Available online at www.sciencedirect.com
SCIENCE DIRECT®
ELSEVIER
Speech Communication 43 (2004) 183–203
SPEECH COMMUNICATION
www.elsevier.com/locate/specom

Auto-induced semantic classes

Andrew Pargellis *[,1], Eric Fosler-Lussier [2], Chin-Hui Lee [3], Alexandros Potamianos [4], Augustine Tsai [5]

Dialogue Systems Research Department, Bell Labs, Lucent Technologies, 600 Mountain Ave., Murray Hill, NJ 07974, USA

Received 18 November 2003; accepted 31 March 2004

Abstract

Advanced computer dialogue agents contain a natural language understanding component that requires knowledge of semantic classes and concepts. These are frequently generated manually for new tasks. We avoid this time-consuming procedure by using a two-step unsupervised clustering process. First, a semantic generalizer automatically induces semantic classes using training data from well-studied applications (domains) for which large transcribed corpora of human–human dialogues are available. Candidate word pairs are grouped into similar semantic groups according to the similarity of their lexical bigram contexts. We show that the proposed algorithms for automatically inducing semantic classes perform very well for typical spoken dialogue applications. We exceed 90% precision for the first 100 cluster assignments for narrowly defined tasks such as a movie information task. For a heterogeneous task such as the text-based WSJ, a precision of only 24% is increased to 73% by including context thresholding and part-of-speech tagging. Second, we determine the degree of domain independence for each class by using concept comparison and projection metrics to rank order semantic classes by degree of domain independence.

Keywords: Computer; Dialogue agent; Natural language understanding; Unsupervised clustering; Lexical bigram context; Domain independence; Spoken dialogue

1. Introduction

In a typical human–machine spoken dialogue interaction over the telephone network, the telephone call is picked up by a telephony server, which passes the audio stream to a speech recognizer, residing on a remote computer. The automatic speech recognizer (ASR) transcribes the audio stream into a string of text, which is then passed on to a computer dialogue agent. The dialogue agent must extract the information content of this text string. This requires teaching the dialogue manager what semantic concepts to expect, and how to extract them from the person's

* Corresponding author. Tel.: +1-201-386-8055.

E-mail addresses: apargellis@aol.com (A. Pargellis), fosler@cis.ohio-state.edu (E. Fosler-Lussier), chl@ece.gatech.edu (C.-H. Lee), potam@kronos.telecom.tuc.gr (A. Potamianos), augustine.tsai@verizon.com (A. Tsai).

[1] Agilix Corp., 2 Church St. South, Suite #401, New Haven, CT 06519, USA.

[2] Department of Computer and Information Science, The Ohio State University, 2015 Neil Ave, Room 395, Columbus, OH 43210, USA.

[3] School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332-0250, USA.

[4] Department of Electronics and Computer Engineering, Technical University of Crete, Chania 73100, Greece.

[5] Yatek Consulting, P.O. Box 494, Piscataway, NJ 08854, USA.

0167-6393/$ - see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.specom.2004.03.002

utterance. This procedure is the content of the research discussed here.

It is challenging and time consuming to train and build the understanding component of the spoken dialogue system for a new domain (Aust and Schroer, 1998; Chu-Carroll, 1999; Chu-Carroll and Carpenter, 1998; Devillers and Bonneau-Maynard, 1998; Issar, 1997; Lamel et al., 1999; Nakagawa, 1998; Narayanan and Potamianos, 2002; Papineni et al., 1999; Potamianos et al., 1999; Seneff et al., 1998). The difficulty arises because a new domain (or, *task*) for an application is frequently poorly understood because there may very little a priori knowledge about grammar structure or the semantic meanings of words. A comprehensive description of the semantics and grammar for such tasks must then be defined manually.

The first step in designing an understanding module for a domain is to obtain a corpus of transcribed utterances; lacking a working spoken dialogue system the system is often bootstrapped from human–human or Wizard-of-Oz dialogues for that domain. [6] The set of semantic classes, where each semantic class is a meaning representation (concept) consisting of a set of words and phrases with similar semantic meaning, must be identified from that data. Some classes, such as those consisting of lists of names from a lexicon, are easy to specify, e.g., *city*, whereas others require a deeper understanding of language structure and the formal relationships (syntax) between words and phrases, e.g., *departure city*. A developer must supply this knowledge manually, or develop tools that automatically (or, semi-automatically) extract these concepts from annotated corpora with the help of language modeling tools. Manually generating concepts has two main limitations, it is time consuming and requires expert knowledge: an inexperienced developer is prone to omitting important components for each group.

In the past decade, a number of algorithms have been developed for automatically extracting, or inducing, semantic classes from corpora of transcribed conversations (Arai et al., 1998; Bellegarda, 1997; Dagan et al., 1997; Gorin and Riccardi, 1997; Jurafsky et al., 1997; McCandless and Glass, 1993; Pargellis et al., 2001a; Pargellis et al., 2001b; Siu and Meng, 1999). These techniques generate classes that are more comprehensive than manually generated ones, but they require large corpora and often suffer from poor performance (misclassification). Recently, several studies (McCandless and Glass, 1993; Siu and Meng, 1999) have shown that statistical processing algorithms can semi-automatically generate concepts from unannotated corpora for a single domain because semantically similar phrases often share similar syntactic environments (Fosler-Lussier and Kuo, 2001). An iterative procedure is typically used to successively generate groups of words and phrases with similar semantic meaning from a corpus consisting of training sentences. This procedure has been tried on human–machine dialogues for limited tasks such as travel information, but not on a large semantically rich corpus such as the (text-based) Wall Street Journal (WSJ).

We use an unsupervised training approach consisting of two complementary procedures. First, we use n-gram statistics to determine the similarity of words (more generally, *phrases*), by looking at the bigram (and in one case, trigram) lexical contexts *within* a single domain (or equivalently, *task*). Phrases that are determined to be the most similar are grouped into the same semantic class (or, concept). Next, we rank the degree of domain independence for various concept groups across pairs of domains in order to validate the process of using concepts *across* domains. Those semantic groups that are considered domain independent will be ported across domains and used to bootstrap the understanding module for a new, untested, task.

Siu and Meng (1999) automatically induced semantic classes by using the Kullback–Leibler distance as a similarity metric. The choice of which metric should be used to group candidate pairs of words and phrases into a semantic class is clearly a critical issue. In this paper, we compare the per-

[6] These are either dialogues between an agent and a user or collected via a Wizard-of-Oz scenario where the user believes that s/he is conversing with a computer agent while in reality a human is playing the role of the agent.

formance of four different metrics used for auto-induction (Pargellis et al., 2001a,b). These metrics are the *Kullback–Leibler* distance, the *Information-Radius* distance, the *Manhattan-Norm* distance, and the *Vector-Product* similarity (Brown et al., 1992; Dagan et al., 1997; Duda et al., 2001; Manning and Schutze, 2000; Pargellis et al., 2001a). We evaluate the relative performance of these metrics for four different application domains: a movie information retrieval service, the Carmen-Sandiego computer game, a travel reservation system, and the WSJ corpus. The first three domains used relatively small, transcribed dialogues between human subjects and agents while the WSJ was a large, text-based corpus. Of the first three domains, the computer game was a Wizard-of-Oz scenario, whereas the movie and travel information services were human–human dialogues. Five human subjects evaluated the metrics subjectively by comparing the quality of members assigned to each semantic class using four different metrics for each of the four domains. The evaluation of the proposed metrics by comparing their performance on tasks with different degrees of semantic complexity is the first contribution of this paper.

Another problem with speech understanding models and algorithms designed for a single task is that they have little generalization power and are not portable across application domains. Our approach is to determine the degree of domain independence of semantic rules and then automatically extend relevant concepts to new domains, which is the second contribution of this paper.

We hypothesize that domain-independent semantic classes (concepts) should occur in similar syntactic (lexical) contexts *across* domains (Pargellis et al., 2001b). We present results for a methodology that rank orders concepts by degree of domain independence. Two metrics, the *concept-comparison* and *concept-projection* metric, are used to measure the portability of a concept from one domain to another.

The ability to automatically induce a concept in one domain and port it to a new domain for which little training data is available could be a powerful tool, aiding developers in building new speech services. Semantic classes, developed for well-studied domains, could be used for a new domain with little modification. By identifying task-independent versus task-dependent concepts with this methodology, a system developer can import data from other domains to fill out the set of task-independent phrases, while focusing efforts on completely specifying the task-dependent categories manually.

A longer-term goal is to build a descriptive picture of the similarities of different domains by determining which sets of semantic classes (concepts) are most closely related across domains. Such a hierarchical structure would enable one to merge phrase structures from semantically similar classes across domains, creating representations for individual concepts that are more comprehensive. More powerful language models could be built than those obtained using training data from a single domain.

The organization of this paper is as follows: In Section 2, we describe the algorithms designed for automatic induction of concepts in a single domain. The various concept class distance metrics are presented and evaluated. Issues related to the context model order, context thresholding and the stopping criterion for the iterative algorithm are investigated. We then investigate how to port concepts across domains using the concept-comparison and concept-projection methods. Results are presented for each algorithm. We conclude the paper with a summary of the main ideas and experimental results.

