Strategies for Teaching "Mixed" Computational Linguistics classes

Eric Fosler-Lussier
Dept. of Computer Science and Engineering
Dept. of Linguistics
The Ohio State University
Columbus, OH 43210, USA
fosler@cse.ohio-state.edu

Abstract

Many of the computational linguistics classes at Ohio State draw a diverse crowd of students, who bring different levels of preparation to the classroom. In the same classroom, we often get graduate and undergraduate students from Linguistics, Computer Science, Electrical Engineering and other departments; teaching the same material to all of these students presents an interesting challenge to the instructor. In this paper, I discuss some of the teaching strategies that I have employed to help integrate students in two classes on automatic speech recognition topics; strategies for a graduate seminar class and a standard "lecture" class are presented. Both courses make use of communal, online activities to facilitate interaction between students.

1 Introduction

As one of the themes of the Teach-CL08 workshop suggests, teaching students of many kinds and many levels of preparation within a single course can be an interesting challenge; this situation is much more prevalent in a cross-disciplinary area such as computational linguistics (as well as medical bioinformatics, etc.). At Ohio State, we also define the computational linguistics field relatively broadly, including automatic speech recognition and (more recently) information retrieval as part of the curriculum. Thus, we see three major variations in the preparation of students at OSU:

1. **Home department:** most of the students taking CL courses are either in the Linguistics or Computer Science and Engineering departments, although there have been students from foreign language departments, Electrical Engineering, Psychology, and Philosophy. Although there are exceptions, typically the engineers have stronger mathematical and computational implementation skills and the non-engineers have a stronger background in the theoretical linguistics literature. Bringing these groups together requires a balancing between the strengths of each group.

2. **Specialization (or lack thereof):** Many of the students, particularly in seminar settings, have particular research agendas that are not traditionally aligned with the topic of the class (e.g., students interested in parsing or computer vision taking an ASR-learning course). Furthermore, there are often students who are not senior enough to have a particular research track, but are interested in exploring the area of the course. Our courses need to be designed to reach across areas and draw on other parts of the curriculum in order to both provide connections with the student's current knowledge base, and allow the student to take away useful lessons even they do not plan to pursue the topic of the course further.

3. **Graduate vs. undergraduate students:** in both the CSE and Linguistics departments at Ohio State, CL (and many other) courses are open to both undergraduates and graduate students. These courses fall far enough down the prerequisite chain that the undergraduates who

Proceedings of the Third Workshop on Issues in Teaching Computational Linguistics (TeachCL-08), pages 36–44,
Columbus, Ohio, USA, June 2008.

enroll are usually very motivated (and consequently do well), but one must keep in mind the differences in abilities and time constraints of each type of student. If the graduate students outnumber the undergraduates, introducing mentoring opportunities can provide a rewarding experience for all concerned.

From a practical perspective, this diversity presents a significant challenge – especially in universities where enrollment concerns drive curricular matters to some degree. Inclusiveness is also a reasonable goal from a financial, not just a pedagogical, perspective. CSE enrollments have declined significantly since the dot-com bust (Vegso, 2008), and while the declines are not as sharp as they once were, the current environment makes it more difficult to justify teaching narrow, advanced courses to only a few students (even if this were the practice in the past).

In this paper, I describe a number of strategies that have been successful in bringing all of these diverse populations into two different classes offered at OSU: a graduate seminar and a undergrad/graduate lecture class. The topic of both classes was statistical language processing, with a significant emphasis on ASR. Sample activities are discussed from each class.

While there are significant differences in the way that each class runs, there are several common elements that I try to provide in all of my classes.:

I first establish the golden rule: primary among my self-imposed rules is to make clear to all participants that all points of view are to be respected (although not necessarily agreed with), and that students are coming to this class with different strengths. If possible, an activity that integrates both linguistic and computer science knowledge should be brought in within the first week of the class; in teaching CSE courses, I tend to emphasize the linguistics a bit more in the first week.

I try to help students to engage with each other: a good way to foster inter- and intra-disciplinary respect is to have the students work collaboratively towards some goal. This can be challenging in a diverse student population setting; monitoring progress of students and gently suggesting turn-taking/mentoring strategies as well as designing activities that speak to multiple backgrounds can help ease the disparity between student backgrounds. Preparing the students to engage with each other on the same level by introducing online preclass activities can also help bring students together.

