CS B553 Assignment 2: Part-of-speech tagging

Spring 2013
Due: Wednesday Feb 13, 11:59PM. (You may submit up to 48 hours late for a 10% penalty.)

In this assignment you'll build a part-of-speech tagger using inference on Bayesian networks models. You'll get experience working with Bayesian networks, implementing the variable elimination algorithm, and estimating Bayes nets parameters from labeled training data.

Please read this document carefully, as it is filled with details that are meant to help you, and you might waste time and effort on issues that are already addressed here. Please start this assignment early and feel free to ask questions on the OnCourse Forum. You may work on this assignment alone or in a partnership with another student in the class. If you work with a partner, please submit only one copy of your source code. To help get you started on the assignment, we've created some data files available via OnCourse.

Background

Natural language processing (NLP) is an important research area in artificial intelligence, with a history dating back to at least the 1950's. The goal of NLP is to algorithmically understand and generate human language. Early work investigated rule-based systems that used linguistic knowledge of human languages to solve NLP, but over the last 10-20 years the most successful systems have used statistical and probabilistic approaches. These approaches do not use detailed linguistic models, but instead learn parameters of simple statistical models by analyzing large corpora of training data.

One of the most basic problems in NLP is *part-of-speech tagging*, in which the goal is to mark every word in a sentence with its part of speech (noun, verb, adjective, etc.). This is a first step towards extracting semantics from natural language text. For example, consider the following sentence:

Her position covers a number of daily tasks common to any social director.

Part-of-speech tagging here is not easy because many of these words can take on different parts of speech depending on context. For example, *position* can be a noun (as in the above sentence) or a verb (as in "They position themselves near the exit"). In fact, *covers*, *number*, and *tasks* can all be used as either nouns or verbs, while *social* and *common* can be nouns or adjectives, and *daily* can be an adjective, noun, or adverb. The correct labeling for the above sentence is:

Her	position	covers	a	number	of	daily	tasks	common	to	any	social	director.
DET	NOUN	VERB	DET	NOUN	ADP	ADJ	NOUN	ADJ	ADP	DET	ADJ	NOUN

where DET stands for a determiner, ADP is an adposition, ADJ is an adjective, and ADV is an adverb.[1] Labeling parts of speech thus involves an understanding of the intended meaning of the words in the sentence, as well as the relationships between the words.

Fortunately, it turns out that a relatively simple Bayesian network can model the part-of-speech problem surprisingly well. In particular, consider the Bayesian network shown in Figure 1. This Bayes net has a set of N random variables $S = \{S_1, ..., S_N\}$ and N random variables $W = \{W_1, ..., W_N\}$. The W variables represent observed words in a sentence; i.e. $\text{Val}(W_i)$ is the set of all words in the English language. The variables in S represent part of speech tags, so $\text{Val}(S_i) = \{\text{VERB}, \text{NOUN}, ...\}$. The arrows between W and S nodes models the probabilistic relationship between a given observed word and the possible parts of speech it can take on, $P(W_i|S_i)$. (For example, these distributions can model the fact that the word "dog" is a fairly common noun but a very rare verb, so $P(W_i = \text{dog}|S_i = \text{NOUN}) > P(W_i = \text{dog}|S_i = \text{VERB}) > 0$). The arrows between S nodes model the probability that a word of one part of speech follows a word of another

[1]If you didn't know the term "adposition", the adpositions in English are prepositions! In many languages, there are postpositions too. But you won't need to understand the lingustic theory between these parts of speech to complete the assignment; if you're curious, you might check out the "Part of Speech" Wikipedia article for some background.

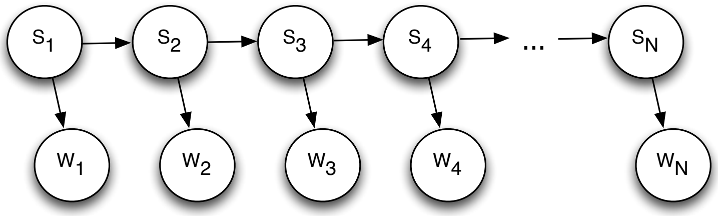

Figure 1: Bayes net for part-of-speech tagging.

part of speech, $P(S_{i+1}|S_i)$. (For example, these arrows can model the fact that verbs are very likely to follow nouns, but are unlikely to follow adjectives.)

