Arabic Morphological Segmentation

Character-Level HMM, BiLSTM, and Transformer Models

Yasser Hifny

Faculty of Computers and Artificial Intelligence
Capital University

1. What Is Arabic Morphological Segmentation?

Task

Split a surface word into morphemes or clitics such as conjunctions, prepositions, articles, stems, and attached pronouns.

وبالمدرسة ⟶ و + ب + ال + مدرسة

- Input: a word represented as a sequence of Arabic characters.
- Output: boundary decisions between adjacent characters.
- Segmentation conventions must be fixed before training and evaluation.

2. Why Arabic Segmentation Is Challenging

- Several clitics may attach to one orthographic word.
- A character sequence can contain conjunctions, prepositions, articles, stems, and pronouns.
- Some boundaries are ambiguous without lexical or contextual information.

$$\text{وسيعطونها} \longrightarrow \text{و} + \text{س} + \text{ي} + \text{عطي} + \text{ون} + \text{ها}$$

Important

The segmentation above is illustrative, not a claim about a unique gold analysis. Use the exact segmentation convention and output of FARASA as the assignment's reference.

3. Character-Level Boundary Labels

For a word with characters $c_1, \ldots, c_n$, predict whether a morpheme boundary follows each character.

$$b_i \in \{0, 1\}, \qquad b_i = 1 \text{ means a boundary after } c_i.$$

وبالمدرسة

Character	و	ب	ا	ل	م	د	ر	س	ة
Boundary after	1	1	0	1	0	0	0	0	1

This illustrative binary labeling corresponds to و + ب + ال + مدرسة; implementations should encode boundaries consistently, including the final character.

4. From Boundary Labels to Segmented Text

Given characters $c_1, \ldots, c_n$ and predictions $\hat{b}_1, \ldots, \hat{b}_n$:

1. Append each character to the current segment.
2. If $\hat{b}_i = 1$ and $i < n$, close the current segment.
3. Continue until the word ends; join segments with the chosen separator.

$$\text{فسيكتبونها} \longrightarrow \text{ف} + \text{س} + \text{يكتبون} + \text{ها}$$

Post-processing

Avoid empty segments, preserve character order, and define how punctuation, digits, diacritics, and whitespace are handled.

5. Learning Formulation

Character sequence:

$$x = (c_1, c_2, \ldots, c_n), \quad c_i \in \mathcal{V}_{char}.$$

Binary target sequence:

$$y = (b_1, b_2, \ldots, b_n), \quad b_i \in \{0, 1\}.$$

- The HMM treats boundary labels as hidden states and characters as observations.
- BiLSTM and Transformer encode character context and predict each boundary with sigmoid.
- Training labels are derived from FARASA-segmented words.

6. HMM at the Character Level

An HMM models a hidden boundary state sequence y given observed characters x.

$$P(x, y) = P(y_1) \prod_{t=2}^{n} P(y_t \mid y_{t-1}) \prod_{t=1}^{n} P(c_t \mid y_t).$$

- Hidden state: boundary decision, e.g. B = boundary after character, N = no boundary.
- Observation: the current Arabic character.
- Transition probabilities capture adjacent boundary patterns.
- Emission probabilities capture character distributions under each state.

وبالمدرسة

7. HMM Parameters and Smoothing

Estimate transition probabilities from labeled training sequences:

$$\hat{P}(y_t = j \mid y_{t-1} = i) = \frac{C(i,j) + \alpha}{C(i) + \alpha K}.$$

Estimate character emissions:

$$\hat{P}(c \mid y = j) = \frac{C(j,c) + \alpha}{C(j) + \alpha |\mathcal{V}_{char}|}.$$

- $C(i,j)$: count of state transition $i \to j$.
- $C(j,c)$: count of character c emitted by state j.
- $K = 2$ states; $\alpha > 0$ is a smoothing constant.
- Use an unknown-character strategy for characters unseen during training.

8. HMM Best-Path Objective

The most likely boundary sequence is

$$\hat{y} = \arg\max_{y} P(y \mid x) = \arg\max_{y} P(x, y).$$

Taking logarithms avoids numerical underflow:

$$\log P(x, y) = \log P(y_1) + \sum_{t=2}^{n} \log P(y_t \mid y_{t-1}) + \sum_{t=1}^{n} \log P(c_t \mid y_t).$$

Why dynamic programming?

There are 2^n possible boundary sequences. Viterbi finds the optimum in $O(nK^2)$ time, where K is the number of hidden states.

