Spring 2026 Natural Language Processing (CSCI-6515)

Azerbaijani Wikipedia Corpus: Tokenization, Heaps' Law, BPE, Sentence Segmentation, and Spell Checking

Kamal Ahmadov
kahmadov24700@ada.edu.az; kamal.ahmadov@gwu.edu
Rufat Guliyev
rguliyev24988@ada.edu.az; rufat.guliyev@gwu.edu

ADA University & George Washington University

February 5, 2026

Motivation

- Azerbaijani is a low-resource language for NLP research
- Wikipedia provides freely licensed, diverse text data (CC BY-SA)
- Goal: build a clean corpus and implement core preprocessing modules
- Outputs are fully reproducible via `bash scripts/run_all.sh`

Dataset Snapshot

- Source: Azerbaijani Wikipedia (MediaWiki API)
- Cleaned corpus stored as CSV: `data/raw/corpus.csv`
- Final size:
  - 623 documents
  - 238,286 tokens
  - 48,151 unique token types

Cleaning Challenges

- Wikipedia markup noise:
  - templates, categories, navigation text, references
- Mixed-language artifacts (English-heavy references)
- Broken punctuation / formatting residue
- Short or empty pages removed to improve quality

Tokenization (Unicode-aware)

We tokenize using Unicode letter categories and preserve Azerbaijani characters.

$$\text{token} \in \{\text{letters}\}^{+} \quad \text{with optional internal } [\prime -]$$

Example:

‘‘S.Rustamov 2.5 df artırsaq’’ $\Rightarrow$ [S.Rustamov, 2.5, df, artırsaq]

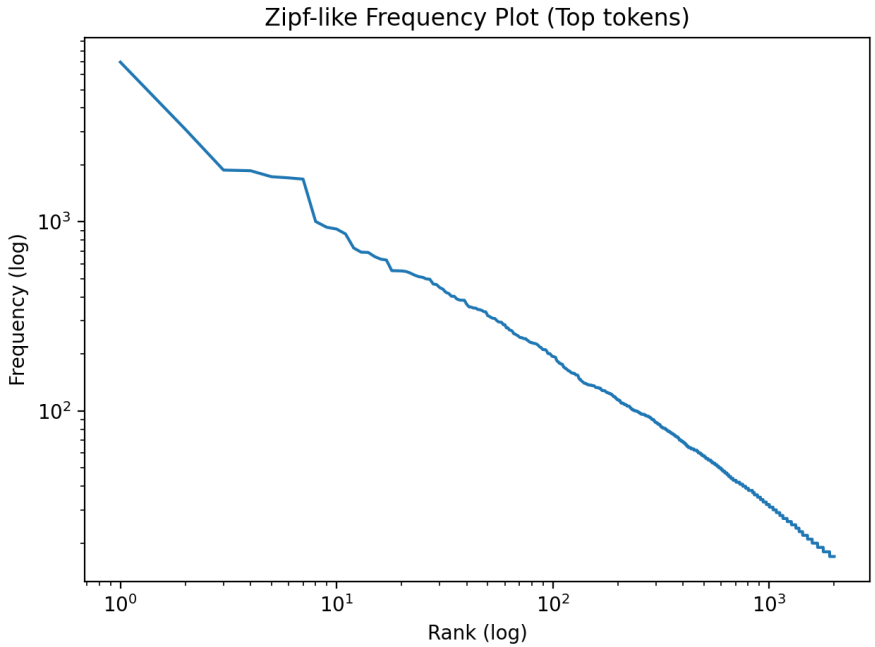
- Handles diacritics: ə, ğ, ö, ş, ü, ı, ç
- Keeps decimals to avoid splitting numeric expressions
- Lowercasing used for vocabulary normalization

Zipf's Law (Rank–Frequency)

- Empirical law for word frequencies
- Frequency decays with rank

$$f(r) \propto \frac{1}{r^s} \quad \Rightarrow \quad \log f(r) = c - s \log r$$

- Near-linear mid section on log–log plot
- Confirms heavy-tailed behavior in our corpus



