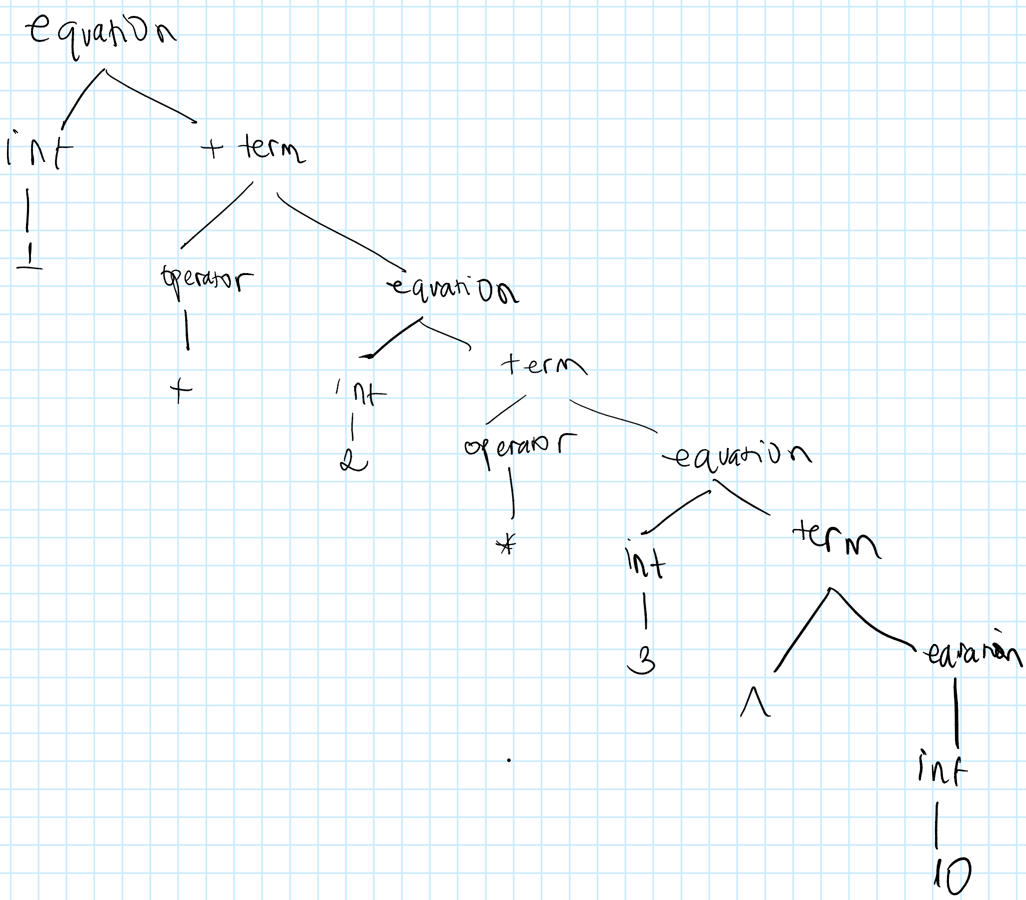
Exercise set 8-9

Tuesday, 28 November 2023 12.05

Exercise 8.2: grammar for equations.
While equations are not natural language, they are featured in many scientific publications and are a useful example for grammars. Propose a context-free grammar that can create equations of the kind used in simple calculators: nonnegative integer numbers, a limited set of variables (x, y, z), operators (+, -, *, /, ^), parentheses, a limited set of functions (log, exp, sin, cos, tan). Example equations that your grammar should be able to generate:

- 1+2*3^10
- 24-(3*x-(z^y))/(3-x)
- sin(2*x-cos(y))^(2-10)

Then, show how an example equation is derived from your grammar: show the derivation tree of applying rules to arrive at the final equation.
Note: in this exercise you do not need to assign probabilities to the rules of your grammar.
Report the definition of your grammar, and the derivation of your example equation.



Exercise 8.3: Chomsky normal form.
The probabilistic context free grammar below is a simplified version of the one used to generate "hmm_sentences.txt" in exercise 8.1. Transform the grammar into Chomsky normal form, so that the possible sentences and their probabilities are the same as in the grammar below. Report the definition of your resulting Chomsky normal form grammar (rules and their probabilities).

S --> STMANY	1.0
STMANY --> S1 .	0.6
STMANY --> S1 , but STMANY	0.4
S1 --> SUBJ QVERB1 QVERB2 OBJ	1.0
SUBJ --> ARTICLE DESC NOUN	1.0
DESC --> ADJECTIVE	0.7
DESC --> ADJECTIVE DESC	0.3
OBJ --> ARTICLE DESC NOUN	1.0
QVERB1 --> can	0.2
QVERB1 --> will	0.5
QVERB1 --> may	0.3
ARTICLE --> a	0.6
ARTICLE --> the	0.4

QVERB1 --> will	0.5
QVERB1 --> may	0.3
ARTICLE --> a	0.6
ARTICLE --> the	0.4
QVERB2 --> explain	0.4
QVERB2 --> help	0.2
QVERB2 --> answer	0.4
ADJECTIVE --> wise	0.3
ADJECTIVE --> friendly	0.5
ADJECTIVE --> insightful	0.2
NOUN --> cat	0.7
NOUN --> dog	0.2
NOUN --> fox	0.1

