NLP Assignment 1 – Text Processing Project Report
Student: Murad Valiyev

Date: 5 February 2026

1. Motivation

The goal of this project was to create a small Azerbaijani-language corpus from Wikipedia and apply standard text processing to it. Azerbaijani (Azeri) is a Turkic language with relatively limited publicly available NLP datasets and tools compared to high-resource languages like English. By building and processing our own corpus, we aimed to:

- Understand how tokenization, frequency statistics, subword methods, sentence segmentation, and spell-checking behave on real Azerbaijani encyclopedic text.
- Explore basic quantitative properties (e.g. Heaps' law) and practical challenges (e.g. diacritic-sensitive errors, common typos).

The corpus was collected from the Azerbaijani Wikipedia because it is freely accessible, high-quality, edited text in the target language. The setting is formal, edited, encyclopedic writing (monologue style, public knowledge-sharing). The project was individual, done for the course, with no external funding.

2. Datasheet / Data Statement

Motivation

To create an Azerbaijani corpus for experimenting with tokenization, frequency analysis, Heaps' law, BPE, sentence splitting, and spell-checking algorithms as part of an NLP course assignment.

Situation

Edited, formal encyclopedic articles written for public dissemination. Originally produced in Azerbaijani Wikipedia (2011–2025, most recent edits 2025).

Language variety

Standard literary Azerbaijani (Latin script, Azerbaijan variety). Minimal dialectal variation.

Speaker demographics

Authors are Wikipedia volunteers (mostly native Azerbaijani speakers from Azerbaijan and diaspora) and some bots like İşçiBot, Turkmenbot who also might be real people with a Botnusername. No age/gender statistics available.

Collection process

- Randomly sampled 3,000 articles via Wikipedia API

  `https://az.wikipedia.org/w/api.php`

  (action=query, list=random, rnnamespace=0).
- Extracted plain text + metadata (title, last edit timestamp, last editor).
- Final corpus (after cleaning): ≈5,423,130 characters, ≈617,554 tokens, ≈93,122 types.
- Pre-processing: removed section headings, wiki markup, templates, HTML tags, normalized multiple spaces/newlines.
- Data is public (CC BY-SA license), no consent required.
- Metadata preserved: title, last edit time, editor username.

Annotation process

No manual annotations. All processing (tokenization, frequency counts, etc.) is done with mainly regex expressions.

Distribution

Original Wikipedia content is CC BY-SA. The derived cleaned corpus is for educational/research use only.

3. Method

Data collection

Used Python's requests library to fetch random articles from az.wikipedia.org API. Saved title, plain text, last edit timestamp and editor username into a CSV file.

Tokenization (Task 1)

Custom rule-based regex tokenizer for Azerbaijani:

- Numbers with decimal/comma + suffix/currency/percentage (154.5$, 50,5%, 2023-cü)
- Time formats (20:00)
- Words and hyphenated compounds (ayrı-ayrı, sərhəd-təhlükəsizlik)
- Case folding with Azerbaijani rules (I → ı, İ → i, then .lower()) Punctuation kept separate when not attached.

Heaps' law (Task 2)

Computed vocabulary growth incrementally over tokens. Fitted V = k × N^β using nonlinear least squares (scipy.optimize.curve_fit).

Byte-Pair Encoding (Task 3)

Simple character-level BPE implementation: started from characters + </w> end-of-word marker, merged top-frequency pairs iteratively (simple version, 25 merges shown).

Sentence segmentation (Task 4)

Rule-based algorithm:

- Protected common abbreviations (Dr., Prof., Cən., və s., etc.) with placeholders
- Split on . ! ? followed by whitespace
- Recombined using heuristic (if next segment starts with lowercase → likely continuation)

Spell checking (Task 5 + Extra)

- Baseline: uniform-cost Levenshtein distance (ins/del/sub cost = 1)
- Weighted version: custom Levenshtein with confusion matrix (lower costs for Azerbaijani-specific errors: a/ə=0.4, i/ı=0.4, o/ö=0.5, u/ü=0.5, ç/c=0.6, etc.) Tested on typical typos: azarbaycan, qarabag, mesele, sehife, etc.

