Context-Sensitive Statistical Language Modeling

Alexander Gruenstein, Chao Wang, and Stephanie Seneff

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

Abstract

We present *context-sensitive dynamic classes* – a novel mechanism for integrating contextual information from spoken dialogue into a class n-gram language model. We exploit the dialogue system's information state to populate dynamic classes, thus percolating contextual constraints to the recognizer's language model in real time. We describe a technique for training a language model incorporating context-sensitive dynamic classes which considerably reduces word error rate under several conditions. Significantly, our technique does not partition the language model based on potentially artificial dialogue state distinctions; rather, it accommodates both strong and weak expectations via dynamic manipulation of a single model.

1. Introduction

The speech recognizer's language model plays a key role in spoken dialogue systems. In most systems, the approach to language modeling can be viewed as being close to one of two extremes: either there is a single large model used to recognize all user utterances, or a series of smaller – often context free – models for each dialogue *state* in the system. Typically, state-based systems demonstrate strong *system initiative*, in which the system guides an information-seeking dialogue in a step-by-step fashion, eliciting the user's response from a narrowly constrained set of alternatives. In contrast, single-language-model systems typically arise as part of an information-state update approach to dialogue, in which the aim is to allow the user to feel as though she could say anything that naturally comes to mind at a particular point in the dialogue. Because the system does not so strongly guide the conversation, it is typically difficult for a dialogue system designer to partition potential information states[1] into all possible dialogue *states* and enumerate their transitions, making it less straightforward to craft state-specific language models in advance – often resulting in the use of a single large model trained on all in-domain data.

While dialogue systems based around information states usually keep rich representations of context in order to *interpret* recognized utterances in context, it is an open question how best to effectively leverage this contextual information to improve recognizer performance without resorting to a state-based approach. In this paper we explore a mechanism for biasing a large n-gram language model based on conversational context. In particular, we develop *context-sensitive dynamic classes* for class n-gram models – dynamic classes with expansions which are updated depending on dialogue context. We train and test our class n-gram models on a corpus of user interactions with the MERCURY flight reservation dialogue system [1] and use the version of MIT's SUMMIT recognizer described in [3]; however the techniques we develop are not particular to that recognizer. We show a large reduction in word error rate under several conditions.

[1] Our definition of *information state* in this paper is meant to be fairly all-inclusive: we mean any detailed representation of conversational context; examples include those implemented in [1] or [2].

2. Previous work

In [2] it is shown how dialogue context can be used to choose among several large segments of a context-free language model (compiled from a unification grammar) in a complex, information-update style dialogue system. The dialogue system's information state is used to determine which subset of the context free grammar should be swapped into the recognizer. If the recognizer fails to produce a hypothesis above a particular confidence threshold, a second pass is made using the larger, more general model. While this technique is powerful, it has the drawbacks that it relies on a hand partitioning of the grammar as well as potentially two recognition passes; additionally, tweaking is required to properly set the confidence level threshold for initiating a second recognizer pass.

Context-sensitive language modeling has also been implemented in systems using n-gram language models. For example, in [4, 5, 6, 7, 8] it is shown how dialogue-state-dependent n-gram models can be used to increase accuracy when they are interpolated with the n-gram model derived from a larger set of in-domain data. The training corpus is subdivided into sub-corpora – one for each dialogue *s*tate. Language models are trained on each sub-corpus, and then interpolated with the larger model trained on all in-domain data. A major difficulty with this approach is that optimal interpolation weights are difficult to set, as perplexity on a held out set does not seem to be a good predictor of word error rate on the test set.

Both techniques require dialogue-system designers to devise a means of partitioning the complex information states of otherwise stateless dialogue systems, so as to choose an appropriate language model for each particular user utterance. In the case of n-gram models, requiring the dialogue-system designer to create definitive segmentations forces the designer to balance the fine-grainedness of the dialogue states with data sparseness. Moreover, as the complexity – and, hence, the number of states – of the dialogue system increases, it becomes increasingly difficult to find the right combination of states to interpolate (automatic clustering has had limited success mitigating some of the difficulty [7, 8]). Similarly, for grammar-based models, the grammar writer must balance writing linguistically motivated rules with domain-specific ones which allow for more optimal subdivisions for recognizer performance. Finally, neither approach reasonably accommodates incorporating highly specific

S1:	How may I help you?
U1:	I'd like to fly from Austin/***$city*** to Oakland/***$city*** on the third/***$digit***.
S2:	Okay, from Boston [*misrecognized*] to Oakland on March third. Can you provide an approximate departure time or airline?
U2:	Not Boston/***$dynsource***, Austin/***$city***.
S3:	Okay, from Austin to Oakland on March third. I've got a flight on American at two o'clock, would that work? Or I've got one on United at four thirty.
U3:	How about the flight at two/***$dyntime***.