2. Auto-induction of semantic classes (single domain)

We propose an iterative procedure for automatically inducing semantic classes, consisting of three main components: a *lexical phraser*, a *semantic generalizer*, and a *corpus parser*. We compare the subjective quality of induced classes by considering: four different metrics for calculating the differences between bigram probability distributions, extending this procedure using trigram contexts, using a minimum threshold on observed contexts in order to isolate statistically

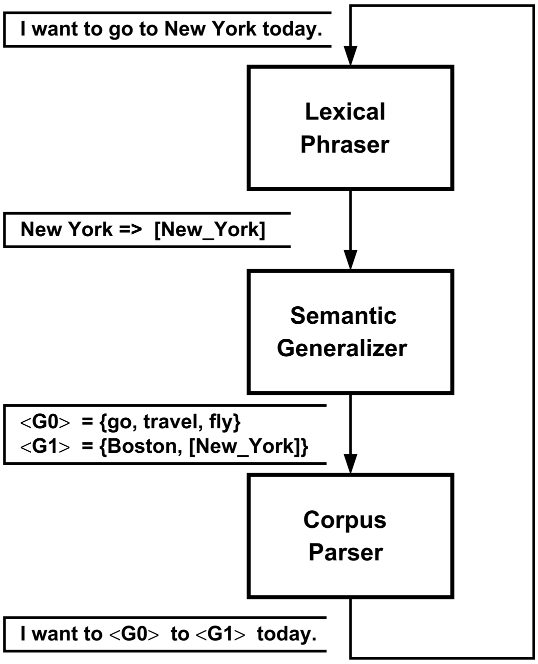

Fig. 1. Iterative procedure used for the auto-induction of semantic classes. The sample sentence is from the Travel domain.

relevant candidates for semantic classes, and tagging each word in the corpus with its part of speech (POS).

The iterative induction process (Jurafsky et al., 1997; McCandless and Glass, 1993; Pargellis et al., 2001a,b; Siu and Meng, 1999) is shown schematically in Fig. 1. There are three main steps to auto-inducing classes, shown by blocks in Fig. 1. First, the *lexical phraser* groups words in a single lexical unit. Next, a *semantic generalizer* generates rules that map words (and concepts) to concepts. Finally, a *corpus parser* re-parses the corpus using the rules generated from the semantic generalizer. On the left-hand side are the successive processed outputs of a typical sentence from the Travel domain.

2.1. Lexical phraser

The top block in Fig. 1 is the lexical phraser. This module generates a list of the most commonly co-occurring lexical phrases (or, sentence-fragments). We generated 25 phrases per iteration for the studies reported here. The results were qualitatively the same when there were between 20 and 40 phrases. We allowed nested, hierarchical phrases and classes were treated the same as words and phrases. Table 1 lists the first 15 phrases, or sentence fragments, from the Travel domain, that were formed from co-occurring words, using the pointwise mutual information measure of Eq. (1).

Table 1
Top 15 phrases, or sentence fragments, obtained for the Travel domain during the first iteration

i'd like ⇒ [i'd_like]
that's fine ⇒ [that's_fine]
thank you ⇒ [thank_you]
[i'd_like] to ⇒ [[i'd_like]_to]
p m ⇒ [p_m]
i need ⇒ [i_need]
i would like ⇒ [i_would_like]
i'll take ⇒ [i'll_take]
make the reservation ⇒ [make_the_reservation]
in the ⇒ [in_the]
what's the ⇒ [what's_the]
[in_the] morning ⇒ [[in_the]_morning]
be fine ⇒ [be_fine]
new york ⇒ [new_york]
could you ⇒ [could_you]

Frequently co-occurring words such as "New York" are chunked into a single phrase, e.g., *New York* => [*New_York*]. Furthermore, we induced hierarchical phrasing by permitting the phraser to operate on its own output.

The lexical phraser groups consecutive words into phrases by using a weighted point-wise mutual information (MI) measure (Manning and Schutze, 2000) to find those lexical entities (referred to as words in the remainder of this paper) that co-occur often. The *n* phrases with the largest MI measure,

$$\mathrm{MI}(w_1, w_2) = p(w_1, w_2)\log\left[\frac{p(w_1, w_2)}{p(w_1)p(w_2)}\right] \tag{1}$$

for the words w_1 and w_2, are kept at each iteration. They are only retained in successive iterations if they are classified into semantic groups in the following, semantic generalizer, module.

We found from studying the three smallest corpora that between 25 and 30 phrases (or more generally, *chunks*) per iteration was a reasonable number for the small corpora used for three of the four domains in this study. For comparison purposes, the same criterion was used for the much larger WSJ corpus. Fewer than 10 chunks meant that certain commonly occurring phrases, such as *I want*, would not be combined. More than 50 chunks created so many nested chunks, such as [[*go_to*]_*Newark*], that entire sentences were frequently combined into a single sentence entity in the smaller domains. This prevented further semantic generalizations for words or sentence fragments (such as *Newark* being a member of a <*city_name*> class, where brackets denote a semantic class label) within these large sentence-level chunks. Table 1 lists the first 15 phrases selected during the first iteration for the Travel domain.

2.2. Semantic generalization

The second block in Fig. 1 shows the semantic generalizer. Grammar rules are generated each iteration, where a rule maps a word, sentence fragment (from the previous block), or previously formed class, into a semantic class whose members share the same meaning. The main criterion for generating such groupings is the lexical or semantic similarity of the left of right-hand context for the members of a group. Consider the following two sentence fragments (taken from the Travel domain).

I want to travel <u>from</u> *Denver* <u>to</u> Dallas.

I wanna go <u>from</u> *Boston* <u>to</u> Pittsburgh on …

The two italicized words occur in the same (bigram) lexical context. Therefore, it is a reasonable hypothesis that they have similar semantic meanings. As we show later, this hypothesis is true as long as a number of other conditions are met. For example, the corpus should have sufficient diversity to prevent dissimilar words from being grouped incidentally.

In this example, city names tend to readily be grouped as they frequently occur in similar environments in the travel domain. Other words, such as *travel* and *go* in the above two sentences, may not be as obvious. Ideally, only one semantic merger would be generated each iteration so that the new semantic group could be incorporated into the corpus immediately. To reduce computational complexity five rules were generated per iteration; no qualitative difference was seen, occasionally the order in generating grammar rules was altered. Recursive rules were not allowed.

2.3. Reparsing the corpus

The corpus is reparsed after semantic generalization. All instances of each of the generalized phrases (words) are replaced with the appropriate class. In the example shown in Fig. 1, the phrase [*New_York*] is merged into the <*city_name*> cluster. Since the computer has no concept of "city_ name", numbers are actually used for each new semantic class as it is created. In this case: [*New_York*] => <*G*1> so all instances of [*New_York*] are replaced with <*G*1> when the corpus is reparsed. This is done for each of the grammar rules generated in the preceding iteration.

2.4. Distance metrics: Bigram contexts

The semantic generalizer pairs words or phrases (generated in the preceding lexical phraser module) according to the similarity of their syntactic environments. We first consider the bigram context, deferring until later (below in Section 3.3.3. Trigram Contexts") the more general trigram context. A candidate word, *w*, is considered with its nearest neighbors in a word sequence

$$\{\cdots \quad v_1^{\mathrm{L}} \quad w \quad v_1^{\mathrm{R}} \quad \cdots\} \tag{2}$$

with v_1^{L} representing the first word in the left context and v_1^{R} representing the first word in the right context. Two probability distributions are calculated, $p^{\mathrm{L}}(v_1^{\mathrm{L}}|w)$ and $p^{\mathrm{R}}(v_1^{\mathrm{R}}|w)$, for the left and right contexts respectively. The right-context bigrams are calculated using the usual word order, and the left-context probabilities are calculated with a reversed-order training corpus using standard n-gram training tools. We used the 1996 version of the CMU toolkit to calculate the n-gram statistics (Clarkson and Rosenfeld, 1997).

We estimate the similarity of two words, w_1 and w_2, as the sum of the symmetric left and right context-dependent distances (Siu and Meng, 1999). This gives the total "distance" between the probability distributions for these two words as

$$d^{\mathrm{LR}}(w_1, w_2) = D^{\mathrm{L}}_{12} + D^{\mathrm{L}}_{21} + D^{\mathrm{R}}_{12} + D^{\mathrm{R}}_{21}, \tag{3}$$

where

$$D^{\mathrm{L}}_{12}(w_1, w_2) \equiv D\left(p^{\mathrm{L}}_1 \| p^{\mathrm{L}}_2\right) \tag{4}$$

is the left-context distance for a given metric D. The "$\|$" symbol refers to the distance, as calculated using some metric, between the two given conditional probability distributions (see Eq. (6) below for an example). Although $\|$ frequently refers to the relative entropy (Kullback–Leibler distance), it refers in general to a distance metric in this report. The conditional probability terms are of the form

$$p^{\mathrm{L}}_1 \equiv p^{\mathrm{L}}_1\left(v^{\mathrm{L}}_1 | w_1\right). \tag{5a}$$

This is the probability that the word v^{L}_1 precedes (is to the left of) the word w_1. The D^{R} distance terms are similar, using the right-context probabilities, p^{R}, as in the term,

$$p^{\mathrm{R}}_2 \equiv p^{\mathrm{R}}_2\left(v^{\mathrm{R}}_1 | w_2\right). \tag{5b}$$

This is the conditional probability that the word v^{R}_1 follows (is immediately to the right of) the word w_2.

