

# The “Casual Cashmere Diaper Bag”: Constraining Speech Recognition Using Examples

Paul Martin

Sun Microsystems Laboratories  
Two Elizabeth Drive  
Chelmsford, MA 01824-4195 USA  
paul.martin@east.sun.com

## Abstract

We describe a new technology for using small collections of example sentences to automatically restrict a speech recognition grammar to allow only the more plausible subset of the sentences it would otherwise admit. This technology is unusual because it bridges the gap between hand-built grammars (used with no training data) and statistical approaches (which require significant data).

## 1 Motivation

Statistical modeling for speech recognition requires a large body of training text; we are addressing the cases where only very modest quantities of “training” data exist.

Our method is relevant when the speech to be recognized contains patterns that can be straightforwardly abstracted from the semantics of the words appearing in them, but where the sentence set to be recognized is too large to cover with manually created rules. Using this technology, a small representative set of utterances (on the order of hundreds of sentences or phrases) is combined with an overly-permissive general grammar to automatically create a much tighter grammar that is specific to the particular domain. The grammar produced is a context free grammar in whatever BNF the speech recognizer of choice requires.

The technique works by using a grammar compiler that accepts grammars composed of rules written using both patterns and restrictions (which can be syntactic or semantic); the Unified Grammar compiler (Martin and Kehler, 1994) provides this foundation. Our approach requires the developer to write grammars with rules which enforce semantic restrictions that test whether the class markings in the lexicon on a phrase head correspond to similar markings on the lexical entries for the possible modifiers of that head. If the grammar writer took the trouble to mark the lexicon appropriately, then these tests, processed by the grammar compiler, would suffice

to build the restricted grammar. Instead, with the new technique, only the semantic markings necessary to understand the “meaning” of the resultant utterance need be marked by the grammar developer; processing of a set of example sentences serves to automatically provide the additional lexical markings needed by the modifier words, and a subsequent grammar compilation reflects these restrictions in the pure BNF grammar used by the speech recognizer.

## 2 Background

Currently, the best speaker-independent continuous speech recognition (SR) is orders of magnitude weaker than a human native speaker in recognizing arbitrary sequences of words. That is, humans do pretty well on clearly spoken sequences of words chosen randomly from a pool of tens of thousands of words, while unconstrained SR systems only do as well when the vocabulary is much smaller, in the range of hundreds of words. When the recognition is to be done over the telephone, the reduced signal-to-noise ratio of the speech data makes this weakness even more dramatic.

### 2.1 Language Models

In order to achieve useful recognition rates, current SR systems impose constraints beyond just a limited vocabulary, either by specifying an exact grammar of the sequences which are allowed or by providing statistical likelihoods for word sequences ( $n$ -gram statistics). The grammars are built by hand as context-free formalisms determining allowable word sequences. The statistical models use tables of the “raw” probabilities of each word (unigram) usually augmented with additional tables of the likelihood of each word given each possible preceding word (bigram) or each possible two preceding words (trigram). These statistical systems have been experimentally extended to include  $n$ -grams where  $n$  is exceeds three, but even for higher  $n$  they generally express only the probability of a word based on the adjacent preceding words.

### 2.1.1 Sentence Grammars

Hand-built grammars can provide exquisitely fine control over the word sequences recognized, but their construction is difficult and painstaking, even for those who are practiced in the art.

### 2.1.2 Statistical Model

Conversely, statistical “grammars” can be built automatically by running an analysis program over an appropriate collection of the kinds of sentences that one wishes to recognize. A prime example of this technology is the ARPA-initiated Wall Street Journal (WSJ) dictation project, where recognizers trained on the text of previously-printed articles from the WSJ are tested by having them recognize text read from a later edition of the WSJ. Unfortunately, the database of WSJ text used in these experiments contained approximately 40 million words, and researchers using this database have indicated that their speech systems work better when they were able to double the size of their training set (Schwartz et al., 1994).

While the recognition achieved on the WSJ with this technique is impressive, the information embodied in the statistical model is so specific there is not much “transfer” to recognizing text that varies in style, even when content and vocabulary are shared. [cite example of NYTimes financial stories and the ads in WSJ not working well]

## 2.2 Command and Control

In the domain of command and control of computer programs, the utterances to be recognized do not correspond directly to any existing body of text that could be used analogously to the WSJ text’s role in training the dictation recognizers. Traditional statistical modeling requires a relatively huge database of example utterances and the models do not include any abstraction of the words, so the actual co-occurrence of words is necessary to count the relative frequency of each. For many applications of speech recognition there simply is not enough training data to support using statistical models.

