Automatic Induction of Language Model Data for A Spoken Dialogue System

Grace Chung †, Stephanie Seneff ‡ and Chao Wang ‡

† Corporation for National Research Initiatives
1895 Preston White Drive, Suite 100, Reston, VA 22209
gchung@cnri.reston.va.us

‡ Spoken Language Systems Group
MIT Computer Science and Artificial Intelligence Laboratory
Stata Center, 32 Vassar Street, Cambridge, MA 02139
{seneff,wangc}@csail.mit.edu

Abstract

When building a new spoken dialogue application, large amounts of domain specific data are required. This paper addresses the issue of generating in-domain training data when little or no real user data are available. The two-stage approach taken begins with a data induction phase whereby linguistic constructs from out-of-domain sentences are harvested and integrated with artificially constructed in-domain phrases. After some syntactic and semantic filtering, a large corpus of synthetically assembled user utterances is induced. The second stage involves sampling the synthetic corpus towards the goal of obtaining data that would be representative of the statistics of application-specific real user interactions. The sampling methods proposed employ an example-based generation framework, a simulated user model and information extracted from development data. Evaluation is conducted on recognition performance in a restaurant information domain. We show that word error rate can be reduced when limited amounts of real user training data are augmented with synthetic data derived by our methods.

1 Introduction

A mounting challenge in the building of any new spoken dialogue application is the collection of user data. At every stage of dialogue system development, real user utterances are important for ensuring adequate coverage and countering sparse data problems, both in the language model and understanding components. To obtain an initial corpus, it is customary to conduct a Wizard-of-Oz data collection and/or solicit plausible inputs from potential users. This is usually followed by successive data collections, in parallel with iterative refinements on each dialogue system component. Such an approach tends to be costly, and more automated methods for obtaining data are critical for lowering barriers to deployment.

This paper presents a methodology for synthesizing language model training data tailored to a spoken dialogue query-based application. A training corpus is derived first in the absence of any in-domain real-user data, and secondly, in combination with a small development set. The objective is to develop automatic techniques for creating artificial data that would be well matched to real user data. Created by running user simulations and transforming a large out-of-domain corpus, the artificial data would ideally be used prior to initial deployment and subsequently as an enhancement to the small initially collected data set.

In our approach, we seek to build a training corpus whose frequency distributions would realistically reflect those of user interactions with the dialogue system. Thus, the data must be similar in style to conversational speech encompassing repairs and disfluencies, while they should also maximize on diversity and coverage in terms of syntactic constructions. Moreover, at the sentence level (e.g., different types of queries), and at the class level (e.g., within-class statistics), distributions should closely approximate those of real user dialogues. To this end, our solution for inducing training data begins with a first stage of taking a restricted set of artificial domain data, and then generatively transforming them into a much richer data set, using templates derived from parsing a previously collected corpus from a different application. This approach does not simply identify sentences that are relevant to the new application in the secondary domain, but exploits as much of the secondary corpus as possible. Essentially domain-specific portions of the sentences in the secondary domain are substituted by phrases of the targeted new application. This process of sentence reconstruction yields a very large variety of patterns harvested from the previous domain, and relies critically on

an intermediate step of extensive syntactic and semantic filtering to produce probable user sentences. A second stage involves sampling this over-generated corpus in order to form a final training set that better approximates the distributions of real application specific dialogue interactions. Two different techniques have been implemented for sampling, *user simulation* and *dialogue resynthesis*, both of which employ an example-based generation mechanism to find semantically related sentences. The details of the sampling stage can be found in (Wang et al., 2005).

This paper focuses on the data induction process of our two-stage approach. The structure of the paper is as follows. Previous related work will be presented in Section 2, followed by an outline of the overall approach in Section 3. Three methods of inducing in-domain data are introduced in Section 4, and Section 5 describes how the over-generated corpus is filtered by imposing domain-specific syntactic and semantic constraints, and down-sampled through user simulations or resynthesis of development data (when available). Further enhancements to the data are also described, including incorporation of meta-level queries from existing corpora and modeling of speech artifacts. Section 6 details recognition experiments in a restaurant information system, followed by discussion and conclusions.

2 Related Work

A recent trend in dialogue system development is a focus on minimizing the time and cost in developing a new dialogue system, particularly with respect to obtaining training data (Fabbrizio et al., 2004; Feng et al., 2003; Fosler-Lussier and Kuo, 2001). But dialogue systems are better trained on large amounts of user data that properly characterize the user interactions (Bechet et al., 2004). Generally, with very little training, researchers have sought to obtain more data by supplementing with alternative text sources such as the Web (Bulyko et al., 2003; Feng et al., 2003; Zhu and Rosenfeld, 2001). Some work has been directed towards selecting from an out-of-domain corpus based on some metric for relevance to the application domain (Klakow, 2000; Iyer and Ostendorf, 1999; Bellagarda, 1998). Alternatively, others have turned to language model adaptation where the parameters of a smoothed language model trained from generic data are tuned based on in-domain observations (Bacchiani et al., 2004; Bertoldi et al., 2001; Rudnicky, 1995). (Fabbrizio et al., 2004) addresses the bootstrapping of out-of-domain data by identifying classes of utterances that are either generic or re-usable in the new application. In the absence of any domain data, one common method is to run a (usually hand-coded) context-free grammar in generative mode (Fosler-Lussier and Kuo, 2001; Popovici and Baggia, 1997; Jurafsky et al., 1994). This is proposed in (Galescu et al., 1998) to combine with a language model whose back-off model is trained on out-of-domain data.