Figure 1: *A dialogue snippet tagged with* strong *context-sensitive dynamic classes.* S *indicates system utterances;* U *indicates user utterances. Note that* **$dynsource** *and* **$dyntime** *are examples of context-sensitive dynamic classes, whereas* **$city** *and* **$digit** *are typical static classes*

contextual information into the language model, such as a particular list of airlines the user might have available to choose from at a given moment.

3. Training a language model with context-sensitive dynamic classes

In this section we present a training process for creating *context-sensitive dynamic classes*. Of particular note is that the dialogue system's information state need not be partitioned to train state-specific language models. Instead, we create a single large n-gram language model which supports dynamically biased classes depending on the current context of the conversation. Moreover, it is capable of accommodating highly specific salient contextual information.

It is common practice to create n-gram language models with *classes*, where semantically similar entities are mapped to one class to allow shared n-gram statistics. The SUMMIT recognizer provides an efficient implementation to dynamically swap in different expansions for these classes at run-time. Typically, these are used in cases where the phrases might be a small subset of a large database – such as the names of restaurants or streets in a particular neighborhood as in [3]. We create *context-sensitive* dynamic classes by considering the *context* (in the form of the information state) in which each utterance in the corpus was uttered. We use a corpus of real users interacting with previous versions of our flight system, which contains dialogue state information associated with each utterance. This information state record is used to tag the corpus with context-sensitive classes in a first pass before the normal set of class tags are applied. We explored two types of context-sensitive dynamic classes which we label *strong* and *weak* respectively. The strong model aims to boost *particular* salient entities likely to be mentioned next, while the weak model controls expectations about the *types* of entities occurring in the discourse.

3.1. *Strong* context-sensitive dynamic classes

Strong context-sensitive classes expand to a small list of phrases which are salient in the current context. For instance, if the system has just offered the user a short list of times available, then phrases which refer to these times will become expansions to a particular *strong* context-sensitive dynamic class called *$dyntime*. We created the following strong dynamic classes for our domain:

- *$dynsource*: the city from which the user is departing

```
:date "March3"
:source "AUS"
:destination "OAK"
:reply_frame {
  :best_departure {
      :departure_time "2:00pm"
      :airline "AA" }
  :second_departure {
      :departure_time "4:30pm"
      :airline "UA" } }
```

$dynsource	`"austin"`
$dyndestination	`"oakland"`
$dyntime	`"two", "two o'clock", "two p.m", "two o'clock p.m", "four thirty", "four thirty p.m"`
$dynairline	`"united", "united airlines" "american","american airlines"`
$dynnthflight	`"the first one", "the first flight", "the second one", "the second flight"`

Figure 2: *A subset of the information state following utterance* S3 *in figure 1 shown with the corresponding strong context-sensitive dynamic class expansions used to recognize* U3

- *$dyndestination*: the city to which the user wants to go
- *$dynairline*: the possible airlines the system has offered as available for that route
- *$dyntime*: the times of the flights the system has offered
- *$dynnthflight*: phrases such as *the first one* or *the second flight* dependent on the number of flights offered

Figure 1 shows a sample dialogue snippet after the user's utterances have been tagged with context-sensitive dynamic classes (as well as with normal classes). Figure 2 shows how the dialogue information state following utterance *S3* gives rise to a particular set of expansions for each dynamic class, which in turn allows the word *two* in utterance *U3* to be tagged with the context-dependent class *$dyntime*. Without such context-sensitivity, *two* would be tagged more generally as *$digit*. Similarly, *Boston* in utterance *U2* would be tagged as *$city* rather than *$dynsource* in the absence of context-dependent classes. As a result, context-sensitive trigram weights are learned such as P(*$city*|[*not,$dynsource*]) and P(*$dyntime*|[*one,at*]). Search paths involving a particular dynamic class are active in the recognizer search exactly when that dynamic class is non-empty: by dynamically populating the classes based on dialogue context, contextually relevant n-grams lend weight to salient phrases in certain contexts, and decrease their weight when they are not contextually salient.