9. Viterbi: Initialization and Recurrence

Let $\delta_t(j)$ be the best log-score of a path ending in state j at position t.

$$\delta_1(j) = \log P(y_1 = j) + \log P(c_1 \mid j).$$

For $t = 2, \ldots, n$:

$$\boxed{\delta_t(j) = \max_i \left[\delta_{t-1}(i) + \log P(j \mid i)\right] + \log P(c_t \mid j)}$$

Store the predecessor:

$$\psi_t(j) = \arg\max_i \left[\delta_{t-1}(i) + \log P(j \mid i)\right].$$

At the final position, select $\hat{y}_n = \arg\max_j \delta_n(j)$ and backtrack with ψ.

10. Viterbi Worked Example: Arabic Word

Consider the word وبالمدرسة. For illustration, use states N (no boundary) and B (boundary after the current character).

$$\delta_t(B) = \max_{i \in \{N,B\}} \delta_{t-1}(i) + \log P(B \mid i) + \log P(c_t \mid B)$$

$$\delta_t(N) = \max_{i \in \{N,B\}} \delta_{t-1}(i) + \log P(N \mid i) + \log P(c_t \mid N)$$

- At every character, compute both scores and save the best predecessor.
- After processing the last character, backtrack to recover all boundary states.
- Convert predicted boundaries into segments, e.g. و + ب + ال + مدرسة.

Numerical scores depend on counts from the training split; this slide presents the exact computation, not fabricated learned values.

11. HMM Limitations

- The first-order Markov assumption only directly conditions on the previous hidden state.
- Emissions depend on a local state and do not naturally encode rich left/right character context.
- Sparse character/state counts require smoothing.
- Independent decoding per word may miss sentence-level information.

Motivation for neural encoders

A bidirectional recurrent network or Transformer can learn contextual character representations, then estimate boundary probabilities with sigmoid outputs.

12. Character Embeddings and BiLSTM

Map each character to a trainable embedding:

$$e_t = E[c_t].$$

A bidirectional LSTM processes the sequence in both directions:

$$\overrightarrow{h}_t = \text{LSTM}_f(e_t, \overrightarrow{h}_{t-1}), \qquad \overleftarrow{h}_t = \text{LSTM}_b(e_t, \overleftarrow{h}_{t+1}).$$

Concatenate contextual states:

$$h_t = [\overrightarrow{h}_t; \overleftarrow{h}_t].$$

فسيكتبونها

Each character representation contains information from both the left and right context.

13. BiLSTM with Sigmoid Boundary Output

For every character position:

$$z_t = w^\top h_t + b, \qquad p_t = \sigma(z_t) = \frac{1}{1 + e^{-z_t}}.$$

Predict a boundary when $p_t \geq \tau$ (often initialized at 0.5, but tuned on validation data).

$$\mathcal{L}_{BCE} = -\sum_{t=1}^{n} \left[b_t \log p_t + (1 - b_t) \log(1 - p_t) \right].$$

- Input: character IDs, optionally with character-level features.
- Output: one probability per character position.
- Mask padding positions so they do not contribute to loss or metrics.

14. BiLSTM Architecture



و + ب + ال + مدرسة ⟶ وبالمدرسة

- Train with binary cross-entropy and Adam.
- Select threshold τ on validation data, not the test set.

15. BiLSTM Training Details

1. Convert each word into a character-ID sequence.
2. Convert the reference segmentation into aligned boundary labels.
3. Pad batches and create masks for true sequence lengths.
4. Optimize binary cross-entropy over non-padding positions.
5. Monitor validation boundary F1 and complete-segmentation accuracy.

Potential imbalance

If boundary labels are much less frequent than non-boundary labels, report class frequencies and consider class weighting or validation-based threshold selection.

16. Transformer Encoder at Character Level

Each character ID is mapped to an embedding and combined with positional information:

$$h_t^{(0)} = E[c_t] + P_t.$$

Self-attention computes context-aware character representations:

$$Q = HW_Q, \quad K = HW_K, \quad V = HW_V,$$

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right) V.$$

- Every character can attend to all other characters in the word.
- Padding masks prevent attention to padded positions.
- A sigmoid classifier predicts each character boundary.

17. Transformer Boundary Classifier

For the final encoder state h_t:

$$p_t = \sigma(w^\top h_t + b).$$

Use binary cross-entropy:

$$\mathcal{L} = -\sum_t m_t \left[b_t \log p_t + (1 - b_t) \log(1 - p_t) \right],$$

where $m_t = 1$ for real characters and 0 for padding.

وسيستقبلونها ⟶ و + س + ي + ستقبلون + ها

The segmentation is an illustrative example; compare the model's output with the exact FARASA reference convention.

18. Transformer vs. BiLSTM

	BiLSTM	**Transformer encoder**
Context	Sequential forward and backward recurrence	Self-attention over all character positions
Parallelism	Recurrence limits parallel computation across time	Character positions processed in parallel
Position	Order is encoded by recurrence	Add positional embeddings or encodings
Long-range links	Information passes through recurrent steps	Direct attention links between positions
Training cost	Often effective on smaller data	Can need more data and compute

Both models use character inputs, sigmoid boundary probabilities, and the same segmentation reconstruction and evaluation protocol.