The Kullback–Leibler (KL) distance has frequently been used for calculating the distances D in Eq. (3) when auto-inducing semantic classes (Dagan et al., 1997; Jurafsky et al., 1997; McCandless and Glass, 1993; Pargellis et al., 2001a,b; Siu and Meng, 1999). This is a straightforward and intuitive measure of the distance between two probability distributions, p_1 and p_2. However, the KL metric is unbounded since it includes ratios whose denominators may approach zero (see Eq. (7) below). This has the consequence that a few terms, or even just one, can dominate the calculation of the KL distance. This is especially an issue for our studies, since we are interested in developing language models for new domains for which there are limited training data and the statistics can be rather poor, with only one or two observations for some extant n-grams. This inspired us (Pargellis et al., 2001a) to compare results obtained using the unbounded KL distance with results for three other bounded metrics (Dagan et al., 1997). The Kullback–Leibler, Information Radius (IR) and Manhattan Norm (MN) are distance measurements. The fourth, the Vector Product (VP), is a similarity measure.

The bigram language model was built using the CMU-Cambridge Statistical Language Modeling Toolkit (Clarkson and Rosenfeld, 1997). Witten-Bell discounting (Jurafsky and Martin, 2000) was applied and out-of-vocabulary words were mapped to the label UNK. The "backwards LM" probabilities, e.g. the left-contexts $p^{\mathrm{L}}_1(v_1|w)$ for the sequences $\cdots v_1\ w \cdots$ in Eq. (5a), were calculated by reversing the word order in the training set.

2.4.1. Kullback–Leibler distance

The KL distance is frequently used because it is a fundamental metric, the relative entropy (Jurafsky and Martin, 2000). In general, one calculates the probability distributions using bigram lexical contexts, as shown by the sample probabilities in Eqs. (5a) and (5b). In this work, we contrast the quality of semantic classes when using bigrams with those induced using trigrams.

The total symmetric KL distance, for bigram lexical contexts, is given by Eq. (3), where D is replaced by K. As a representative example, the right, bigram-context distance, K^{R}_{12}, between two candidate words, w_1 and w_2, is defined over the vocabulary V as

$$\begin{aligned} K^{\mathrm{R}}_{12} &\equiv K\left(p^{\mathrm{R}}_1(w_1) | p^{\mathrm{R}}_2(w_2)\right) \\ &= \sum_{v^{\mathrm{R}}_1 \in V} p^{\mathrm{R}}_1\left(v^{\mathrm{R}}_1 | w_1\right) \log\left(\frac{p^{\mathrm{R}}_1\left(v^{\mathrm{R}}_1 | w_1\right)}{p^{\mathrm{R}}_2\left(v^{\mathrm{R}}_1 | w_2\right)}\right), \end{aligned} \tag{6}$$

where the sum is over all words in the vocabulary, V. As discussed above, a problem can occur if the logarithm ratio is very small, such as in cases where the statistics are poor. This is especially an issue when developing language models for new domains for which there are limited training data.

2.4.2. Information-radius (Shannon–Jannsen) distance

The IR distance is similar to the KL distance (Dagan et al., 1997), but is bounded because the denominator for the logarithmic ratio is the average of the two probabilities being considered. The total symmetric IR distance $I^{\mathrm{LR}}(w_1, w_2)$ is given by Eq. (3) (where we replace the symbol D, for a generic distance, with I). A representative term is

$$I^{\mathrm{L}}_{12} = \sum_{v \in V} p^{\mathrm{L}}_1(v|w_1) \log \left(\frac{p^{\mathrm{L}}_1(v|w_1)}{\frac{1}{2}(p^{\mathrm{L}}_1(v|w_1) + p^{\mathrm{L}}_2(v|w_2))} \right) \tag{7}$$

with a maximum distance log(2) for each of the four terms. In Eq. (7), $v \equiv v^{\mathrm{L}}_1$ for the left-context. This shorthand notation is also used in the next two sections.

2.4.3. Manhattan-norm (L1) distance

The Manhattan-norm $M^{\mathrm{LR}}(w_1, w_2)$ is just the absolute value of the difference of the two distributions and is given by a similar equation to Eq. (3)

$$M^{\mathrm{LR}}(w_1, w_2) = M^{\mathrm{L}}_{12} + M^{\mathrm{R}}_{12}. \tag{8}$$

It has an upper bound of four, the left and right context sums each being between zero and two. The left-context dependent term is

$$M^{\mathrm{L}}_{(12)} = \sum_{v \in V} \left| p^{\mathrm{L}}_1(v|w_1) - p^{\mathrm{L}}_2(v|w_2) \right|, \tag{9}$$

where $M_{12} \equiv M_{21}$. This distance is also referred to as the “L1 distance”.

2.4.4. Vector product similarity

This metric is a similarity measure, rather than a difference measure. This is the vector product of two vectors, each vector being a sequence of bigram probabilities. The total distance $V^{\mathrm{LR}}(w_1, w_2)$ is similar to that given in Eq. (11). Each term is bounded by zero (no similarity) and one (identical vectors). The left-context vector product is

$$V^{\mathrm{L}}_{1,2} = \frac{\sum_{v \in V} p^{\mathrm{L}}_1(v|w_1) p^{\mathrm{L}}_2(v|w_2)}{\sqrt{\sum_{v \in V} p^{\mathrm{L}}_1(v|w_1)^2 \sum_{v \in V} p^{\mathrm{L}}_2(v|w_2)^2}} \tag{10}$$

and V^{LR} has an upper bound of two, which is the value for which two words are the closest match.

3. Experimental results for several domains

Table 2 contains the corpus statistics for the four domains used in our study. Two domains (Pargellis and Potamianos, 2000; Pargellis et al., 2001a) are corpora from adult human–human conversations: Movie, an information retrieval task; and Travel, an air, hotel, and car reservation system. A third domain: Carmen-Sandiego, a children’s computer game, was a Wizard-of-Oz scenario where children subjects believed they were conversing with a machine (Narayanan and Potamianos, 2002). These first three corpora were small; each corpus contained less than 2500 sentences and fewer than 20,000 words. The fourth domain consists of a subset of 6920 sentences (about 150,000 words) extracted from the Wall Street Journal (WSJ) corpora. This WSJ corpus consists of many topics ranging in size from two sentences (about 40–50 words) to several dozen sentences (about 1000 words). The WSJ was included in this study to investigate the limitations of automatic concept induction when dealing with a large semantically heterogeneous corpus. In Table 2, the set size for each feature is shown; bigrams and trigrams are only included for extant word sequences. A cutoff threshold of three was used for bigrams and trigrams. All word sequences were excluded if there were fewer than three extant bigrams (trigrams).

The types of sentences were very different for each domain. Representative examples of sentences for each domain are shown in Table 3. The *Carmen* domain is a corpus collected from a Wizard-of-Oz study for children playing the Carmen-Sandiego computer game. The vocabulary is limited; sentences are concentrated around a few basic requests and commands. The children interacted in a multimodal environment where they saw images on a

Table 2
Statistics for the four domains: *Carmen-Sandiego*, *Movie*, *Travel*, and *Wall Street Journal*

Feature	CS	Movie	Travel	WSJ
Sentences	2416	2500	1451	6920
Words	12,128	16,386	7811	152,526
Unigrams	433	583	764	13,219
Bigrams	256	368	278	11,441
Trigrams	334	499	240	6484

Table 3
Sample sentences for the four domains discussed in this paper

Carmen-Sandiego: children's game (spoken)
Tell me about the suspect
Can I go look at the clues please

Movie: information retrieval (spoken)
Where is Dumb and Dumber playing near Naperville?
What comedies are playing at Showplace Twelve?
When is Citizen Kane playing near Centerville tonight?
What's playing at Ogden Six in Naperville?

Travel: transaction (spoken)
Sure, that's fine
I need to go to San Francisco and then on to Seattle
I'd like to return on Friday, June twelfth
Okay, I need a flight to Washington DC from Pittsburgh on . . .

Wall Street Journal: news stories (text)
Neither Mr. Rosenfield nor officials of John Blair could be . . .
Sterling also fell to two point nine three three eight marks from . . .
Mcorp, which has twenty one point nine billion dollars in assets, . . .
Revenue for the quarter rose forty percent to ten point nine . . .

monitor (that changed in response to their statements) as they traveled through parts of the United States in search of clues. The *Movie* domain is a collection of open-ended questions from adults but of a limited nature, focusing on movie titles, show times, and names of theaters and cities. The sentence structure and lexical contexts were somewhat confused by the presence of movie titles in their queries. Unlike other domains, locations were usually theater names and only rarely were city names. The *Travel* domain consisted of human–human dialogues where callers phoned an agent in order to make flight, car and hotel reservations. Some of their sentences were very long and complex. This corpus is similar to the ATIS corpus, composed of natural speech. The vocabulary, sentence structures, and tasks are generally more diverse than in the other two domains. They frequently combined multiple sentences (consisting of different dialogue acts) into one sentence. The *Wall Street Journal* (WSJ) domain was a collection of financial news articles, consisting of very long sentences written by professional editors. This domain was very open-ended and contained statements that were far more diverse than the other three domains in this study.

3.1. Example: Auto-induction for the travel domain

Table 4 shows a subset of the rank-ordered list of word pairs, ranked according to the Kullback–Leibler (KL) metric (distances shown in the third column). These data are from the first iteration for the Travel domain. The system exhaustively ranks all statistically relevant pairs, so the pairs at the end of the list are representing words that do not share any similar lexical context.

