity}(S) + \lambda_3 P_{Slot\,Rest}(S) + \lambda_4 P_{PropPlaus}(S)$$

However, we may prefer to divide the sources of evidence into more fine-grained groups (e.g. one source of evidence might be proposition plausibility for *row* events, and another might be proposition plausibility for *run* events).

Pros and cons—The big advantage to this method is that we can incorporate semantic evidence simply but not absolutely. The semantic evidence is balanced against all the other evidence. It is also very easy to implement.

If we learn the parameter weights from some training dataset—and the training data is big enough and distributed similarly to unseen data—then this method cannot hurt either. That is, the optimal weight parameters for those sources of evidence which are not reliable or which are not helpful, will be determined to be zero. Another advantage to this method is that it gives us an easy way to manually adjust the importance of various sources of evidence. This could be useful in an interactive application or in a domain specific application where we know more about the knowledge sources.

The disadvantage to this technique is that a linear combination may not capture interdependence of knowledge sources, and may drive their weights too high or too low. Additionally, if the contribution of the other knowledge sources in non-linear they may not be able to be used efficiently, if at all.

6.4.3 In a maximum entropy model

Maximum entropy modeling (MEM) is a technique for combining diverse sources of evidence into a single model. The principle that underlies this technique is that the model makes no assumptions and has no bias for or against any unseen information.

This model appears to be particularly well-suited to our own assumptions—that the KB is incomplete and that the NLU may be imperfect. Thus, MEMs would appear to be ideal for this task.

Maximum entropy models can be used in the standard conditional framework for computing *P(w|h)*, the probability of a word given its history (Rosenfeld, 1996). Whole sentence maximum entropy models (Rosenfeld, 1997) perhaps provide a better framework in which to apply our techniques because the semantic representations we identify may span words or the entire utterance.

A whole-sentence maximum entropy probability, is defined as follows. Here s is a sentences $P_0(s)$ is the standard *n*-gram language model score, $f_i(s)$ are features of the sentence, λ_i are the feature weights, and *Z* is simply a normalizing constant.

$$P_{ME}(s) = \frac{1}{Z} \cdot P_0(s) \cdot e^{\sum_i \lambda_i f_i(s)}$$

The key to maximum entropy modeling is the way in which the λ_i parameters are determined. These are chosen such that the expected value of $f_i(s)$ in the probability distribution is the same as the expected value in the training data, and beyond that constraint, all other features values and combinations occur as uniformly as possible.

The knowledge-base evidence comes in several forms. It could be a direct representation of a semantic conflict with the knowledge base. For instance, we could define a binary feature $f_{semantic\_conflict}(s)$ that is equal to one if we found some language that produced a meaning representation that is inconsistent with the knowledge base, and zero otherwise. Additionally, we could have a feature for when we find a semantic representation that is consistent with the knowledge base $f_{KB\ consistent}(s)$.

Perhaps even more interesting would be features generated from the meaning representation themselves. For instance, if "we row a placement bureau" occurs in a hypothesis, then the semantic representation: *row(we, placement bureau)* is created. This could be used directly as a feature, as in $f_{row(we,\ placement\ bureau)}=0/1$. This sort of feature resembles the "constituent trigram features" in Zhu et al. (1999). The advantage of working in a knowledge base is that we can generalize the each component of this proposition. For instance, *row(we, placement bureau)* could generalize into any of the following, and more: *row(person, placement bureau)*, *row(we, agency)*, *row(person, agency)*.

Another feature form *admire*(*Mr. Sciarra*, *Tony Provenzano*), features might generalize to: *admire*(*person*, *Tony Provenzano*), *admire*(*Mr. Sciarra*, *person*), and *admire*(*person*, *person*). The alternate hypothesis for this particular sentence *admire*(*Mr. Sciarra*, *twenty pro ovens on all*), could generalize to *admire*(*person*, *oven*) or *admire*(*person*, *kitchen appliance*). We can even generalize the verb in this case, and end up with something like *expressEmotionFor(person, kitchen appliance)*.

Most previous work using maximum entropy models in ASR has used word-level or syntactic features. Roni Rosenfeld, who began this work on incorporating semantic and sentence-level features into a maximum entropy model (and a member of this thesis committee), concluded in 2001 that many word-level features contribute negligibly to the performance of a maximum entropy language model because these word-level features do not occur frequently enough to have a significant impact. He wrote, in 2001:

> *Thus, we need to concentrate on more common features. An ideal feature should occur frequently enough, yet exhibit a significant discrepancy. "Does the sentence make sense to a human reader?" is such a feature… It is, of course, AI-hard to compute. However, even a rough approximation of it may be quite useful.*

Our "snippets of sense" are exactly this sort of feature.

Pros and cons—Although maximum entropy models seem particularly well suited to our goals, there are certainly drawbacks to this approach. For one, they are more difficult to implement. Perhaps most importantly, MEMs require computationally intensive training and evaluation. Practical speed considerations may limit significantly the features we could use. In that case, the question of whether our models will be effective lies in how much "bang for the buck" we can get out of each feature. We may have an advantage over lexical features, which only apply to individual words, but how much so remains to be seen.

6.5 Overall system architecture

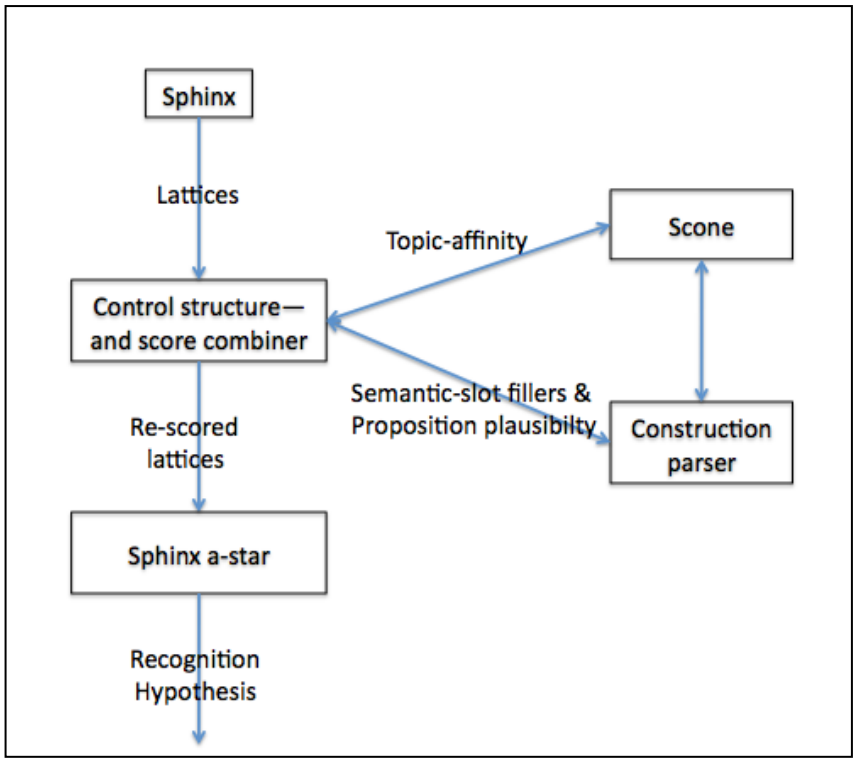
For the prototype implementation for this research, the specific software components we will use are:

- The Scone Knowledge representation system
- Our in-house construction grammar parser
- Sphinx3

The construction parser is the component that translates from words to the semantic representation. For all the techniques that identify construction matches, the construction grammar parser sits between Scone and Sphinx.

We have local expertise for both Sphinx and Scone. Sphinx is a well-known open-source speech recognition system[6]. Scone is a knowledge representation system that has been under development by Scott E. Fahlman and the Scone research group for several years[7]. Scone has been used as the knowledge representation system in a number of CMU projects and non-CMU projects.

Our initial work will be in n-best re-ranking and lattice rescoring. For n-best re-ranking, the semantic processing is a post-processing step. When re-scoring lattices, it is an intermediate step. Lattices are produced by Sphinx, re-scored by our techniques, then given back to Sphinx to determine the optimal path through the lattice. In the end, we are left with architectures that look like the following. Figure 20 shows the overall architecture for lattice re-scoring, and Figure 21 shows the architecture for *n*-best re-ranking.