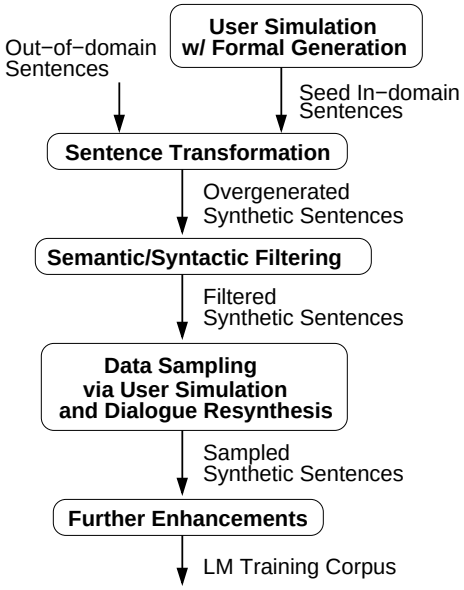

Figure 1: *A schematic depicting successive steps towards the automatic induction of language model data with seed in-domain data from out-of-domain data. The seed data are synthetic, obtained exclusively through simulations.*

In contrast, our method assembles entirely new utterances by inserting artificially constructed in-domain phrases into templates from another unrelated domain. Furthermore, we believe that obtaining the appropriate sentence level statistics by sampling through simulated dialogue interactions would produce higher quality data. Stochastically generated user simulations are increasingly being adopted to train dialogue systems (Levin et al., 2000; Scheffler and Young, 2000), particularly for selecting and evaluating dialogue strategies (Lopez-Cozar et al., 2003; Lin and Lee, 2001; Araki and Doshita, 1996; Hone and Baber, 1995). The method described here uses simulations as one method for pre-selecting training utterances to shape the training corpus statistics.

3 Approach

Figure 1 illustrates the multiple steps proposed in this paper. We begin with generating an initial seed corpus in our target domain; examples are given in a Boston restaurant information system. This domain data (13,000 sentences) was obtained by running the dialogue system in simulated user mode, as previously described in (Chung, 2004). The simulations utilized a stochastic user model that, from the system reply frame, determined a user response, represented as a key-value (KV) string. From the KV representation, the system generated user utterances

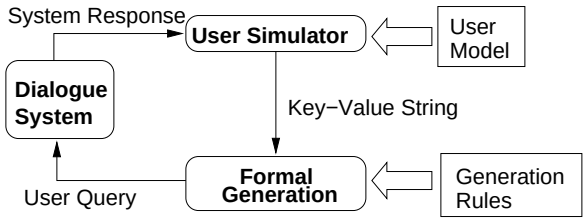

Figure 2: *A schematic showing the process of generating synthetic data by the formal generation method from a key-value representation using user simulation.*

by way of formal generation rules (Baptist, 2000). The technique of inducing data from first principles using formal generation will be outlined in Section 4.1.

Following the creation of a seed corpus, phrases extracted from these in-domain utterances, together with a previously collected flight reservation corpus of 31,000 utterances (Seneff, 2002), undergo a transformation to yield synthetic sentences in the new domain. Two specific methods for the transformation will be described: an automatic template generation and substitution method (Section 4.2) and a formal transformation method (Section 4.3). The resultant set of over-generated and artificially constructed sentences is successively reduced first by selecting on legal syntactic parses, and then filtering on semantic relationships. The subsequent steps then address the process of data sampling to refine the data distribution to better match the statistics expected in realistic user dialogue interactions. The resulting sampled data are then further enhanced with generic meta-level queries and speech artifacts modeling.

4 Domain Data Induction

In this section, we describe three different methods for inducing synthetic corpora for a new domain. The first method involves formal generation from first principles, using a language generation system, and based on a simple semantic representation of the sentence contents. The other two methods involve transforming user queries from another "source" domain into queries appropriate for the new "target" domain – the first method substitutes phrasal units from the target domain into utterances obtained from the source domain, whereas the second method utilizes formal generation rules to "translate" queries from one domain to the other.

4.1 Formal Generation Method

The formal generation method works within a user simulation framework to generate in-domain sentences in the absence of any in-domain or out-of-domain data, as illustrated in Figure 2. During simulation, the end-to-end dialogue system continuously operates in text-mode with a user simulator, described in (Chung, 2004), and the formal rule-based generation system (Baptist, 2000). The system response, encoded in a frame-based meaning representation, known as the system reply frame, is used by the simulator to generate a next putative user query in the form of a key-value string. The formal generation component converts the KV specification into a surface string using trivial generation rules crafted by hand. The rules can support multiple patterns for a specific template, thus adding some variations to the generated surface strings. Table 1 shows an example simulated dialogue.