Table 5 shows some of the classes induced in the Travel domain using the VP similarity metric. The classes shown are some of the first 40 classes induced. Most of the classes are reasonably well defined, matching human judgment. The classes formed predominately correspond to common travel concepts such as dates, place names, and company names. Some of the class members are misclassified, such as <G14> = {*airport*, *seventeenth*}. These are word combinations occurring in a similar lexical bigram-context. In the case of <G14>, the most common lexical context was {. . . *the w/s*}, where */s* is the end of sentence marker and *w* is the slot for one of the two words. This indicates that the bigram context is sometimes too local to capture semantic similarity.

Table 4
Ranked pair list of words and phrases obtained for the Travel domain in the first iteration

Word 1	Word 2	Distance
[i'd_like]	[i_want]	1.1748
[thank_you]	thanks	1.3775
[i_want]	[i_would_like]	1.4103
[i'd_like]	[i_would_like]	1.4298
[i_need]	[i_want]	1.5077
[that_would]	that'll	2.2726
[i_need]	[i_would_like]	2.3923
april	march	2.5909
. . .	. . .	. . .
five	what	116.6524
[that's_fine]	then	134.4937
could	don't	149.0310
just	take	186.7840

The distances shown were calculated using the KL metric.

Table 5
Selected classes for the Travel domain, auto-induced using the VP similarity metric

Class label	Members
<G0>	second, third, sixth, ninth
<G1>	fourteenth, twentieth
<G2>	last, latest, first
<G3>	Boston, Newark, [San_<G21>]
<G4>	[I'd_like], [I_want], [I_need]
<G5>	fourth, fifth
<G14>	airport, seventeenth
<G21>	Antonio, Diego
<G22>	eighteenth, [twenty_<G5>]

In addition, some classes with members of the same semantic meaning are not merged by the time the 40th class was formed. This lack of generalization is also an error. For example, the <G0>, <G1> and <G5> classes each contain only ordinal members and should therefore be merged into a single "ordinal" class. In fact, many of these classes are merged in later iterations. An example is the <G22> class that has begun to combine the ordinal class, <G5> with another ordinal.

Members are assigned to classes and the merging of classes generates a hierarchy of semantic classes. For example, Table 5 shows some members for a "city" class, <G3>. This class is hierarchical because the "San [Name]" city group (<G21>) is merged with Boston and Newark. This class <G3> forms a subgroup class (cities with airports). In addition, all these cities are in the United States; cities from other countries would form other subgroups. The generalizer will steadily form new classes and merge classes together. Eventually, the system begins to over-generalize classes, initially by clauses, eventually at the sentence level. The only easy way to prevent over-generalization is to only allow terminal classes (no mergers allowed). We do not know of a general means to prevent over-generalization while allowing "beneficial" mergers.

3.2. Comparison of the four metrics

We compared the four distance metrics by asking human subjects to subjectively evaluate the quality of semantic classes that were induced by each of the different metrics. The human subjects were asked to evaluate the first 40 classes generated for each metric for each of the four domains in this study. Five naive evaluators were used; they were all high-school students employed at Bell Labs for the summer. They used English as their native tongue, were not familiar with any of the four domains, and had no linguistic expertise. They individually judged classes one-at-a-time except in merger cases where they had to check members of the two classes being merged. The evaluators labeled each terminal and merger rule for each metric and domain. Terminal rules are rules that do not group classes into other classes. Therefore, "*seventy* = > *<G0>*" is a terminal rule, while "*<G1>* = > *<G0>*" is a merger rule. Each rule was given a 0 (bad rule), 1 (good rule), or 0.5 (not clear). The agreement between labelers can be estimated using the standard kappa-evaluation statistic (Cohen, 1960), where $\kappa = 1$ means labelers are in complete agreement and $\kappa = -1$, complete disagreement. A simplified version of the κ statistic is

$$\kappa = \frac{P(a) - P(c)}{1 - P(c)}, \tag{11}$$

where $P(a)$ is the probability of agreement in labeling and $P(c)$ is the probability of agreement by chance. For N possible choices, $P(c) = 1/N$. Since the evaluators had three choices, $P(C) = 1/3$. In our studies, the average value of κ, averaged over all pairs of evaluators and all metrics and domains, is $\kappa = 0.85$. This value ranged from a minimum of 0.74 for the Carmen domain to a maximum of 0.94 for the WSJ task. This indicates that the labelers are mostly in agreement. The high number of $\kappa = 0.94$ for the WSJ domain is simply because the semantic class assignments were in general very poor for this large and diverse corpus and the human subjects gave almost all assignments a score of 0 (bad rule).

Table 6 shows the number of accurately assigned elements (precision) for each of the four metrics for the four domains studied. The forty classes typically included about 110 terminal rules for merging single words and phrases, and about 15 rules for merging two existing classes together.

Table 6
Precision of cluster assignments (%) using each of the four metrics for each of the four domains studied

Domain	IR	KL	MN	VP
Carmen	72.8	69.4	72.2	69.3
Movie	91.6	90.4	89.5	87.7
Travel	77.9	74.7	78.3	73.9
WSJ	20.1	11.1	23.9	23.2

Values are shown after the first 40 iterations (about 110 rules).

The main exception was the WSJ corpus, which had almost no class-class mergers, due to the large size of the WSJ corpus. There was no appreciable difference observed between a metric's ability to merge individual words and phrases into an existing class, and its ability to merge two existing classes.

The quality of words and phrases grouped into semantic classes was worst for the WSJ corpus, with all four metrics misclassifying more than 76% of the semantic class members. All the metrics use bigram-contexts for probability computation, so phrases of the type, $\{\ldots \textit{the } w \textit{ of} \ldots\}$, classify all words w in the same class. In such cases, the words are members of a broad part-of-speech class. An example using the above phrase (WSJ) is the pair of singular nouns (*daughter*, *string*). This problem was not as serious in the other three domains because they consisted of more constrained tasks. Each of the corpora was for a focused topic, with a limited number of query types, a limited vocabulary (less than 800 words), a limited number of semantic classes, and each semantic class had a precise and specific semantic meaning (such as <*city_name*>).

The results are best for all four metrics when inducing classes from the Movie corpus. This is because queries were limited to three types of WH-questions: what, when, where. For example, a typical when-request for this domain was, *When is Lion King playing at Northgate theatre?* Most questions of this type are asked in almost the same way every time, *When is <Movie_Title> playing at <Theater_Name>?* Therefore, the generated classes are very tight and well defined.

Overall, the bounded metrics perform better than the unbounded KL distance. For example, the IR distance is always better than the KL distance because the denominator for the IR terms is an average of the two probability distributions. The poor performance of the VP-similarity may be due to the limited number of extant bigrams.

The data shown in Table 6 are a bit misleading because of errors in the precisions due to the limited sizes of the corpora used to calculate the distances. Therefore, a better method of presenting the data is in a graph of the precision plotted against the number of rules. Fig. 2 shows data taken from the Travel domain. Three out of the five students mentioned above evaluated the quality of the first 200 terminal rules. Ratings were aggregated as more rules were added; the scores from the three subjects were averaged every 10 rules. The precision generally decreases with increasing number of rules as the closest pairs of words and phrases are removed from the corpus. This means that the raters found that rules produced in initial iterations of the semantic clustering were of generally higher quality than later rules.

It can be seen that the IR and MN metrics consistently out-perform the other two. The data for less than 50 rules is quite noisy because terms can be misassigned to a class simply because of the finite data size. The KL distance is especially poor for more than 100 rules. This is expected because the statistics begins to be much worse for those latter assignments when there are fewer extant bigrams used in the sums.

3.3. Additional considerations

The data shown in Table 6 indicate that the fraction of misclassified class members is large enough that manual post-processing would be required. This is especially true for the heterogeneous WSJ corpus, where the precision is less than 24% for all distance metrics tested.

Therefore, we investigated three additional features in order to improve the quality of induced classes. First, some extant n-grams occurred only once or twice so candidate words and phrases were not considered if the number of bigram and trigram extants was below a minimum *context threshold*, in both the right and left contexts. Second, it was observed that a number of words were

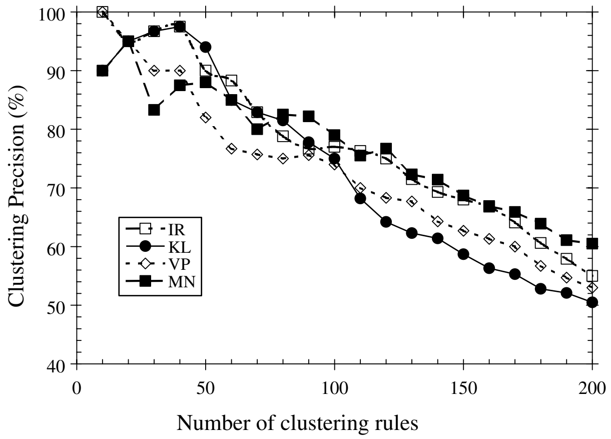

Fig. 2. Clustering precision (%) of auto-induced classes versus number of rules generated for the Travel domain. Four metrics are compared: Kullback–Leibler (●), information radius (□), Manhattan norm (■), and vector product (◇).

merged into classes even when the part-of-speech (POS) was different for the two words. It seems reasonable to use *part-of-speech tags* as an auxiliary filter, so that word pairs such as (*Sunday*, *where*), as occasionally occurred in the Travel domain, would be eliminated as possible candidates. Third, studies from the WSJ corpus shows that the bigram contexts may be too local and that it is reasonable to use *trigram contexts* as a means of improving class assignments.

As shown in the following sections, context thresholding and POS tagging dramatically improved the precision of the auto-induced class members. Even in the worst-case scenario with the large heterogeneous WSJ corpus, the precision increased from about 24% (first 100 rules) to 73%.