[6] http://cmusphinx.sourceforge.net/

[7] http://www.cs.cmu.edu/~sef/scone/

Figure 20 - Lattice re-scoring

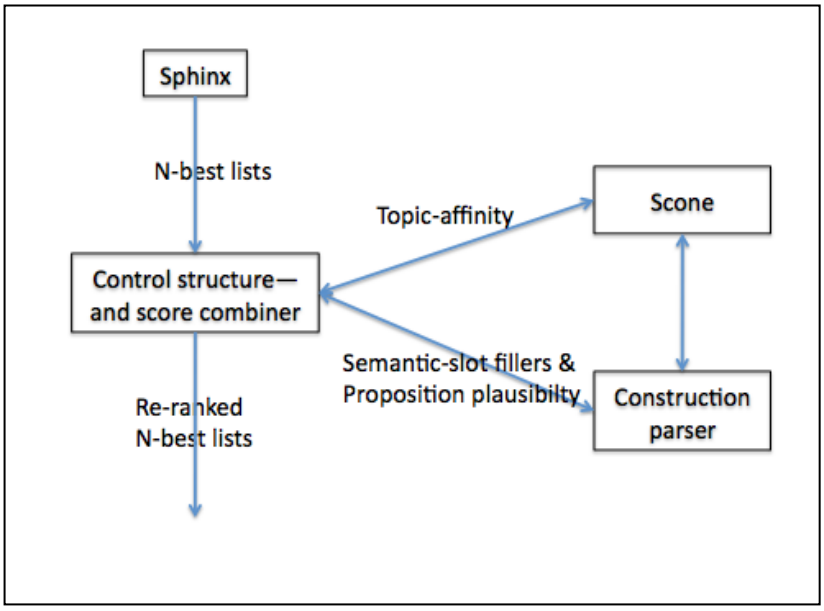


Figure 21 - N-best re-scoring and re-ranking

7 Methodology

In the end, this research is about using human-like common sense to improve ASR. Because it is both about a very human process (common sense) and a very quantitatively-based process (ASR), it will be important to be guided by, and evaluate with, both qualitative and quantitative measures.

For both qualitative and quantitative methods of evaluation, we will very early-on establish a test-bed of examples. This might consist of 100 examples from each dataset that we know we may be able to have some positive effect on (as well as others that we know we may have a neutral or negative effect upon). We can use this test bed as we go to get a sense of how each new technique we try affects the examples in this test bed. Since the test bed will be small, we can more easily perform qualitative evaluations without having to look at thousands of n-best lists to see what is working, not working, and why. We may be able to use some results from the Amazon Mechanical Turk experiments to help construct this test bed.

7.1 Data sets

For speech data, we plan to evaluate these techniques on the following datasets:

- Wall Street Journal HUB-4
- SWITCHBOARD-I
- And, a domain-specific dialog-based dataset from an operational dialog system, such as CMU's "Let's Go" (or perhaps the "Jupiter" corpus).

A domain such as "Let's Go" or ATIS would be a valuable dataset for evaluation because we may be able to additionally perform a semantic evaluation (i.e. are the correct database queries generated?). In general, for each of these, we will divide the dataset into training and test sets.

In addition to speech recognition datasets, there are other sources of data that we may be able to use to evaluate and guide this research. For instance, since the goal is to check if some hypothesized utterance makes sense or not, we could simply evaluate the ability to identify meaningful vs. nonsensical sentences. Meaningful sentences are easy to get from text corpora, and nonsense sentences can be generated automatically with an n-gram language model, or acquired by using a service like Amazon Mechanical Turk.

7.2 Evaluation

The primary focus of the quantitative evaluation will be evaluating the performance of re-ranking n-best lists. We would like to be able to show quantitative performance evaluation in lattice rescoring and decoding, but the engineering challenges may limit the extent of this.

In the end, we would like to be able to show improvement in word error rate (WER). But, along the way, and in very difficult datasets, we will evaluate our techniques using change in rank on n-best lists—at a minimum this evaluation will give us an indication of whether we are going in the right direction or not.

Another interesting measure of success will be: what fraction of human performance can we achieve? That is, we can run our techniques on the examples that people (on Amazon mechanical Turk) are able to do well, and see how much of that improvement we can automate. It may be the case that the proportion of human performance we are able to achieve is small, but it would also not be very surprising since we would assume that a real person's knowledge is much larger than our knowledge bases. However, we may be able to extrapolate how much closer we can get to human performance by adding more knowledge, more constructions, or more sophisticated methods for automatically judging sense.

8 Preliminary Results

In this section, I describe some of our preliminary results. These just begin to scratch the surface of this work. Some of the experiments we have planned are listed in the future work section of this proposal.

Here I present three small sets of results, in increasing levels of automation:

1) No automation—an informal study of how people use common sense listening to speech.
2) Partial automation—we automatically recognize speech, then ask people to use their own common sense to select the "best."
3) A non-speech human experiment—we ask subjects to identify whether a sentence was written by a human, or generated randomly by an n-gram LM.
4) A fully automated implementation of using distance-based "topic affinity," as described in section 5, for n-best re-ranking.

8.1 Semantics in human speech perception

We performed an informal experiment on human audio perception, driven by a specific mistake produced by our ASR system, Sphinx (Placeway, 1996). The example is taken from the Wall Street Journal dataset (Paul, 1992). In this example from the data, a man reads:

"The **pulp** will be used to produce newsprint."

And, Sphinx recognizes this to be:

"The **Pope** will be used to produce newsprint."

What Sphinx recognized seems so implausible, we figured no human could ever possibly make this mistake. So, we wondered what would happen if we turned this around. That is, what would happen if we actually played audio of someone speaking: "The **Pope** will be used to produce newsprint." Would people hear "pulp" or "Pope"?

To do this, we found another person speaking the word "Pope" in the dataset, and spliced the audio for "Pope" into the original audio where "pulp" is. This resulted in an audio file of someone speaking "The Pope will be used to produce newsprint." This splicing operation is illustrated in figure 22. The resulting track is everything that is underlined.

"The pulp will be used to produce newsprint"
+ "Pope & Talbot said its chairman and chief executive officer, Peter T. Pope, made the earnings projection in a presentation to New York securities analysts."

= "The Pope will be used to produce newsprint"

Figure 22 - "Pope" spliced into the audio for "pulp."

We posted these two tracks: the original ("the pulp will be…"), and the modified version ("the Pope will be…") on Amazon Mechanical Turk[8], and asked people to listen and type what they heard. (These were posted several weeks apart to avoid, hopefully, having the same people see both audio clips).

We had 20 people transcribe each audio clip. Perhaps not surprisingly, most, but not all, people when given the track with "Pope" wrote that they heard "pulp." Somewhat more surprising were the people who, when played the track with "pulp," heard "Pope." The complete results are shown in table 4 in a confusion matrix.

Although people err in both directions, they mostly hear "pulp" in either case. The distribution appears to reflect what is actually spoken (i.e. more people do hear "Pope" when "Pope" is what is actually played).

Heard\Played	Pulp	Pope
Pulp	18	16
Pope	2	4

Table 4 - What people hear when played "pulp" and "Pope" in the context of: "The ___ will be used to produce newsprint."

Although this experiment was informal and not completely controlled, it does appear to indicate that people in fact do use their own common sense to understand speech. That is, people appear to hear "pulp" more often than "Pope" because "pulp" makes more sense semantically.

Another interesting aspect to these results is that there does seem to be a slight tendency for people, when they believe they heard "Pope" to misrecognize the latter part of the sentence, as shown in the following two responses:

"the **pope** will be used to produce this print"
"the **pope** will be used to produce visprint"

[8] http://www.mturk.com/

I related this experiment to Jont Allen, an expert in the field of speech perception. Speech perception is the study of how people hear and understand speech. He hypothesized that under ideal acoustic listening conditions subjects would be able to hear "Pope" correctly, but the worse the quality of the audio, the more likely people would be to use their own background knowledge to come up with "pulp" (Allen, 2009). And, indeed, by performing this experiment on MTurk, we had no control over the listening conditions, and so one would expect that the listening conditions were often not ideal. Additionally, subjects were paid a very small amount of money, so one would not expect them to have much incentive to listen carefully.

There has been research along these lines in the field of speech perception (Garnes and Bond, 1976; and Bronckhorst, 1993). Human perception of speech is not the focus of this work. However, as a general principle we do seek to mimic human behavior. Perhaps a good introduction to human speech perception, and a very fun read, is Allen (2005).

That is as opposed to approaching speech recognition from a signal processing perspective, or acoustic modeling, or language modeling. Thus, it may be very useful to gather information and ideas from the linguistic and psychological studies of human speech perception.

8.2 Human performance on n-best re-ranking

In this, we investigate how well people are able to perform the very tasks that we seek to automate. That is, rather than attempt to re-rank n-best lists automatically, we have people "re-rank" them.

Re-ranking n-best lists is far from actually hearing and understanding speech. The subjects in this are given neither the original audio nor the context in which the speech occurred (sentences before or after). Without this information, we should not expect people to perform this task perfectly, and they do not. However, people are often able to do this well. Human performance on this task effectively gives us an upper-bound on the performance we could automate.

We ask subjects to "re-rank" 5-best lists. The audio for these 5-best lists was taken from the Wall Street Journal (HUB-4) corpus (Paul, 1992). The audio was recognized by Sphinx, to produce 5-best lists. Subjects performed this task on Amazon Mechanical Turk. We had 10 subjects re-rank each *5*-best list.