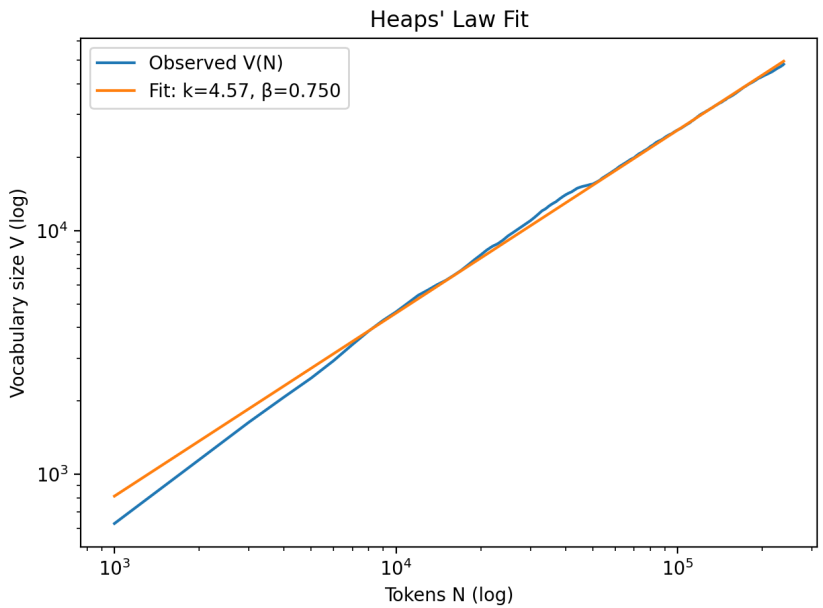
Heaps' Law (Vocabulary Growth)

$$V(N) = kN^{\beta} \quad \Rightarrow \quad \log V = \log k + \beta \log N$$

- Sample (N, V) every 1000 tokens
- Fit $\log V$ vs. $\log N$ by linear regression

$$k = 4.57, \qquad \beta = 0.750$$

- $\beta > 0.5$ indicates fast vocabulary growth
- Encyclopedic topics + named entities increase diversity



Byte Pair Encoding (BPE)

BPE learns subword units by repeatedly merging the most frequent symbol pairs.

$$(a, b) = \arg\max_{(x,y)} \text{freq}(x\, y) \quad \Rightarrow \quad xy \rightarrow \langle xy \rangle$$

- Merges: 5000 $\rightarrow$ Subword vocab: 5239
- Reduces OOV rate and helps with morphology

Example:

$$\texttt{azrbaycanlılardan} \Rightarrow \texttt{azrbaycan} + \texttt{lı} + \texttt{lardan}$$

Sentence Segmentation (Rule-based)

We detect sentence boundaries using punctuation + context rules.
Boundary candidates:

$$p \in \{., !, ?\}$$

Do **NOT** split when:

decimal: `3.14`, `50.5`

initial: `A. Mlikli`, `S.Rustamov`

abbrev: `prof.`, `Dr.`

Split when:

$$\text{quote close} + \text{space} + \text{Uppercase} \Rightarrow \text{boundary}$$

Segmentation Struggles (Real Wikipedia Text)

- Period does not always mean sentence end:
  - initials: `S.Rustamov`
  - abbreviations: `prof.`
  - decimals: `2.5 df`
- Quotes and guillemets: punctuation inside quotation marks
- Lowercase continuation after punctuation (avoid over-splitting)
- Trade-off: fewer false splits vs missing some true boundaries

Spell Checking: Levenshtein Distance

Baseline: Levenshtein distance via dynamic programming.

$$D(i,j) = \min \begin{cases} D(i-1,j)+1 & \text{delete} \\ D(i,j-1)+1 & \text{insert} \\ D(i-1,j-1)+[x_i \neq y_j] & \text{substitute} \end{cases}$$

- Given misspelled token w, search candidates $c \in V$
- Rank by smallest $D(w,c)$ (with pruning / max distance)

Weighted Edit Distance (From Confusions)

We replace uniform substitution costs with learned costs from character confusions.