3.2. *Weak* context-sensitive dynamic classes

We also experimented with *weak* context-sensitive dynamic classes. We define these as occurring in cases when the dialogue context provides knowledge of the *type* of information the user might soon supply, but with less specificity than in the *strong* case above. Specifically, we have experimented with expectations regarding *airlines* and *locations*. Using the same procedure as above, we have tagged occurrences of *airlines*, *cities*, *city state* combinations, *states*, *city country* combinations, and *countries* in each case with a different class tag depending on whether the utterance occurred in one of three conditions:

S1:	How may I help you?
U1:	I'd like to fly from Austin/***$city2*** to Oakland/***$city2*** on the third.
S2:	Okay from Boston [*misrecognized*] to Oakland on March third. Can you provide an approximate departure time or airline?
U2:	Not Boston/***$city3***, Austin/***$city3***
S3:	Okay, from Boston to Oakland on March third. Can you provide an approximate departure time or airline?
U3:	United/***$airline1*** in the afternoon.

Figure 3: *A dialogue snippet tagged with weak classes.*

C1: The current dialogue context is such that the user has just been prompted for one of the class members: for example, the system has prompted the user for an airline.

C2: The system has just prompted with *How may I help you?*

C3: All other contexts.

Figure 3 provides an example dialogue with the weak classes tagged. We note that, for example, *Austin* and *Oakland* in *U1* have been tagged as *$city2* indicating that they appear in condition C2, while *United* in *U3* has been tagged as *$airline1* to indicate that it falls under condition C1.

Unlike *strong* classes, the class expansions for the *weak* classes are not manipulated at run time. Instead, particular pretrained *weak* classes are *enabled* or *disabled* at run time depending on dialogue context. Exactly one of the conditions enumerated above must hold at all times in the dialogue, and that condition determines which of the three variants is enabled at any given time, and which two are disabled. This allows us to learn context-dependent n-gram weights. For instance, *$city3* occurs relatively infrequently compared to both of *$city1* and *$city2*; and all three have distinct co-occurrence statistics.

4. Evaluation

We evaluated the strong and weak techniques independently using a corpus of 26,886 utterances collected over several years of real user interaction with various versions of the MERCURY flight reservation system [1]. We trained with 24,815 utterances and tested on 2,071, creating trigram language models with a base vocabulary size of 1,586 (excluding class expansions noted below). We manipulated two relevant variables in creating the static language model baselines to produce four different conditions. First, we varied how the *expansion weights* for several of the classes were calculated, controlling how likely each phrase within a single class is to appear. We compared three different ways of calculating the expansion weights: the first looked at how often each expansion appeared in the training corpus, the second gave uniform weights to each expansion, and the third weighted *$city* and *$city_state* based on each city's population. In tandem with varying the expansion weights, we manipulated the *class vocabulary size* of the *$city* and *$city_state* classes. The conditions are summarized as follows:

CM Expansion weights for *$city*, *$city_state*, and *$airline* based on **c**orpus statistics; **m**edium vocabulary size of: 516 city names, 329 city_states, 68 airlines

UM Expansion weights **u**niform; **m**edium vocabulary.

PL Expansion weights for *$city*, *$city_state* based on **p**opulation, *$airline* uniform; **l**arge vocabulary: 16,956 city names, 25,334 city_states, 68 airlines.

UL Expansion weights **u**niform; **l**arge vocabulary.

We chose these conditions because they exhibit problems commonly confronted by dialogue-system designers. Creating dialogue systems tends to be an iterative process: it is often necessary to collect data using a more limited pilot system in order to later deploy a more capable dialogue system. For example, while the data collected from an initial version of a flight reservation system may only include a few cities, in order to deploy a more robust version in the future with more cities, one can use a proxy, such as population, to estimate class-expansion weights until actual usage data is collected.

