Concrete Assignments for Teaching NLP in an M.S. Program

Reva Freedman
Department of Computer Science
Northern Illinois University
DeKalb, IL 60115
freedman@cs.niu.edu

Abstract

The professionally oriented computer science M.S. students at Northern Illinois University are intelligent, interested in new ideas, and have good programming skills and a good math background. However, they have no linguistics background, find traditional academic prose difficult and uninteresting, and have had no exposure to research. Given this population, the assignments I have found most successful in teaching Introduction to NLP involve concrete projects where students could see for themselves the phenomena discussed in class. This paper describes three of my most successful assignments: duplicating Kernighan et al.'s Bayesian approach to spelling correction, a study of Greenberg's universals in the student's native language, and a dialogue generation project. For each assignment I discuss what the students learned and why the assignment was successful.

1 Introduction

Northern Illinois University is a large public university (25,000 students) located in the farm-oriented exurbs of Chicago, about 60 miles west of the city. Most of the undergraduate computer science majors and about a third of the M.S. students come from the hi-tech corridor west of Chicago or small towns near the university. The remaining M.S. students are international students, currently mostly from India.

This paper discusses my experiences in two semesters of teaching Introduction to NLP and three semesters of teaching an NLP unit in an Introduction to Artificial Intelligence course. Because the students have no background in linguistics and are not used to reading the type of academic prose found in the textbook (Jurafsky and Martin, 2000), the most successful units I have taught involved concrete assignments where students could see for themselves the phenomena discussed in class. Successful assignments also did not assume any background in linguistics, even such basic notions as part of speech.

To provide an overview of the field, each year the NLP course contains three segments: one on a statistical approach to NLP, one on syntax, and one on a logic-based approach. The three segments are also chosen to include topics from phonology and morphology, syntax, and pragmatics. The specific content changes from year to year in an effort to find topics that both represent current issues in the field and capture the students' imagination.

This paper describes three of the most successful assignments. For each one, I describe the assignment, the topics the students learned, and why it was successful. The three assignments are: duplicating Kernighan et al.'s Bayesian approach

Proceedings of the Second ACL Workshop on Effective Tools and Methodologies for Teaching NLP and CL, pages 37–42,
Ann Arbor, June 2005.

to spelling correction, a study of Greenberg's universals in a language other than English (usually the student's native language), and a dialogue generation project using my research software.

2 Background

2.1 Student demographics

Most of the students taking Introduction to NLP are graduate students, although undergraduates are eligible if they have had three semesters of C++ programming. Graduate students in cognitive science-related fields, such as psychology or linguistics, are eligible if they have taken one semester of programming and are willing to teach themselves about trees. I actively recruit non-computer science students because it makes the course more interesting. In addition to providing a broader spectrum of interests, they tend to be more outgoing. They tend to be more willing to answer questions in class, and also to ask questions in class, which many of the computer science students will not do.

The preferred career path among the students is to obtain a programming job in local industry, preferably in a hi-tech area. However, among both undergraduates and graduate students, a few continue their education. One minority student with no previous experience in research became interested and is now planning to apply to a PhD program. In general, students take the course out of a desire to do something different from the normal operating systems, networking and database courses. An occasional student also takes the course because it fits in their schedule or because it doesn't have onerous prerequisites.

In general, the international students have good to near-native competence in spoken English, although a few cannot not follow my lectures, and some do not have sufficient writing skills for an essay exam. All could read my lecture notes without difficulty. Both among the international students and local students, many do not have sufficient experience with formal academic prose to understand the textbook (Jurafsky and Martin, 2000). Students' first languages have included Telugu, Hindi/Urdu, Nepali, Chinese (Mandarin), and Bulgarian.

2.2 Student background

Koedinger (2001), in his research on tutoring systems for high school mathematics, gives the following as his fundamental principle: "the student is not like me." In particular, student background frequently did not include the following items:

1) Parts of speech
2) Basic English grammar
3) Relationships between languages and language families
4) Practical issues, such as the importance of transliteration and glossing
5) Philosophical issues, such as the fact that there is no single authoritative grammar of a natural language or that one language is not more difficult than another in an absolute sense

However, students were talented at and enjoyed programming. Most students also had a good math background. Finally, they were enthusiastic about learning new things, as long as it involved concrete examples that they could work out and a sample problem with a solution that they could use as a model.

3 Spelling correction

3.1 Background

The goal of the first section of the course was to show the students the power of statistical methods in NLP. In this section, students were asked to duplicate the calculations used in Kernighan et al.'s (1990) Bayesian approach to spelling correction, as explained in Section 5.5 of the textbook.

Kernighan et al. choose as the preferred correction the one that maximizes $P(t|c)P(c)$, where t is the typo and c is a candidate correction. Candidate corrections are generated by assuming that errors involve only one letter or the transposition of two adjacent letters. To reproduce this calculation, students need the confusion matrices provided in the original paper, a source of unigram and bigram data, and a source for word frequencies.