I try to allow students to build on previous knowledge via processes other than lecturing: a lecture, presented by either a student or a professor, is a "one-size-fits-all" solution that in a diverse population can sometimes either confuse unprepared students, bore prepared students, or both. Interactive in-class and out-of-class activities have the advantage of real-time evaluation of the understanding of students. This is not to say that I never lecture; but as a goal, lecturing should be short in duration and focused on coordinating understanding among the students. Over the years, I am gradually reducing the amount of lecturing I do, replacing it with other activities.

By putting some simple techniques into place, both students and I have noticed a significant improvement in the quality of classes. In Section 2, I describe improvements to a graduate seminar that facilitated interaction among a diverse group of participants. The most recent offering of the 10-week seminar class had 22 participants: 14 from CSE, 7 from Linguistics, and one from another department. In my informal evaluation of background, 13 of the 22 participants were relatively new to the field of computational linguistics ($<$ 2 years experience). Student-directed searching for background materials, pre-posing of questions via a class website, and blind reviewing of extended project abstracts by fellow students were effective strategies for providing common ground.

Section 3 describes improvements in a lecture-style class (Foundations of Spoken Language Processing) which has a similarly diverse participant base: the most recently completed offering had 7 CSE and 3 Linguistics Students, with the undergrad/graduate student ratio 3:7. Devoting one of the two weekly sessions to in-class group practical exercises also bolstered performance of all students.

2 Seminar structure

In developing a graduate seminar on machine learning for language processing, I was faced with a seri-

ous challenge: the previous seminar offering (on sequential machine learning) two years prior was not as inspiring as one would hope, with several students not actively participating in the class. This happened in part because students were permitted to suggest papers to read that week, which usually came from their own research area and often had esoteric terminology and mathematics. There was nothing wrong with the papers *per se*, but many of the students were not able to bridge the gap from their own experience to get into the depths of the current paper. While I thought having students partially control the seminar agenda might provide ownership of the material, in practice it gave a few students control of the session each time. In the more recent offering, this problem was likely to be exacerbated: the increased diversity of backgrounds of the students in the class suggested that it would be difficult to find common ground for discussing advanced topics in machine learning.

In previous seminars, students had given computer-projected presentations of papers, which led to rather perfunctory, non-engaged discussions. In the offering two years prior, I had banned computerized presentations, but was faced with the fact that many students still came unprepared for discussions, so the sessions were somewhat hit-and-miss.

In sum, a reorganization of the class seemed desirable that would encourage more student participation, provide students the opportunity to improve their background understanding, and still cover advanced topics.

2.1 A revised seminar structure

The previous instantiation of the seminar met twice weekly for 1 1/2 hours; in the most recent offering the seminar was moved to a single 2 1/2 hour block on Fridays. Each week was assigned a pair of student facilitators who were to lead the discussion for the week. The instructor chose roughly four papers on the topic of the week: one or two were more basic, overview papers (e.g., the Rabiner HMM tutorial (Rabiner, 1989) or Lafferty *et al.*'s Conditional Random Fields paper (Lafferty et al., 2001)), and the remaining were more advanced papers. Students then had varying assigned responsibilities relating to these papers and the topic throughout the week. Out-of-class assignments were completed using discussion boards as part of Ohio State's online course management system.

The first assignment (due Tuesday evening) was to find relevant review articles or resources (such as class or tutorial slides) on the internet relating to the topic of the week. Each student was to write and post a short, one-paragraph summary of the tutorial and its strengths and weaknesses. Asking the students to find their own "catch-up" resources provided a wealth of information for the class to look at, as well as boosting the confidence of many students by letting them find the information that best suited them. I usually picked one (or possibly two) of the tutorials for the class to examine as a whole that would provide additional grounding for class discussions.

The second assignment (due Thursday evening at 8 pm) was for each student to post a series of questions on the readings of the week. At a minimum, each student was required to ask one question per week, but all of the students far exceeded this. Comments such as "I totally don't understand this section" were welcome (and encouraged) by the instructor. Often (but not exclusively) these questions would arise from students whose background knowledge was sparser. In the forum, there was a general air of collegiality in getting everyone up to speed: students often read each others' questions and commented on them inline. Figure 1 shows a sample conversation from the course; many of the small clarifications that students needed were handled in this manner, whereas the bigger discussion topics were typically dealt with in class. Students often pointed out the types of background information that, if discussed in class, could help them better understand the papers.