19. Evaluation Metrics

Compare predicted boundary decisions with the reference boundaries.

$$P = \frac{TP}{TP + FP}, \qquad R = \frac{TP}{TP + FN}$$

$$F_1 = 2\frac{PR}{P + R} = \frac{2TP}{2TP + FP + FN}.$$

- *TP*: predicted boundary is a reference boundary.
- *FP*: predicted boundary is not in the reference.
- *FN*: reference boundary was missed.
- Also report exact segmentation accuracy: fraction of words whose entire segmentation matches the reference.

Evaluation unit

State clearly whether F1 is calculated over character boundaries, segmented morphemes, or complete words. Boundary-level F1 is the most direct metric for the binary formulation used here.

20. Fair Experimental Protocol

1. Split articles/documents into train, validation, and test sets before extracting words to reduce leakage.
2. Build all vocabularies and model choices from the training split only.
3. Tune thresholds and hyperparameters on validation data.
4. Evaluate each model once on the held-out test set.
5. Use the same FARASA reference labels and boundary-level metric for all models.

Report

Data split sizes, character vocabulary, maximum length, optimizer, epochs, selected threshold, random seed, and model-selection rule.

21. Results Table: Fill with Measured Test Scores

Model	Boundary Precision	Boundary Recall	Boundary F1
Character HMM + Viterbi	–	–	–
Character BiLSTM + sigmoid	–	–	–
Character Transformer + sigmoid	–	–	–

No fabricated results

The figures are intentionally blank: actual F1 values require running the three models on a fixed, accessible dataset split with FARASA-derived reference segmentations.

22. Error Analysis

Inspect errors by category, not only by aggregate F1:

- Missed boundary: a clitic is merged into the stem.
- Spurious boundary: a stem is split incorrectly.
- Unseen or rare character sequences.
- Orthographic variants, diacritics, punctuation, and noisy text.
- Ambiguous segmentation or inconsistency in the reference convention.

وبكتابهم ⟶ و + ب + كتاب + هم

23. Assignment: SANAD + FARASA

Goal: Build three Arabic morphological segmentation models using character-level input.

1. Obtain the SANAD Arabic news dataset and create document-level train/dev/test splits.
2. Run FARASA segmentation on each split to create reference segmentations, respecting its license and usage requirements.
3. Convert each segmented word into character-level binary boundary labels.
4. Train a character HMM with Viterbi, a BiLSTM with sigmoid, and a Transformer encoder with sigmoid.
5. Compare boundary precision, recall, F1, and exact word segmentation accuracy.

24. FARASA Reference Segmentation

FARASA provides an Arabic word segmentation module and a web API. Access to downloadable packages/API requires registration; follow the official usage conditions.

`https://farasa.qcri.org/segmentation/`

Reference generation

Input: زار الطالب الجامعة.
Apply FARASA to obtain the segmentation output, then convert the output into boundary labels without changing its convention.

FARASA documentation describes standalone JAR/source options and a REST API. Cite the tool/version and record any normalization or post-processing used.

25. Assignment: HMM Implementation

- Hidden labels: N and B; observations: Arabic characters.
- Estimate initial, transition, and emission probabilities from training data.
- Apply additive smoothing and define handling for unseen characters.
- Implement Viterbi with backpointers and decode each word.
- Reconstruct segmented words and compute boundary-level F1.

Required analysis: Explain where the first-order HMM assumption helps and where it limits contextual predictions.

26. Assignment: Neural Models

BiLSTM

- Character embedding → bidirectional LSTM → dense layer → sigmoid.

Transformer

- Character embedding + positional encoding → Transformer encoder → dense layer → sigmoid.

For both models:

- Use masked binary cross-entropy, validation-based early stopping, and deterministic seeds.
- Tune the boundary threshold on validation data.
- Report parameter count, training time, and test metrics.

27. Assignment Deliverables

1. Dataset preparation and FARASA labeling script.
2. Reproducible training and evaluation code for all three approaches.
3. Description of boundary-label conversion and reconstruction.
4. Table of precision, recall, F1, exact segmentation accuracy, and runtime.
5. Error analysis with at least 20 Arabic examples.
6. Short report discussing data leakage, limitations, and differences among models.

28. Summary

- Arabic segmentation can be formulated as character-level boundary prediction.
- HMM + Viterbi provides a probabilistic baseline with dynamic programming.
- BiLSTM learns bidirectional sequential context and predicts boundaries with sigmoid.
- Transformer self-attention contextualizes all characters and predicts boundaries with sigmoid.
- Reliable comparison requires consistent FARASA labels, document-level splits, and measured test F1.

و + ب + ال + مدرسة

Thank You

Arabic Morphological Segmentation