3.2 Assignment

Students are given some misspelled words and possible corrections, such as the following examples from Kernighan et al:

misspelled word	possible corrections
ambitios	ambitious
	ambitions
	ambition

For each of these misspelled words, students are asked to do the following:

a) Use the method described by Kernighan et al., or equivalently in section 5.5 of the text, to find the probability of each possible correction.

b) Use their preferred spell checker (Microsoft Word, Unix ispell, etc.) to generate possible corrections for the same misspelled words.

The following questions are asked for each misspelled word:

• Is the most probable correction according to Kernighan's algorithm the same as the one suggested by your program?

• Which additional possible corrections (i.e., non-single-error corrections or non-single word corrections) does your program generate?

• Which of Kernighan's possible corrections does your program omit?

Since Kernighan's original paper omits the unigram and bigram count matrices, I provide a file with this information. Students are encouraged to find a source for word frequencies on the Web. As one option, I suggest they use any search engine (e.g., Google), after class discussion about the approximations involved in this approach.

Students are also given two summary questions to answer:

• A former student, Mr. I. M. Kluless, says: I don't see the point of using the frequency of potential corrections in the corpus (i.e., the prior probability) as part of Kernighan's algorithm. I would just use the likelihood of a given error. How would you answer Mr. Kluless? (One way to think about this question is: what would happen if you left it out?)

• Another former student, Ms. U. R. Useless, says: I don't see the point of using the likelihood of a given error as part of Kernighan's algorithm. I would just use the prior probability. How would you answer Ms. Useless?

3.3 Results

Students enjoyed this assignment because it was straightforward and used mathematics they were familiar with. They were uniformly surprised to discover that spelling correction is generally done today using Bayesian concepts rather than by dictionary lookup alone. They were also surprised to learn that learn that results were largely independent of the corpus chosen. Students who already knew Bayes' theorem learned about an application completely different from the ones they had used in other courses.

The majority of students used my suggestion to approximate word frequencies in a corpus by page counts in Google. They were surprised to learn that in spite of the number of ways in which the web differs from an ideal corpus, the volume of data available ensures that accurate results are still obtained. The better students searched the web for corpora they preferred, including the works of Shakespeare and an online interface to the British National Corpus (http://sara.natcorp.ox.ac.uk/lookup.html).

4 Syntax and language universals

4.1 Background

The second section of the course had as its goal to teach the students some basic aspects of syntax. I started with parts of speech and basic concepts of context-free grammars. I then introduced unification grammars as a way of obtaining more power with fewer rules.

As a way of showing the syntactic variation among languages, I also introduced some of Greenberg's word order universals (Greenberg, 1966), following the exposition in Baker (2001). Although identifying the most probable underlying word order (SVO, etc.) of an unknown language can involve significant linguistic intuition, I did not expect students to achieve that goal. Rather, I used Greenberg's ideas to make students think

about the rules they were generating instead of generating S --> NP VP by rote. Additionally, the use of multiple languages contributed to the university's goal of introducing ideas of internationalization and diversity in classes where feasible.

4.2 Assignment

The students were asked to prepare a 15-minute class presentation showing two or three interesting phenomena of one of the languages of the world. Most students used their native language.

They were asked to include the following information:

- Where the language fits in Greenberg's classification (SVO, etc.)
- One or more syntactic phenomena that make the language interesting
- A grammar fragment (a set of CFG rules, possibly with unification-based features) illustrating one of the chosen phenomena

They could show several interesting phenomena with a short implementation of one, a complex phenomenon and a longer fragment of grammar, or one interesting phenomenon and multiple ways to implement it.

For each example they used, they were required to show the original transliterated into the Roman alphabet, a morpheme-level analysis, and a translation into English.

As a template, I gave a presentation using a language none of them had been exposed to, modern Hebrew. The four sample phenomena I presented were: a) there is no indefinite article, b) nouns and adjectives must agree in gender and number, c) adjectives follow the noun, and d) the definite article is attached to every adjective in an NP as well as to the noun.

In addition to providing an example of the scope required, the presentation also introduced the students to conventions of linguistic presentation, including interlinear display of transliteration, morpheme analysis, and translation. One slide from my presentation is shown below:

```
he- khatul    ha- gadol
DET cat-M-S   DET big-M-S
"the big cat"

he- khatulim   ha- g'dolim
DET cat-M-PL   DET big-M-PL
"the big cats"
```

4.3 Results

This assignment was useful for ensuring that students had a basic grasp of many elements of syntax covered in Section II of the textbook, including parts of speech, context-free grammars, and unification grammars. Second, the class presentations provided students concrete examples of some major syntactic concepts that all languages share, as well as some of the differences. Finally, this assignment enabled students to learn about and present some of the core linguistic features of their native language.

5 Dialogue generation

5.1 Background

The third segment of the course had as its goal to show how a logic-based approach is useful in NLP. Since some of my previous work involves implementing dialogue software using a logic-based approach, dialogue systems was a natural choice for this segment.