3.3.1. Brill part-of-speech tagging

We used part-of-speech (POS) tagging to try to improve the subjective quality of auto-induced classes. POS tagging is a process that assigns a lexical-class marker (or, part of speech) to each word of the corpus. Since the inception of the speech understanding and analysis, people have manually created rules for tagging. There are two steps, the first steps is to mark a word based on a pre-constructed dictionary. A set of disambiguation rules is used to select the candidate tags. These rules are derived from the corpus. In order to automate the tagging process, Jelinek (1985) proposed a Markov model based stochastic tagger. This tagger assigns a tag t_1 to word w_1 based on the maximization of the conditional probability product,

$$p(w_1|t_1)p(t_1|t_2,t_3,\ldots). \tag{12}$$

Although the stochastic methodology alleviates the burden of manually constructing rules, it suffers from an incomplete capture of linguistic information. In 1992, Brill proposed a trainable rule-based tagger (Brill, 1992); not only does it achieve comparable performance as the stochastic tagger, but it also encodes the linguistic information directly into the rules. Each word in a corpus is assigned a tag from the set of Penn Treebank POS tags, e.g., VBD for verbs in past participle form and CC for conjunctions.

We used the POS tagger to annotate each word in a corpus consisting of 6292 sentences taken from WSJ news articles. These tags are used in two ways. First, the language model uses the statistics of the tagged words when calculating the n-gram probabilities. Therefore, a word that occurs with two different POS senses is treated as two separate words. Second, the tags are used in a pre-filtering process. Only those word pairs, which have the same tags, are added to the pair list of words for consideration as candidates to be merged into the same class.

In adding the tags, the language model is altered because a word with different POS senses will be considered as separate words and the statistics for each word/POS pair will be calculated separately. For example, the word *executive* may be used with two different meanings, as seen in the two sentences

... the/DT *executive*/JJ department/NN of/IN ...

... the/DT *executive*/NN who/WP joined/ VBD ...

where each word is tagged with its POS and the word *executive* occurs as an adjective in the first case and noun in the second. Only those words with the same POS tag are considered as candidates for a merger.

3.3.2. Context thresholding

Context thresholding gave the largest improvement in precision. Context thresholding was implemented by only adding words to the word-pair candidate list when a minimum number of bigram and trigram contexts occurred. For example, a threshold of "three" requires a word to have at least three extant bigrams and trigrams. This eliminates singletons, such as cases where the human subject uses a single word to answer a system query, and in general, restricts candidates to those words well represented in a broader lexical context.

In Fig. 3, we combine the effects due to context thresholding and POS tagging. In both cases, with or without POS tags, the context thresholding gave an enormous improvement in precision for the WSJ corpus. Data shown are for a subset of the WSJ corpus using the Kullback–Leibler distance metric. Data are shown for two thresholds, 0 (open symbols, ○ and □), and 3 (solid symbols, ● and ■). Data points for the case without POS tags (described next section) are connected by dotted lines. A context threshold of three means that a word is not considered for merging into a semantic group unless it occurs in the corpus at least three times with two other words (e.g. bigram and trigram context thresholds) in the right and left contexts. Increasing the required minimum number of extant bigrams (and trigrams) consistently increases the precision of the auto-induced classes. This suggests that it is critical to gather enough statistics to determine the similarity between words; using completely backed-off probabilities in the KL distance measurement leads to fallacious groupings. For purposes of clarity, the curves are not shown for intermediate threshold values (1 and 2) but they lie in between the two sets shown.

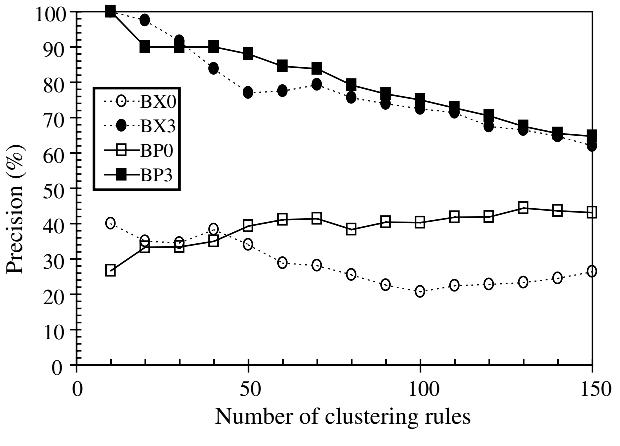

Fig. 3. Clustering precision (%) versus number of rules generated from the WSJ corpus. Two pairs of data are shown, one pair for each of two context thresholds: 0 (open symbols, ○ and □), 3 (filled symbols, ● and ■). The data shown by circles (○ and ●) are without POS tags and the data shown by squares (□ and ■) are using a part-of-speech tagged corpus.

Even requiring only one extant right- and left-context for each word is a big improvement over no extants. In the absence of POS tagging, the precision for the first 100 rules increases from about 21% (open circle) to 43% (curve not shown for only one extant). Requiring three extants for all candidate words increases the precision to about 73% (filled circles). The threshold may be increased further, but then the number of available candidates begins to decrease, so that the precision is higher for the first rules, but then rapidly decreases with later rules.

The precision of the POS tagged data (data points connected by solid lines in Fig. 3) was about 20% higher than that for the non-tagged data for the case where no extant bigrams or trigrams were required. However, in the cases where extant thresholds for bigram and trigram contexts were required, the precision of induced classes was independent of the POS tagging. This indicates that if extant bigrams and trigrams are required, in the right and left contexts, then the lexical context is sufficient, and implicitly acts as a type of POS tagger.

3.3.3. Trigram contexts

Finally, we explored the effect of extending the context range from bigram to trigram. This was inspired because our preliminary studies of the WSJ corpus showed that the precision was very low, on the order of 25% for the first 100 rules. A typical example observed in the Travel domain was the pair of words $\{airport, seventeenth\}$ in the "noun cluster". The most common lexical context that contained these words was $\{\ldots\ the\ \_ < /s >\}$, where $< /s >$ is the end-of-sentence marker and _ is one of the two words. In fact, the WSJ news domain had the most frequently occurring "POS groups". An example is the noun-cluster $\{string, daughter\}$ obtained from the two sentences

> … because of the *string* of losses incurred …

> … and since the *daughter* of the firm's founder …

where both *string* and *daughter* occur in the same left and right bigram contexts. Both these cases indicated the bigram context is too local to capture semantic similarity. This led us to consider using trigrams for the lexical context when calculating the distances between pairs of words.

We follow a procedure similar to that discussed above for the bigram contexts. In the more general trigram context, we consider a candidate word, w, in a word sequence

$$\{\ldots \quad v_2^{\rm L} \quad v_1^{\rm L} \quad w \quad v_1^{\rm R} \quad v_2^{\rm R} \quad \ldots\} \tag{2$'$}$$

with $v_1^{\rm L}$ and $v_2^{\rm L}$ words in the left context and with $v_1^{\rm R}$ and $v_2^{\rm R}$ words in the right context. The right-context probabilities and left-context probabilities are calculated as was done for the bigram case.

Extending the calculations using bigram contexts, we extend the two probability distributions shown above for the bigram case, with

$$p_1^{\rm L} \equiv p_1^{\rm L}(v_2^{\rm L} v_1^{\rm L} | w_1) \tag{5a$'$}$$

and

$$p_2^{\rm R} \equiv p_2^{\rm R}(v_1^{\rm R} v_2^{\rm R} | w_2) \tag{5b$'$}$$

when using the trigram contexts. The above two equations require conditional probabilities for the bigram and trigram contexts. We used the 1996 version of the CMU toolkit to calculate the n-gram statistics.

Similarly, Eq. (6) for the KL distance using bigram contexts, can be extended to the more general trigram-context case,

$$K_{12}^{\rm R} \equiv K\left(p^{\rm R}(v_1^{\rm R} v_2^{\rm R} | w_1) \| p^{\rm R}(v_1^{\rm R} v_2^{\rm R} | w_2)\right) = \sum_{v_1^{\rm R} \in V} \sum_{v_2^{\rm R} \in V} p^{\rm R}(v_1^{\rm R} v_2^{\rm R} | w_1) \log\left(\frac{p^{\rm R}(v_1^{\rm R} v_2^{\rm R} | w_1)}{p^{\rm R}(v_1^{\rm R} v_2^{\rm R} | w_2)}\right), \tag{6$'$}$$

where the sums are over all words in the vocabulary, V, but in the two positions to the right of the word w under investigation. Noting the identity,

$$p(v_1^{\rm R} v_2^{\rm R} | w) \equiv p(v_1^{\rm R} | w) p(v_2^{\rm R} | w v_1^{\rm R}) \tag{13}$$

gives the distance,

$$K_{12}^{\mathrm{R}} \equiv \sum_{v_1^{\mathrm{R}} \in V} p^{\mathrm{R}}(v_1^{\mathrm{R}}|w_1) \log\left(\frac{p^{\mathrm{R}}(v_1^{\mathrm{R}}|w_1)}{p^{\mathrm{R}}(v_1^{\mathrm{R}}|w_2)}\right) + \sum_{v_1^{\mathrm{R}} \in V} p^{\mathrm{R}}(v_1^{\mathrm{R}}|w_1) \sum_{v_2^{\mathrm{R}} \in V} p^{\mathrm{R}}(v_2^{\mathrm{R}}|w_1 v_1^{\mathrm{R}}) \times \log\left(\frac{p^{\mathrm{R}}(v_2^{\mathrm{R}}|w_1 v_1^{\mathrm{R}})}{p^{\mathrm{R}}(v_2^{\mathrm{R}}|w_2 v_1^{\mathrm{R}})}\right), \quad (14)$$

where the first term is the familiar bigram context (see Eq. (6)). The second term is used in addition to the first term when considering trigram contexts.

