CS 489/698 Introduction to Natural Language Processing
Assignment 1: Linguistic Cryptography

Instructors: Freda Shi and Victor Zhong
Due Date: Monday, February 9th, 2026 at 11:59 PM ET (Waterloo Time)

Winter 2026

1 Overview

In this assignment, you will act as a "*linguistic cryptographer*." You are provided with encrypted text corpora from 6 known languages: Finnish, Hungarian, Thai, Turkish, Vietnamese, and Western Peripheral Nahuatl. All these languages have some known linguistic features documented in the WALS Database.[1] The provided texts are in (or are transliterated to) Latin script. The text in Latin scripts has been subjected to a **character-level perturbation**. You will have three main tasks:

- **Task 1 (36 points)**: Train a tokenizer with 1,000 tokens for each language. Your tokenizer will be evaluated based by its compress ratio on held-out data.
- **Task 2 (42 points)**: You will be given six mixtures of languages (after perturbation) with different ratios of each language. Your task is to (a) provide the estimated proportions of each language in the mixture, and (b) provide a 2,000-token tokenizer for the mixture. Your tokenizer will be evaluated by the compression ratio on held-out data.
- **Task 3 (22 points)**: Identify the source language of each encrypted corpus, and the encryption function used. Your answers will be evaluated based on the "correctness", approximated by the similarity of your proposed languages to the ground truth languages.

  For more details, see the tasks below in Section 3.

2 The Dataset

You are provided with six text files: `lang_01.txt` through `lang_06.txt`, each from one of the six languages mentioned above. Each line in the files should be considered separately—note that they do not necessarily correspond to sentences; instead, they should be considered as a chunk of text extracted from a larger document.

- **Content:** Text in six languages, sourced from a reliable linguistic corpus.
- **Preprocessing:**

  1. If the canonical writing system is not Latin (e.g., languages like Chinese, Hindi, and Japanese), the text will be transliterated to Latin script. If the original language does not use spaces to

[1] `https://wals.info/feature`

separate words (e.g., Chinese), spaces will be inserted between words following a widely accepted standard.

2. The texts have been lowercased.
3. Only lowercase letters `a-z` and spaces are retained. If there is diacritic marking in the original text, it will be removed—for example, `é` becomes `e` and `ç` becomes `c`. All punctuation, digits, and special characters will be removed.

- **Perturbation:** Each language has been encrypted using a **simple substitution cipher** at the character level; that is, each character in the original text has been replaced by another character (possibly itself) consistently throughout the text. Different languages may have different substitution mappings, but the same mapping is used for all text within a single language throughout this assignment. The mapping could be any bijection over the set of characters `a-z`, with spaces left unchanged.
- **Knowledge about the original languages:** The languages fall into three typological categories based on their morphological structure of inflection.

  1. **Prefixing:** Inflectional information appears at the *start* of words.
  2. **Suffixing:** Inflectional information appears at the *end* of words.
  3. **(Almost) no inflection:** Isolating languages with minimal morphological changes.

  You may refer to the WALS Database Feature 26A (`https://wals.info/feature/26A`) for details of each language.

3 Tasks

3.1 Task 1 (36 points + 8 bonus points): Train a Tokenizer

Train a tokenizer on each of the six languages. You may implement the algorithm from scratch or use a standard library, but your tokenizer should be able to be represented as a fixed vocabulary of V tokens.

- Set the target vocabulary size $V = 1,000$.
- Please include all characters `a-z` and space in your vocabulary, which will take up 27 slots from the total vocabulary size, to make sure that every piece of text can be tokenized.

  The compression ratio is defined as:

$$\text{Compression Ratio} = \frac{\text{Size of original text (in characters)}}{\text{Size of tokenized text (in tokens)}}$$

A higher compression ratio is considered better as it indicates that the tokenizer is more effective at representing the text with fewer tokens.

Expectation: We will set up a baseline method on the Kaggle leaderboard, which uses the simple BPE algorithm. If your compression ratio is not lower than the baseline, you will receive full marks (6 points) for that language. In addition, you will earn up to 8 bonus points in total for better compression ratios on held-out data in our competition. If your compression ratio is lower

than the baseline, your score will be scaled linearly based on the following formula:

$$\text{Score} = 6 \times \left(1 - \frac{\text{Baseline Compression Ratio} - \text{Your Compression Ratio}}{\text{Baseline Compression Ratio} - 1}\right)$$

In the tokenization process, each line is tokenized independently; that is, you do not need to consider cross-line tokenization. More detailed decoding instructions can be found on the Kaggle competition page.

Note: The held-out data may contain words that do not appear in the training data. This is normal in real-world applications, as a corpus with limited size can hardly cover all possible words in a language, but a good tokenizer should still be able to effectively tokenize unseen words by breaking them down into known subword units.

3.2 Task 2: (42 points + 8 bonus points): Mixture Analysis and Tokenizer Training

You are provided with six mixture files: `mix_01.txt` through `mix_06.txt`; each line is from an individual language. Each mixture file contains text from two or more of the six languages, combined in different proportions. Your tasks are:

1. **Estimate Language Distributions (24 points):** For each mixture file, estimate the proportion of each of the six languages present in the mixture. There will be no code-switching within a line; that is, each line belongs to exactly one language. You will estimate two distributions: one is based on line count, and the other is based on word (tokens separated by whitespace) count, each summing to 100%.

