

To create an AI model that can add harakat (vowel marks) to Arabic text with high accuracy (like 99%), you'll need to follow several steps. Here's an outline to help you get started:

1. Understand the Problem:

- **Input:** Arabic text without harakat.
- **Output:** Arabic text with harakat.
- Harakat is important for correct pronunciation, and adding it requires context understanding of the text.

2. Choose the Right Model Architecture:

- Use **Sequence-to-Sequence models** (like **Transformer-based models** such as **BERT** or **T5** for Arabic).
- A **character-level model** is more effective because harakat is added to individual characters rather than words.

3. Prepare the Dataset:

- You need a parallel corpus (text without harakat and text with harakat) to train your model.
- **Datasets you can use:**
 - **Arabic-Indic Corpus:** Includes texts with harakat and their corresponding versions without.
 - **Quranic Texts:** The Quran has fully harakated Arabic, making it a useful resource.
 - **Tashkeela Corpus:** A large dataset of Arabic sentences with harakat.
 - **Custom Data:** You can collect data from books, articles, or other sources and manually add harakat (though this might be time-consuming).

4. Preprocessing the Data:

- **Tokenization:** Split the text into characters or sub-words.
- **Normalization:** Standardize the text to remove any inconsistencies (e.g., normalizing Arabic characters).
- For training, each sentence without harakat should be mapped to its harakatized counterpart.

5. Training the Model:

- **Train a Transformer Model** (like **BERT**, **T5**, or **GPT**):
 - Fine-tune a pre-trained model on your dataset.
 - Use **Masked Language Models (MLM)** or **causal language models** for training.
- **Supervised Learning:** Use the harakatized text as the target output.
- **Data Augmentation:** Apply different augmentation techniques (e.g., shuffling, random masking) to make the model more robust.

6. Evaluation Metrics:

- **Accuracy:** Calculate character-level accuracy to compare the model's output to the expected harakatized text.
- **Precision, Recall, and F1 Score:** Use these metrics to evaluate the model's performance on individual harakat.
- **BLEU Score:** Can be used to evaluate how close the model's output is to the reference text.

7. Improve Accuracy:

- **Fine-tuning:** Experiment with hyperparameters (e.g., learning rate, batch size, number of layers).
- **Use Pre-trained Models:** Start with a pre-trained Arabic transformer model (like **AraBERT** or **Arabic T5**) and fine-tune it for your task.
- **Post-Processing:** After generating the harakat, you can use rule-based post-processing to correct common errors.

8. Testing and Validation:

- Test the model on unseen data (out-of-domain data).
- Ensure the model performs well across different text domains (formal, informal, poetry, etc.).

9. Deploying the Model:

- Once the model achieves good performance, you can deploy it as an API or integrate it into a web app.
- Consider using frameworks like **Flask** or **FastAPI** for deployment.

10. Tools and Libraries:

- **TensorFlow/PyTorch:** For building and training your model.
- **Hugging Face Transformers:** For accessing pre-trained transformer models.
- **Keras:** For easy implementation and model training.
- **NLTK/spacy:** For text preprocessing.

Would you like help diving deeper into any of these steps or getting specific code examples?