Rather than actually have subjects "re-rank" the lists, we simply asked them to choose which one they thought was most likely to have been spoken. We randomized the order of the list, so there was no information available about which of the five might have been the best acoustic match. When the "correct" answer was not in the top-*5*, we added it.

8.2.1 Re-ranking 5-best lists

For these experiments, human subjects were told that five sentences are the top-5 outputs of a speech recognizer. They are asked to choose the one they think is "most likely to have been spoken by a person." The specific prompt we give them is:

> *"The following five sentences are the top-5 candidates generated by a speech recognition system. The sentences do not contain punctuation. Please read all five sentences and choose the sentence that appears most likely to have been spoken by a person. We do not provide the original audio or the context in which the sentence was spoken.*
>
> *Often the difference between candidates is a single word. Please read all five carefully and determine the *best*. Carefully choose the one that seems: 1)*

meaningful, 2) plausible, and 3) grammatically correct. In rare cases, more than one will meet all three of these criteria. In such cases, please enter the number of all such sentences in the text box at the bottom.

Note: Please do not attempt this HIT if you will not carefully read and consider each sentence. Responses that appear to be "random" will be rejected.

Please choose the sentence that you believe most likely to have been spoken by a person:"

Here is one 5-best list presented to subjects. For this, all 10 subjects correctly chose #5.

1.THERE IS HOWEVER **ONE SUPPORTING** CONGRESS FOR INCREASED FUNDING
2.THERE IS HOWEVER **ONE SUPPORT IN** CONGRESS FOR INCREASED FUNDING
3.THERE IS HOWEVER **WHITE SUPPORT IN** CONGRESS FOR INCREASED FUNDING
4.THERE IS HOWEVER **WON SUPPORT IN** CONGRESS FOR INCREASED FUNDING
5.THERE IS HOWEVER **WIDE SUPPORT IN** CONGRESS FOR INCREASED FUNDING (100%)

Figure 23 - Five best list for WSJ example 422c0413. Correct answer is #5.

In some examples, the lack of context destroys the human subject's ability to correctly identify what was spoken as in the following 5-best list. Half of the subjects correctly chose #1, the other half incorrectly chose #4.

1.I DON'T KNOW ANYTHING BAD ABOUT THE **GUY HE SAYS** (50%)
2.I DON'T KNOW ANYTHING BAD ABOUT THE **GUY YOU SAYS**
3.I DON'T KNOW ANYTHING BAD ABOUT THE **GUIDE SAYS**
4.I DON'T KNOW ANYTHING BAD ABOUT THE **GUIDE HE SAYS** (50%)
5.I DON'T KNOW ANYTHING BAD ABOUT THE **GUYS SAYS**

Figure 24 - Another WSJ 5-best list. The correct #1 is only identified by half of the subjects.

We had 10 people look at each of 50 such 5-best lists, which gave us about 500 data points (actually 487, due to some non-responses). Of the 487 responses, people are able to choose the "correct" one, i.e. the reference transcription, 87% of the time. Considering the difficulty of examples like in figure 24, this seems rather good.

The interesting question though, is how well this compares with the ASR system's performance. We have not yet performed a "fair" comparison of this yet, but that will be one of the first steps in this research. The reason the experiments we have done are not fair is that we artificially add the correct answer to the 5-best lists. Thus, when people choose the correct transcription, they improve the WER dramatically.

As shown in table 5, in this "unfair" experiment the average WER goes from 17.9% to 1.9%. (The average WER rate for humans is computed across all instances; thus, it is an average WER over 487 examples, rather than 50).

Human average WER: 1.953% (9.512405 / 487)
System average WER: 17.879% (8.939663 / 50)

Table 5 - Human vs. ASR system performance on WSJ 5-best lists.

Similar work by Brill et al. (1998) found that in a "fair" experiment, human n-best re-rankers are able to reduce overall WER. Using the same dataset as we use here (WSJ), Brill et al. found that people were able to bring the WER down from 13.2% to 10.1%, by choosing the best of the 10-best. For their dataset, the Oracle WER is 7.9%, so the human in the loop is able to bring the error rate down by an absolute 3.1%, which is 58.5% of the best possible 5.3% absolute WER reduction. Allowing people to manually edit their choice (that is, add or delete words of their choice), brought the WER down to 9.2% (75.5% of the way to the Oracle error rate)

Their results for the Switchboard and Broadcast news datasets were similar to those of the WSJ dataset. But the absolute and the relative improvements were not as pronounced. The complete set of relevant results from that paper are shown in figures 25 and 26

	Switchboard	Broadcast News	WSJ
ASR	43.9%	27.2%	13.2%
Oracle	32.7%	22.6%	7.9%
Human selection	42.0%	25.9%	10.1%
Human edit	41.0%	25.2%	9.2%

Figure 25 - Summary of Brill et al. (1998) results-WER: Recognizer, Oracle, and human

	Switchboard	Broadcast News	WSJ
Human selection	17.0%	28.3%	58.5%
Human edit	25.9%	43.5%	75.5%

Figure 26 - Human gain relative to recognizer and oracle.

It is worth noting that the Brill study was on 75 10-best lists, for each dataset. In addition, only four human subjects looked at each 10 best.

In this paper, they additionally asked participants to identify as many of 15 supplied decision criteria that they used to arrive at their decision. These criteria range from "argument structure" to "determiner choice" to "world knowledge." The results report how often each criterion was used, as well as the possible change in WER for each criterion. The conclusion from this data include, "world knowledge was used frequently… there were many linguistic proficiencies that the person used as well."

One other interesting result they report is the strategy subjects claimed to use. "…All participants essentially used the same strategy. When all hypotheses appeared to be equally bad, the highest-ranking hypothesis was chosen."

8.2.2 Re-ranking 5-best lists, without stop words

Here we repeat the experiment described in the previous section, but this time we omit stop words from each hypothesis on each 5-best list. One example is shown in figure 27. Although the word order of the content words is retained, this experiment (hopefully) in effect measures how well people can use topic affinity to choose the correct hypothesis (since they are looking at a "bag of concepts"). The prompt given to subjects is similar to the previous one, but describes that the words are the content words only.

Not surprisingly, choosing the correct one is harder for people to do when the stopwords are removed. In figure 27, the correct answer at #2 is derived from is "We need to make economically superior fish," and none of the 10 subjects chose it.

1.NEED MAKE ECONOMICALLY SUPERIOR **FINNISH (10%)**
2.NEED MAKE ECONOMICALLY SUPERIOR **FISH**
3.NEED MAKE ECONOMICALLY SUPERIOR **FINISHED (10%)**
4.NEED MAKE ECONOMICALLY SUPERIOR **FINISH (70%)**
5.NEED MAKE ECONOMICALLY SUPERIOR **FETCH (10%)**

Figure 27 - A 5-best list with stop words removed. #2 is correct.

Of the approximately 500 specific choices made in this experiment, the "correct" answer is identified 41% of the time. This is better than completely random (20% would be completely random), but ultimately the word error rate goes up! These lists were also randomized, so even when completely baffled subjects could not default to the acoustic best.

Human average WER: 14.734% (73.66993 / 500)
System average WER: 12.975% (6.4876485 / 50)

Table 6 - Human vs. ASR system performance on WSJ 5-best lists, with stopwords removed.

This experiment is worth repeating. I do believe that this indicates that "topic affinity" may be one of the weakest methods I am proposing, but I do not believe this result indicates that it is hopeless. Rather, I think this is another hint that topic affinity must be used very carefully if it is to be used effectively.

8.3 Identifying human- and *n*-gram-generated sentences

The ultimate goal of this work is to "weed out" the ASR hypotheses that do not make sense. However, ASR hypotheses can be incorrect in many ways, and labeled speech data is not as abundant as text data. Thus, we simulate the process of identifying nonsense by looking at real human-written sentences, and sentences randomly generated by an n-gram LM. Identifying that an n-gram generated sentence is "nonsense" is similar to identifying an incorrect ASR hypothesis as nonsense.

Before trying to automate this, we had human subjects perform the task. For this experiment, we gave human subjects sentences that were either taken from the WSJ reference transcriptions (in other words, "real" human-generated English sentences) or were generated randomly using an n-gram language model. Examples of these are shown in table 7.

Human-written: "Most hadn't expected as big a drop in the soybean estimate"
N-gram-generated: "He agreed to adopt economic terms of how to the defendant's leader fifteen and one eighth Friday."

Table 7 - Examples of human-written sentences and n-gram-generated sentence used in this experiment.

Our subjects were able to correctly decide if a sentence is human- or n-gram- generated 78% of the time (excluding instances when people reported they are unsure). A further break-down of subjects' choices is shown in table 8 as a confusion matrix.

Judged\Actual	Human	N-Gram	
Human	**887 (.87)**	317 (.31)	1204
N-Gram	108 (.11)	592 (.58)	700
Unsure	20 (.02)	110 (.11)	130
	1015	1019	

Table 8 - Confusion matrix of human subjects' responses when asked if a sentence was written by a human or randomly generated by a computer.

