Massachvsetts Institvte of Technology
Department of Electrical Engineering and Computer Science
6.863J/9.611J Natural Language Processing
Laboratory 2: Competitive Grammar Writing
Handed out: March 19, 2018 **Due: April 2, April 9, April 11, 2018 (see section 1.1)**

Abstract

In this assignment, you will compete in teams of three to create the best context-free grammar for English sentences over a fixed vocabulary. We have specified 27 sentences that your grammar must be able to parse, but you are not limited to generating the grammatical structures found in those sentences. After your grammars are finalized, sentences generated from each team's grammar will be distributed back to you for judging. Team grammars will then be tested against all of the sentences judged grammatical by you and others.

There is no writeup for this assignment, nor do you need to write any code (though you may if you wish).

1 Introduction and timeline

During CGW, your team will design a context-free grammar for English sentences over a limited vocabulary. Your grammar is split into two subgrammars, S1 and S2. S1 is the handmade grammar you are expected to customize to generate and parse a wide variety of English sentences. S2 is a fallback grammar that accepts any string, but with very low probability for any particular string. You will adjust the relative weights of these two subgrammars to match your confidence in S1. (See section 2.)

There are two rounds to the competition, the beta and final rounds, and two benchmarks, the base set and the adversarial set.

The base set is a list of 27 sentences constructed by the course staff. By the beta submission deadline, your S1 subgrammar should be able to parse at least 14 (i.e., half) of the base set sentences. By the final deadline, your S1 should be able to parse all of the sentences in the base set. Both the beta and final base set evaluation are part of your grade, guaranteeing you some points regardless of your performance in the competition.

The adversarial set is a set of sentences generated from the teams' grammars and vetted by humans for grammaticality. There is no adversarial evaluation during the beta round. After the beta deadline, a sample of sentences will be generated from each team's S1 and shared with the entire class. You can use this sample (after making your own judgements about grammaticality) to improve your grammar. After the final deadline, another sample will be generated from each team's S1, and the resulting sentences will be distributed amongst the class to be judged for grammaticality. The grammatical sentences form the adversarial set, which will be used to evaluate S1 recall (the fraction of sentences parsed by your S1 subgrammar) and the cross-entropy against your full grammar (both S1 and S2, weighted as you chose).

The CGW files will be distributed on Athena. We will create directories in the 6.863 locker for each team that only that team (and the course staff) can access. You will be e-mailed when your team directory becomes available. At each deadline, the course staff will take the current grammar files from that directory as your team's submission. (Consider backing this directory up into your home directory and/or using version control.)

There is no writeup for this assignment.

1.1 Important dates

March 19: Handout released. Team directories accessible and pre-populated with starter grammar files.
April 2: Beta **due** at midnight. Generated sentences available shortly after the deadline.
April 9: Grammars **due** at midnight and evaluated on base set. Sample sentences assigned for judging.
April 11: Sentence judgments due. Final evaluation completed shortly thereafter.

2 Structure of the grammar

You have been given five grammar rule files which make up a single context-free grammar. The grammar consists of two subgrammars, S1 and S2, in files named `S1.gr` and `S2.gr`. Each subgrammar has an associated vocabulary grammar file, `S1_Vocab.gr` and `S2_Vocab.gr`, which map different sets of nonterminals to the same fixed vocabulary (terminals). Finally, `Top.gr` contains rules from the start symbol to S1 and S2, allowing you to express your confidence in S1 by adjusting the relative weights between the S1 and S2 subgrammars.

S1 S1 is the primary grammar. We expect you to spend most of your time working on S1, and we will assess S1's recall and grammaticality during the competition. We have provided you with a very simple S1 grammar to start with, using a very simple set of part-of-speech tags in `S1_Vocab.gr`, but you are free to modify the contents of both files. The only constraints are that the root symbol of S1 must be `S1` (our scripts rely on this) and that you may only use words from the allowed words list.

S2 S2 is the fallback subgrammar (sometimes called a "backoff grammar" in the literature). S2 exists to ensure that the combined grammar can parse any string over our vocabulary. You are allowed to modify S2 and its vocab grammar if you wish (apart from adding new terminals), but it is very important that it continues to parse all sentences, because part of the evaluation will be to compute the cross-entropy of the adversarial set against your grammar. If the adversarial set contains a sentence your grammar cannot parse, the cross-entropy will be infinite and you will receive no credit for that portion of the evaluation. S2 is an insurance policy against that outcome.

