



Cairo University - Faculty Of Engineering  
Computer Engineering Department  
Natural Language Processing - Fall 2025



# Final Project Submission Document

Arabic Diacritization Pipelines

Submitted to

**Eng. Omar Samir**

Submitted by

Name	Section	BN	ID
Ahmed Hamed	1	3	9220027
Akram Hany	1	13	9220158
Amir Anwar	1	14	9220166
Somia Saad	1	26	9202666

December 10, 2025

## Pipeline Architectures

We implemented two main pipeline architectures:

1. Pipeline A — Character-Level Sequential Models
2. Pipeline B — Hierarchical Word + Character Model with Attention

### 1 Pipeline A — Character-Level Encoder + Diacritic Classifier

This pipeline treats the task as a sequence labeling problem at the character level.

#### 1. Data Preprocessing

**1.1 Data Cleaning** We removed all characters outside the allowed vocabulary:

- Arabic letters
- Arabic diacritics
- Spaces
- Standard punctuation
- Padding symbol

Any invalid tokens (HTML tags, English characters, digits, unsupported Unicode symbols) were stripped.

**1.2 Sentence Tokenization** We split each sentence into smaller segments based on punctuation to reduce excessively long sequences and stabilize LSTM gradients.

**1.3 Character/Diacritic Separation** Each Arabic character was decomposed into:

- Base letter
- Associated diacritic (or PAD if none)

This enabled independent modeling of diacritics as categorical labels.

**1.4 Encoding** We performed:

- Character encoding: letter  $\rightarrow$  ID
- Diacritic encoding: diacritic  $\rightarrow$  ID
- Sequence padding: fixed-length representation for batching

#### 2. Feature Extraction

We implemented and evaluated several feature families:

**2.1 Trainable Embeddings (BiLSTM / BiLSTM-CRF / CNN-CRF)**

- Learned jointly with the model during training
- Embedding dimension tuned experimentally (128–256)

**2.2 Skip-gram Word Embeddings (CRF Model Only)** Skip-gram was used in the pure CRF model to provide word-level distributional semantics. *Hyperparameters:*

- Embedding dim: 100
- Window size: 5
- Workers: 4
- Min count: 1
- Epochs: 10

**2.3 CNN Character Encoder** (*for CNN-CRF model*) We extracted n-gram-like morphological features using a shallow CNN:

- Kernel sizes: (2, 3, 4)
- Filters per kernel: configurable
- ReLU activation + max-pooling

**2.4 HMM Features** For the HMM model, we computed:

- **Initial probabilities**
- **Transition probabilities** between diacritics
- **Emission probabilities**  $p(\text{letter}|\text{diacritic})$

### 3. Models

We trained four main models under Pipeline A.

#### 3.1 Bidirectional LSTM

A standard sequence labeling model using only trainable embeddings.

##### Hyperparameters

- Embedding dim: 128
- Hidden dim: 256
- Layers: 3
- Dropout: 0.2
- Batch size: 32
- Epochs: 5
- Learning rate: 0.001

#### 3.2 Conditional Random Field (CRF)

Uses skip-gram embeddings + handcrafted contextual features.

**Skip-gram hyperparameters:** Embedding dim: 100, Window size: 5, Min count: 1, SG = 1, Epochs: 10.

**CRF hyperparameters:**

- Algorithm: L-BFGS
- C1: 0.1 (L1 regularization)
- C2: 0.1 (L2 regularization)
- Max iterations: 100
- Context window:  $\pm 2$  characters

#### 3.3 CNN-CRF Model

Combines CNN character encoder + skip-gram embeddings + CRF inference.

- CNN extracts morphological features
- CRF ensures valid label transitions

**CNN hyperparameters:** Character embedding dim: 30, Filters: 50, Kernel sizes: 2, 3, 4, CNN output dim: 150, Batch size: 512.

### 3.4 Bidirectional LSTM-CRF

This hybrid model adds a CRF decoding layer on top of the BiLSTM output, allowing explicit modeling of dependencies between consecutive diacritics.

**Hyperparameters:** Same as BiLSTM **except:** Batch size: 128, Epochs: 6. *This model ultimately achieved the best performance.*

### 3.5 Hidden Markov Model (HMM)

A baseline generative model trained using Initial probability estimation, Transition matrix, and Emission matrix. Inference performed using Viterbi decoding.

## 2 Pipeline B — Hierarchical Word + Character Model

### 4.1 Overview

Pipeline B implements the **D2 (Two-Level Diacritizer)** architecture from the paper "*Effective Deep Learning Models for Automatic Diacritization of Arabic Text*". Unlike Pipeline A which processes text character-by-character, this architecture separates word-level and character-level processing.

**Key insight:** Diacritization requires both local morphological features (within-word patterns) and global syntactic context (cross-word dependencies like grammatical agreement).