We repeated this same experiment, but removed stop-words to see how performance is affected when they are removed. Here are two examples:

Human-written: "Olympia York declined comment"
N-gram-generated: "Curtiss scaling plan claims generate Quaker"

Table 9 - Examples human- and n-gram- generated sentences after the stop-words have been removed.

Without stop-words, people are able to identify whether the source of the words was a human or an n-gram 57% of the time (excluding "unsure" responses). Fifty-seven percent is greater than random (which would be 50% in this case), but not very much greater. Thus, it appears that people are able to have some success at this task, but that their accuracy is very limited.

Below is the break-down of the results. Interestingly, it appears that much of the non-randomness comes from identifying n-gram generated sentences as n-gram generated sentences. This makes sense because one would expect that the content words might vary greatly between the beginning and end of an *n*-gram generated sentence, and that abrupt changes in topic would signal to a person that it is n-gram generated rather than human-generated.

Judged\Actual	Human	N-Gram	
Human	**438 (.43)**	383 (.28)	721
N-Gram	**487 (.48)**	**573 (.56)**	1060
Unsure	92 (.09)	160 (.16)	252
	1017	1016	

Table 10 - Confusion matrix of human subjects' responses when asked if a sentence was written by a human or randomly generated by a computer, with stop-words removed.

Finally, we briefly examine how accuracy at this task varies as a function of sentence length. We found (not surprisingly) that human accuracy increases as sentence length increases. This makes sense, because in a very long n-gram generated sentence, the sentence may shift topics several times by the end, thereby exposing its nonsensicality. Additionally, the longer the sentence, the more opportunities arise for the *n*-gram LM to make grammatical mistakes.

We analyzed the data of responses both when subjects were given stop-words and when the stop-words were removed. Figure 28 shows accuracy for each of these as a function of length. When given stop-words, accuracy appears to level off at length around 17 words. By length 22 words, sentences without stop-words approach the performance on sentences with stop-words, but have not quite reached the same level.

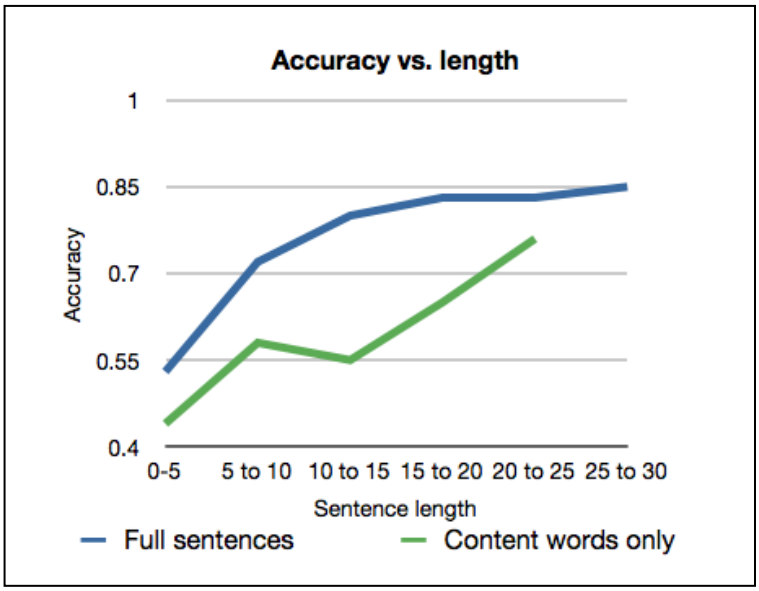


Figure 28 – Accuracy identifying human- vs. n-gram generated sentences by length

8.4 Link-counting topic affinity with WordNet

From the human experiments described in the previous two sections, it should be clear that topic-affinity (or perhaps any technique that only uses a "bag of words/concepts") is difficult even for people to perform. So, it perhaps should not be surprising that this is also a difficult task to automate.

We attempted to do this using a simple link-counting based measure of topic affinity. To obtain broad coverage, we import the WordNet (Miller, 1990) word hierarchy into Scone. We incorporated these distance-based scores into the ASR process using a linear score interpolation between the topic-affinity and the ASR score, then performed *n*-best re-ranking.

We use the following formula to compute a score for each candidate. Here H is a hypothesized speech transcription, $P_{Sphinx}(H)$ is the probability assigned by Sphinx to H, and $D_{KB}(H)$ is the average pairwise distance in the KB for H, and lambda is the scaling factor

$$Score(H) = \log(P_{Sphinx}(H)) + \lambda \cdot D_{KB}(H)$$

Using a WordNet-based KB, with this very simplistic measure of topic-affinity, "out-of-the-box" does not yield an improvement in ASR performance. That is, the "optimal" parameter lambda was determined to be zero. Others have found this to be the case as well (Pucher, 2006).

One example of why this happens is the following. In one sentence, about "states sponsoring ***seat/seed*** capital funds," there is a shorter distance from the incorrect word *seat* to *capital*, than from the correct word *seed* to *fund*. The KB represents that the *seat* of a government is usually at the *capital*. But, financial *seed funds* are more distant from *financial capital* in the KB.

To understand how well a KB that does not have these problems performs in ASR, we create "better" KB in which distances *are* correlated with correctness. We do this by modifying the KB such that correct words have artificially shorter distances among them. In particular, we artificially "shorten" the links on paths among the correct words.

To do this we find all the paths between correct concepts, and multiply the length of each edge on those paths by a small number, such as 0.5. The degree to which the quality of the KB is improved is determined by how small a multiplier we use. In the extreme, if we multiply the lengths by zero, then all correct hypotheses will have a score of zero, a "perfect" score in this technique.

Setting the multiplier coefficient (alpha) to zero yields a "perfect" KB in which the distance scores for correct hypotheses are guaranteed to be zero. Setting the coefficient to one means we are simply using distances in the original unmodified KB. Rather than optimize the combining parameter (lambda) for each version of the KB, we plot the results for a range of values. Higher values of lambda indicate a larger contribution of the KB distance score to the final re-ranking score.

Figure 29 shows the possible gains and losses in performance over a range of values for both lambda and alpha. The *x*-axis measures the relative change in KB-CER (error rate of words in the KB) from the baseline ASR (Sphinx). (Setting lambda to zero yields mean the KB is unchanged and is our baseline). Not surprisingly, smaller values of *alpha* and larger values of *lambda* achieve a substantial decrease in KB-CER. At the extreme, KB-CER deceases by 18.5%. This is 41% of the best possible decrease of 51% (shown in table 1). When alpha is large, we observe an increase in error rate.

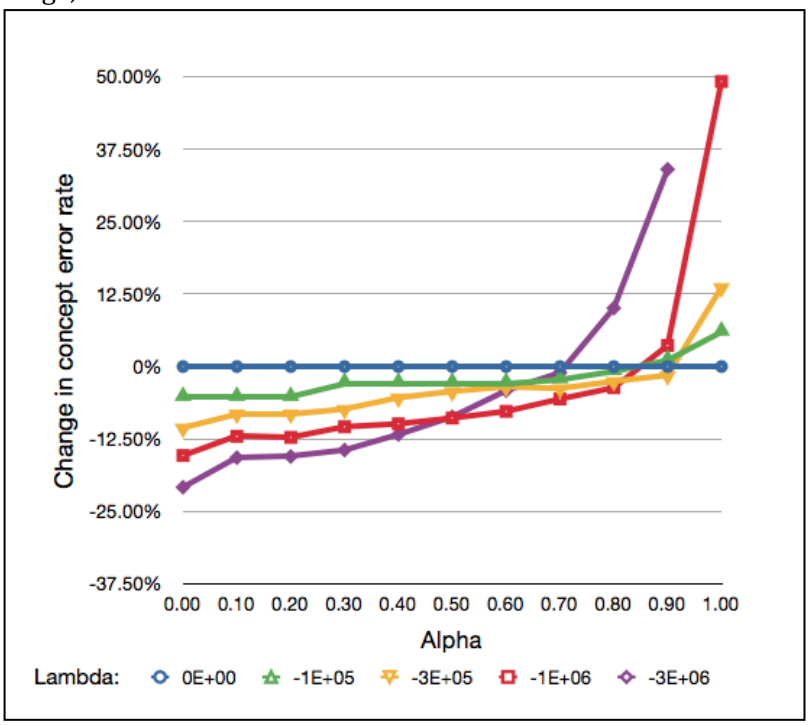


Figure 29 - Change in KB-CER, varying KB "quality" with alpha.

9 Recent related work

Section two described semantics in ASR up to around 1990. Since the early 1990's, work related to this comes from a number of fields, primarily from dialog system research, and from language modeling for speech recognition. Dialog systems and ASR ultimately have very similar goals, but approach the problem in different ways. In statistical language modeling, recent work has been aimed at incorporating more information about the structure and meaning of the language into the statistical LM. In dialog systems, recent work related to this proposal has been on using information about the domain and task semantics to improve overall system performance.

