CS 784: Assignment 1 (100 Points)

Instructor: Freda Shi
Due Date: Wednesday, February 12, 2025, 11:59 pm (ET; Waterloo Time)

Instructions

- Submit the code solutions (§ 1) in a file named `code.zip` to the `Assignment 1 Code` Dropbox on `LEARN`. The file should contain a file named `process_childes.*` containing the main code and a file named `test_process_childes.*` containig test cases, where * is the file extension.

  You may use any programming language you like.

  - If you use Python, please include any non-default libraries you use in a standard `requirements.txt` file that is compatible with `pip install -r requirements.txt`, and make sure your code runs well with Python 3.12. We also offer a few standard unit tests in Python, which are can be executed with the command "`python -m unittest discover ./`" once you have the `unittest` library installed.
  - If you use another programming language, please include a `README.md` file with instructions on how to compile and run your code.

- Submit your paper review (§ 2) in a file named `review.pdf` to the `Assignment 1 Paper Review` Dropbox on `LEARN`.
- You have a 3-day (72 hours) grace period to submit the assignment with no penalty.

  No late assignment will be accepted after February 15, 2025, 11:59 pm (ET).
- This assignment is to be done individually.

1 Programming (50 points)

There has been a recent trend of research of training language models with cognitively plausible data [cf. 2, *inter alia*],[1] analyzing the language acquisition process of language models, and drawing hypotheses and inferences on the language acquisition process of humans and/or the connection between human and machine language acquisition. The CHILDES database [1], which contains transcripts of child language acquisition studies, offers a rich source of data for this line of research. However, CHILDES has a lot of additional information that is not natural language exposed to children, and it is necessary to preprocess the data and extract the main content before training language models on it.

In this part of the assignment, you will implement a data preprocessing pipeline for the CHILDES database from the TalkBank project, extracting the main transcript contents.[2] The original data are in the CHAT format, a plain text format that stores transcripts of child language acquisition studies with lots of additional annotations such as grammars and actions. Below is a quick snapshot of the CHAT format.

```
@UTF8
@PID:    11312/c-00015218-1
@Begin
@Languages:     eng
@Participants:  CHI Target_Child , MOT Mother
@ID:    eng|Bates|CHI|1;08.|female|TD|MC|Target_Child|||
@ID:    eng|Bates|MOT||female|||Mother|||
@Types: cross, toyplay, TD
*MOT:   what's that ?
%mor:   pro:int|what~cop|be&3S pro:dem|that ?
%gra:   1|2|SUBJ 2|0|ROOT 3|2|PRED 4|2|PUNCT
%act:   holds object out to Amy
*CHI:   yyy .
%gpx:   looks at chicken
%act:   holds nesting cups
%xpho:  wi
*MOT:   it's a chicken .
%mor:   pro:per|it~cop|be&3S det:art|a n|chicken .
%gra:   1|2|SUBJ 2|0|ROOT 3|4|DET 4|2|PRED 5|2|PUNCT
*CHI:   yeah .
```

[1] The cognitive plausibility of language models is usually referred to being trained on a corpus that is similar to the input that humans (or children, depending on the main objective) receive, in terms of both token numbers and lexical complexity.

[2] https://talkbank.org/

All lines starting with @ (header information) or % (additional annotations, e.g., %act = actions, %gpx = gestures, %gra = grammars) are to be completely removed. For lines starting with *, you will refer to the CHAT manual for the detailed description of the format at https://talkbank.org/manuals/CHAT.html. **Only Sections 8.4, 8.6, and 10.2 are directly relevant for this assignment**; however, you are encouraged to check out other sections as an excellent example of linguistic data annotation. Below is the desired output of the preprocessing pipeline for the above data in the CHAT format into a markup language format.

```
<MOT> what's that ?
<CHI> <unk> .
<MOT> it's a chicken .
<CHI> yeah .
```

Your task is to implement a data preprocessing pipeline that handles the cases mentioned in Sections 8.4, 8.6, and 10.2 of the CHAT manual. **You are highly encouraged to use regular expressions**; however, this is not a requirement. Below are the some instructions for each individual category, in addition to the manual.

- **8.4 - Unidentifiable Material**: Substitute all the unidentifiable material from the transcripts with the unknown-word token <unk>.

  Note: a token is only considered unidentifiable material if it appears as a single token wrapped with word boundaries—for example, yyyy is not unidentifiable material, but yyy is.

- **8.6 - Incomplete and Omitted Words**: Remove the special markers (i.e., parentheses) for incomplete words to retain its completed form, and **remove** all the omitted word markers. For example,
  - (Incomplete)

    `*MOT: I been sit(ting) all day .` → `*MOT: I been sitting all day .`
  - (Omitted)

    `*CHI: I want &=0to go.` → `*CHI: I want go.`

  There will not be any naturally existing parentheses in the text, as they are speech transcripts. You may safely assume that the parentheses are only used for marking incomplete words.

- **10.2 Paralinguistic and Duration Scoping**: Remove all annotations for paralinguistic events (indicated by square brackets with marker symbols `=!`, `!!`, `!`, `#`) that appear in the transcripts. For example,
  - (No angle brackets)

    `*CHI: that's mine [=! cries].` → `*CHI: that's mine .`

Note that this case is more complicated than simply removing the square brackets, their may be angle brackets before indicating the durations. If that happens, you should also handle the angle brackets. For example,

- (Angle brackets)

  `*CHI: <that's mine> [=!  cries] .` → `*CHI: that's mine .`

Additionally, some paralinguistic events may happen without any vocalization (denoted by `&=action`), and they should be removed as well. For example

- (No content)

  `*CHI: &=cries .` → `*CHI: .`

  This indeed looks strange, but we will detect empty lines and remove them in other components of the pipeline.