The S2 subgrammar we have provided you to start with is a simple bigram model over the part-of-speech tags defined in its vocabulary grammar (which happens to be the same as the S1 grammar). Thus it will produce a right-branching parse tree for any sentence, but the probability it assigns to any particular sentence will be very low because the probability mass is spread over all strings made from the (given and fixed) terminal vocabulary.

Part-of-speech tags `S1_Vocab.gr` and `S2_Vocab.gr` are lists of rules mapping preterminals to the vocabulary. You can consider the preterminals as part-of-speech tags for the words. The starter files use a very simple tag system in which most of the words are lumped into one tag, `Misc`. In the starter files, both grammars use the same part-of-speech tag set, but this is not a requirement (that's why there are two files).

You should develop a more precise tag set for use with your grammar. The precise set you choose depends on the kinds of phenomena you wish to capture. They need not be purely syntactic tags; for example, you might tag proper nouns more granularly as persons, locations, or specific objects. You may wish to research the tags used by corpora[1][2][3]. A description of the Penn Treebank part-of-speech tagset is provided on Stellar.

Note that a word can appear on the right-hand side of any number of rules, so a word can be tagged with more than one part of speech.

Top `Top.gr` is a very simple file containing two rules: one that expands `START` to `S1` and one that expands `START` to `S2`. The weights of these rules allow you to express your confidence in S1. If you are highly confident in your S1 grammar, you should weight the rule expanding to `S1` highly (as it is by default). If you are less confident, you can allocate more probability mass to S2.

If you weight S1 highly, your combined grammar will assign a very low probability to any sentences S1 cannot parse. On the other hand, if you weight S2 too highly, then your combined grammar will do a poor job of predicting the other teams' sentences, because S2 divides its probability mass over all possible strings (making any individual sentence very unlikely). If you do a good job on S1, you probably want to favor S1 over S2, but perhaps not as extremely as the starter file does.

[1] `https://web.archive.org/web/20170813005658/www.scs.leeds.ac.uk/ccalas/tagsets/brown.html`
[2] `https://web.archive.org/web/20130517134339/http://bulba.sdsu.edu/jeanette/thesis/PennTags.html`
[3] `https://web.archive.org/web/20130902000829/http://www.mozart-oz.org/mogul/doc/lager/brill-tagger/penn.html`

Putting the grammars together To form the full grammar, simply concatenate the individual grammar files:

```
cat Top.gr S1.gr S1_Vocab.gr S2.gr S2_Vocab.gr > GRAMMAR.gr
```

This is exactly what our evaluation scripts will do[4]. You can then use `GRAMMAR.gr` with `randsent` or `parse`, as normal.

To test S1 individually, you can concatenate just `S1.gr` and `S1_Vocab.gr`, then manually add a rule expanding `START` to `S1`. With `parse`, you can instead pass `-s S1` to specify an alternate start symbol. It should not be too difficult to modify your `randsent` from the warmup to use a start symbol specified on the command line. (To test S2 individually, just replace all instances of S1 in this paragraph with S2.)

Note that because the grammars files will be concatenated, you should be careful that the nonterminal symbols you use in S1 and S2 do not conflict. (The starter files get away with having `S1_Vocab.gr` and `S2_Vocab.gr` define the same symbols because the files are identical, effectively doubling the weights of all repeated rules, which does not affect the probabilities.)

3 Athena files

As in previous labs, you must run `add 6.863; setup 6.863` to access files and run programs from the course locker.

3.1 Team directories

Each team has a directory under `/mit/6.863/spring2018/cgw/teams/` which only they (and the course staff) can access. At the beta and final round deadlines, we will take the grammar files in the team directories as each team's submission.

3.2 Data files

`/mit/6.863/spring2018/cgw/data/` contains some data files that may be useful during the competition:

- `allowed_words` is an exhaustive list of the allowed terminal symbols. Your grammar must not generate sentences containing any words not in this list (remember that case is significant). Conversely, your S2 subgrammar should parse any sentence made out of this vocabulary.
- `base-set.sen` contains the base set sentences, one per line.
- `starter/` contains a copy of the starter files, for reference. For example, if you accidentally modify your S2 subgrammar such that it cannot parse all sentences over our vocabulary, you can copy `S2.gr` and `S2_Vocab.gr` into your team directory.
- After the beta round deadline, `beta-sample/` will contain anonymized samples of sentences generated from each team's grammar. These sentences will not be judged for grammaticality, but they may be a useful reference in developing your own grammar.