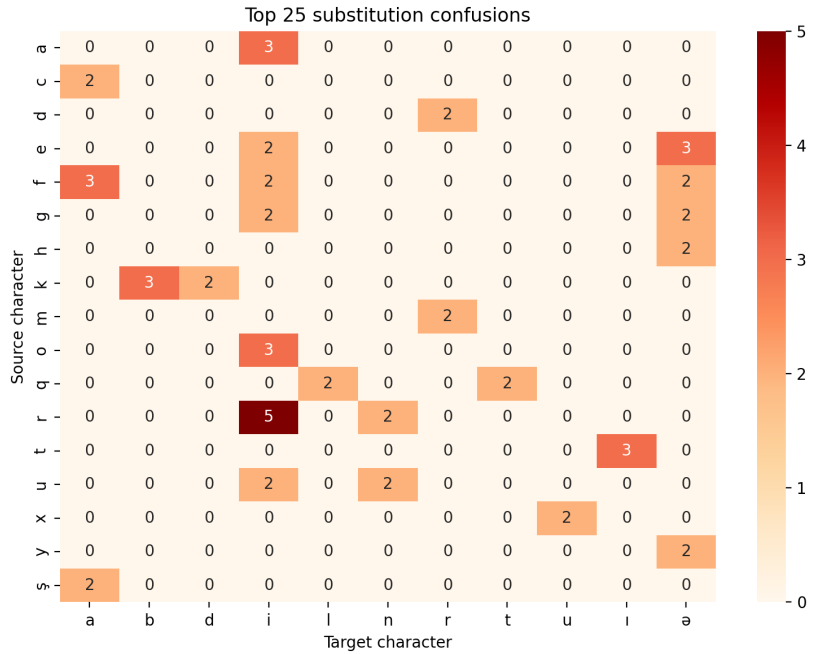
$$\text{cost}(a \to b) = \frac{1}{\epsilon + P(b|a)}$$

Weighted recurrence:

$$D(i,j) = \min \begin{cases} D(i-1,j) + w_{\text{del}}(x_i) \\ D(i,j-1) + w_{\text{ins}}(y_j) \\ D(i-1,j-1) + w_{\text{sub}}(x_i, y_j) \end{cases}$$

Weighted Edit Distance (Interpretation)

- Confusions learned from synthetic misspellings
- Common substitutions get lower penalty
- Diacritic and keyboard-adjacent errors favored
- Improves ranking of realistic corrections



Evaluation Metrics

We evaluate spellchecker ranking using Accuracy@k.

$$\text{Acc@k} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}\{y_i \in \text{TopK}(x_i)\}$$

Results on 1000 synthetic misspellings:

$$\text{Acc@1} = 0.637, \qquad \text{Acc@5} = 0.801$$

- Acc@1 = strict correction success
- Acc@5 = practical usefulness of suggestions

Key Outputs (Reproducibility)

- Run everything:
  - `bash scripts/run_all.sh`
- Plots:
  - `outputs/plots/zipf.png`, `outputs/plots/heaps.png`
- BPE:
  - `outputs/bpe/merges.txt`, `bpe_summary.json`
- Spellcheck:
  - `spell_eval.json`, `confusion_heatmap.png`

Conclusion

- Built a clean Azerbaijani Wikipedia corpus (623 docs)
- Verified Zipf-like distribution and fitted Heaps' law
- Trained BPE subword model for robust tokenization
- Implemented rule-based sentence segmentation with edge-case handling
- Developed spellchecker with weighted edit distance; Acc@5 = 0.801

Future Work

- Improve cleaning and Azerbaijani-only filtering earlier
- Replace synthetic gold data with manual annotations
- Context-aware spell correction using language model scoring
- Explore neural sentence segmentation as stronger baseline

Questions?