Fig. 4 shows that the trigram context did not measurably improve the precision of auto-induced classes. The KL distance metric was used for the two pairs of data shown, one pair for each of two context thresholds (0 and 3). The circles connected by dotted lines are for data based on bigrams and the square symbols connected with solid lines are for data taken using trigrams. At best, the trigram statistics only improved performance marginally by a few percent over the corresponding bigrams. It is likely that with our WSJ corpus was too small and the extant trigram counts were too few to make much of a difference. Doing a similar study using a much larger corpus may show an improvement.

3.4. Class hierarchy tree

The semantic generalizer iteratively creates classes from candidate pairs until the system eventually runs out of candidates. Initially, the merger rules are terminal rules, where words and phrases are merged into new or existing semantic classes. In later iterations, most mergers are between classes. Eventually, all classes are merged into a single "sentence-level" class.

The history of these mergers enables one to obtain a class hierarchy tree. In the studies reported here, we were primarily interested in identifying the "leaves" of the semantic class tree, whose members were solely determined based on terminal rules. We tried to derive a "stopping criterion", the point where enough merger rules have been generated, identifying semantic classes of interest to a developer of a natural language understanding system. We have not yet found a good general stopping criterion; for this work, we chose to evaluate the first m groups provided by the semantic generalizer. In our experience, at least with small semantically homogeneous domains, reasonable results are obtained by stopping after $m = 40$ groups have been generated, containing about 110 members.

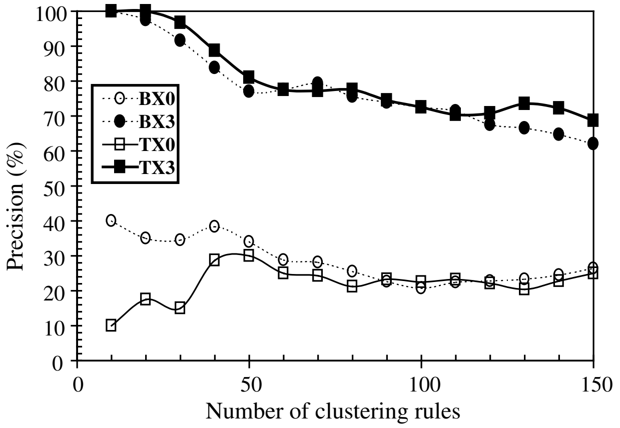

Fig. 4. Clustering precision (%) versus number of rules generated from the WSJ corpus. The data shown by circles (○ and ●) are as in Fig. 3 using a bigram lexical context. The data shown by squares (□ and ■) are for the same training data using a trigram lexical context. No POS tags were used for these data.

4. Porting concepts across domains

In the previous section, we discussed some ways to automatically generate semantic classes from training data for a single domain. In this section, we wish to discuss ways for determining which semantic classes are domain-independent. These classes could be built using data from one domain, and then ported to a second domain that is poorly understood, for which little data is available.

Table 7 contains some of the more common phrases that were shared between the three domains, Carmen, Movie, and Travel. The hyphens indicate no other phrases commonly occurred. The members in these classes were manually selected and ranked according to their frequency in the respective corpus. Although the lexical contexts for the words and phrases are not shown, this table does show that some classes, such as the <WANT> and <YES> classes, contain very similar members. Other classes, such as the <LOCATION> class, contain words that are in similar lexical contexts, but the subset of words greatly varies from one domain to another. Therefore, it is likely that some concepts, such as <WANT> may be portable across domains, whereas other concepts such as <HOTELS> may not.

In the results reported here, we validate our metrics using sets of predefined, manually generated classes. We use two different statistical measurements to estimate the similarity of different domains. Fig. 5 shows a schematic representation of the two methods for the *movie* information domain (which encompasses semantic classes such as <*LOCATION*>, <*THEATER NAME*>, and <*GENRE*>), and the *travel* information domain (with concepts like <*LOCATION*>, <*AIRLINE*>, and <*MONTH*).

The *concept-comparison* metric, shown at the top of Fig. 5, estimates the similarities for all possible pairs of semantic classes from two different domains. Each concept is evaluated in the lexical environment of its own domain. This method should help a designer identify which concepts could be merged into larger, more comprehensive classes and which concepts are useful for many tasks.

The *concept-projection* metric is quite similar mathematically to the concept-comparison metric, but it determines the degree of task (in)dependence for a single concept from *one* domain by comparing how that concept is used in the lexical environments of different domains. Therefore, this method should be useful for identifying the degree of domain-independence for a particular concept. Concepts that are specific to the new domain will not occur in similar syntactic contexts in other domains and will need to be fully specified when designing the speech understanding system.

In order to evaluate these metrics, we decided to compare manually construct classes from a number of domains. We were hoping that the metrics would give us a rank-ordered list of the defined semantic classes, from task independent to task dependent. The evaluation was informal, relying on the experimenter's intuition of the task-dependence of the manually derived concepts.

Table 7
Common phrases for the three concepts, <WANT>, <YES>, and <LOCATION> for the three domains, Carmen, Movie, and Travel

Class	Carmen	Movie	Travel
WANT	I'd like	I would like	I'd like
	I would like	–	I need
	I want	–	I'll need
YES	Okay	Okay	Okay
	Yeah	Yes	Yes
	Good	Fine	That's fine
LOCATION	Alabama	Centerville	Pittsburgh
	Idaho	Warrenville	Boston
	Iowa	Aurora	Cleveland

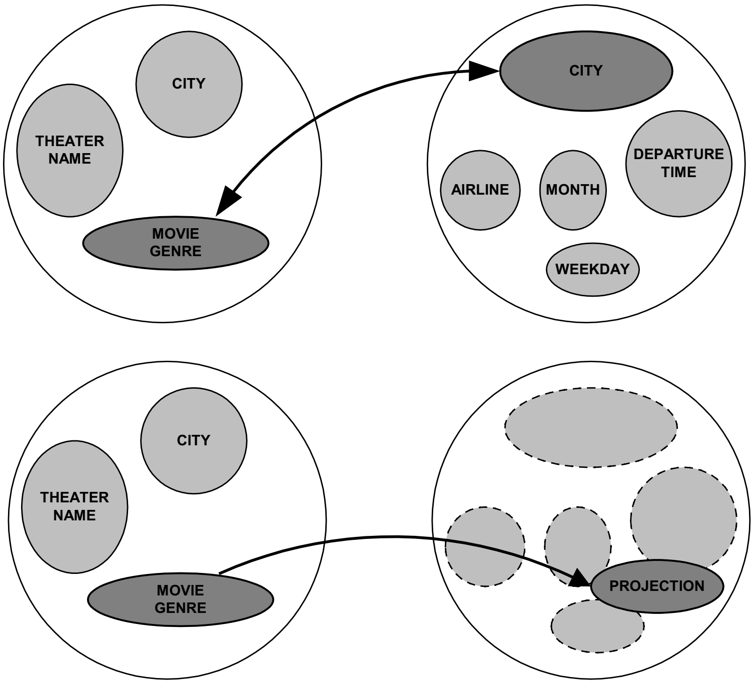

Fig. 5. Pictorial view of the extension of the <GENRE> concept from the *Movie* domain (left) to the *Travel* domain (right). Top: *comparison* method. Bottom: *projection* method.

As an initial baseline test of the validity of our proposed metrics, we calculate the KL distances for the *Travel* and *Carmen* domains using hand-selected semantic classes. A concept class was used only if there were at least 15 tokens in that class in the domain's corpus. As in the studies above, the CMU Toolkit was used to calculate the conditional n-gram probabilities.

4.1. Concept-comparison method: theory

The comparison method compares how well a concept from one domain is matched by a second concept in another domain. For example, suppose (top of Fig. 5) we wish to compare the two concepts, $<$ GENRE $>= \{comedies|westerns\}$ from the *Movie* domain and $< LOCATION >= \{san\ francisco|newark\}$ from the *Travel* domain. We do this by comparing how the phrases *san francisco* and *newark* are used in the *Travel* domain with how the phrases *comedies* and *westerns* are used in the *Movie* domain. In other words, how similarly are each of these phrases used in their respective tasks?

We develop a formal description by considering two different domains, d_a and d_b, containing M and N semantic classes (concepts) respectively. By the way, in the most general case, a "domain" could consist of concepts obtained for a merger of two or more previously studied domains. The respective sets of concepts are $\{C_{a1}, C_{a2}, \ldots, C_{am}, \ldots, C_{aM}\}$ for domain d_a and $\{C_{b1}, C_{b2}, \ldots, C_{bn}, \ldots, C_{bN}\}$ for domain d_b. These concepts could have been generated either manually or by some automatic means. We find the similarity between all pairs of concepts across the two domains, resulting in $M \times N$ comparisons; two concepts are similar if their respective bigram contexts are similar. In other words, two concepts C_{am} and C_{bn} are compared by finding the distance between the contexts in which the concepts are found. The metric uses a left and right context bigram language model for concept C_{am} in domain d_a and the parallel bigram model for concept C_{bn} in domain d_b to form a probabilistic distance metric.