User:	Okay help me please.
System:	I know about nearly one thousand restaurants in the Greater Boston area. I have information for over one hundred cities, predominantly Boston, Cambridge, Brookline, Somerville, and Newton.
User:	Please give me the restaurant that serves seafood food.
System:	There are 41 Seafood restaurants. They are predominantly in The South End, North End, and Back Bay.
User:	Find the cheapest restaurant.
System:	There are 4 inexpensive Seafood restaurants. Alamo Roast Beef and Seafood, Captain Quarters Seafood, Dolphin Seafood Too, and Grill Fish.
User:	Well where is the second one?
System:	Captain Quarters Seafood is located at 1871 Revere Beach Parkway in Everett.
User:	Please show me the price range for captain quarters seafood.
System:	The price range of Captain Quarters Seafood is inexpensive.
User:	Can you please tell me the telephone number of Elbow Room in Brighton?
System:	The phone number of The Elbow Room is 617-738-9990

Table 1: *Example dialogue between a simulated user and the restaurant information dialogue system.*

A large corpus can be generated by running the simulation process over many thousands of dialogues. The corpus generated by the formal method typically contains very well-formed sentences; however, the linguistic richness is limited by the rules created by the developer.

4.2 Template-Based Transformation Method

The template-based transformation method aims to induce in-domain sentences from available out-of-domain corpora. Essentially, the objective of this method is to capitalize on the diverse *syntactic constructions* and *spontaneous speech phenomena* found in the out-of-domain data, replacing the domain-specific phrases with alternatives appropriate in the new application domain. This step will massively over-generate possible sentences that will be refined using various filtering methods.

In our experiments, we attempt to transform a large corpus of flight reservation sentences into the restaurant

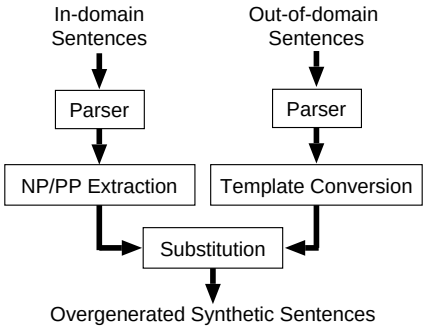

Figure 3: *A schematic showing steps towards generating synthetic data by substituting the NPs and PPs of one domain into the templates derived from a second domain.*

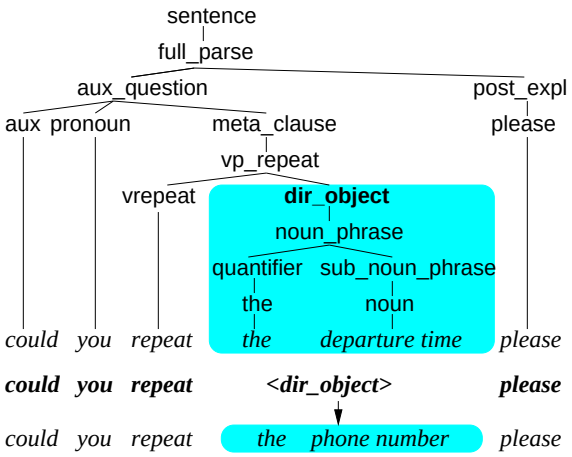

Figure 4: *An example illustrating the transformation of a flight-domain sentence "Could you repeat the departure time please?" into a restaurant domain sentence "Could you repeat the phone number please?" The phrase "the departure time" is substituted by "the phone number" in the* `dir_object` *slot in the template.*

information domain. Each steps of the transformation method is shown in Figure 3. We take advantage of some seed target domain sentences obtained via the formal generation method described in Section 4.1. The seed restaurant sentences are parsed, and all the noun phrases (NPs) and prepositional phrases (PPs), in their various sequential orderings, are gathered under the non-terminal categories in which they occur. Similarly, the flight sentences are parsed, and the locations of NPs and PPs are replaced by non-terminal tags, yielding a set of templates. Some of the non-terminal categories in which NPs and PPs occur are: `direct_object`, `subject`, and `predicate_adjective`. By exhaustively substituting the phrases for each non-terminal category of the target domain into the templates, new artificial sentences incorporating the syntactic constructs of the flight domain and the semantic content of the restaurant domain are synthesized. Figure 4 illustrates the process with an example.

Our initial seed restaurant domain synthetic data yielded 6800 examples of NPs and PPs, and the flight reservation domain yielded 1000 unique sentence templates. Because of the vast number of combinations possible, we terminated the sentence generation procedure after randomly creating 450k *unique* sentences. Some typical sentences from the original restaurant data and the example transformations are shown in Table 2. In comparison with the original artificial data, created from a rule-based method, the new synthetic data are richer, embodying more of the characteristic features of human-computer interactions found in real data. In particular, this template-based approach is able to harvest domain-independent speech artifacts that are embedded within the domain-dependent queries. As a result, we found that the newly constructed data compared with the seed data encompass many more novel phrases that constitute repeats, repairs, greetings and corrections.

4.3 Formal Transformation Method

Another technique that is feasible for inducing sentences for a new application from a secondary domain is to develop formal generation rules which essentially perform a "translation" from one domain to another. The method we propose here reuses a machine translation capability for paraphrasing user queries from a second language back into English. The same set of generation rules that translate one language to another is now modified so that they replace certain semantic concepts from the secondary (flight) domain to those of the new target domain (restaurants).

The language generation capability has some sophisticated mechanisms; for instance, it is possible to control generation from source and destination such that only one of them maps to "in_city" while the other maps to "on_street" or "in_region." Thus we prevent an anomalous query that includes two references to cities. Any flight-domain predicates that are difficult to translate can simply be omitted from the generation rules. Some example query transformations through formal generation are shown in Table 3. A disadvantage of this approach is that it requires manual expertise to develop the formal rules.