9.1 Related work in ASR and language modeling

Most of the work related to this in language modeling extends the modeling of word n-grams to also include linguistic structure and semantic affinity. For instance, class-based language models statistically (or explicitly) generalize words into syntactic or semantic classes (Brown, 1992). Structured language models incorporate syntactic structure into the language model (Jelinek and Chelba, 1999; Chelba and Jelinek, 1999; Khudanpur and Wu, 2000; Wang 2004). The goal of structured language models is that they are able to incorporate some long-distance dependency, based on grammatical relations, into word probability computations.

Rosenfeld (2001) makes a compelling case in favor of incorporating linguistic and semantic evidence into a whole-sentence exponential model. Some variations on this use shallow semantic information, such as semantic role labels (Balakrishna, 2006), or task-specific classes and semantic parses (Erdogan, 2005). All of the above methods have been shown to contribute positively, to some degree, to ASR performance. Additionally, word co-occurrence-based semantic affinity has been used in language modeling (Coccaro and Jurafsky, 1998). There has been little, if any, work on using incorporating an explicit semantic representation into the language model.

9.2 Related work in dialog systems and domain-specific speech applications

Many dialog systems and other domain- or task-specific speech applications have a similar goal to that described in this proposal. In those systems, the system has some background knowledge that it can use to constrain or bias the recognizer in favor of language that "makes sense" in the context of the specific application. In other words, the system has a model of its own capabilities and its own domain at any particular time, and it can use that model to prefer to recognize speech that describes what the system can do at any given time.

For instance, when a semantic grammar is used the grammar parser can search the word lattice for speech recognition hypotheses that are both acoustically probable and conform to the semantic grammar as much as possible (Gavalda, 2004).

Additionally, sometimes information in a back-end database can be used to bias the recognizer in favor of data in the database. For instance, an automated telephone directory assistant (automated '411') can use its database of names, to only recognize those names (Kellner, 1997; Souvignier, 2000). Switching among, and combining, different language models (either statistical or syntactic) in different dialog contexts is also possible (Lemon, 2004).

Usually, the language understanding (SLU) components of these systems are critical. But, it seems that very often, these SLU components take an all or nothing approach to influencing ASR. That is, they either fully constrain the recognizer with a grammar, or they

do not constrain it, and perform SLU in post-processing. Most of the recent work in SLU is summarized in Wang et al. (2005) and De Mori (2007).

9.3 Related work in natural language understanding (NLU)

In addition to all the work that has been specifically aimed at ASR and dialog systems, there is a long history of research on natural language understanding in general. This perhaps begins around 1960, with Bert Green et al.'s system called BASEBALL. This system allowed a user to query a database of baseball statistics in simple natural language (Green et. al, 1986).

There has been too much work in this area to survey in this proposal. Perhaps all of the important early work is included in Barbara Grosz et al.'s collection of "Readings in Natural Language Processing" (1986). This volume includes papers describing one of the earliest (perhaps the earliest) dialog systems, "GUS" (Bobrow et al., 1977). As well as a number of important early NLU systems such as LIFER (Hendrix et al., 1977), and LUNAR (Nash-Webber). There is too much work in this area to include a complete survey in this proposal.

10 Progress and future/proposed work

Before describing my detailed proposed plan and timeline, I will briefly sketch the components and work that are underway or have been completed.

10.1 Work in progress and completed

Up to now, we have made a lot of progress on an initial implementation, and integration of Scone and our construction parser with Sphinx. We have also experimented with distance-based topic affinity, and a few small-scale studies. Specifically, we have:

- Developed and implemented an interface between Scone, the construction parser, and Sphinx 3. (interface to both n-best lists and word lattices).
- Adapted the Scone group's construction parser to be suitable for this work.
- Performed an in-depth exploration of link-counting based topic affinity to re-rank n-best lists.
- Began some smaller informal studies including: slot restrictions, proposition plausibility, and lattice pruning/rescoring, and human performance on Amazon Mechanical Turk

Additionally, we now have a draft formalism, and implementation of topic affinity, semantic slot restrictions, and proposition plausibility, as described in this proposal. Some of the other components and work that have been completed or are significantly in progress are listed below.

- Tools
    - Construction parser—The construction grammar parser we are using was originally developed by CMU undergraduate Jiquan Ngiam. It is based on the Earley parsing algorithm, is written in Lisp, and ties very cleanly into Scone. The parser has been under development by Gabe Levi and me to make it more suitable for this work and for use in speech recognition (e.g. by allowing the parser to skip disfluencies)
- Draft formalism

    - I have attempted to describe most of the techniques and ideas in this proposal in detail. We have not defined a rigorous formalism in this proposal, since some of the ideas are in flux and will be modified. However, we at least rudimentary implementations of many of the ideas and techniques described in this proposal. Our formalism is well-defined enough, for the current work, but will be extended and formalized.
- Sense machines
    - Link-counting based topic affinity
    - Semantic slot constraints check—including similar methods using noun-phrase constructions, and non-role-filler knowledge constraints.
    - Instance-count-based proposition plausibility
- Knowledge bases
    - A few hand-crafted knowledge bases for specific examples in the WSJ data
    - Imported WordNet into Scone
    - Numerous other Scone knowledge bases are available as well (e.g. a movie knowledge base, cooking, and facts and statistics about countries)
- Integration with ASR
    - Locations
        - N-best lists: pruning, re-scoring, and re-ranking
        - Word lattices: pruning, and re-scoring
    - Combining as
        - Hard evidence
        - Linear interpolation
- Initial results (as described in section X), including:
    - Human performance studied on Amazon Mechanical Turk
    - Link-count based topic affinity
    - Pilot study on: Topic affinity, slot restrictions, and proposition plausibility

10.2 Future and proposed work

Throughout this proposal I have attempted to:

- define the overall goal of using "sense" as represented in a knowledge base in automatic speech recognition,
- propose a number of methods that may accomplish this, and
- describe the initial results that we have found.

The focus going forward will be to really explore this space. Among the biggest questions to answer are:

- Which techniques work? Which do not? Why?
- What kinds of semantic errors are produced by a speech recognizer? And, what kinds of semantic techniques would fix those errors?
- What resources are needed to accomplish all of these? What knowledge? What grammatical constructions?
- What methods for combining semantic evidence with statistical and acoustic evidence work best? How do numerous sources of evidence interact?
- How do the answers to these questions change for different kinds of speech? Conversational vs. formal? General vs. domain- or task-specific?

The answers to some of these questions are only possible through implementation, such as the first: which sense machines work? Other answers will be comprised of results, e.g.

which work best? The answers to most of these questions will be answered with both implementation and experimentation.

As much as possible, I would like to answer these questions and explore this space in a data-driven way. For instance, I would like to be able to say technique (or construction) #X improves recognition on Y% of examples in dataset Z, and improves those by Q percent.

Similarly, we may be able to extend this analysis of the performance to more specific parts of the overall system. For example, "the part of the knowledge base on foods and drinks hurts performance on dataset X, but the part of the KB on buses and transportation helps." Identifying the components, knowledge, evidence, and techniques that contribute at this level of detail will not always be possible, but whenever possible we will do this to better understand precisely what works and what does not.

It will not always be possible to proceed this way, guided by data. In those cases, we will proceed using qualitative, anecdotal, manual evaluation to investigate the performance and behavior of each technique by hand and through trial and error. In the end, we would like to have both a qualitative and a quantitative analysis of each technique.

Much of the work going forward will be centered on the following larger tasks:

- Developing more techniques (or "sense machines") and methods for combining their evidence with each other and with acoustic and LM scores.
- Extensive experimental evaluation of each sense machine, as well as each method of combining scores, and how they all interact.
- Manual analysis of data (mistakes) and of human performance, to determine which additional sense machines may improve performance.
- Implementing the additional framework necessary to accomplish these tasks, to use the desired data, and to set up an appropriate experimental test-bed.

Additionally, the specific parameters and techniques that we will adjust, investigate, and evaluate include:

- Which constructions can be used effectively to generate useful evidence for or against recognition hypotheses?
- What knowledge in the knowledge base will be most useful? What parts of the KB? Why?
- What are the characteristics of each method of combining knowledge-based evidence with ASR evidence, on different kinds of data (e.g. domain-specific vs. conversational)?
- What can human-performance tell us about what is possible to automate and how to automate it?

As in all research, it is not always possible to predict exactly what one will do, what will work, and in which direction the results will bring the work. Below is my best guess of the work and experiments that need to be done, as fine-grained as I can predict now, in the *ideal* order that these would be performed in, if they could all be performed sequentially without distraction, and how much time each might take alone. Following this list, I will list a coarser and more likely sequence of work in a calendar-format where the tasks overlap with each other and other work.