   As an example, for the following corpus:

   ```
   (Finnish) w1 w2 w3
   (Hungarian) w4 w5
   (Finnish) w6 w7 w8 w9
   ```

   By line count, the distribution is 66.7% (2 out of 3) Finnish and 33.3% (1 out of 3) Hungarian.

   By word count, the distribution is 77.8% (7 out of 9) Finnish and 22.2% (2 out of 9) Hungarian.

   Your estimates will be evaluated based on the Jensen-Shannon divergence (JSD) between your estimated distributions and the ground-truth distributions. A lower JSD indicates a better estimate. The JSD is defined as:

   $$JS(P||Q) = \frac{1}{2}KL(P||M) + \frac{1}{2}KL(Q||M)$$

   where $M = \frac{1}{2}(P + Q)$ and $KL(P||Q)$ is the Kullback-Leibler divergence from distribution P to distribution Q:

   $$KL(P||Q) = \sum_i P(i) \log_2 \frac{P(i)}{Q(i)}$$

   You might have recognized that JSD is bounded between 0 and 1, with 0 indicating identical distributions and 1 indicating completely disjoint distributions.

Each mixture file is worth 2 points for line count-based distribution and 2 points for word count-based distribution, totaling 4 points per mixture file and 24 points for all six mixture files.

Expectation: We will have a simple baseline method that only uses data offered in this assignment. If your method achieves a lower JSD than the baseline in one setting, you will receive full marks (2 points) for that setting; otherwise, the score will be scaled linearly based on the following formula:

$$\text{Score} = 2 \times \left(1 - \frac{\text{Your JSD} - \text{Baseline JSD}}{1 - \text{Baseline JSD}}\right)$$

2. **Train a Tokenizer for Each Mixture (18 points):** Train a tokenizer on each mixture file with a target vocabulary size of $V = 2,000$. Again, please include all characters `a-z` and space in your vocabulary, which will take up 27 slots from the total vocabulary size. Your tokenizer will be evaluated based on the compression ratio on held-out data with the same language distributions (in terms of both line count and word count) as the given mixture file.

   Expectation: A BPE tokenizer that only uses data offered in this assignment will be able to obtain full marks. You will earn up to 8 bonus points for better compression ratios on held-out data in our competition.

3.3 Task 3 (22 points): Language and Cipher Identification

For each of the six encrypted language files, identify:

- The source language (from the list of six languages provided): 1.5 points. If your guess of the source language is not exactly correct, you will receive 0.5 point if your guess shares the same WALS 26A feature with the ground-truth language. You will get the additional 2 points if you identify all six languages correctly.
- The character-level substitution cipher used for encryption: 1.5 points. You will receive partial credit based on the number of correct character mappings you identify. For example, if the ground-truth mapping is `a->b`, `b->c`, `c->d`, ..., `z->a` and your proposed mapping is `a->a`, `b->c`, `c->d`, ..., `z->b`, you will receive 24/26 points for correctly identifying 24 out of 26 character mappings (except the target of a and z). You will get the additional 2 points if you identify all six cipher mappings perfectly.

4 Submission Instructions

Please submit your code in a zip file named `submissions.zip` on LEARN. The submission zip should contain the following files/folders:

- `task1_submission.csv`
- `task2_1_submission.csv`
- `task2_2_submission.csv`
- `task3_submission.csv`

  All these `csv` files should be in exactly the same format as described on Kaggle.

- `code/`: a subfolder that contains your code to produce the submission files.
- `README.md`: a brief text description of what you've done. You are encouraged but not required to include execution instructions here for your code. It is okay if you use some heuristics or manual inspection to help with the tasks, e.g., by looking at the encrypted language and guess which language it's in by word length impression—if so, please document them here. If you use additional data or AI assistants, please also document here. We may contact you for further clarification if we fail to (roughly) reproduce your results based on your description and code.

Note: Please make sure that all csv files are named exactly as listed above and included in the root directory of the zip file, not in a subdirectory; otherwise you will receive 0 points for this assignment.

4.1 Kaggle Submissions

You are encouraged to make multiple submissions to the Kaggle platform for this assignment to verify your methods and estimate their performance. For detailed formats, please see the Kaggle competition page (linked on the course material page `https://waterloo-nlp.github.io/intro-to-nlp-material/`).
Upon your submission, please set your both Kaggle username and the team name to `Anonymous`.

- To set your Kaggle username, go to your Kaggle account settings page (`https://www.kaggle.com/settings`), click on "Your Profile" on the top right, click on "Edit your public profile", and then click on the pencil icon on the top right of your profile page to edit your username.
- To set your team name, go to the competition page, click on the "Team" tab on this competition page, and edit your team name.

5 Tips

- We recommend the Google Colab Notebook environment for this assignment, as it provides free access to powerful compute resources. You can also use your local machine or any other cloud service of your choice.
- You may use any resource that is available publicly on the web to help with this assignment.
- Results from Task 3 may be helpful for Tasks 1 and 2, and vice versa.
- Be creative and have fun!