The facilitators of the week then worked Thursday evening to collate the questions, find the ones that were most common across the attendees or that would lead to good discussion points, and develop an order for the discussion on Friday. Facilitators started each Friday session with a summary of the main points of the papers (10-15 minutes maximum) and then started the discussion by putting the questions up to the group. It was important that the facilitators did not need to know the answers to the questions, but rather how to pose the questions so that a group discussion ensued. Facilitators almost always

Student 1: After reading all of these papers on [topic], astoundingly, a few of the concepts have started to sink in. The formulas are mostly gibberish, but at least they're familiar. Anyhow, I have only mostly dumb questions....

- [Paper 1]:
  - Anyone want to talk about Kullback-Leibler divergence?
  - We've see this before, but I forget. What is an l_2 norm?
  - What's the meaning of an equal symbol with a delta over it?
  - When it talked about the "SI mode", does that mean "speaker independent"?
- [Paper 2]:
  - In multiple places, we see the where we have a vector and a matrix, and they compute the product of the transpose of the vector with the matrix with the vector. Why are they doing that?
- [Paper 3]:
  - I came away from this paper feeling like they gave a vague description of what they did, followed by results. I mean, nice explanation of [topic] in general, but their whole innovation, as far as I can tell, fits into section [section number]. I feel like I'm missing something huge here.
- [Paper 4]:
  - So, they want to maximize $2/|w|$, so they decide instead to minimize $|w|^2/2$. Why? I mean, I get that it's a reciprocal, so you change from max to min, and that squaring it still makes it a minimization problem. But why square it? Is this another instance of making the calculus easier?
  - What are the 'sx' and 'si' training sentences?

Student 2: *But why square it? Is this another instance of making the calculus easier?* I think so. I think it has to do with the fact that we will take its derivative, hence the ^{2} and 1/2 cancel each other. And since they're just getting an argmax, the ^{2} exponent doesn't matter, since the maximum x^2 can be found by finding the maximum x.
Student 3: 'sx' are the phonetically-compact sentences in the TIMIT database and 'si' are the phonetically-diverse sentences.
Student 4: Ah thanks for that; I've wondered the same thing when seeing the phrase "TIMIT si/sx"
Student 5: Oh, so 'si' and 'sx' do not represent the phones they are trying to learn and discern?

Figure 1: Conversation during question posting period in online discussion forum. Participants and papers have been anonymized to protect the students.

had something to contribute to the conversation; releasing them from absolutely needing to be sure of the answer made them (and other participants) able to go out on a limb more in the discussion.

I found that since the instructor usually has more background knowledge with respect to many of the questions asked, it was critical for me to have a sense of timing for when the discussion was faltering or getting off track and needed for me to jump in. I spent roughly a half hour total of each session (in 5-10 minute increments) up at the blackboard quickly sketching some material (such as algorithms unknown to about half of the class) to make a connection. However, it was also important for me to realize when to let the control of the class revert to the facilitators.

The blackboard was a communal workspace: in some of the later classes students also started to get up and use the board to make points, or make points on top of other students drawings. In the future, I will encourage students to use this space from the first session. I suspect that the lack of electronic presentation media contributed to this dynamism.

2.2 Class projects

The seminar required individual or team projects that were developed through the term; presentations took place in the last session and in finals week. Three weeks prior to the end of term, each team submitted a two-page extended abstract describing their

What single aspect of the course did you find most helpful? Why?
Discussions.
Very good papers used.
Style of teaching atmosphere.
Just the discussion style.
Informal, discussion based.
The project.
[Instructor] really explained the intuitions behind the dense math. The pictorial method to explain algorithms.
The breadth of NLP problems addressed.
Instructor's encouragement to get students involved in the classroom discussion.
Interaction between students, sharing questions.
Reading many papers on these topics was good training on how to pull out the important parts of the papers.

What single change in the course would you most like to see? Why?
There are a lot of papers – keeps us busy and focused on the course, but it may be too much to comprehend in a single term.
More background info.
None.
I think it is highly improved from 2 years ago. Good job.
Less emphasis on ASR.
Have some basic optional exercises on some of the math techniques discussed.
Less reading, covered at greater depth.
Make the material slightly less broad in scope.
More quick overviews of the algorithms for those of us who haven't studied them before.