5 Data Selection and Enhancement

After the sentence transformation step has taken place, a large corpus of synthetic data is generated and the next phase is to apply a series of refinements on the data quality. These refinements are based on the criteria that utterances should be well-formed in the target domain and

Seed sentences with embedded NPs/PPs:
1. Are there *any Asian restaurants on Richmond street*.
2. Give me *some information on Atasca*.
3. I would like *cheap Mexican food*.
4. Give me *the telephone number*.

Source sentence in flight domain:
1. Also list *any flights from Atlanta to Boston on Thursday morning*.
2. Ok hi i'm looking for *the cheapest flight from Atlanta to Boston*.
3. I mean i only want *the arrival time*.
4. Say *it* again please.

Newly synthesized sentences from templates:
1. Also list *any Asian restaurants on Richmond street*.
2. Ok hi i'm looking for *some information on Atasca*.
3. I mean i only want *cheap Mexican food*.
4. Say *the telephone number* again please.

Table 2: *Examples of how seed sentences in the target restaurant domain are transformed to richer synthetic sentences using templates of a source flight domain. NPs/PPs (italics) are shown (top box) in the original target domain data, (middle box) in the original source sentences from the flight domain, and (bottom box) slotted into the templates from the flight data.*

F:	What meals does the first flight serve?
R:	What credit cards does the first restaurant offer?
F:	Show me flights from Boston to Phoenix via Dallas.
R:	Show me restaurants in Chinatown in Boston that accept credit cards.
F:	I'd like to go to Denver.
R:	I would like to eat in Chinatown.

Table 3: *Example transformations produced via formal generation rules translating flight domain utterances (F) into restaurant domain utterances (R).*

they should be representative of real user interactions. Additional enhancements with meta-level queries and incorporation of noise models in language modeling data will also be discussed.

5.1 Syntactic and Semantic Filtering

While we have not quantitatively measured the similarity between the flight domain and the restaurant domain data, we do assume that the two applications, being quite different, do not share many common query types. Hence the methods described above are likely to generate many sentences that are not appropriate for the new application. For example, in the template-based method, we only replace NPs and PPs, thereby preserving the verb phrases of the flight domain. Thus extensive filtering is necessary to remove irrelevant or improbable sentences.

One obvious approach to filtering is based on syntactic constraints. That is: remove sentences that fail to produce full parse trees under the grammar for the new domain. As for removing unlikely semantic relationships, we have devised a method for filtering based on semantic constraints. The semantics of a sentence are encoded by using a parser to convert it into a hierarchical frame. Each sentence maps to a clause type captured at the top level of the frame, and subsequent descendent sub-frames capture topic-predicate relationships. An example is shown in Table 4.

```
{c request
   :pred {p describe
          :topic {q restaurant
                  :pred {p pred_cuisine :topic "asian" }
                  :pred {p on
                         :topic {q street_name
                                 :name "richmond"
                                 :street_type "street" }}}
```

Table 4: *Example semantic frame for sentence: "Describe any Asian restaurants on Richmond Street."*

In the semantic filtering phase, the first training step is the compilation of all the topic-predicate relationships of the target domain, extracted from the semantic hierarchies. The second filtering step is the parsing and semantic frame construction of the new sentences, and deletion of those containing previously unrecorded topic-predicate relationships. The initial training step processes the original seed data using an algorithm that produces a single tree-like hierarchical frame, storing all observed vertical semantic relationships up to a predetermined depth. At three levels deep, all observed parent-child-grandchild relationships involving clause/topic/predicate sub-frames are documented in a training reference frame. When the new synthetic sentences are parsed, the (parent-child-grandchild) sub-frame tuples from the semantics are compared with those in the training reference frame. If a tuple has not been previously observed, the entire sentence is deleted from the training corpus.

```
{c request
   :pred {p describe
          :topic {q restaurant ..}
          :topic {q pronoun ..} ..}
   :pred {p give
          :topic {q phone_number ..}
          :topic {q address ..} .. }
   :pred {p tell
          :pred {p indir ..}
          :topic {q price_range ..} ..}}
```

Table 5: *A portion of the automatically derived reference frame (depth $n = 3$) that is used in semantic filtering. Shown are some relationships captured under the* `request` *clause.*

Table 5 displays a portion of a training reference frame. Although the trained reference frame, derived from the original seed set (7k sentences), is quite sparse in semantic relationships, this kind of filtering is a crude way

for eliminating sentences with constructs from the flight domain that are not appropriate for the restaurant domain. Generally, novel subject-verb-object relationships that tend to be improbable or nonsensical are rejected, whereas semantic relationships consistent with the seed data are preserved. This does presume then that the seed training data has adequate coverage of the basic query types of the domain, although it does not necessitate semantic or syntactic diversity in the seed data.

Examples of filtered or rejected sentences are depicted in Table 6. Shown are the semantically malformed sentences that have been output from the template-based transformation method but have failed the semantic constraints imposed by the training reference frame. To counter sparsity in the seed data, the developer can enrich the filtered data by manually relaxing the semantic constraints. That is, in several iterative stages, some legitimate but novel semantic relationships are added to the reference frame so that more synthetic sentences would pass the semantic constraints. In the end, the 450k synthetic sentences are reduced to 130k sentences via syntactic and semantic filtering.