<u>List of things to do:</u>

- Obtain, process, and decode two additional datasets:
    - Switchboard (< 1 week)

    - "Let's Go"? (At least one week?) (possible alternative: Jupiter corpus)
- Perform a "fair" evaluation of human performance on each dataset, on MTurk (~1 week per dataset)
    - Ideally: post *all* 5-best lists (or 10-best) on MTurk, measure accuracy, change in WER, etc.
    - Most likely: post a subset of the data; sample from example where "correct" answer is in the top n; extrapolate an upper bound.
- Perform some additional oracle analysis of the currently implemented sense machines (1 week – 1 month; I think this is important, but it can also be spread out)
    - How much of the human upper-bound performance can each obtain in the current implementation? How well would they perform given a "perfect" or an "oracle" KB? (i.e. "cheating with constructions")
- Perform (possibly informal and manual) analysis of the upper-bound of alternate methods (e.g. Google n-grams, and syntactic parses) (a few days)
    - E.g., look at sample of mistakes made by Sphinx (esp. those that people can fix manually) and see which ones might have been nailed by Google n-grams, and which ones would not have been. Try to answer question: what would it take to correct the mistakes that Google n-grams cannot correct?
- Re-post some of the mistakes on MTurk, and ask people to identify (< 1 week):
    - Where is the mistake (which words? Which phrases?)
    - What kind of mistake is it? (Semantic? Syntactic? Pragmatic? Other?)
- Collect additional "non-speech" datasets (i.e. "sense" vs. "nonsense", rather than "was spoken" vs. "was not spoken") (a few weeks)
    - For each dataset, use the statistical LM to generate random "sentences"—these become "negative" examples, and reference transcriptions become "positive" examples (this is one dataset per corpus).
    - Next, have people evaluate sensicality of both positive and negative examples of above dataset (or a subset of)—this becomes the second dataset per corpus.
    - These two datasets differ from standard *n*-best lists in terms of the granularity of evaluation they provide—for instance correct/incorrect or sensical/nonsensical utterances on an n-best list are likely to be much closer in meaning than the utterances in these two datasets.
- New methods for measuring topic affinity (1-2 months?)
    - Implement and experiment with identifying concept co-occurrences that aid ASR: both *individual* concept co-occurrence and *type* co-occurrence, as much as the KB will allow (using WordNet to start)—this is effectively a concept and concept super-type long-distance language model.
    - Repeat with concept triplets in addition to concept pairs
    - Many issues here: information-gain style selection of "good" pairs, using these exactly as Roni's "trigger pairs" in a maximum entropy model.
- Repeat previous bullet as non-long distance LM—using actual concept (or type) n-grams (optional) (<1 week; easy if the previous bullet is completed)
    - Not a critical experiment but closely related, and easily implementable, with respect to several experiments—results will be a very useful baseline and/or sanity check against other methods.
- If possible: Repeat similar experiments as the above two bullets, but with Latent Semantic Analysis rather than a real KB (1-2+ weeks; this would be very interesting to try but is not the highest priority).
- Evaluate all sense machines (esp. non-topic affinity machines) both independently and together on the three speech datasets, and the three "non-speech" datasets. (easy once

the framework is set up; ~1 week to get a basic implementation framework functional, then it'll just be a matter of pushing a button for each sense machine and each dataset).
    - Use this to determine which sense machines "work" and which do not.
    - In addition to the performance of each sense machine, we may be able to evaluate the contribution or performance of different parts of the KB or different constructions being used.
    - Doing this we effectively: vet the sense machines, and vet the knowledge base
- Compete in SemEval competition—filling FrameNet slots
    - Compete against or with Noah's students—interesting collaboration opportunity
    - Requires importing parts or all of FrameNet
- Add knowledge and constructions by hand throughout and as needed.
- Write speech recognition decoder in *Lisp*—this is part of Bhiksha Raj's course on "Design and Implementation of Speech Recognition Systems," that I will take in Spring 2010, not a core part of this research. However:
    - This will be a sub-optimal decoder, but will give us a chance to actually try out some of these techniques in the decoder easily (since it'll be in Lisp)
    - Whether or not these techniques are successful, we may be able to determine experimentally what might happen if we could successfully integrate into the decoder (perhaps via "cheating" style experiments)—i.e. could sense in the decoder actually give us a good n-best list (as opposed to the correct answer being at rank #200k).
- More detailed experimentation of linear combination methods of combining scores
- Design/explore/implement whole sentence maximum entropy models.
- All the necessary work for the additional dataset (e.g. "Let's Go")
    - Data analysis
    - Writing appropriate knowledge bases (e.g. locations/neighborhoods in Pittsburgh), and constructions (e.g. "Yinz", "needs washed").
- Extend 10-example dataset/test bed to ~100 examples (MTurk may aid us in finding good test bed examples)
    - Set up end-to-end testing environment
    - These can be from the WSJ dataset to start (maybe later use other datasets)
    - Write sense modules for them.
    - Establish a baseline, oracle performance, etc.
    - These examples will give us a small-scale quantitative evaluation if all else fails
- Continue writing questions and reading from all areas (philosophy, linguistics, etc.)
    - Since semantics in ASR has not really been explored, if we can tie all the related bits in other fields to our work, then we can really claim this space.
- Possibly, look into using other KR systems: Cyc? OWL? FrameNet?
- Additional cheating experiments. Rather than try to perform perfect WSD, context, anaphora, etc.:
    - Do cheating experiments (possibly on manually tagged data) where it is assumed that these other NLP jobs have already been performed perfectly.
- As we go, as time permits, we can implement and compare all of the methods to some of the alternatives, such as:
    - Google n-grams
    - Syntax-based approached (e.g. structured LM).
    - Latent semantics

10.3 Schedule

Twelve-eighteen months:

Date	Task(s)
February 2010	• Determine a "fair" upper-bound on human performance • Determine additional computational upper-bound analysis on implemented techniques • Decode Switchboard data
Jan-Feb 2010	• New methods for topic affinity • Generate non-speech datasets (i.e. sense vs. nonsense) • Post more data on MTurk (e.g. have people label *why* it doesn't make sense)
Feb-Mar 2010	• In-depth exploration of slot restriction semantics. Import/use FrameNet and submit results on non-speech data to SemEval
April 2010	• Set up our test-bed, begin to do more comprehensive evaluations of each technique
May 2010	• Experiment with plugging in the techniques into the decoder • Write more sense machines • Try to obtain and use "Let's Go" dataset
June-July 2010	• Internship (to be determined).
August 2010	• Develop and formalize how to use maximum entropy in this framework
Sept. 2010	• "Learn" the good/bad knowledge and constructions from data
Oct. 2010	• Implement and compare to non-KB alternatives (syntax, latent semantics)
Nov. 2010	• Investigate using more complex knowledge, such as complex dialog states
Dec. 2010 +	• Writing + Everything else not included/anticipated

11 Conclusion

In this proposal, I have attempted to make a case for using semantics directly in automatic speech recognition. I motivated this with the history of speech recognition, including some of the possible reasons that spoken language understanding technologies have not been more closely integrated with automatic speech recognition.

In order to use semantics in ASR, I propose to model the meaning in the spoken language with a symbolic knowledge base. I have described some of the knowledge we will model in a symbolic KB and how it is represented. In addition, I have described how to extract snippets of meaning from the spoken language, that is to convert from words in speech to a structured representation of their meaning.

I have presented three methods in particular for using the meaning extracted from speech in the speech recognizer. These are topic affinity, semantic type restrictions, and proposition plausibility. I propose to use these three techniques, as hard constraints on the speech recognizer, as linearly combined evidence, and as features in a maximum entropy model. I have described how to do this through *n*-best list re-ranking, lattice re-scoring, and within an ASR decoder.

Finally, I described our overall approach to the methodology of this research, some of the initial results we have obtained, and the plan going forward.

I believe the time is right for this research. The work will be a challenge, but even if we can make a little progress, I believe it will open new doors and help to set the stage for more work in this area and will help create new ideas and methods for using semantics in language technologies.

References

Allen, Jont. Articulation and Intelligibility. Morgan & Claypool Publishers, 2005.

Allen, Jont. Personal communication, December 16, 2009.

Bahl, Lalit R.; Jelinek, Frederick; Mercer, Robert L., "A Maximum Likelihood Approach to Continuous Speech Recognition," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol.PAMI-5, no.2, pp.179-190, March 1983.

Baker, J., "The DRAGON system--An overview," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol.23, no.1, pp. 24-29, Feb 1975.

Bobrow, Daniel G., Ronald M. Kaplan, Martin Kay, Donald A. Norman, Henry S. Thompson Terry Winograd. "GUS, A Frame-Driven Dialog System." 155-173 8 Artificial Intelligence 2. 1977.

Bohus, D., and Rudnicky, A., "Sorry, I Didn't Catch That! - An Investigation of Non-understanding Errors and Recovery Strategies," in Proceedings of *SIGdial-2005*, Lisbon, Portugal, 2005.

Bohus, D. and Rudnicky, A. I. 2009. "The RavenClaw dialog management framework: Architecture and systems." *Comput. Speech Lang.* 23, 3 (Jul. 2009), 332-361.

Bronkhorst, Adelbert W., Arjan J. Bosman, and Guido F. Smoorenburg. "A model for context effects in speech recognition", Journal Acoustical Society if America. 93, 499, 1993.