We can use this model to perform part-of-speech tagging as follows. Given a sentence with N words, we construct a Bayes Net with $2N$ nodes as above. The values of the variables W are just the words in the sentence, and then we can compute marginal distributions over the variables in S to estimate the part-of-speech of each word. For example, to estimate the part-of-speech of the first word in the example above, we would first compute a marginal distribution over S_1,

$$P(S_1|W) = P(S_1|W_1 = \text{Her}, W_2 = \text{position}, ..., W_{13} = \text{director})$$

and then to declare a part of speech for the first word, we could choose the part of speech s_1^* that maximizes this marginal probability,

$$s_1^* = \arg\max_{s_1} P(S_1 = s_1|W).$$

Writing your own part-of-speech tagger

Your goal in this assignment is to implement a part-of-speech tagger, using Bayesian networks.

Language. You can use any general-purpose programming language of your choice. You can also use libraries for common data structures and algorithms (e.g. trees, vectors, sorting algorithms, etc.), as long as you implement the Bayes net learning and inference (described below) yourself. (So, for example, using the C++ Standard Template Library for convenient data structures is fine, but using a Bayes net library or a Part-of-speech tagging library is not.)

Data. To help you get started, we've prepared a large corpus of labeled training and testing data, consisting of over 1 million words and nearly 50,000 sentences. This dataset is available through OnCourse. The file format of the datasets is quite simple: each line consists of a word, followed by a space, followed by one of 12 part-of-speech tags: ADJ (adjective), ADV (adverb), ADP (adposition), CONJ (conjunction), DET (determiner), NOUN, NUM (number), PRON (pronoun), PRT (particle), VERB, X (foreign word), and . (punctuation mark). Sentence boundaries are indicated by blank lines.[2]

Step 1: Learning. In the first step, you'll need to estimate the conditional probability tables encoded in the Bayesian network above, namely $P(S_1)$, $P(S_{i+1}|S_i)$, and $P(W_i|S_i)$. To do this, use the labeled *training* corpus file we've provided.

Step 2: Naïve inference. Your goal now is to label new sentences with parts of speech, using the probability distributions learned in step 1. To get started, consider the simplified Bayes net in Figure 2. To

[2]This dataset is based on the Brown corpus. Modern part-of-speech taggers often use a much larger set of tags – often over 100 tags, depending on the language of interest – that carry finer-grained information like the tense and mood of verbs, whether nouns are singular or plural, etc. In this assignment we've simplified the set of tags to the 12 described here; the simple tag set is due to Petrov, Das and McDonald, and is discussed in detail in their 2012 LREC paper if you're interested.

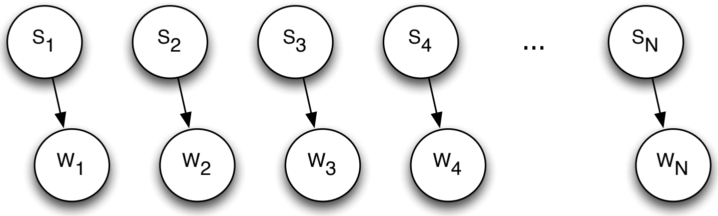

Figure 2: Simplified Bayes net for Step 2.

perform part-of-speech tagging, we'll want to estimate the most-probable tag s_i^* for each word W_i,

$$s_i^* = \arg\max_{s_i} P(S_i = s_i|W).$$

Computing the marginal distribution $P(S_i|W)$ on the more sophisticated Bayes net of Figure 2 is particularly simple. (To see this, try running the Variable Elimination algorithm by hand on a sample problem and see what happens!) Implement part-of-speech tagging using this simple model.