4. Experiments & Results

Task 1 – Tokenization & frequencies
Corpus size after cleaning: 5,423,130 characters, 617,554 tokens, 93,122 types.

Type-Token Ratio ≈ 0.151 (low, expected for large corpus with repetition).

Top 20 most frequent tokens (clean, meaningful):

və (19,651), ilə (5,408), bu (4,683), bir (4,447), olan (2,950), üçün (2,728), də (2,519), azərbaycan (2,479), sonra (2,468), ildə (2,410), kimi (2,300), isə (2,070), o (2,019), edir (1,803), tərəfindən (1,771), da (1,765), onun (1,738), görə (1,714), çox (1,513), idi (1,509)

(Note: "the" appears due to English special terms in mainly citations of Wikipedia articles.)

Task 2 – Heaps' law
Fitted parameters:
k ≈ 10.40

β ≈ 0.683

β = 0.683 is within the typical range for natural languages (0.4–0.8). It indicates reasonable vocabulary growth for a medium-sized, topically diverse corpus.

Task 3 – BPE
The first 1000 merges produced frequent pairs and syllables: n</w>, ə</w>, i</w>, r</w>, ər, və</w>, də</w>, ının</w>, indən</w>, etc.

To sum up, the implementation behaved correctly for the merges and corpus size.

Task 4 – Sentence segmentation
The abbreviation-protected splitter handled Wikipedia text reasonably well.
Correctly avoided splitting on Dr., Prof., Cən., və s., etc.

Some limitations exist: There may be some over-splitting on edge cases, though these are acceptable for a rule-based approach, there may be rare edge cases may not have been fully accounted for.

Task 5 & Extra — Spell Checking

A spell-checking system was evaluated using a test set of **common Azerbaijani typos**, including *azarbaycan*, *qarabag*, *mesele*, and *sehife*. Two approaches were compared: a baseline Levenshtein distance model with uniform edit costs and a weighted edit distance model using a corpus-learned confusion matrix.

Baseline Levenshtein Distance

The baseline Levenshtein model generally produced reasonable candidate corrections; however, due to its **uniform cost assignment**, it occasionally ranked non-phonetic or linguistically implausible matches higher. After preprocessing fixes, correct targets were still recovered, for example:

- *azarbaycan* → **azərbaycan** (distance = 1)
- *qarabag* → **qarabağ** (distance = 1)
- *sehife* → **səhifə** (distance = 1)

Despite these correct matches, the lack of language-specific weighting limited the model's ability to prioritize phonetic and diacritic variations consistently.

Weighted Edit Distance Model

The weighted model demonstrated clearer improvements for **Azerbaijani-specific spelling errors**, as it incorporates a confusion matrix learned from corpus statistics. Common diacritic and vowel substitutions were penalized less heavily, resulting in more appropriate rankings:

- *azarbaycan* → **azərbaycan** (weighted distance ≈ 0.40, ranked highest)
- *qarabag* → **qarabağ** (weighted distance = 0.50)

- *sehife* → **səhifə** (weighted distance = 0.80)

This approach more accurately reflects typical orthographic and phonetic confusions in Azerbaijani.

Error Analysis and Limitations

A direct comparison shows that the weighted model outperforms uniform Levenshtein distance on **diacritic and vowel confusions**, such as `ə/a`, `ı/i`, `ö/o`, and `ğ/g`. However, the system does not incorporate any **language-model prior or contextual information**, and candidate ranking relies solely on edit distance and word frequency.

5. Contributions

Individual project.

All tasks (data collection, cleaning, implementation, report) completed by Murad.

References

- Gebru et al. (2020). Datasheets for Datasets.
- Bender & Friedman (2018). Data Statements for Natural Language Processing.
- Wikipedia API (az.wikipedia.org/w/api.php)
- Levenshtein distance (1965), BPE (Sennrich et al., 2016)