Figure 2: Comments from student evaluation forms for the seminar class

work, as if for a conference submission.

Each abstract was reviewed by three members of the class using a standard conference reviewing form; part of the challenge of the abstract submission is that it needed to be broad enough to be reviewed by non-experts in their area, but also needed to be detailed enough to show that it was a reasonable project. The reviews were collated and provided back to authors; along with the final project writeup the team was required to submit a letter explaining how they handled the criticisms of the reviewers. This proved to be an excellent exercise in perspective-taking (both in reviewing and writing the abstract) and provided experience in tasks that are critical to academic success.

I believe that injecting the tutorial-finding and question-posting activities also positively affected the presentations; many of the students used terminology that was developed/discussed during the course of the term. The project presentations were generally stronger than presentations for other classes that I have run in the past.

2.3 Feedback on new course structure

The student evaluations of the course were quite positive (in terms of numeric scores), but perhaps more telling were the free-form comments on the course itself. Figure 2 shows some of the comments, which basically show that students enjoyed the dynamic, interactive atmosphere; the primary negative comment was about how much material was presented in the course.

After this initial experiment, some of my colleagues adopted the technique of preparing the students for class via electronic discussion boards for their own seminars. This has been used already for two CL seminars (one at Ohio State and another at University of Tübengen), and plans for a third seminar at OSU (in a non-CL setting) are underway. The professors leading those courses have also reported positive experiences in increased interaction in the class.

All in all, while the course was clearly not perfect, it seems that many of the simple strategies that were put into place helped bridge the gap between the backgrounds of students; almost all of the students found the class a rewarding experience. It is not clear how this technique will scale to large classes: there were roughly 20 participants in the seminar (including auditors who came occasionally); doubling the number of postings would probably make facilitation much more difficult, so modifications might be necessary to accommodate larger classes.

3 Group work within a lecture class

I have seen similar issues in diversity of preparation in an undergraduate/graduate lecture class entitled "Foundations of Spoken Language Processing." This class draws students from CSE, ECE, and Linguistics departments, and from both undergraduate and graduate populations.

3.1 Course structure

In early offerings of this class, I had primarily presented the material in lecture format; however, when I taught it most recently, I divided the material into weekly topics. I presented lectures on Tuesday only, whereas on most Thursdays students completed group lab assignments; the remaining Thursdays were for group discussions. For the practical labs, students would bring their laptops to class, connect wirelessly to the departmental servers, and work together to solve some introductory problems.

The course utilizes several technologies for building system components: MATLAB for signal processing, the AT&T Finite State Toolkit (Mohri et al., 2001) for building ASR models and doing text analysis[1], and the SRI Language Modeling Toolkit (Stolcke, 2002) for training n-gram language models. One of the key ideas behind the class is that students learn to build an end-to-end ASR system from the component parts, which helps them identify major research areas (acoustic features, acoustic models, search, pronunciation models, and language models). We also re-use the same FST tools to build the first pieces of a speech synthesis module. Componentized technologies allow the students to take the first step beyond using a black-box system and prepare them to understand the individual components more deeply. The FST formalism helps the Linguistics students, who often come to the class with knowledge of formal language theory.

The group activities that get students of varying backgrounds to interact constitute the heart of the course, and provide a basis for the homework assignments. Figure 3 outlines the practical lab sessions and group discussions that were part of the most recent offering of the course.

Weeks 1 and 3 offer complementary activities that tend to bring the class together early on in the term. In the first week, students are given some speech examples from the TIMIT database; the first example they see is phonetically labeled. Using the Wavesurfer program (Sjölander and Beskow, 2000), students look for characteristics in spectrograms that are indicative of particular phonemes. Students are then presented with a second, unlabeled utterance that they need to phonetically label according to a pronunciation chart. The linguists, who generally have been exposed to this concept previously, tend to lead groups; most students are surprised at how difficult the task is, and this task provokes good discussion about the difference between canonical phonemes versus realized phones.