Step 3: Max marginal inference. Now, implement part-of-speech tagging using the Bayes net of Figure 1. You'll again estimate the most-probable tag s_i^* for each word W_i,

$$s_i^* = \arg\max_{s_i} P(S_i = s_i^*|W),$$

but using the Bayes net of Figure 1 to compute $P(S_i|W)$. Use the Variable Elimination algorithm to do this. Note that you don't need to implement the completely general Variable Elimination algorithm because we know the structure of the Bayes net ahead of time – it's a simple chain. Work a few simple examples out on paper to determine the best variable elimination ordering and figure out how to implement variable elimination on a chain.

Hint: You might start by figuring out how to compute the marginal distribution for the last word, $P(S_N|W)$. Then figure out how to compute $P(S_{N-1}|W)$, $P(S_{N-2}|W)$, etc.

Step 4: Sampling. The technique of step 3 gives a *single* best labeling for each word of a sentence. This answer may often be correct, but since our Bayes net model is a simple statistical model and not a linguistic model of English, the best labeling according to our model may not always be the right answer. An alternative approach is to produce multiple high-likelihood labelings, which could then be presented to a user or another algorithm for verification. A straightforward way of doing this is to *sample* from the distribution $P(S|W)$ instead of finding the single max-marginal labeling.

One can show that under the Bayes net of Figure 1, sampling from $P(S|W)$ can be achieved by:

1. Calculating the marginal distribution of the first word, $P(S_1|W)$.
2. Sampling from $P(S_1|W)$ to yield a label s_1.
3. Now sample from $P(S_2|W_2, S_1 = s_1)$ to yield a label s_2.
4. Continue step 3 until we produce all labels up to s_N.

Implement this technique and output 5 candidate labelings by sampling from the distribution 5 times.

Step 5: Evaluation. We've provided a separate test corpus file in addition to the training corpus file. Run the three versions of your part-of-speech tagger on the test corpus file, and give report quantitative accuracy statistics in your report. To do this, use the training corpus to learn the conditional probabilities (Step 1). Apply each of the three part-of-speech taggers to the test corpus, hiding the correct part-of-speech labels

```
[djcran@raichu djc-sol]$ ./label bc.train bc.tiny.test
--------------
Considering sentence: Her position covers a number of daily tasks common to any social director .
Ground truth: DET  NOUN VERB DET  NOUN ADP  ADJ  NOUN ADJ  ADP  DET  ADJ  NOUN .
Naive:        DET  NOUN VERB DET  NOUN ADP  ADV  NOUN ADJ  PRT  DET  ADJ  NOUN .
Bayes:        DET  NOUN VERB DET  NOUN ADP  ADJ  NOUN ADJ  ADP  DET  ADJ  NOUN .
Sample 1:     DET  NOUN NOUN DET  NOUN ADP  ADJ  NOUN ADJ  ADP  DET  ADJ  NOUN .
Sample 2:     DET  NOUN VERB DET  NOUN ADP  ADJ  NOUN ADJ  ADP  DET  ADJ  NOUN .
Sample 3:     DET  NOUN NOUN DET  NOUN ADP  ADV  NOUN ADJ  ADP  DET  ADJ  NOUN .
Sample 4:     DET  NOUN VERB DET  NOUN ADP  ADJ  NOUN ADJ  PRT  DET  ADJ  NOUN .
Sample 5:     DET  NOUN VERB DET  NOUN ADP  ADJ  NOUN NOUN ADP  DET  ADJ  NOUN .
--------------
Considering sentence: She started to brush the dirt and bits of leaves off her clothes .
Ground truth: PRON VERB PRT  VERB DET  NOUN CONJ NOUN ADP  NOUN ADP  DET  NOUN .
Naive:        PRON VERB PRT  NOUN DET  NOUN CONJ NOUN ADP  VERB PRT  DET  NOUN .
Bayes:        PRON VERB PRT  VERB DET  NOUN CONJ NOUN ADP  NOUN ADP  DET  NOUN .
Sample 1:     PRON VERB PRT  VERB DET  NOUN CONJ NOUN ADP  NOUN ADP  DET  NOUN .
Sample 2:     PRON VERB ADP  NOUN DET  NOUN CONJ NOUN ADP  NOUN ADP  DET  NOUN .
Sample 3:     PRON VERB ADP  NOUN DET  NOUN CONJ NOUN ADP  VERB PRT  DET  NOUN .
Sample 4:     PRON VERB ADP  NOUN DET  NOUN CONJ NOUN ADP  NOUN ADP  DET  NOUN .
Sample 5:     PRON VERB ADP  NOUN DET  NOUN CONJ NOUN ADP  NOUN ADP  DET  NOUN .
--------------
PERFORMANCE SUMMARY
Total words: 28  sentences: 2
Part 2, Naive graphical model:
      Words correct: 82.1429%
  Sentences correct: 0.0%
Part 3, Bayes net:
      Words correct: 100.0%
  Sentences correct: 100.0%
Part 4, Sampling:
(Here, sentences correct is fraction for which at least one sample is completely correct)
      Words correct: 89.2857%
  Sentences correct: 100.0%
```