### 4.2 Preprocessing Differences

- **Sliding Window Segmentation:** Sentences are split into overlapping segments of  $T_s = 10$  words with  $\text{stride} = 5$ .
- **Word-Level Tokenization:** Text is tokenized at the word level. Words below  $\text{min\_freq} = 2$  are mapped to  $\langle \text{UNK} \rangle$ .
- **Character-Diacritic Separation:** Combined diacritics (e.g., shaddah + vowel) mapped to one of 15 classes.

### 4.3 Feature Extraction

- **Trainable Word Embeddings:** Dim 128.
- **Trainable Character Embeddings:** Dim 32.
- **Contextual Word Features (BiLSTM):** 2-layer BiLSTM, Hidden dim 256.
- **Contextual Character Features (BiLSTM):** Character embeddings concatenated with parent word context, processed by 3-layer BiLSTM (Hidden 512).
- **Cross-Level Attention Features:** Scaled dot-product attention over word encodings. For each character, attends to **all other words**.

### 4.4 Model Architecture

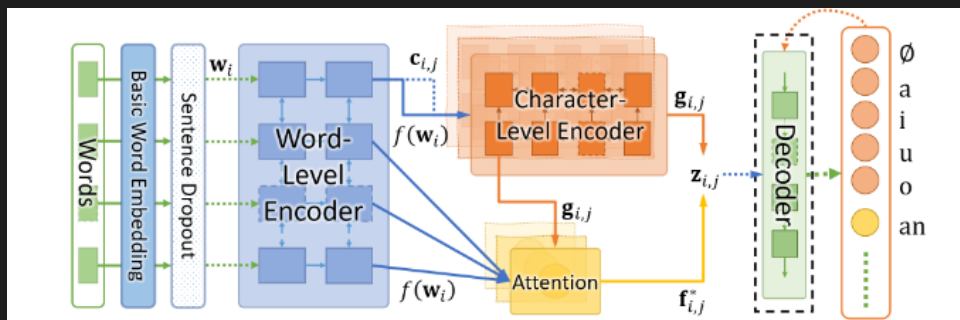


Figure 1: Architecture of the D2 Hierarchical Model

Table 1: Component Specification

Component	Specification
Word Encoder	2-layer BiLSTM, hidden=256, word_dropout=0.2
Char Encoder	3-layer BiLSTM, hidden=512, input=[char_emb; word_ctx]
Attention	Scaled dot-product, Q=char, K/V=words, self-masked
Classifier	Linear(1024 $\rightarrow$ 15 classes)

Table 2: Training Hyperparameters

Parameter	Value
Batch size	128
Learning rate	0.002
Optimizer	Adam
LR scheduler	ReduceLROnPlateau (factor=0.5, patience=1)
Dropout	0.25 (vertical), 0.2 (word/input)
Early stopping	3 epochs patience

## 4.5 D2 Experiments

Exp	Acc	CE Acc	W-L	C-L	W-Hid	C-Hid	Drop	W-Drop	Batch	LR	MaxW	Str	Notes
1	96.1	92.8	2	2	256	512	0.25	0.2	32	0.002	10	5	Baseline config
2	95.0	93.0	2	2	256	512	0.25	0.0	32	0.002	10	5	No word dropout
3	96.87	95.0	2	3	256	512	0.25	0.2	32	0.002	10	5	+1 char layer
4	81.0	-	2	5	256	512	0.25	0.2	32	0.002	10	5	Overfitted
5	96.98	94.77	2	3	256	512	0.25	0.2	128	0.002	20	15	Larger batch
6	96.97	94.81	2	4	256	512	0.25	0.2	128	0.002	20	15	+1 char layer
7	<b>97.2</b>	<b>95.37</b>	4	4	<b>256</b>	<b>512</b>	<b>0.25</b>	<b>0.2</b>	<b>128</b>	<b>0.002</b>	<b>15</b>	<b>15</b>	<b>Best</b>

Table 3: D2 Model Experiments (W-L: Word Layers, C-L: Char Layers, Str: Stride)

### Best D2 Model Results:

- Overall Accuracy: **97.20%**
- Case Ending Accuracy: **95.37%**
- DER (Diacritic Error Rate): **2.80%**

## 5. Evaluation

### Accuracy Evaluation

Table 4: Case Ending Accuracy

Model	Accuracy
BiLSTM	95.69
CRF	83.82
CNN-CRF	85.17
<b>BiLSTM-CRF</b>	<b>95.75</b>
HMM	45.81
D2 Model	95.37

Table 5: Overall Performance

Model	Accuracy
BiLSTM	97.73
CRF	82.33
CNN-CRF	83.67
<b>BiLSTM-CRF</b>	<b>97.80</b>
HMM	61.41
D2 Model	97.20

## 6. Final Model Selection

We selected the **Bidirectional LSTM-CRF** model as our final submission.

**Reasons for selection:**

- It achieved the **highest accuracy** among all tested models.
- It obtained the **lowest Diacritic Error Rate (DER)**.
- CRF decoding significantly reduced invalid or unlikely diacritic sequences.
- Better generalization to rare diacritics and complex morphological patterns.