Brown, P. F., deSouza, P. V., Mercer, R. L., Pietra, V. J., and Lai, J. C. 1992. "Class-based n-gram models of natural language." *Computational Linguistics*, 18, 4 (Dec. 1992), 467-479.

Bryant, John. "Best-Fit Constructional Analysis." Ph.D. dissertation. University of California at Berkeley, 2008.

Buhler, Dirk; Minker, Wolfgang; and Elciyanti, Artha. "Using Language Modelling to Integrate Speech Recognition with a Flat Semantic Analysis," *SigDial*, 2005.

Burton, Richard R. "Semantic grammar: a technique for efficient language understanding in limited domains." Ph.D. dissertation. University of California at Irvine, 1976.

Carbonell, J. G. and Hayes, P. J. 1983. "Recovery strategies for parsing extragrammatical language." *Computational Linguistics*. 9, 3-4 (Jul. 1983), 123-146.

C. Chelba, D Engle, F Jelinek, V Jimenez, S. "Structure and performance of a dependency language model." *Eurospeech*, 1997.

Chelba, Ciprian and Frederick Jelinek. "Recognition Performance of a Structured Language Model," In proceedings of *Eurospeech*, 1999.

Coccaro, Noah and Daniel Jurafsky. "Towards Better Integration of Semantic Predictors in Statistical Language Modeling." In *ICSLP*, 1998.

Croft, William A. *Radical Construction Grammar: Syntactic Theory in Typological Perspective*. Oxford: Oxford University Press, 2001.

CyCorp. http://www.cyc.com/.

Das, Dipanjan, Nathan Schneider Desai Chen Noah A. Smith, "Probabilistic Frame-Semantic Parsing." In submission to *HLT-NAACL*, 2010.

De Mori, R., "Spoken language understanding: a survey," In IEEE Workshop on *Automatic Speech Recognition and Understanding (ASRU)*, 2007.

Dowding, John; Jean Mark Gawron, Doug Appelt, John Bear, Lynn Cherny, Robert Moore, and Douglas Moran, "Gemini: a natural language system for spoken-language understanding." *ACL*, 1993.

J. Earley, "An efficient context-free parsing algorithm," *Communications of the Association for Computing Machinery*, 13:2:94-102, 1970.

Fillmore, Charles, Paul Kay and Catherine O'Connor. "Regularity and Idiomaticity in Grammatical Constructions: The Case of let alone." *Language* 64: 501-38, 1988.

Garnes, S., Bond, Z.S. "The relationship between acoustic information and semantic expectation". Phonologica 1976. Innsbruck. pp. 285-293, 1976.

Gavaldà, M. "Soup: a parser for real-world spontaneous speech." In *New Developments in Parsing Technology*, eds. H. Bunt, J. Carroll, and G. Satta, Eds. Kluwer Academic Publishers, Norwell, MA, pp. 339-350, 2004.

Goldberg, Adele. *Constructions: A Construction Grammar Approach to Argument Structure*. Chicago: University of Chicago Press, 1995.

Goldberg, Adele. Constructions at Work: The Nature of Generalization in Language. Oxford: Oxford University Press, 2006.

Green, B., Wolf, A., Chomsky, C., and Laughery, K. "BASEBALL: an automatic question answerer." In *Readings in Natural Language Processing*, B. J. Grosz, K. Sparck-Jones, and B. L. Webber, (eds.). Morgan Kaufmann Publishers, San Francisco, CA, 545-549, 1986.

B. J. Grosz, K. Sparck-Jones, and B. L. Webber, (eds.) *Readings in Natural Language Processing*. Morgan Kaufmann Publishers Inc. 1986.

Hayes, P. J., Hauptmann, A. G., Carbonell, J. G., and Tomita, M. "Parsing spoken language: a semantic caseframe approach." In Proceedings of the *11th Conference on Computational Linguistics*, 587-592, 1986.

Hemphill, C. T., Godfrey, J. J., and Doddington, G. R. 1990. "The ATIS spoken language systems pilot corpus." In Proceedings of the *Workshop on Speech and Natural Language*, 1990.

Huang, Xuedong; Alex Acero; Hsiao-Wuen Hon. *Spoken Language Processing: A Guide to Theory, Algorithm and System Development.* Prentice Hall, 2001.

F. Jelinek. "Continuous Speech Recognition by Statistical Methods." IEEE Proceedings 64:4: 532-556. 1976.

Jelinek, Frederick, and Ciprian Chelba, "Putting Language into Language Modeling," In Proc. of *Eurospeech*, 1999.

Katz SM, J. L. Gauvain, L. F. Lamel, G. Adda, and J. Mariani. "Estimation of probabilities from Sparse data for the language model component of a speech recognizer." *Int. J. Pattern Recognition & A.I*, volume 8, 1987.

Kawahara, Tatsuya, "New Perspectives on Spoken Language Understanding: Does Machine Need to Fully Understand Speech?" *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2009.

Kellner, A., Rueber, B., Seide, F., and Tran, B. 1997. "PADIS—an automatic telephone switchboard and directory information system." Speech Communication. 23, 1-2 (Oct. 1997), 95-111.

Khudanpur, Sanjeev, and Jun Wu. "Maximum entropy techniques for exploiting syntactic, semantic and collocational dependencies in language modeling," in *Computer Speech and Language*, 14(4):355-372, 2000.

D. H. Klatt. "Review of the ARPA speech understanding project." *Journal of the Acoustical Society of America*, v62, Issue 6, 1977.

Kuo, Hong-Kwang Jeff, and Yuqing Gao. "Maximum entropy direct models for speech recognition." *IEEE Transactions on Audio, Speech, and Language Processing*, May 2006.

Langacker, Ronald. *Cognitive Grammar*. Oxford University Press, 2008.

Lemon, O. and Gruenstein, A. "Multithreaded context for robust conversational interfaces: Context-sensitive speech recognition and interpretation of corrective fragments." *ACM Trans. Computer-Human Interacttion.* 11, 3 (Sep. 2004), 241-267, 2004.

Lieberman, Henry, Alexander Faaborg, Waseem Daher, and José Espinosa. "How to Wreck a Nice Beach You Sing Calm Incense." In *International Conference on Intelligent User Interfaces (IUI)*, 2005.

Lochbaum, K. E. "A collaborative planning model of intentional structure." Computational Linguistics. 24, 4 (Dec. 1998), 525-572, 1998.

G.A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. Miller, "WordNet: An on-line lexical database," International *Journal of Lexicography,* Vol. 3, pp 235-244. 1990.

Miller, Tim, Lane Schwartz, and William Schuler. "Incremental Semantic Models for Continuous Context-Sensitive Speech Recognition," 2008.

Nadas, A., "Estimation of probabilities in the language model of the IBM speech recognition system," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol.32, no.4, pp. 859-861, Aug 1984.

Paul, D. B. and Baker, J. M. 1992. "The design for the wall street journal-based CSR corpus." In Proceedings of the *Workshop on Speech and Natural Language,* 1992.

Placeway, P., Thayer, E., Stern, R., Siegler, M., Seymore, K., Rosenfeld, R., Ravishankar, M., Raj, B., Parikh, V., Jain, U., Eskenazi, M., Chen, S. "The 1996 HUB-4 Sphinx-3 system." DARPA Spoken Systems Technology Workshop, February, 1997

Price, P.; Fisher, W.M.; Bernstein, J.; Pallett, D.S., "The DARPA 1000-word resource management database for continuous speech recognition," *Acoustics, Speech, and Signal Processing* (ICASSP), pp.651-654, 1988.

Pucher, Michael, "WordNet-based semantic relatedness measures in automatic speech recognition for meetings". In Proceedings of *The 45th Annual Meeting of the Association for Computational Linguistics* (ACL), 2007.

Raj, Bhiksha. Class lecture slides. "Automatic Speech Recognition in an Hour." Carnegie Mellon University, Pittsburgh, PA. Nov 5, 2009.

Reddy, D.; Erman, L.; Neely, R., "A model and a system for machine recognition of speech," *IEEE Transactions on Audio and Electroacoustics*, vol.21, no.3, pp. 229-238, Jun 1973.

Rosenfeld, Ronald. "A Maximum Entropy Approach to Adaptive Statistical Language Modeling." *Computer, Speech and Language* 10, 187--228, 1996.

Rosenfeld, Ronald. "A Whole Sentence Maximum Entropy Language Model." In Proc. *IEEE Workshop on Automatic Speech Recognition and Understanding*, 1997.

Rosenfeld, Roni, Stanley F. Chen and Xiaojin Zhu. "Whole-Sentence Exponential Language Models: a Vehicle for Linguistic-Statistical Integration." *Computers Speech and Language*, 15(1), 2001.

Selting, Margret, "Lists as embedded structures and the prosody of list construction as an interactional resource," *Journal of Pragmatics—Special Issue: Diversity and Continuity in Conversation Analysis*. Volume 39, Issue 3, March 2007, pp. 483-526.