Since C_{am} is the label for the mth concept in domain d_a, we use W_{am} to denote the set of all words or phrases that are grouped together as the mth concept in domain d_a, i.e., all words and phrases that get mapped to concept C_{am}. As an example, $C_{am} = <\mathrm{LOCATION}>$ and $W_{am} = \{san\ francisco|newark\}$. Similarly, w_{am} denotes any element of the W_{am} set, i.e., $w_{am} \in W_{am}$.

In order to calculate the cross-domain distance measure for a pair of concepts, we first replace in the training corpus d_a all instances of phrases $w_{am} \in W_{am}$ with the label C_{am} (designated by $w_{am} => C_{am}$ for $m = 1, \ldots, M$ in domain d_a and $w_{bn} => C_{bn}$ for $n = 1 \ldots N$ in domain d_b). Then a relative entropy measure, the Kullback–Leibler (KL) distance, is used to estimate the similarity between any two concepts (one from domain d_a and one from d_b). The KL distance is computed between the bigram context probability density functions for each concept. This KL distance is similar to the metric used in (Pargellis et al., 2001b), except that it uses domain-dependent probability distributions; the previous work cited only considers probability distributions within one domain.

We calculate the left and right language models, p^{R} and p^{L}. The left context-dependent bigram probabilities are of the form $p_a^{\mathrm{L}}(v|C_{am})$ where $v \equiv v_1^{\mathrm{L}}$ using the nomenclature (for one domain) above in Eqs. (5a) and (5a′). This can be read as "the probability that a word v is found to the *left* of any word in class C_{am} in domain d_a (*i.e.*, the ratio of counts of $\ldots v\, C_{am} \ldots$ to counts of $\ldots C_{am} \ldots$ in domain d_a). Similarly, the right context probability $\left(p_a^{\mathrm{R}}(v|C_{am})\right)$ is the probability that v occurs to the *right* of class C_{am} (equivalent to the traditional bigram grammar).

From these probability distributions, we can define KL distances by summing over the vocabulary V for a concept C_{am} from domain d_a and a concept C_{bn} from d_b. The *left* KL distance is given as

$$\begin{aligned} D^{\mathrm{L}}_{am,bn} &\equiv D\left(p_a^{\mathrm{L}}(C_{am})\|p_b^{\mathrm{L}}(C_{bn})\right) \\ &= \sum_{v \in V}\left[p_a^{\mathrm{L}}(v|C_{am}) \log \frac{p_a^{\mathrm{L}}(v|C_{am})}{p_b^{\mathrm{L}}(v|C_{bn})}\right] \end{aligned} \tag{15}$$

and the right context-dependent KL distances are defined similarly.

We need to define a common vocabulary V for the two domains since the KL distance is calculated by summing over all words in a vocabulary V. We consider two ways to combine the two domains' vocabularies, V_a and V_b, their union, $V_\cup = V_a \cup V_b$, or their intersection, $V_\cap = V_a \cap V_b$.

The distance d between two concepts, C_{am} and C_{bn} is computed as the sum of the left and right context-dependent symmetric KL distances (Siu and Meng, 1999). Specifically, the total symmetric distance between two concepts C_{am} and C_{bn} is

$$d(C_{am}, C_{bn}|d_a, d_b) = D^{\mathrm{L}}_{am,bn} + D^{\mathrm{L}}_{bn,am} + D^{\mathrm{R}}_{am,bn} + D^{\mathrm{R}}_{bn,am}. \tag{16}$$

The distance between the two concepts C_{am} and C_{bn} is a measure of how similar their respective domains' lexical contexts are within which they are used (Fosler-Lussier and Kuo, 2001; Siu and Meng, 1999). If our hypothesis is correct, similar concepts should have smaller KL distances. They may even be the same concept, and a developer could extend to a new task any set of phrases obtained for that concept from previously developed domains.

Larger distances indicate a poor match, possibly because one or both concepts are domain-specific. The comparison method enables us to compare two domains directly as it gives a measure of how many concepts, and which types, are represented in the two domains being compared. KL distances cannot be compared for different pairs of domains since they have different pair probability functions. Therefore, the absolute numbers are not meaningful, although the rank ordering within a pair of domains is.

4.1.1. Concept-comparison method: experiment

Table 8 shows the symmetric KL distances from the concept-comparison method for a few representative concepts. The minimum distances are in bold for cases where the difference is less than four and more than 15% from the next lowest KL distance and multiple entries within 15% are in bold. The union vocabulary V_{U} was used in the summations.

Three of the concepts shown here are shared by both domains, <LOCATION>, <WANT>, and

Table 8
Comparison of hand-selected concepts for the Travel and Carmen tasks

CLASSES: travel domain	CLASSES: Carmen Domain			
	<LOC'N>	<GREET>	<WANT>	<YES>
<CARDINAL>	5.52	5.46	5.87	5.18
<LOCATION>	**2.72**	**3.15**	3.24	**2.92**
<MONTH>	5.60	5.69	6.03	5.51
<WANT>	3.34	2.54	**0.91**	2.45
<WEEKDAY>	4.41	4.52	5.08	4.33
<YES>	3.23	2.43	3.43	**2.09**

<YES>. The <LOCATION>, <WANT> and <YES> concepts have the expected cross-domain KL minima, but <LOCATION>, <GREET>, and <YES> are confused with each other in the *Carmen* task. This occurs because people frequently used these words by themselves in single-word sentences. In addition, children participating in the *Carmen* task frequently prefaced a <WANT> query with the words "hello" or "yes", so that <GREET> and <YES> were used interchangeably. The <CARDINAL> (numbers) and <MONTH> concepts are specific to *Travel* and they have relatively large KL distances (above five) for all concepts in the *Carmen* domain. The <WEEKDAY> category also has a similarity to several *Carmen* classes because words from these classes are frequently used in single-word sentences such as: "hello," "yes", "Monday", or "Boston". These results indicate that single-word sentences should be ignored when comparing concepts across domains.

4.2. Concept-projection method: theory

The projection method investigates how well a single concept from one domain is represented in another domain. If the concept for a movie type is $<GENRE>=\{comedies|westerns\}$, we want to compare how the words *comedies* and *westerns* are used in both domains. In other words, how does the context, or usage, of each concept vary from one task to another? The projection method addresses this question by using the KL distance to estimate the degree of similarity for the same concept when used in the bigram contexts of two different domains.

As with the comparison method, the projection technique uses KL distance measures, but the distributions are calculated using the same concept for both domains. Since only a single semantic class is considered at a time for the projection method, the probability density functions (pdf's) for both domains are calculated using the same set of words from just one concept, but using the respective language models (LMs) for the two domains. A semantic class C_{am} in domain d_a fulfills a similar function as in domain d_b if the bigram contexts of the phrases $w_{am} \in W_{am}$ are similar for the two domains. In the projection formalism, we replace words according to the two rules: $w_{am} => C_{am}$ for both the d_a and d_b domains. Therefore, both domains are parsed for the same set of words $w_{am} \in W_{am}$ in the "projected" class, C_{am}. Following the procedure for the concept-comparison formalism (Eqs. (15) and (16) above), the left-context dependent KL distance $D^{\mathrm{L}}_{am,bm}$ is defined as

$$D^{\mathrm{L}}_{am,bm} \equiv D\big(p^{\mathrm{L}}_a(C_{am})\|p^{\mathrm{L}}_b(C_{am})\big) = \sum_{v\in V}\left[p^{\mathrm{L}}_a(v|C_{am})\log\frac{p^{\mathrm{L}}_a(v|C_{am})}{p^{\mathrm{L}}_b(v|C_{am})}\right] \qquad (17)$$

and the total symmetric distance

$$d(C_{am},C_{am}|d_a,d_b) = D^{\mathrm{L}}_{am,bm} + D^{\mathrm{L}}_{bm,am} + D^{\mathrm{R}}_{am,bm} + D^{\mathrm{R}}_{bm,am} \qquad (18)$$

measures the similarity of the same concept C_{am} in the different lexical environments of the two domains, d_a and d_b.

A small KL distance indicates a domain-independent concept that can be useful for many tasks, since the C_{am} concept exists in similar syntactical contexts for both domains. Larger distances indicate concepts that are probably domain-specific and do not occur in any context in the second domain. Therefore, projecting a concept across domains should be an effective measure of the similarity of the lexical realization for that concept in two different domains.

4.2.1. Concept-projection method: experiment

Table 9 shows the KL distances when the concepts in the *Travel* domain are projected into the other two domains, *Carmen* and *Movie*. In this case, each domain's corpus is first parsed only for the words w_{am} that are mapped to the C_{am} concept being projected. Then the right and left bigram LMs for the two domains are calculated. The results show that the ranking is the same for both domains for the first three concepts: <WANT>, <YES>, <LOCATION>.

Note that for the *Travel* < = > *Carmen* comparisons, the projected distances (Table 9, *Travel* = > *Carmen*) are almost the same as the compared distances (Table 8) for these first three classes. This suggests these concepts are domain independent and could be used as prior knowledge to bootstrap the automatic generation of semantic classes in new domains (Pargellis et al., 2001b). This is not too surprising since the most common phrases in these three classes (shown above for each domain in Table 7) are quite similar. The <WANT> concept is the most domain-independent since people ask for things in a similar way. The <LOCATION> class is composed of different sets of cities, but they are encountered in similar lexical contexts so the KL distances are small. The sets of phrases in the respective <YES> classes are similar, but they also share a similarity to members of a semantically different class, <GREET>.