In the third week, students are asked to recreate the spectrograms by implementing the mel filterbank equations in MATLAB. Engineering students who have seen MATLAB before tend to take the lead in this session, but there has been enough rapport among the students at this point, and there is enough intuition behind the math in the tutorial instructions, that nobody in the previous session had trouble grasping what was going on with the math: almost all of the students completed the follow-on homework, which was to fully compute Mel Frequency Cepstral Coefficients (MFCCs) based on the spectrogram code they developed in class. Because both linguists and engineers have opportunities to take the lead in these activities they help to build groups that trust and rely on each other.

The second week's activity is a tutorial that I had developed for the Johns Hopkins University Summer School on Human Language Technology (supported by NSF and NAACL) based around the Finite State Toolkit; the tutorial acquaints students with

[1] Subsequent offerings of the course will likely use the OpenFST toolkit (Riley et al., 2008).

Week	Lecture topic	Group activity
1	Speech production & perception	Group discussion about spectrograms and phonemes; groups use Wavesurfer (Sjölander and Beskow, 2000) to transcribe speech data.
2	Finite state representations	Use FST tools for a basic language generation task where parts of speech are substituted with words; use FST tools to break a simple letter-substitution cipher probabilistically.
3	Frequency analysis & acoustic features	Use MATLAB to implement Mel filterbanks and draw spectrograms (referring back to Week 1); use spectral representations to develop a "Radio Rex" simulation.
4	Dynamic Time Warping, Acoustic Modeling	Quiz; Group discussion: having read various ASR toolkit manuals, if you were a technical manager who needed to direct someone to implement a system, which would you choose? What features does each toolkit provide?
5	HMMs, EM, and Search	The class acts out the token passing algorithm (Young et al., 1989), with each group acting as a single HMM for a digit word (one, two, three...), and post-it notes being exchanged as tokens.
6	Language models	Build language models using the SRILM toolkit (Stolcke, 2002), and compute the perplexity of Wall Street Journal text.
7	Text Analysis & Speech Synthesis	Use FST tools to turn digit strings like "345" into the corresponding word string ("three hundred and forty five"). This tutorial grants more independence than previous ones; students are expected to figure out that "0" can be problematic, for example.
8	Speech Synthesis Speaker Recognition	Group discussion on a speaker recognition and verification tutorial paper (Campbell, 1997)
9	Spoken Dialogue Systems	Quiz; General discussion of any topic in the class.
10	Project presentations over the course of both sessions	

Week	Homework Task
2	Rank poker hands and develop end-to-end ASR system, both using finite state toolkit.
3	Finish Radio Rex implementation, compute MFCCs.
4	Replace week 2 homework's acoustic model with different classifier/probabilistic model.
5	Implement Viterbi algorithm for isolated words.
6	Lattice rescoring with language models trained by the student.
7	Text normalization of times, dates, money, addresses, phone numbers, course numbers.

Figure 3: Syllabus for Foundations of Spoken Language Processing class with group activities and homeworks.

various finite state operations; the two tasks are a simplified language generation task (convert the sequence "DET N V DET N" into a sentence like "the man bit the dog") and a cryptogram solver (solve a simple substitution cipher by comparing frequencies of crypttext letters versus frequencies of plaintext letters). The students get experience, in particular, with transducer composition (which is novel for almost all of the students); these techniques are used in the first homework, which involves building a transducer-based pronunciation model for digits (converting "w ah n" into "ONE") and implementing a FST composition chain for an ASR system, akin to that of (Mohri et al., 2002). A subsequent homework reuses this chain, but asks students to implement a new acoustic model and replace the acoustic model outputs that are given in the first homework. Similarly, practical tutorials on language models (Week 6) and text analysis (Week 7) feed into homework assignments on rescoring lattices with language models and turning different kinds of numeric strings (addresses, time, course numbers) into word strings.

Using group activities raises the question of how to evaluate individual understanding. Homework assignments in this class are designed to extend the work done in-class, but must be done individually. Because many people will be starting from a

group code base, assignments will often look similar. Since the potential for plagiarism is a concern, it is important that the assignments extend the group activities enough that one can distinguish between group and individual effort.