### 2.2.1 Automating the Lands’ End catalog

We discovered the need for some new method to restrict a speech recognizer when we attempted to implement an automated customer service agent to interact with users wanting to browse and order items from an online catalog. Lands’ End Direct Merchants provided a collection of “video assets” from one of their catalogs for this experiment. A typical “page” illustrated and described an item or a collection of related items, and might have associated with it additional information such as a video clip, color and size pages, and indications of the pages that are specializations of this page. We prototyped a speech-controlled application which allows a user to interact with the automated agent us-

ing speech through the telephone while viewing the video on a television<sup>1</sup>. Allowing a free conversational dialogue and supporting a large subset of the myriad ways an untrained caller might describe the catalog items overwhelmed our speech recognizer.

## 3 How Restrictive is a grammar?

Writing a grammar to allow a user to make queries about the contents of this computerized catalog was the concrete example that drove our new approach.

### 3.1 What do users say?

We collected examples of what users said to an expert human service representative in a “Wizard of Oz” experiment (Yankelovich, forthcoming). Besides the action words and phrases (“can you show me <itemPhrase>?” or “what <itemPhrase> do you carry?”) in a shopping query, the user commonly supplied a phrase that names or describes the item of interest.

**D:** I’d like a soft-sided attache.

<displays luggage page>

**D:** The canvas line.

**C:** How about kids?

**B:** Can I see the squall jacket?

**C:** Could I see the men’s clothes?

<displays menswear page>

**C:** Dress shirts.

**S:** Could we switch to children’s clothing.

**L:** Let’s look at some casual dresses.

**M:** I’d like to see the sweaters please.

**S:** I’m looking for things from bed and bath.

**B:** Let’s go back to sweaters.

**B:** Can I go back to the main screen.

**L:** I’ll go back to the womens.

**A:** I’m looking for a blazer and slacks and skirts to go with it.

**C:** I need a flat sheet and a fitted sheet in queen.

Example queries users said to “Wizard” system.

#### 3.1.1 A grammar to collect semantics

We implemented the prototype Lands’ End system using our SpeechActs (Martin et al., 1996) system, collecting the relevant semantics from utterances with a simple grammar specifying the allowable phrases.

One over-simplified grammar of such “item specification” phrases would allow any basic item (such as

<sup>1</sup>In a real installation, the television would be connected to a pay-per-view channel or a cable system such as in a hotel

“pants”) to be modified by any combination of meta-style, pattern style, color, size, gender, wearer’s age, fabric type, fabric style, and maker’s name. A particular sweater could be referred to as “the petite women’s medium dusty sage jewel-neck cashmere fine-knit ‘drifter’ sweater”. While no one would ever spontaneously utter this monster, we cannot predict which portion of these options will be used in any given utterance. Such an accepting grammar works just fine for extracting the meaning from a written form of the item description, and in fact, is used in the Lands’ End system to identify what items are displayed on each “page” of the video-accessible catalog.

```
{!eNounPhrase/nosize := [determiners]
  [style/styl1][preModifiers][style/sty2]
  [sem=style-name][sem=fabric-style]
  [sem=material][style/sty3]
  !eNoun [postModPhrase];
head !eNoun;
fabric := material.root;
fabric ^= postModPhrase.material;
index := postModPhrase.index;
fabric-style := fabric-style.root;
fabric-style ^= postModPhrase.fabric-style;
genderCat := preModifiers.genderCat;
genderCat ^= postModPhrase.genderCat;
.....}
```

Example UG rule allowing many possible modifiers

### 3.2 Semantic grammar is too loose for SR

Unfortunately, the perplexity of the grammar produced by the cross product of all these choices is so large that the word accuracy of the speech recognition becomes uselessly low. Phrases that no user would ever utter are “heard” by the SR engine; the “casual cashmere diaper bag” mentioned in the title of this paper refers to one of the more outrageous combinations that pass the muster of this weakly-constraining grammar.

#### 3.2.1 Marking semantic “agreement”

If the lexical entry for every modifier were marked with a feature containing the set of things it could realistically modify (or, better yet, the set of classes of things), then the grammar could be written to allow only the “reasonable” combinations and to rule out the ridiculous ones that should be omitted to reduce the perplexity. With a grammar compiler that accepts such restrictions based on features in the lexicon, such a markup appears to be a possible solution. The grammar writer could create and record classes of basic items, noting that “chinos” and “jeans” were “tough clothing” and then only allowing them to be associated with fabrics appropriate for “tough” clothes. This strategy would block combinations such as “lace chinos” but allow “silk blouses” and “denim jeans.”

The biggest disadvantage of requiring a grammar writer to figure out and record the features that determine allowable modifiers is the large amount of detailed work required to make such annotations. If these markings could be derived automatically from some pre-existing or easily-created data, then the task would be much reduced, and the cost of adding new items to the catalog would be much smaller. (In the particular case of modeling a catalog, the effort required to accommodate each subsequent revision of the items carried is a primary concern<sup>2</sup>)

```
genderCat = 'womens
fabric = 'cotton
meta-style = 'casual
catalogtype = 'pants
style = 'chino
```

Example indexing of an item page described as “women’s chino slacks” and as “casual cotton pants.”