1. Is the phone number interested in a restaurant in Boston?
2. Do i read any restaurant?
3. Does the number get in Chinatown?
4. May i use any Chinese food?
5. What neighborhood is the price range?
6. Does their street address make any Chinese food?

Table 6: *Example synthetic sentences that fail the semantic filter. Originally, the sentences are induced by substituting restaurant NPs and PPs into flight domain templates.*

5.2 Data Sampling

Although the data generated as described above cover many variations in syntactic and semantic constructs, it is expected that the frequency distributions in the patterns will not reflect those found in real user data because the data were not gathered in dialogue interaction. For any dialogue system, the proportions of query types, at the sentence level, will depend both on the functionality of the system as well as user behavior. Moreover, the raw data do not encode appropriate within-class statistics, for instance, the lesser prior likelihood of querying for Burmese cuisine versus Chinese cuisine. To gather such statistics, the approach taken here is to reshape the training data by sampling from the raw corpus, utilizing dialogue-level information. The techniques utilized in the sampling technology will only be summarized briefly here; details can be found in (Wang et al., 2005).

The sampling technology relies heavily on an example-based generation method for selecting semantically related sentences. There are two primary components: a collection of sentences indexed by lean syntactic and semantic information encoded as key-value pairs, and a retrieval mechanism to select a candidate from the indexed corpus given a KV specification. Compiled from the raw synthetic data set, the indexed corpus is the pool of synthetic sentences from which we shall sub-sample. During retrieval, the selected candidate sentence can either be used directly, or further processed by substituting the values of certain keys. Two different configurations have been invoked for sampling, which we term *user simulation* and *dialogue resynthesis*. We will be applying both these methods in our experiments.

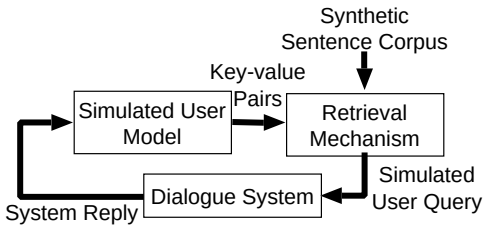

Figure 5: *A schematic showing the dialogue system configuration for collecting a dialogue corpus with a simulated user model, and a retrieval mechanism sampling from raw induced data.*

5.2.1 Simulated User Interaction

The first method, depicted in Figure 5, is conducted by running the dialogue system in simulation mode through thousands of text dialogues. At each dialogue turn, a simulator generates a KV query which is used to retrieve an appropriate sentence from the synthetic corpus, to serve as the user input in continuation to a conversation.

Embodying a parameterized user model, the function of the simulator ((Chung, 2004)) is to stochastically determine an appropriate follow-on user query, given a number of information sources, including the dialogue history and the system response. In total, the contents of the system reply frame, frequency counts of application database and the parameters of the user model are all taken into account when the simulator populates the contents of its output KV frame. After a retrieval mechanism selects a sentence for the particular KV frame, the utterance is then sent to the dialogue system to push the dialogue interaction forward.

As a result, the probabilistic distributions of the embedded semantic content will be shaped by complex interactions of the user model with the system's dialogue strategies. Prior probabilities of within class distributions, estimated from frequency counts of database instances, will further influence the semantic content of the final training corpus.

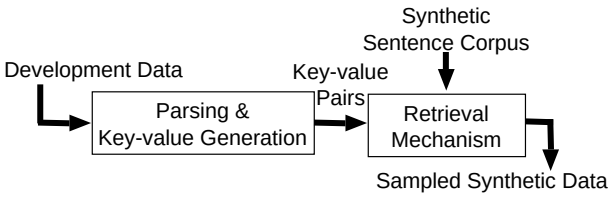

Figure 6: *A schematic showing the configuration for sampling an induced data set using development data dialogue re-synthesis.*

5.2.2 Development Data Dialogue Re-synthesis

The second method for sampling data assumes the availability of a small amount of development data. Figure 6 describes the process of sub-selecting the synthetic corpus via dialogue re-synthesis. Rather than running a closed-loop dialogue system, we simply drive the selection with the semantics of the development data. The entire development corpus is parsed and converted into a KV representation as input to a retrieval mechanism, which directs the selection process on the indexed corpus of synthetic sentences. This technique enables the development data to act as a user model to generate similar but novel dialogues from the synthetic data. The expected training corpus would embed more realistic user behavior, but at the same time, the harvested sentences will contain a richer variety of sentence constructs than those found in the small development set.

5.3 Meta Queries

There is a core component of all spoken dialogue systems that involves so-called "meta queries," which include greetings ("hello"/"good-bye"), dialogue navigation requests such as "start-over," "scratch-that," and "repeat," as well as "help" requests. There are, for example, a surprising number of different ways to say "good-bye," for example, "That's it for now, thank you very much!", and one would expect at least one "good-bye" query in each real-user dialogue. Rather than incorporating such activities into the simulated user model, we decided instead to simply harvest a set of 1158 meta-query sentences from our previous data collection efforts in the flight and weather domains, and augment all of our simulated query corpora with these utterances.