Phenomena discussed in lecture included the concepts of speech act and discourse intention, the relationship between syntactic form and intention, direct and indirect speech acts, and a short introduction to dialogue act classification.

As a counterbalance to the more theoretical material from Greenberg, this section included some information about current commercial uses of NLP. Students were asked to read an article from the popular press (Mount, 2005) describing experiences with currently available commercial systems.

I used my own software, APE (Freedman, 2000), a domain-independent dialogue plan interpreter based on reactive planning concepts. APE uses a rule-based macro language implemented in Common Lisp. It is a hierarchical task network (HTN) style planner, achieving each goal via a series of subgoals. APE's high-level planning loop alternates between waiting for user input and planning responses. It executes plan operators until a primitive, non-decomposable one

is obtained. In addition to *elicit* and *inform*, plans can also include primitives to query and update APE's internal knowledge base, thus giving the system a "mind." Primitives are added to a buffer until a primitive requiring a response from the user is received. At that point the operators in the buffer are used to build the output text. Goals are represented using first-order logic without quantifiers, with full unification used for matching.

APE provides two ways to change a plan in progress. The author can instruct the system either to switch from one method of satisfying a goal to another or to add new goals at the top of the agenda, possibly replacing existing goals. The latter facility is particularly useful in dialogue generation, since it allows the system to prompt the user after an error. This feature makes APE more powerful than the pushdown automaton one might use to implement a context-free grammar. In addition, APE is obviously more powerful than the finite-state machines often used in dialogue generation.

Use of APE allows students to generate realistic hierarchically structured conversations with a reasonable number of rules.

5.2 Assignment

Sample code presented in class involved looking up data in a database of presidents' names. The sample system prompted the user for input, then provided answers, error messages, and re-prompts as appropriate. As an illustration of the power of the approach, I also demonstrated some of my research software, which showed conversations embedded in a variety of front-end GUIs.

For the assignment, students were asked to choose their own topic. They were asked to choose a problem, then provide a database layout and draw a graph showing the possible conversations their system could generate. Finally, they were asked to implement the code. At the end of the semester, students made a five-minute presentation to the class showing their application.

5.3 Results

Students greatly enjoyed this assignment because it involved the activity they enjoyed most, namely programming. Even though it was qualitatively different from other algorithms they had learned, they had no trouble learning the unification algorithm, both iterative and recursive versions, because they were experienced in learning algorithms. For most students in our program, this project will be their only experience with a non-imperative programming language.

Students were not bothered by the fact that the sample software provided included some features not discussed in class. In fact, some of the better students studied these features and used them in their own programs.

Every student mastered the basics of logic programming, including how to choose between alternatives, establish a default, implement multi-step and hierarchical procedures, interact with the user, and access an external database. They also learned how to use unification along with multiple first-order relations to access and update a database. The weaker students simply used the sample software as a guide, while the stronger ones mastered the underlying concepts and wrote more creative code.

Student projects ranged the gamut, including a system for running a car dealership, a game identifying movie directors, and an interactive system providing health information.

6 Conclusions

Teaching NLP to students for whom this will be the only exposure to the topic, and possibly the only exposure to a research-oriented topic, can be a successful and enjoyable experience for both students and teacher. With good organization, students can do useful projects even in one semester.

One factor that has increased student satisfaction as well as their mastery of the material is the use of concrete assignments where students can see for themselves concepts described in class. Three such assignments I have successfully used involve duplicating Kernighan et al.'s Bayesian approach to spelling correction, a study of Greenberg's universals in the student's native language, and a dialogue generation project using my research software. Each of these assignments is used in one of the three segments of the course: statistical approaches to language, introduction to syntax, and logic-based approaches to NLP.

Acknowledgments

Michael Glass of Valparaiso University graciously supplied the unigram and bigram counts needed to implement Kernighan et al.'s (1990) spelling correction algorithm.

References

Baker, M. (2001). *Atoms of Language: The Mind's Hidden Rules of Grammar*. New York: Basic Books.

Freedman, R. (2000). Using a Reactive Planner as the Basis for a Dialogue Agent. In Proceedings of the Thirteenth Florida Artificial Intelligence Research Symposium (FLAIRS 2000), Orlando.

Greenberg, J. (1966). Some universals of grammar with particular reference to the order of meaningful elements. In *Universals of language*, ed. J. Greenberg, pp. 73–113. Cambridge, MA: MIT Press. 2nd ed.

Kernighan, M., Church, K., and Gale, W. (1990). A spelling correction program based on a noisy channel model. In COLING '90 (Helsinki), v. 2, pp. 205–211. Available online from the ACL archive at http://acl.ldc.upenn.edu/C/C90/C90-2036.pdf.

Koedinger, K. (2001). The Student is Not Like Me. In Tenth International Conference on Artificial Intelligence in Education (AI-ED 2001). San Antonio, TX. Keynote address. Slides available online at http://www.itsconference.org/content/seminars.htm.

Mount, I. (2005). Cranky Consumer: Testing Online Service Reps. *Wall Street Journal*, Feb. 1, 2005.