Table 1 compares the overall word error rates achieved in each condition by the baseline static language model with those produced by the weak and strong models respectively. It further breaks down the statistics by looking at the word error rates achieved on relevant subsets of the test set discussed below. In all four conditions, both the strong and the weak models outperform the static baseline, with the weak model leading to greater reductions than the strong. The relative drop in overall word error rate ranges from 1% in the CM condition to 17% in the UM case. The results show that the dynamic models make the most gains under the conditions in which the class expansion probabilities are not estimated from the training data.

4.1. Utterances targeted by *strong* classes

In addition to the overall word error rate in each condition, we were particularly interested in the effect of the strong context-sensitive classes on the utterances in the test set for which they are *targeted*. By *targeted*, we mean simply utterances for which the dialogue context supplied expansions to some or all of the dynamic classes, making them non-empty. In the case of the *strong* dynamic classes, it is these utterances in particular which the dynamic classes are meant to be "sensitive" to – that is, it is our hypothesis that these classes are helpful because they isolate and boost n-gram statistics involving salient phrases; hence when the classes are populated based on conversational context, it is likely that expansion phrases of these classes will be uttered. We use the terminology that such utterances are *targeted*, while calling all other utterances in the discourse *untargeted*. In figure 1, the user's utterances would be classified as follows:

U1: Not targeted by any strong class.

U2: Targeted by *$dynsource* and *$dyndestination* only.

U3: Targeted by *$dynsource*, *$dyndestination*, *$dyntime*, and *$dynairline* only.

We note that in those examples some of the dynamic class expansions appear in the actual utterances (*Boston*, and *two*), but this is not a requirement. Indeed, *U3* is targeted by *$dynairline* despite the fact that neither *United* nor *American* appears in the utterance. Targeted utterances are those for which we expect that it is more likely that salient phrases will appear, not utterances in which they necessarily do appear (*i.e.* they are *not* chosen by an "oracle").

Table 1 compares results over two subsets of strong *targeted* utterances: those targeted by one or both of *$dynsource* or *$dyndestination* as well as those targeted by one or more of *$dyntime*, *$dynairline*, or *$dynnthflight*. We draw this distinction because *$dynsource* and *$dyndestination* tend to be less specific indicators of the context of the utterance than the other classes. Indeed, throughout much of a typical conversation, either *$dynsource* or *$dyndestination* will be non-empty, as the source and destination of a flight are usually settled on immediately. Hence, while targeted utterances in this category comprised 60% of the test set, they saw only a 1.5% drop in word

				Strong: *source*, *destination*				Strong: *time*, *airline*, *nthflight*				Weak: *location, airline*			
	Overall WER			*Targeted*		*Untargeted*		*Targeted*		*Untargeted*		*C1 & C2*		*C3*	
	static	**strong**	**weak**	**static**	**strong**	**static**	**strong**	**static**	**strong**	**static**	**strong**	**static**	**weak**	**static**	**weak**
CM	17.8	17.7	**17.6**	19.5	**19.2**	16.0	16.0	18.6	**16.0**	**17.7**	17.8	14.7	**14.4**	19.5	**19.3**
UM	25.2	24.4	**20.8**	29.4	**27.8**	**20.6**	20.7	32.0	**25.3**	**24.1**	24.2	18.9	**17.0**	28.7	**22.9**
PL	27.1	26.7	**26.0**	30.1	**29.1**	**23.9**	24.0	30.4	**27.6**	26.6	**26.5**	**23.6**	27.2	29.1	**25.4**
UL	46.7	45.0	**42.1**	52.5	**50.1**	40.4	**39.4**	52.9	**47.1**	45.7	**44.6**	38.4	**34.8**	51.2	**46.1**

Table 1: *Comparison of word error rates of static baseline language models to ones with context-sensitive dynamic classes over four different conditions. Each condition's overall word error rate is shown, as well as the rates among two sets of targeted and untargeted utterances in the strong case, and one subset in the weak case. Bolded results indicate the best score among each comparison set.*

error rate. Conversely, *$dyntime*, *$dynairline*, and *$dynnthflight* tend to be non-empty most often just after the system has offered the user a selection of multiple flights. Therefore they are strongly salient, and the user is highly likely to invoke one of the phrases in these dynamic classes to select a flight. Thus, while targeted utterances in this category constituted only 13% of the test set, they saw a 14.4% decrease in error rate in the CM case, accounting for much of the overall improvement. This indicates that in a system where we were able to target a larger number of utterances we would likely see a larger reduction in overall word error rate.