Souvignier, B.; Kellner, A.; Rueber, B.; Schramm, H.; Seide, F., "The thoughtful elephant: strategies for spoken dialog systems," *Speech and Audio Processing, IEEE Transactions on*, vol. 8, no.1, pp. 51-62, Jan 2000.

Steels, Luc; and Joachim De Beule. "A (very) Brief Introduction to Fluid Construction Grammar," *Third International Workshop on Scalable Natural Language Understanding* (ScaNaLU), 2006.

Thomae, M. Fabian, T. Lieb, R. Ruske, G. "A one-stage decoder for interpretation of natural speech." Proceedings of the *International Conference on Natural Language Processing and Knowledge Engineering*, 2003.

Ward, Nigel. "Second Thoughts on an Artificial Intelligence Approach to Speech Understanding." *Fourteenth Spoken Language and Discourse Workshop Notes* (SIGSLUD-14), 1996.

Wang, Richard C. and William W. Cohen. "Character-Level Analysis of Semi-Structured Documents for Set Expansion." In Proceedings of *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2009.

Wang, W., Stolke, A; Harper, M. "The use of a Linguistically Motivated Language model in conversational speech recognition." *ICASSP*, 2004.

Ward, W. "Understanding spontaneous speech. In Proceedings of the *Workshop on Speech and Natural Language* at *Human Language Technology Conference*. Association for Computational Linguistics, Morristown, NJ, 137-141, 1989.

Ward, W. and Issar, S. "Recent improvements in the CMU spoken language understanding system." In Proceedings of the *Workshop on Human Language Technology*, 1994.

Wang, Ye-Yi; Li Deng; Acero, A., "Spoken language understanding," *Signal Processing Magazine, IEEE* , vol.22, no.5, pp. 16-31, Sept. 2005.

Warren, R.M. "Restoration of missing speech sounds". Science 167: 392–393, 1970.

W3C. "OWL 2 Web Ontology Language Document Overview," October 27, 2009. http://www.w3.org/TR/owl2-overview/

Zhu, Xiaojin, Stanley Chen and Ronald Rosenfeld. "Linguistic Features for Whole Sentence Maximum Entropy Language Models." In Proc. *Eurospeech,* 1999.

Appendix A – Sample Code

For completeness sake, here is an example construction rule, and the Scone Lisp code that defines and instantiates a *row* event.

```
(define-construction
    ("nominalized transitive verb construction"
               {S construction}
             ((v1 {genitive form})
            (v2 {action with patient} instance)
            (v3 "for" string)
            (v4 {scone-noun} ))
               (v1 v2 v3 v4)
             ((*c-head* v2 ))
               :base-score 1.0)

  (r-eval (if (can-x-be-the-y-of-z? v1 {transitive-agent} v2)
            (x-is-the-y-of-z (new-indv nil v1) {transitive-agent} v2)
      (error "invalid agent role")))

  (r-eval (if (can-x-be-the-y-of-z? v4 {patient} v2)
      (x-is-the-y-of-z (new-indv nil v4) {patient} v2)
      (error "invalid patient role")))
  nil)
```

In Scone, this action, its constraints, and this particular individual are specified as follows:

```
(new-type {row} {event})
(the-x-of-y-is-a-z {agent} {row} {person})
(the-x-of-y-is-a-z {patient} {row} {boat})

(new-indv {row-34} {row})
(the-x-of-y-is-z {agent} {row} {Fred})
(the-x-of-y-is-z {patient} {row} {boat-22})
```

Appendix B – Examples of semantics errors in the data

051c030v
1. Commitments to purchase mortgages from leaders felt the three point six billion dollars from six point five billion dollars
2. The commitments to purchase mortgages from leaders felt the three point six billion dollars from six point five billion dollars
3. Commitments to purchase mortgages from leaders fell to three point six billion dollars from six point five billion dollars

053c030g
1. But regulators say they received about seventy complaints industrywide just during April and are considering finding some airlines
n. but regulators say they received about seventy complaints industrywide just during April and are considering fining some airlines

22gc0212
1. Those retailers with stronger inventory and financial controls employees found their profit margins benefited.
2. Those retailers with stronger inventory and financial controls and place found their profit margins benefited.
3. Those retailers with stronger inventory and financial controls and placed found their profit margins benefited.
4. Those retailers with stronger inventory and financial controls in place found their profit margins benefited.***

001c0q0h
REF: for nineteen eighty SEVEN THE institute PREDICTS GROSS sales to investors of stock bond AND income funds will total one hundred eighty eight point three billion dollars
HYP: for nineteen eighty SEVENTY *** institute CRITICS GROW sales to investors of stock bond AN income funds will total one hundred eighty eight point three billion dollars

001c0q0n
REF: the rise came AMID optimism about the possibility of an agreement to reduce the U. S. budget deficit
HYP: the rise came EMIT optimism about the possibility of an agreement to reduce the U S budget deficit

001o0r04
REF: the law covers a wide range of *** ANTI -HYPHEN competitive activity and isn't limited to the effects of MERGERS
HYP: the law covers a wide range of THE ANTE HI-FI competitive activity and isn't limited to the effects of MURDERS

001c0q0p

REF: investment bankers AND TRADERS **** generally decide to switch firms after receiving THEIR year END bonuses
HYP: investment bankers IN TRADE EARS generally decide to switch firms after receiving THEY'RE year AND bonuses

001o0r0s
REF: superior industries international FELL three and one quarter to fifteen and one half
HYP: superior industries international FELT three and one quarter to fifteen and one half

001o0r0y
REF: but for white women ,COMMA ** a single -HYPHEN parent family had no impact on * EVENTUAL education
HYP: but for white women CALM UP a single HEIFER parent family had no impact on A VENGEFUL education PERIOD

002c0q0u
REF: volume was three hundred ninety point seven million SHARES compared with five hundred eighty nine million friday
HYP: volume was three hundred ninety point seven million CHAIRS compared with five hundred eighty nine million friday

002o0r0o
REF: all settlement talks CEASED after the filing
HYP: all settlement talks DECEASED after the filing

00ao0r0g
REF: the KING didn't indicate what portion of revenue would come from oil and what would come from government FEES
HYP: the KEY didn't indicate what portion of revenue would come from oil and what would come from government HE'S

00cc0q03
REF: so MIGHT the next FRENCH election
HYP: so MY the next FRIDGE election

00co0r0b
REF: meanwhile ,COMMA *** **** UNILEVER ,COMMA which has been mentioned AS A potential RIVAL BIDDER for ROBINS ,COMMA rose FOUR AND three quarters TO two hundred forty
HYP: meanwhile COMMON YOU KNOW LIVER COMMON which has been mentioned IS * potential REBEL BETTER for ROBIN SCALLOPED rose FOR *** three quarters ** two hundred forty

00co0r0j
REF: TRADERS say the rule already has had some impact on the market
HYP: TRAINER say the rule already has had some impact on the market

00dc0q0i
REF: U. S. CAR sales FELL ten POINT one **** PERCENT IN EARLY november FROM a year ago

HYP: U S FAST sales FELT ten FOUR one THIS AND THAT REALLY november **** a year ago

002o0r07
REF: american farmers also STEPPED UP production as soon as the program *** **
EXPIRED ,COMMA and world crop prices SANK
HYP: american farmers also STEP THE production as soon as the program MIX BY OR COMMON and world crop prices THINK

Hyp. The servant in this case is government
Ref. The serpent in this case is government

1. I don't know anything bad about the guide he says
2. I don't know anything bad about the guy he says

1. I want them to see the possibilities for July
2. I wanted to see the possibilities for July
3. I want them to see the possibilities for joy **

1. Britain's favored Spain as a foreign destination followed by France the U. S. Greece and Italy
2. Britain's favored Spain as a foreign destination followed by France the U. S. grease and Italy
...
8. Britons favored Spain as a foreign destination followed by France the U. S. Greece and Italy

1. The fifty one year old Mr. Seow are on makes no secret of his admiration for twenty pro ovens on all
N. The fifty one year old Mr. Sciarra makes no secret of his admiration for Tony Provenzano

1. Mr. Harness said the turmoil hasn't affected the date to day operations of Forstmann Leff which currently manages four billion dollars in client assets.
2. Mr. Harnisch said the turmoil hasn't affected the day to day operations of Forstmann Leff which currently manages four billion dollars in client assets.

1. I've never lied about my age and I never will knock out would she went on
N. I've never lied about my age and I never will knock on wood she went on

1. We being dog for N. T. R. and isn't simple
N. Weaving dog fur into yarn isn't simple

1. Then he treats it would agree conditioner to add fragrance
2. Then he treats it with agree conditioner to add fragrance ***

1. In American stock exchange composite trading Friday the east the shares closed at four point eight seven five dollars down twelve point five cents.
10. In American stock exchange composite trading Friday T. E. C. shares closed at four point eight seven five dollars down twelve point five cents.