5.4 Noise Models

It is typically the case that so-called "non-speech events" are prevalent in spoken dialogue user queries. In our definition, these include the filled-pause words, such as "um" and "er," as well as laughter, coughs, and other forms of extraneous noise. We have developed a set of acoustic models that are specific to these kinds of sounds, and have painstakingly labeled them in our training corpora for the flight and weather domains. Careful modeling of these events, both acoustically and linguistically, can lead to significant improvements in speech recognition accuracy (Hazen et al., 2001).

Our parser removes such events from the text string before attempting to parse an utterance. As a consequence, they are omitted from the simulated restaurant-domain sentences that are transduced from a flight domain corpus, using either the formal transformation or the template-based method. Of course, they are also missing from the formally generated sentences in our original seed corpus. Hence, we sought a way to reintroduce them with an appropriate distribution over induced corpora.

Our approach was to develop a simple statistical model directly from a large flight domain corpus. Observing that non-speech events tend to be concentrated at the beginning and end of a sentence, we decided to compute statistics for three "positional" specifications: "beginning," "middle," and "end," where "middle" means simply occurring anywhere in the utterance except the very beginning or the very end. We also decided to collapse all such events into a single class, to simplify the modeling aspects. We could then run our generated corpus through a procedure which would randomly insert a noise word at the beginning, exact middle, or end, of an utterance, with an appropriate frequency, given the measured statistic. We selected individual noise words based on their observed unigram frequencies in the flight corpus. There were also a certain number of utterances in the flight corpus that contained *only* noise words, and we added a corresponding proportion of "noise-only" utterances to the synthetic corpus. A "non-word" class was then included in the class bigram to recapture appropriate statistics from the enhanced corpus.

6 Experiments and Results

In this section, we describe the results of several experiments that were conducted on a test set of 520 utterances that were obtained through a data collection effort involving 72 telephone conversations between naive users and the system. Users were asked to interact with a Boston-based restaurant information system via telephone. No specific instructions were provided to the subjects other than a brief introduction to the basic system capability. We excluded recordings which did not contain any speech, but the evaluation data includes utterances (5.4%) with out-of-vocabulary words as well as artifacts such as noise, laughter, etc. The data were transcribed manually to provide reference transcripts.

During the course of system development, we have also collected over 3000 sentences from developers interacting with the system, either via a typed interface or in spoken mode. This set of developer/expert data is probably not representative of real data from naive users. Nevertheless, they are of high quality both in terms of the syn-

Configuration	Num Utts	WER (%)
Baseline(F&F)	203	32.1
Raw Sim	134,526	30.7
Sampling via User Simulation		
(1) Formal Generation	13,352	22.8
(2) Flight Utts: Transform	7,622	24.5
(3) Flight Utts: Template	10,807	22.0
(4) All (1+2+3)	28,564	20.1

Table 7: *Results of recognition experiments using synthetic data to train the recognizer language model. Recognition was performed on a test set of 520 spontaneous naive user queries. (Note: WER = word error rate. F&F = utterances solicited from friends and family before the system existed. Raw Sim = automatically induced data using template-based transformation, prior to applying any sampling methods. See text for discussion.)*

tactic constructs and the semantic content of the queries. These data can thus serve both as a benchmark against which to reference our synthetic corpus performance and as templates from which to guide a sub-selection process.

The recognizer configuration was kept exactly the same for all experiments reported below, except for the language model training data. The recognizer uses word class n-gram language models, with vocabulary and classes automatically generated from the natural language grammar used for parsing, utilizing techniques described in (Seneff et al., 2003). In the deployed system, the recognizer utilizes a dynamic class for the restaurant names, which is adjusted based on dialogue context (Chung et al., 2004). However, for the off-line experiments conducted here, the recognizer is configured to uniformly support all the known restaurant names in the database, under a static NAME word class. The vocabulary size is about 2500 words, with 1100 of these words being unique restaurant names. The acoustic models are trained on about 120,000 utterances previously collected from telephone conversations in the weather and flight information domains.

In the following two sections, we first describe a series of experiments intended to assess the quality of a number of different sets of synthetic data, in the absence of any real user data. We then describe a set of experiments intended to enhance recognition performance of an initial system trained on developer data, by manipulating and/or augmenting the training utterances with additional similar data induced through our synthesis techniques.

6.1 Synthetic Data Only

Table 7 reports word error rates (WERs) for a series of experiments assessing the effectiveness of various sets of synthetic data for speech recognition. These results were compared with a baseline system (F&F) which utilized a set of just 200 made-up sentences that were solicited from "friends and family" prior to system development. Friends were asked to propose queries that they would likely ask a hypothetical restaurant system. Such preliminary data collection efforts in the absence of a functioning dialogue system represent one rudimentary method for jump-starting data collection. It realized a rather high WER of 32.1%.

The "raw" set is a set of over 130,000 utterances that is induced by the template-based method using 30,000 flight domain templates. These have been filtered on syntactic and semantic constraints but have not been downsampled via simulation. In spite of its large size, it only improves slightly over the baseline performance.