Another group activity that supports a homework assignment is the Token Passing tutorial (Week 5). The Token Passing algorithm (Young et al., 1989) describes how to extend the Viterbi algorithm to continuous speech: each word in the vocabulary is represented by a single HMM, and as the Viterbi algorithm reaches the end of an HMM at a particular timeframe, a token is "emitted" from the HMM recording the ending time, word identity, acoustic score, and pointer to the previous word-token. The students are divided into small groups and each group is assigned a digit word (one, two, ...) with a particular pronunciation. The HMM topology assumes only one, self-looping state per phone for simplicity. The instructor then displays on the projector a likelihood for every phone for the first time frame. The groups work to assign the forward probabilities for the first frame. Once every group is synchronized, the second frame of data likelihoods is displayed, and students then again calculate forward probabilities, and so forth. After the second frame, some groups ("two") start to emit tokens, which are posted on the board; groups then have to also consider starting a new word at the third time step. The activity continues for roughly ten frames, at which point the global best path is found. Including this activity has had a beneficial effect on homework performance: a significantly higher proportion of students across all backgrounds correctly completed an assignment to build an isolated word decoder in this offering of the class compared to the previous offering.

Some of the activities were more conceptual in nature, involving reading papers or manuals and discussing the high-level concepts in small groups (Weeks 4 and 8), with each group reporting back to the class. One of the skills I hope to foster in students is the ability to pick out the main points of papers during the reports back to the main group; I am still thinking about ways to tie these activities into strengthening the project presentations (Week 10).

For the next offering of the class in the upcoming quarter, I would like to reuse the ideas developed in the seminar to reduce the amount of lecturing. The strategy I am considering is to give the students the old lecture slides as well as readings, and have them post questions the evening before class; we can then focus discussion on the points they did not understand. This will likely require the instructor to seed the online pre-discussion with some of the important points from the slides. These changes can be discussed at the workshop.

3.2 Feedback

Student evaluations of the course were very positive; in response to "what single aspect of the course did you find most helpful?," half of the students chose to respond, and all of the responses focused on the utility of the hands-on practicals or homeworks. Anecdotally, I also felt that students were better able to retain the concepts presented in the course in the most recent offering than in previous offerings.

4 Summary

In trying to serve multiple populations of students with different aims and goals, I have found that activities can be designed that foster students' development through team problem-solving and small group work. Online resources such as discussion boards and tutorials using software toolkits can be effectively deployed to minimize the discrepancy in preparations of the students.

Moving away from lecture formats (either in lecture class or seminar presentations) has been helpful in fostering cross-disciplinary interaction for both seminar and lecture classes. I have found that active learning techniques, such as the ones described here, provide more immediate feedback to the instructor as to what material is understood and what material needs extra emphasis.

Acknowledgments

The author would like to thank the anonymous students who agreed to have their conversations published and whose comments appear throughout the paper, as well as Mike White for providing input on the use of the seminar strategies in other contexts. This work was supported in part by NSF CAREER grant IIS-0643901. The opinions and findings expressed here are of the author and not of any funding agency.

References

J.P. Campbell. 1997. Speaker recognition: A tutorial. *Proceedings of IEEE*, 85:1437–1462.

J. Lafferty, A. McCallum, and F. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 18th International Conference on Machine Learning*.

M. Mohri, F. Pereira, and M. Riley, 2001. *AT&T FSM Library*$^{\text{TM}}$ *– General-Purpose Finite-State Machine Software Tools*. AT&T, Florham Park, New Jersey. Available at http://research.att.com/fsmtools/fsm.

M. Mohri, F. Pereira, and M. Riley. 2002. Weighted finite-state transducers in speech recognition. *Computer Speech and Language*, 16(1):69–88.

L. Rabiner. 1989. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2).

M. Riley, J. Schalkwyk, W. Skut, C. Allauzen, and M. Mohri. 2008. OpenFst library. www.openfst.org.

K. Sjölander and J. Beskow. 2000. Wavesurfer – an open source speech tool. In *Proceedings of ICSLP*, Beijing.

A. Stolcke. 2002. SRILM — an extensible language modeling toolkit. In *Proc. Int'l Conf. on Spoken Language Processing (ICSLP 2002)*, Denver, Colorado.

J. Vegso. 2008. Enrollments and degree production at us cs departments drop further in 2006/2007. http://www.cra.org/wp/index.php?p=139.

S. Young, N. Russell, and J. Thornton. 1989. Token passing: a simple conceptual model for connected speech recognition systems. Technical Report TR-38, Cambridge University Engineering Department, Cambridge, England.