4.2. Subsets based on applicable *weak* conditions

Finally, table 1 also compares the word error rates produced by the weak language models on utterance subsets formed by determining which of the three conditions enumerated in section 3.2 were active during recognition of a particular utterance. The first subset includes all utterances under conditions C1 or C2, when we expect the user is likely to name an airline or location. The second subset includes just those recognized when condition C3 held, indicating that based on dialogue context the user is unlikely to name an airline or location. Here, the weak model outperformed the static baseline in all comparisons except under subset C1/C2 evaluated in the PL condition. The largest gains generally came under condition C3 – there is, for example, a 20% relative reduction in word error rate in the UM case. This is to be expected, as conditions C1 and C2 are similar to the baseline in that airlines and locations are fairly common in both; conversely, in C3, locations and airlines are *not* likely to be mentioned, thus leading the weak model to discount the weights on a large swath of lexical items.

Further analysis of the anomalous PL case shows almost all of the increase in word error rate came in condition C1. We hypothesize that the population model may heavily weight a few cities which appear very infrequently in the test set. As the weak model under C1 boosts the probability of seeing a city over the baseline, it may be that under C1 we actually exacerbate the problems caused by our poor expansion probability model.

5. Conclusions and future work

We have demonstrated two techniques for training and deploying *context-sensitive dynamic classes* in n-gram language models which leads to a reduction in word error rate compared with standard class n-gram models. The techniques, which are fairly simple to implement, employ context provided by the dialogue system's information state, which is already available for other utterance processing tasks typically occurring at later stages. The primary advantage of the *strong* technique is that it effectively integrates precise contextual expectation into the language model in real time. Both approaches free the dialogue system designer from having to build several small models which must be carefully interpolated. In addition, the techniques we have presented are easily scalable: new context-sensitive dynamic classes can be added to the system without necessitating a repartitioning of the dialogue flow into distinct states. Finally, the approach is fully compatible with the interpolation and backoff techniques discussed in section 2; combining these techniques could lead to greater payoffs.

One major piece of work left to be done is in successfully combining the *strong* and *weak* models. There are some difficulties in doing this correctly, as both weak and strong expectations will simultaneously exist for the same utterance: for instance, we presented here both strong and weak classes that capture information about airlines. If the system has a strong expectation about a particular set of airlines, it would be wrong to swamp this likelihood by the weaker expectation that the user may be likely to say any airline at this point. We have done pilot work on combining the models, but have not yet worked out the proper technique for balancing them.

6. Acknowledgments

Thanks to Dr. Lee Hetherington and Dr. T.J. Hazen for their generous support with technical issues involving the recognizer. This research is funded in part by an industrial consortium supporting the MIT Oxygen Alliance.

7. References

[1] S. Seneff, "Response planning and generation in the MERCURY flight reservation system," *Computer Speech and Language*, vol. 16, pp. 283–312, 2002.

[2] O. Lemon and A. Gruenstein, "Multithreaded context for robust conversational interfaces: context-sensitive speech recognition and interpretation of corrective fragments," *ACM Transactions on Computer-Human Interaction*, 2004.

[3] G. Chung, S. Seneff, C. Wang, and L. Hetherington, "A dynamic vocabulary spoken dialogue interface," in *Proceedings of Interspeech*, 2004, pp. 327–330.

[4] R. A. Solsona, E. Fosler-Lussier, H.-K. J. Kuo, A. Potamianos, and I. Zitouni, "Adaptive language models for spoken dialogue systems," in *Proceedings of ICASSP*, 2002.

[5] A. Aaron *et al.*, "Speech recognition for DARPA communicator," in *Proceedings of ICASSP*, 2001.

[6] K. Visweswariah and H. Printz, "Language models conditioned on dialogue state," in *Proceedings of Eurospeech*, 2001, pp. 251–254.

[7] F. Wessel, A. Baader, and H. Ney, "A comparison of dialogue-state dependent language models," in *Proceedings of ESCA Workshop on Interactive Dialogue in Multi-Modal Systems*, 1999, pp. 93–96.

[8] W. Xu and A. Rudnicky, "Language modeling for dialog system," in *Proceedings of ICSLP*, 2000.