#### 3.2.2 Using example sentences

In the Lands’ End example, we already have item descriptions which are part of their standard catalog database. We use these descriptive phrases both to navigate to the item or item collection (such as “men’s jackets”) the user has requested and to verify that the semantic grammar and lexicon will accept the phrases used by the catalog designers. Any new version of the catalog will necessarily already have these phrases created for it; using them additionally for grammar restriction almost automates the update chore for new editions of the catalog.

If the grammar were written incorporating tests to require the lexical markings indicating allowable modifiers, then it would reject any phrase that lacked the needed marks. If such a grammar were used with a “bare” lexicon (one lacking these modifier markings), it would not support parsing the page descriptors, and would compile into a speech grammar allowing only bare item names, devoid of any modifiers. We addressed this problem by adding the ability to switch the restrictions on or off, and then turning them off when parsing the (written) page descriptors. (See the example of switched tests in a grammar rule.)

#### 3.2.3 Automating the markup

Indexing and then processing the results of all the page descriptor parses provides the information content needed to automatically mark up the lexicon with the compatibility results derived from the page descriptors. Once the lexicon has been enhanced with this information, the restrictions can be turned on while the unified grammar is used by the speech recognizer. In our system, we compile the unified grammar to produce BNF reflecting the

<sup>2</sup>We don’t mind working hard once in a while, but we do not want a new career updating this catalog.

restrictions, but logically these restrictions could be applied “on the fly” by a speech recognizer or used in post-processing to choose among the n-best alternatives from a less restricted SR. Regardless of how it is implemented, the resultant grammar will not allow “lace jeans” simply because no page description phrase mentions any such thing.

```
{leNounPhrase/nosize := [determiners]
  [style/sty1][preModifiers][style/sty2]
  [sem=style-name][sem=fabric-style]
  [sem=material][style/sty3]
  leNoun [postModPhrase];
head leNoun;

leNoun.cat-type *= material.cat-type-set;

fabric := material.root;
fabric ^= postModPhrase.material;
index := postModPhrase.index;
fabric-style := fabric-style.root;
fabric-style ^= postModPhrase.fabric-style;
genderCat := preModifiers.genderCat;
genderCat ^= postModPhrase.genderCat;
.....}
```

The \*= operator applied to leNoun is the switched test operator in this example grammar rule.

## 4 Logic versus reality

### 4.1 A mysterious deafness

One final problem must be addressed to make this scheme actually useful; there are sure to be some “reasonable” combinations of modifiers and basic items that the catalog makers just do not include in their catalog. If there were “canvas jackets” and “denim jeans” in the catalog but no “denim jackets,” then unless jeans and jackets shared a common “kind of thing” property on which to base the grammar restrictions, the restricted grammar could not hear the phrase “denim jacket”. Presented with those sounds, it would probably produce something like the “d’women jacket” pronunciation of “the women[’s] jacket”, but it could not “hear” what the user actually said. This would be baffling to a naive user of the system, especially since rephrasing his request to include “a jacket made of denim” would also fail.

### 4.2 Filling in the gaps

To fix this shortcoming, the examples that generate the automatic marking of the lexicon must be augmented to include the logical extensions of the actual database of “real” items. When proposing this approach, Nicole Yankelovich loosely described it as “listing all the things that *aren’t* in the catalog”. Of course, taking this literally would be an unbounded task and would defeat the whole goal of restricting the grammar; such a list would include the infamous

cashmere diaper bag! What we really needed was a listing of the things that one might logically expect to find but which do not exist in this particular catalog. In our Lands’ End example, we created pages of “missing” items and associated these explicitly missing pages as phantom pages under their logical parent pages in the catalog. These phantom pages serve to attach the information we give the customer when we report the omission. With this addition to the scheme, the user can be “heard” asking for a denim jacket, and will receive a helpful response.

We attach explicit helpful messages to some phantom pages (“Sorry, but the jackets do not come in denim, only Polartec, Thinsulate, and wool”) and otherwise generate a message indicating the query was heard, but no such item is in *this* catalog.

## 5 System Details

The restrictions computed by this scheme must be applied to the speech recognizer if any reduction in perplexity is to be achieved. Testing restrictions during SR or selecting “semantically” among the n-best are both possible implementations. Neither works with currently available SRs; these SRs use BNF grammars and do not deliver semantically distinct alternatives for n-best (Hemphill, 1993; Smith and Bates, 1993).