Table 9
Projection of hand-selected concepts from the Travel domain into the Carmen and Movie domains

Travel classes	Carmen	Movie
<WANT>	1.093	1.766
<YES>	2.138	3.417
<LOCATION>	2.718	4.174
<ORDINAL>	2.988	4.994
<CARDINAL>	4.139	9.931
<MONTH>	4.277	19.209
<DAYPERIOD>	4.498	4.542
<HOTEL>	8.790	9.916
<WEEKDAY>	10.423	9.346

The small KL distances between the two classes, <GREET> and <YES>, may be a bit surprising since they have different semantic meanings. However, there are some concepts that are semantically quite different, yet tend to be used similarly by people in natural speech. It was frequently observed, for the Travel domain, that people would say things like: "Yes, I need a round trip ..." and "Hello, could I have a ticket from ...". In these examples, words such as "hello" and "yes" are really used as grounding terms, indicating the speaker is aware they may now make a request. Therefore, the comparison and projection methodologies also identify similarities between groups of phrases based on how they are used by people in natural speech, and not according to their definitions in standard lexicons.

Future work will address such issues as comparing and projecting auto-induced classes, the soft classification of words and phrases to multiple concepts, and the development of a task hierarchy. Using these techniques, a designer would be able to collect some training sentences, use known statistical techniques to automatically generate semantic classes, and then import additional classes from previously studied tasks that are identified to be similar.

We conclude that both our proposed formalisms for comparing concepts across domains are good measures for ranking concepts according to their degree of domain independence. These metrics could form an extremely powerful tool with which to build understanding modules for new domains.

5. Summary

We have presented results for auto-inducing semantic classes from corpora by grouping candidate word pairs according to the similarity of their lexical bigram contexts. Four different distance metrics were used to determine the similarity of word pairs obtained from corpora in each of four different domains. Our results from the four semantic similarity metrics proposed for auto-inducing semantic classes indicate that the Manhattan-norm and Information-Radius distances are best at classifying words and phrases into semantic groups. Good classification results have been demonstrated for three semantically homogeneous domains using these context-based similarity metrics. However, for the large semantically heterogeneous Wall Street Journal (WSJ) corpus the bigram-context alone did not provide adequate information for auto-inducing semantic classes.

We therefore examined the impact on precision using context thresholding, part-of-speech tagging, and trigram contexts. Context thresholding has the greatest impact on the precision of auto-induced classes from an open-ended corpus such as news articles in the WSJ. The precision of the first 100 terminal rules is about 21% for no context thresholding, increasing to 73% when at least three extant bigrams and trigrams are required. Part-of-speech (POS) tags improve precision, especially for lower levels of context thresholding. The precision increases from 21% to 40% when POS tags are used with no context thresholding, with a further increase to 75% when combined with context thresholding. Perhaps surprisingly, trigram contexts had almost no improvement over the bigram contexts. This is possibly due to poor statistics arising from the small corpus size. Further research is needed to investigate other syntactic and lexical features that indicate semantic similarity, including a comparison of trigram versus bigram precisions using larger corpora.

The concept-comparison or concept-projection methodologies are useful for identifying domain independent phrases. These phrases may then be used to bootstrap auto-induction of more semantic classes for new domains for which there is not much data. The concept comparison technique should be able to identify classes that are candidates for merging. The comparison and projection studies in this report used manually generated classes. Future research needs to be done in the area of porting auto-induced classes across domains.

Acknowledgements

All of the work described in this article was done at the Multimedia Center in Bell Laboratories, Lucent Technologies during the years 2001 to 2002. We gratefully acknowledge the assistance we received from many people at Bell Laboratories, AT&T Research Labs, and the speech community in general. We had especially fruitful discussions with Robert Carpenter, Jennifer Chu-Carroll, Hong-Kwang Jeff Kuo, Helen Meng, Joseph Olive, Alex Rudnicki, and Richard Sproat. Finally, we thank the two reviewers for their careful reading of the manuscript and useful suggestions.

References

Arai, K., Wright, J.H., Riccardi, G., Gorin, A.L., 1998. Grammar fragment acquisition using syntactic and semantic clustering. In: Proc. Fifth Internat. Conf. on Spoken Language Processing, Sydney, Australia, vol. 5, pp. 2051–2054.

Aust, H., Schroer, O., 1998. Application Development with the Philips Dialog System. In: Fujisaki, H. (Ed.), Proc. Internat. Symp. on Spoken Dialogue, Sydney, Australia. pp. 27–34.

Bellegarda, J.R., 1997. A latent semantic analysis framework for large-span language modeling. In: Proc. Fifth European Conf. on Speech Comm. and Tech., Rhodes, 1997, pp. 1451–1454.

Brill, E., 1992. A simple rule-based part of speech tagger. In: Proc. Fifth Darpa Workshop on Speech and Natural Language, San Mateo, CA, Feb 1992, pp. 112–116.

Brown, P.F. et al., 1992. Class-based n-gram models of natural language. Comput. Linguist. 18 (4), 467–479.

Chu-Carroll, J., 1999. Form-based Reasoning for Mixed-Initiative Dialogue Management in Information-Query Systems. In: 6th European Conference on Speech Communication and Technology, Budapest, Hungary.

Chu-Carroll, J., Carpenter, B., 1998. Dialogue management in vector-based call routing. In: Proc. ACL and COLING, Montreal, pp. 256–262.

Clarkson, P.R., Rosenfeld, R., 1997. Statistical Language Modeling Using the CMU-Cambridge Toolkit. In: Proc. Fifth European Conf. on Speech Comm. and Tech.

Cohen, J., 1960. A coefficient of agreement for nominal scales. Educational and Psychological Measurement 20, 307–320.

Dagan, I., Lee, L., Pereira, F., 1997. Similarity-Based Methods for Word-Sense Disambiguation. In: Proc. 35th Annual Meeting of the ACL, with EACL 8.

Devillers, L., Bonneau-Maynard, H., 1998. Evaluation of Dialog Strategies for a Tourist Information Retrieval System. ICSLP'98, Sydney, Australia; 1–4 December 1998.

Duda, R.O., Hart, P.E., Stork, D.G., 2001. Pattern Classification. John Wiley & Sons, Inc, New York.

Fosler-Lussier, E., Kuo, H.-K.J., 2001. Using semantic class information for rapid development of language models within ASR dialogue systems. In: Proc. IEEE Internat. Conf. on Acoustics, Speech and Signal Proc., Salt Lake City.

Gorin, A., Riccardi, G., Wright, J.H., 1997. How May I Help You? Speech Commun. 23, 113–127.

Issar, S., 1997. A Speech Interface for Forms on WWW. In: 5th European Conference on Speech Communication and Technology, Rhodes, Greece.

Jelinek, F., 1985. Markov Source modeling of text generation. In: Skwirzinski, J. (Ed.), Impact of Processing Techniques on Communication.

Jurafsky, D., et al., 1997. Automatic detection of discourse structure for speech recognition and understanding. In: Proc. IEEE Workshop on Speech Recognition and Understanding, Santa Barbara.

Jurafsky, D., Martin, J.H., 2000. Speech and Language Processing. Prentice Hall, Upper Saddle River.

Lamel, L., Rosset, S., Gauvain, J.L., Bennacef, S., 1999. The LIMSI ARISE system for train travel information. In: Proc. IEEE Internat. Conf. on Acoustics, Speech and Signal Proc., Phoenix, Arizona.

Manning, C.D., Schutze, H., 2000. Foundations of Statistical Natural Language Processing. The MIT Press, Cambridge.

McCandless, M.K., Glass, J.R., 1993. Empirical acquisition of word and phrase classes in the ATIS domain. In: Proc. Third European Conf. on Speech Comm. and Tech., Berlin, pp. 981–984.

Nakagawa, S., 1998. Architecture and evaluation for spoken dialogue systems. In: Proc. 1998 Internat. Symp. on Spoken Dialogue, pp. 1–8.

Narayanan, S., Potamianos, A., 2002. Creating conversational interfaces for children. IEEE Trans. Speech and Audio Proc. 10 (2), 65–77.

Papineni, K.A., Roukos, S., Ward, R.T., 1999. Free-flow dialogue management using forms. In: 6th European Conference on Speech Communication and Technology, Budapest, Hungary.

Pargellis, A.N., Potamianos, A., 2000. Cross-domain classification using generalized domain acts. In: Proc. Sixth Internat. Conf. on Spoken Lang. Proc., Beijing, vol. 3, pp. 502–505.

Pargellis, A.N., Fosler-Lussier, E., Potamianos, A., Lee, C.-H., 2001. Metrics for measuring domain independence of semantic classes. In: Proc. 7th European Conf. on Speech Communication and Technology, Aalborg, Denmark.

Pargellis, A.N., Fosler-Lussier, E., Potamianos, A., Lee, C.-H., 2001. A Comparison of four metrics for auto-inducing semantic classes. In: Proc. Automatic Speech Recognition and Understanding Workshop, Madonna di Campiglio.

Potamianos, A., et al., 1999. Design Principles and Tools for Multimodal Dialog Systems, Interactive Dialogue in Multimodal Systems. Kloster Irsee, Germany.

Seneff, S., et al., 1998. Galaxy-II: A Reference Architecture for Conversational System Development, ICSLP'98, Sydney, Australia.

Siu, K.-C., Meng, H.M., 1999. Semi-automatic acquisition of domain-specific semantic structures. In: Proc. Sixth European Conf. on Speech Comm. and Tech., Budapest, vol. 5, pp. 2039–2042.