Figure 3: Sample output.

from these taggers. Then compare the estimated tags produced by the algorithms to the correct ("ground truth") labels provided in the test file, and report the accuracy in terms of percentage of correctly-labeled words and percentage of correctly-labeled sentences.

Example. Figure 3 shows an example of an output from the program you might write, using the bc.train corpus for training, and the `bc.tiny.test` corpus for testing. We've included several test files (`bc.tiny.test`, `bc.small.test`, and `bc.test`); please report results in your report on the large `bc.test` file.

Important hints and warnings

Design decisions. You'll likely encounter some decisions as you write your programs that are not specified in this assignment. Feel free to make some reasonable assumptions in these cases, documenting them in your report, or to ask on the OnCourse forum for clarification.

Academic integrity. The goal of this assignment is to help you learn about Bayesian networks and part-of-speech taggers. **We thus require that you (and your partner) design and implement the part-of-speech tagger yourselves.** You and your partner may discuss the assignment with other people at a high level, e.g. discussing general strategies to solve the problem, talking about language syntax and features, etc. You may also consult printed and/or online references, including books, tutorials, etc., but you must cite these materials in the documentation of your source code and make it explicitly clear exactly which parts of your code were influenced by which sources. **However, the code that you (and your partner, if working in a group) submit must be your own work, which you personally designed**

and wrote. You may not share written code with any other students except your own partner, nor may you possess code written by another person who is not your partner, either in whole or in part, regardless of format. Please note that copying or modifying code from a student, from online sources, or from a textbook is a violation of the academic integrity policy. As described in the course syllabus and the IU Academic Handbook, University policy requires us to investigate suspected cases of academic integrity, to assign grade penalties, and to report the case to the Office of the Dean of Students for possible additional sanctions (up to and including expulsion from the University).

That said, the course staff wants you to succeed and is very happy to help if you have questions about the assignment. Please stop by office hours, write a question on the OnCourse forum, or set up an appointment to meet with us if you have concerns or questions.

What to turn in

Turn in two files, via OnCourse:

1. A .zip archive file containing your source code. Make sure your code is thoroughly debugged and tested. Make sure your code is thoroughly legible, understandable, and commented. Please use meaningful variable and function names. Cryptic or uncommented code is not acceptable; we can't grade your assignment unless we can understand how it works!

2. A separate text file or PDF with a brief report about your work. Explain, at a high level, how you implemented it, what design decisions and other assumptions you made, any problems you had to overcome, etc. Give credit to any source of assistance (students with whom you discussed your assignments, instructors, books, online sources, etc.). Also give the accuracy details requested above (on the results of your testing on our sample datasets).

Extra credit. We may give extra credit to assignments that implement more sophisticated features than the ones described here. Make sure to describe these extra features in your documentation file.