### 5.1 Compiling restrictions in the SR grammar

The tool we use to impose these restrictions is a compiler capable of converting a grammar composed of patterns and calculated “semantic” restrictions into two compiled grammars: one for use in a speech recognizer and one to parse the recognized words and produce a structure representing the relevant semantics of the sentence. The Unified Grammar and its associated tools fill this requirement, providing a generally adequate approximation to this ideal compiler. The ideal compiler would turn the patterns and restrictions into just patterns, and do so without expanding the compact notation of the original grammar into some “rolled-out” form that is too large for the SR to use; this compactness requirement rules out any approach which enumerates the acceptable sentences of the grammar. The Unified Grammar compiler produces a patterns-only grammar that also reflects the restrictions by pre-computing these tests, when possible, to create more specific patterns reflecting the constraints. It omits restrictions that are too complex for it to effect, thus allowing all the good utterances and possibly some bad ones as well.

### 5.2 Processing steps

To implement the example-based restrictions, the Unified Grammar language was extended to include

tests that could be disabled or enabled with a global switch, and then the following processing was used:

1. Disable the feature restrictions and compile the Unified Grammar to produce a semantic grammar.
2. Parse the (written) page descriptors with the “relaxed” semantic parser, building an index of all the parses which can be used later to locate the related pages of the catalog.
3. Reprocess this index to extract the information about existing modifier types and use this information to add the implied markings to the lexicon.
4. Turn the global switch to enable the lexical restrictions and compile the Unified Grammar again to produce the speech recognition grammar. The compiler will use the enhanced lexicon while applying the restrictions now enabled, and this will produce a “tight” speech recognition grammar.
5. Use the restricted grammars for both speech recognition and semantics extraction when running the catalog with users, so that the system can “hear” and process “canvas diaper bag” but not “cashmere diaper bag”.

## 6 Applicability and application of this approach

### 6.1 Where does this approach work?

It is important to note that this approach depends on there being some simple way to indicate in a grammar what sort of “agreement” is required between the parts of a phrase, and that a relatively rich example set illustrating the “good” agreements also must be available. General language processing lacks one or both of these requirements, so this approach must be understood as having relevance only where the ratio of example data is high relative to the variability that must be supported in the spoken language being processed.

Our example case used a “binary” decision paradigm, completely ruling out combinations which did not match up with criteria from the example set; by using likelihood weighting instead of rigid exclusion, a more flexible system could be built.

### 6.2 Semantic leverage

Clearly, categorization of the items used in this technique improves both the simplicity and the generality of the restrictions that can be generated. For example, using the fact that “diaper bag” and “book bag” are types of luggage, we can write the restriction rules to record and test the markings on their “type” rather than their “species”, and thus get information about the appropriate modifiers for “duf-

fel bag” without having ever “seen” sentences about duffel bags.

## 7 Conclusions

We have presented an implemented scheme which significantly reduces the perplexity of the speech recognition task in cases where the perplexity arises from allowing semantically irrelevant grammatical constructions. This method is applicable where there is a modest collection of relevant sample sentences to support building the restrictions by example. This method is applicable only in certain classes of speech, but in those cases it can automate the otherwise quite tedious task of manually marking semantic restrictions for a grammar.

## Acknowledgments

This work is part of a larger effort within Sun Microsystems Labs prototyping tools to make the use of computer speech more practical. Thanks to my co-Principal Investigator Nicole Yankelovich and to Stuart Adams, Eric Baatz, and Andrew Kehler for their contributions.

## References

- Charles Hemphill. 1993. Dagger: Directed acyclic graphs of grammars for enhanced recognition, user’s guide and reference manual. Technical report, Texas Instruments, May.
- Paul Martin and Andrew Kehler. 1994. Speechacts: A testbed for continuous speech applications. In *Proceedings of the AAAI Workshop on Integration of Natural Language and Speech Processing*, pages 65–71, Cambridge, MA, August. MIT Press.
- Paul Martin, Fredrick Crabbe, Stuart Adams, Eric Baatz, and Nicole Yankelovich. 1996. Speechacts: A spoken language framework. *IEEE Computer*, 29(7):33–40.
- R. Schwartz, L. Nguyen, F. Kubala, G. Chou, G. Zavaliagkos, and J. Makhoul. 1994. On using written language training data for spoken language modeling. In *Proceedings of the Human Language Technology Workshop*, pages 94–98, Plainsboro, NJ, March. Morgan Kaufmann.
- Graeme W. Smith and Madeleine Bates. 1993. Voice activated automated telephone call routing. In *Proceedings of the Ninth IEEE Conference on Artificial Intelligence for Applications*, Orlando, FL, March.
- Nicole Yankelovich. forthcoming. Using natural dialogs as the basis for speech interface design. In Susann Luperfoy, editor, *Automated Spoken Dialogue Systems*. MIT Press.