S --> STMANY 1.0
STMANY --> S1 STMANY' 1.0
STMANY' --> ε 0.6
STMANY' --> BUT STMANY 0.4
S1 --> SUBJ QVERB1 QVERB2 OBJ 1.0
SUBJ --> ARTICLE DESC NOUN 1.0
DESC --> ADJECTIVE DESC' 1.0
DESC' --> ε 0.7
DESC' --> DESC 0.3
OBJ --> ARTICLE DESC NOUN 1.0
QVERB1 --> can 0.2
QVERB1 --> will 0.5
QVERB1 --> may 0.3
ARTICLE --> a 0.6
ARTICLE --> the 0.4
QVERB2 --> explain 0.4
QVERB2 --> help 0.2
QVERB2 --> answer 0.4
ADJECTIVE --> wise 0.3
ADJECTIVE --> friendly 0.5
ADJECTIVE --> insightful 0.2
NOUN --> cat 0.7
NOUN --> dog 0.2
NOUN --> fox 0.1
BUT --> , but 1.0

ε represents the non-string value, since I have combined it in the previous one
Rules: combining the string with the same as the previous, from shorter to longer string

Exercise 9.2: Can you explain the Inside-outside algorithm better than an LLM?
Use a large language model (AI chatbot; see previous exercises for examples of what could be used) to generate an explanation of how the inside-outside algorithm works in PCFGs. Then modify the LLM's answer to add new facts or details the LLM did not state and to fix errors or other issues in the LLM's answer.

In your solution, report what LLM you used, report the original LLM answer, and report your modified answer with the added and modified parts highlighted.

I would choose CHATGPT for this question:
Question:
an explanation of how the inside-outside algorithm works in PCFGs
CHATGPT:
The inside-outside algorithm is used to learn the probabilities of production rules in a Probabilistic Context-Free Grammar (PCFG) from a set of trees, typically obtained from a corpus. PCFGs extend context-free grammars by associating probabilities with each production rule.

Here’s a simplified step-by-step explanation of the inside-outside algorithm:

1. Initial Probabilities:

Initialize the probabilities for each production rule in the PCFG.
These probabilities could be initialized uniformly or using some other method (e.g., randomly).
2. Inside Probabilities:
For each node in each parse tree, calculate the "inside" probabilities.
The "inside" probability of a node is the probability of generating the subtree rooted at that node given the production rules.
3. Outside Probabilities:
Calculate the "outside" probabilities for each node in each parse tree.
The "outside" probability of a node is the probability of generating the words before and after that node, given the rest of the parse tree.
4. Update Probabilities:
Use the inside and outside probabilities to update the probabilities associated with each production rule in the PCFG.
This step involves adjusting the probabilities based on how likely certain rules are given the observed trees.
5. Repeat:
Iterate through steps 2-4 until convergence (when the probabilities stabilize or a certain stopping criterion is met).
Key Formulas:
Inside Probabilities:
Inside probability of a node X:
For terminal nodes: 1 (if the word is generated), 0 (if the word is not generated)
For non-terminal nodes:
∑
Probability of all possible productions of X
∑Probability of all possible productions of X
Outside Probabilities:
Outside probability of a node X:
For root node: 1
For non-root non-terminal nodes:
Probability of parent node
×
Probability of generating sibling nodes
Probability of parent node×Probability of generating sibling nodes
Advantages and Applications:
The inside-outside algorithm is used in natural language processing tasks such as parsing and language modeling.
It helps in estimating the probabilities of different grammar rules, which is crucial for tasks like syntactic parsing and language generation.
Keep in mind that this is a simplified overview of the inside-outside algorithm in PCFGs. The actual implementation involves more intricacies and optimizations to efficiently learn the grammar probabilities from the given parse trees.

Modify answer

The inside-outside algorithm is a method used to learn the probabilities of production rules in a Probabilistic Context-Free Grammar (PCFG) from a set of trees, typically obtained from a corpus. PCFGs extend context-free grammars by associating probabilities with each production rule.
Here’s a more detailed step-by-step explanation of the inside-outside algorithm:

1. **Initial Probabilities**: Initialize the probabilities for each production rule in the PCFG. These probabilities could be initialized uniformly or using some other method (e.g., randomly).
   This step is crucial as it sets the initial state for the learning process.
2. **Inside Probabilities**: For each node in each parse tree, calculate the “inside” probabilities. The “inside” probability of a node is the probability of generating the subtree rooted at that node given the production rules.
   This step involves traversing the parse tree from the leaves (terminal nodes) to the root.
3. **Outside Probabilities**: Calculate the “outside” probabilities for each node in each parse tree. The “outside” probability of a node is the probability of generating the words before and after that node, given the rest of the parse tree.
   This step involves traversing the parse tree from the root to the leaves.
4. **Update Probabilities**: Use the inside and outside probabilities to update the probabilities associated with each production rule in the PCFG. This step involves adjusting the probabilities based on how likely certain rules are given the observed trees.
   This is the core of the learning process where the model adjusts itself to better fit the data.
5. **Repeat**: Iterate through steps 2-4 until convergence (when the probabilities stabilize or a certain

stopping criterion is met).
This iterative process allows the model to continuously learn and improve its predictions.