Systems 1–4 are all trained on data devoid of any restaurant-specific real-user utterances, but all of them involve user simulation runs. The training utterances for System 1 were generated from first principles, using only formal generation rules in simulated dialogues. Systems 2 and 3 both involve transformations from flight domain utterances. System 2 uses formal generation for translating flight queries into restaurant-domain queries, coupled with user simulation, whereas System 3 is trained on data that is sampled on the template-induced "raw" data set, as described above. Systems 1–3 all yield substantial improvements over the results for the "raw" simulated data, with the template-based approach being the most effective. In particular, when the template-induced data are sampled, WER is reduced by 28.3%, (30.7% to 22.0%). Evidently, as intended, sampling the raw training data through a strategy of user simulations has yielded higher quality training data, because the probability distributions, as estimated by the simulated user model, are much closer to those of real dialogue interactions.

When all three of these sets are combined into a single large set, (System 4), the performance improves further, yielding a recognition error rate of just over 20%. It should be noted that, although the formal method performs relatively poorly by itself, the WER increases to over 21% if data from only Systems 1 and 3 are included in the training corpus. The difference in WER is tested for statistical significance using *matched pairs segment word error test* (Gillick and Cox, 1989), and a significance at a level of 0.03 is established. Hence, the formal transformation method seems to add some novel coverage beyond what the other two sets offer.

For all of these experiments except the F&F, the training data were augmented with the meta queries harvested from the flight and weather data, and the synthetic data were manipulated to insert non-speech events.

Configuration	Num Utts	WER (%)
Dev Only	3497	19.1
Dev + Metas	4691	18.2
Dev + Metas + Noise	4753	18.0
Dev + Metas + Noise + Sim (2x)	9131	17.2

Table 8: *Results of recognition experiments augmenting developer data to train the recognizer language model. Recognition was performed on a test set of 520 spontaneous naive user queries. (Note: WER = word error rate. "Metas" indicates the additional meta-level queries. "Noise" indicates the application of noise models. "Sim" indicates simulated data via user simulation, followed by dialogue resynthesis. See text for discussion.)*

6.2 Augmenting Developer Data

Table 8 summarizes the results of several experiments involving the available corpus of nearly 3500 utterances harvested from developer interactions with the system. By themselves, these utterances (typed plus spoken) yielded a word error rate (WER) that was one percent lower (19.1 versus 20.1) than the best result achieved from data that are entirely artificially generated. A question worth asking, however, is whether these developer data can be improved upon through augmentations with meta queries from previous domains, simulations of noise events, and/or reduplication through dialogue resynthesis to yield variants derived from our corpus of simulated data. As can be seen from the table, each of these augmentations led to further reductions in the WER.

The best performing system, at WER 17.2%, combines the developer data utterances with a synthetic data set. The synthetic data set is obtained by (1) induction via the template-based approach from flight utterances followed by syntactic/semantic filtering, (2) downsampling by user simulation, and finally (3) further downsampling guided by the semantic content of the developer utterances (dialogue resynthesis). Two consecutive runs of dialogue resynthesis are conducted, resulting in two different utterances for each developer utterance. The overall relative improvement achieved by all of these augmentations, compared to the original *Dev Only* system, is 9.9%. This WER difference is significant at the level of $P = .001$. In other experiments we conducted, it was found that combining with synthetic data without applying the dialogue resynthesis did not outperform the system using the "augmented" developer data (*Dev* + *Metas* + *Noise*).

These results suggest that real user data, even when derived from developer/expert users, can be valuable for training a dialogue system. Combining simulated data with developer data has enhanced performance even further. The simulated data clearly add coverage by capturing more novel queries in syntactic constructions and semantic content through the process of induction from flight utterances and simulated dialogue interactions. But this final simulated set also maintains a set of sentence level statistics that directly approximates user interactions of developers. This seems to be better than only using the user model of the simulator.

A further examination of the development set shows that it covers many non-grammatical constructs that are plausible spoken inputs and cause parse failures. Also included are some out-of-domain queries and sentences with unknown words, found in 3.2% of the sentences. These are not modeled by the template-based induction method because the method uses the same parser to derive the meaning representation, and filters out illegal parses such that none of the induced sentences are intended to be out-of-domain or contain unknown words.

In one final experiment, we ascertain a possible lower bound on word error by training on the transcriptions of the test set. This "oracle" condition achieved a 12.2% WER. We can deduce that further manipulations on the language model training data, whether in terms of quantity or quality, while holding the acoustic model constant, would be unlikely to outperform this system.

7 Conclusion

This paper has described novel methods for automatically inducing language modeling data for a new spoken dialogue system. The methodology we implemented involves a step of generatively inducing a large corpus of artificial sentences assembled via a process of parsing and deconstructing out-of-domain data. This is followed by syntactic and semantic filtering of illegal sentences. A final step is concerned with sampling the corpus based on either simulated dialogues or semantic information extracted from development data.

Our experiments have shown that reasonable performance can be obtained in the absence of real data, simply by using synthetic training data. We have demonstrated a method for assembling linguistically varied sentences for a new application by harvesting the sentence constructs found inside queries of a secondary, essentially unrelated, domain. In addition, we have also shown that a training corpus can be refined by incorporating statistics that estimate user interaction with the system. This can be achieved without user data via a user simulation strategy. On the other hand, collecting even some expert dialogue data, typed or spoken, can be beneficial, and the sentence-level distributions of expert users interactions can be exploited to generate even better synthetic data.