There are some additional cases in the CHAT manual that uses squared brackets (coupled with other marker symbols) for other purposes—you need to leave them untouched.

Before you start, please read through the CHAT manual for more example cases. Please do not change any content that is not covered in the above three cases.

You will implement the **processer** and **unit tests** for the above three cases. If you program with Python, you can check out all the `TODO` comments in the provided code; if you use another language, please include a `README.md` file with brief instructions on how to compile and run your code. Your grade will be determined by the correctness of your code and the quality of your unit tests. Specifically, your grade will have the following components:

- Processer implementation (25 points)
  - (3 points) 3 test cases for 8.4.
  - (3 points) 3 test cases for 8.6.
  - (10 points) 10 test cases for 10.2.
  - (9 points) 9 test cases to ensure the pipeline works correctly.
- Unit tests (25 points)
  - (3 points) Write 3 unit tests for 8.4, and ensure the pass of your code.
  - (3 points) Write 3 unit tests for 8.6, and ensure the pass of your code.
  - (10 points) Write 5 unit tests for 10.2, and ensure the pass of your code.
  - (4 points) Write 2 unit tests for the pipeline, and ensure the pass of your code.
  - (5 points) Brief explanation of the unit tests: why you choose these test cases, and how they possibly cover the edge cases. Please **include this as comments in your**

test code. For example, it might be a bad idea for 8.4 to have two test cases for `xxx xxx` and `xxx xxx xxx`, as they are likely to be both addressed with most reasonable implementations; however, it might be a good idea to have two test cases for `xxx yyy` and `yyy www`, as they have covered different perspectives.

- Extra credit (up to 10 points): You will receive 5 extra credits for every correct test case that fails Freda's implementation.

There are 3 existing test cases in the Python code skeleton for your reference, which will not be counted towards the required number of test cases. Feel free to include more test cases if you think they are necessary.

Hint: You are very much encouraged to check out the original CHILDES data to understand the data format and find possible corner cases for your implementation.

2 Paper Review (50 points)

Write a review for the paper

Word Acquisition in Neural Language Models

Tyler A. Chang, Benjamin K. Bergen

In TACL (2022)

`https://aclanthology.org/2022.tacl-1.1/`

Your review should include the following parts:

- Summary of the paper (10 points)
- Strengths of the paper (10 points)
- Weaknesses of the paper (10 points)
- Detailed comments, suggestions, and questions for the authors (20 points)

Each part will be graded separately by the following criteria:

$$\textit{Your Grade}(\%) = \min\left\{F_1(\textit{Your Review}, \textit{Freda's Review} \cup \textit{Filtered Review from Class})/0.6, 100\right\},$$

where F_1 is the F_1 score between your review and the union of Freda's review and a good peer review. *Filtered Review from Class* refer to the review arguments from the class that are endorsed by Freda. *Freda's Review* $\cup$ *Filtered Review from Class* will be a set of unique "ground-truth" arguments for the review of this paper. While all arguments will be weighted equally when calculating precision, the arguments will be weighted differently when calculating your

recall based on the importance and levels of detail.[3] Your review will also be considered as a set of arguments, so you are encouraged (though not required) to write in bullet points.

Everyone might misunderstand or simply miss some points, so you will receive full marks if your review gets higher than 60% F_1 score compared to the ground-truth arguments.

Below is an example of grading. Suppose you raised 7 points about the paper (1-7), the "ground-truth" set consists of 6 points (1, 2, 3, 7, 8, 9). In the "ground-truth" set, points 1, 2 are weighted 2 and points 3, 7, 8, 9 are weighted 1. Then your recall will be

$$\text{Recall (weighted)} = \frac{2+2+1}{2+2+1+1+1+1} = \frac{5}{8} = 0.625.$$

Your precision will be

$$\text{Precision (unweighted)} = \frac{4}{7} = 0.571.$$

Your F_1 score will be the harmonic mean of the recall and precision

$$F_1 = \frac{2 \times 0.625 \times 0.571}{0.625 + 0.571} = 0.597.$$

That is, you will receive $0.597/0.6 = 99.5\%$ of the marks for this part.

The ACL instruction (`https://2023.aclweb.org/blog/review-acl23/`) to reviewers might be helpful. You may also find a few good reviews Freda received in the past on `LEARN`, as well as an example review she wrote for a recent paper.

Please note that this is a journal paper published in 2022, so please avoid suggestions that involve work **in or after 2021**, such as comparing to GPT-4 performance.

References

[1] Brian MacWhinney. The CHILDES Project: Tools for analyzing talk. Third Edition. Mahwah, NJ: Lawrence Erlbaum Associates, 2000.

[2] Kanishka Misra and Najoung Kim. Generating novel experimental hypotheses from language models: A case study on cross-dative generalization. arXiv:2408.05086, 2024.

[3] There may be even some arguments weighted zero; that is, you will not be penalized for missing them through recall, but you will be rewarded for mentioning them through the precision score.