3.3 Tools

`/mit/6.863/spring2018/cgw/tools/` contains a few tool scripts. Each script begins with brief comments explaining its use. You can copy any of these scripts into your team directory if you wish to customize them.

[4]So don't store anything important in a file named `GRAMMAR.gr`, as it will be overwritten.

- `pos-tagger.py` is a part-of-speech tagger. Given a list of sentences (such as `base-set.sen`) and a grammar file (either `S1_Vocab.gr` or `S2_Vocab.gr`), the script will print the tag(s) of each symbol in the sentence and the possible sequences of tag nonterminals that can generate the sentence. You can use these sequences to identify patterns to model in your S1 subgrammar.
- `drawtrees.py` visualizes trees (parenthesized like `parse`'s output) from a sentence file provided on standard input. As this is a graphical application, you need X forwarding enabled (see the "Notes on Using Athena" handout on Stellar).
- `best-top-weights.py` finds the weights for your S1 and S2 subgrammars that minimize the cross-entropy on a test set. For example, using the base set and the starter grammar files, it prints `best S1 weight is 7, S2 is 93, obtained cross-entropy 8.79831`. As you extend your S1 subgrammar, the weights will begin to shift towards S2.
- `train-grammar.py` reads parenthesized trees from a file and prints the relative frequency of each rule that was invoked to build the tree. You can use this to help you tune the weights of your rules. If you pass the `-i` flag to specify a grammar file, the script will attempt to update the weights for you and write a new grammar file (though the script does not always succeed, so you should double-check the file manually).

Note that automatically optimized weights are only as good as the test set they came from. If you use only the base set to tune your weights, you will overfit and do poorly in the competition!

4 Things to think about

Here are some ideas for linguistic phenomena you may wish to model in your grammar:

- Subject-verb agreement: "we ride" not "we rides"
- Number agreement: "five coconuts" not "five coconut", or "a coconut" not "a coconuts"
- Verbs can take different numbers of arguments. Some verbs are intransitive, i.e., they take no arguments after the verb: *Arthur spoke the king .* is not well-formed, but *Arthur spoke .* is fine. Some verbs take just one argument, perhaps implicit: *Arthur rode the horse .* or *Arthur rode ..* Some verbs take two arguments, like *gave* (but there might not be any examples like this in your fixed vocabulary).
- Specificity of verbs: for instance, `suggest` usually takes an argument that stands for some proposition, e.g., *Arthur suggested that the king ride to Camelot .*, although it can also take just a Noun Phrase (NP): *Arthur suggested the king ..* Other verbs cannot take a propositional argument, e.g., *Arthur rode that the king .* is not well-formed.
- Some verbs can take question-phrases but not declarative propositions as their arguments: *Arthur carried what Guinevere rode .* but not: *Arthur carried that Guinevere rode.* (Compare: *Arthur knows what Guinevere rode*/*Arthur knows that Guinevere rode.*)
- Some conjunctions pair up "neither" with "nor" and "either" with "or", but not "neither" with "and".

5 Grading

5.1 Beta

- **Base set S1 recall (10%):** Your S1 grammar must parse 14 of the 27 sentences in the base set.

There is no adversarial evaluation during the beta round.

5.2 Final

- **Base set S1 recall (30%):** Your S1 subgrammar must parse all 27 sentences in the base set.[5]
- **S1 grammaticality (30%):** The fraction of the sentences generated from your S1 subgrammar that were judged grammatical by humans. (Those grammatical sentences are added to the adversarial set.)
- **Adversarial set S1 recall (15%):** The fraction of the sentences in the adversarial set that your S1 subgrammar can parse.
- **Adversarial set cross-entropy (15%):** The cross-entropy of the adversarial set against your grammar (both S1 and S2).

Note that recall and grammaticality are Boolean (a sentence is either parsed or not), but cross-entropy takes into account the weights in your grammar.

[5]To prevent grammars from hardcoding the base set sentences, we will also test against similar sentences with nouns, verbs and/or adjectives replaced by other words of those classes.