We believe that the formal method of transforming flight-domain utterances into restaurant domain utterances is potentially quite powerful, and could be significantly improved by carefully extending and refining the generation rules. We invested only about a one-day effort

thus far, and many of the generated sentences have deficiencies involving remaining flight-domain dependencies which could be eliminated through careful rule modifications. However, this is a time-intensive process that requires expertise, and in that sense this method has disadvantages over the more straightforward method of NP and PP substitution into templates.

8 Acknowledgements

The research at MIT was supported by an industrial consortium supporting the MIT Oxygen Alliance. The research at CNRI was supported in part by SPAWAR SSC-SD. The content of this paper does not necessarily reflect the position or policy of the Government, and no official endorsement should be inferred.

References

Araki M. and Doshita S. 1996. "Automatic evaluation environment for spoken dialog systems," in *Workshop on Dialog Processing in Spoken Language Systems*, Budapest, Hungary.

Bacchiani M. et al. 2004. "Language model adaptation with map estimation and the perceptron algorithm," *Proc. HLT*, Boston, MA.

Baptist L and Seneff S., 2000. "Genesis-II: A versatile system for language generation in conversational system applications," *Proc. ICSLP*, Beijing, China.

Bechet F. et al. 2004. "Mining spoken dialogue corpora for system evaluation and modeling," *Proc. EMNLP*, Barcelona, Spain.

Bellagarda J. 1998. "Exploiting both local and global constraint for multispan language modeling," *Proc. ICASSP*, II, pp. 677–680.

Bertoldi N. et al. 2001. "From broadcast news to spontaneous dialogue transcription: Portability issues," *Proc. ICASSP*, I, pp. 37–40.

Bulyko I. et al. 2003. "Getting more mileage from web text sources for conversational speech language modeling using class-dependent mixtures," *Proc. HLT*, Edmonton, Canada.

Chung G. 2004. "Developing a flexible spoken dialog system using simulation," *Proc. ACL*, Barcelona, Spain.

Chung G. et al. 2004. "A dynamic vocabulary spoken dialogue interface," *Proc. ICSLP*, Jeju, Korea.

Fabbrizio G. D. et al. 2004. "Bootstrapping spoken dialog systems with data reuse," *Proc. SIGDIAL*, Cambridge, MA.

Feng J. et al. 2003. "Webtalk: mining websites for automatically building dialog systems," *Proc. IEEE ASRU*, Virgin Islands.

Fosler-Lussier E. and Kuo H.-K. J. 2001. "Using semantic class information for rapid development of language models within ASR dialogue systems," *Proc. ICASSP*, Salt Lake City, Utah.

Galescu L. et al. 1998. "Rapid language model development for new task domains," *Proc. LREC*, Granada, Spain.

Gillick L. and Cox S. 1989. "Some statistical issues in the comparison of speech recognition algorithms," *Proc. ICASSP*, Glasgow, Scotland.

Hazen T. J. et al. 2001. "FST-based recognition techniques for multi-lingual and multi-domain spontaneous speech," *Proc. Eurospeech*, Aalborg, Denmark.

Hone K. and Baber C. 1995. "Using a simulation method to predict the transaction time effects of applying alternative levels of constraint to user utterances within speech interactive dialogs," *ESCA Workshop on Spoken Dialog Systems*, Hanstholm, Denmark.

Iyer R. and Ostendorf M. 1999. "Relevance weighting for combining multi-domain data for n-gram language model," *Computer, Speech and Language*, 13(3):267–282.

Jurafsky. D. et al. 1994. "The Berkeley restaurant project," *Proc. ICSLP*, pp. 2139–2142.

Klakow D. 2000. "Selecting articles from the language model training corpus," *Proc. ICASSP*, III, pp. 1905–1698.

Levin E. et al. 2000. "A stochastic model of human-machine interaction for learning dialogu strategies," *IEEE Trans. on Speech and Audio Processing*, 8, pp. 11–23.

Lin B. S. and Lee L. S. 2001. "Computer-aided analysis and design for spoken dialog systems based on quantitative simulations," *IEEE Trans. on Speech and Audio Processing*, 9(5).

Lopez-Cozar R. et al. 2003. "Assessment of dialog systems by means of a new simulation technique," *Speech Communication*, 40.

Popovici C. and Baggia P. 1997. "Language modelling for task-oriented domains," *Proc. Eurospeech*, pp 1459–1462..

Rudnicky A. 1995. "Language modeling with limited domain data," *Proc. ARPA Spoken Language Technology Workshop*,

Scheffler K. and Young S. 2000. "Probabilistic simulation of human-machine dialogs," *Proc. ICASSP*, Istanbul, Turkey.

Seneff. S. 1992. "TINA: A natural language system for spoken language applications," *Computational Linguistics*, 18(1).

Seneff. S. 2002. "Response planning and generation in the MERCURY flight reservation system," *Computer Speech and Language*, 16.

Seneff S, et al. 2003 "Automatic induction of n-gram language models from a natural language grammar," *Proc. Eurospeech*, Geneva, Switzerland.

Wang C. et al. 2005. "Language model data filtering via user simulation and dialogue resynthesis," *submitted to Proc. Eurospeech*, Lisbon, Portugal.

Zhu X. and Rosenfeld R. 2001. "Improving trigram language models with the world wide web," *Proc. ICASSP*, I, pp 533–536.