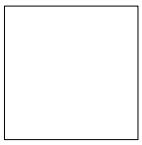

Chapitre 3 [1]

DÉVELOPPEMENT D'UN SYSTÈME DE SYNTHÈSE VOCALE

1 .

2traitement du langage naturel
(front-end)

"Toute vie est une concaténation d'éphémères."

Alfred Edward Kahn (1917-2010)

3.1introduction

Dans ce chapitre, nous décrirons l'architecture de notre système TTS à travers lequel nous avons tenté de surmonter les lacunes des systèmes examinés. La contribution de ce chapitre est double : d'une part, nous contribuons à l'amélioration du module TTS NLP (front-end) par l'utilisation et le développement de trois composants : un système de diacritisation, un convertisseur G2P et un module de prédiction des pauses. D'autre part, nous contribuons à l'amélioration du module DSP par une nouvelle approche de sélection d'unités basée sur un inventaire d'unités non-uniformes. Une attention particulière est accordée au choix des unités acoustiques de base adoptées pour cette tâche. En outre, nous proposons un ensemble de filtres de présélection contextuelle pour optimiser la complexité temporelle de l'algorithme de sélection d'unités en rejetant les unités de la base de données qui ne correspondent pas au contexte droit et gauche de l'unité cible. Enfin, nous discutons des caractéristiques que nous avons adoptées pour le calcul de la fonction de coût et l'utilisation d'une pénalité téléphonique pour maximiser le coût des unités susceptibles de produire des artefacts aux points de concaténation.

la ponctuation fournie avec le texte d'entrée. Il est donc nécessaire d'aborder cette limitation dans ce travail en implémentant un module qui traite les pauses implicites (ponctuation) et les coupures de phrases. Ces points sont abordés dans la section 3.2.2.

Comme nous l'avons vu au point 2.3.1, le bloc frontal typique (NLP) comprend quatre composants principaux : un normalisateur, un phonétiseur (G2P) et un module d'extraction de caractéristiques pour construire les vecteurs de caractéristiques linguistiques et prosodiques.

Pour la langue arabe, l'utilisation d'un module de diacritisation est obligatoire (Section2.6.1). Ceci conclut qu'un module de diacritisation doit être implémenté et incorporé dans le frontal du système TTS. Néanmoins, il n'est pas nécessaire de réinventer la roue, car il existe de nombreux outils de diacritisation pour la langue arabe. Nous devrions plutôt étudier le choix d'un diacriticien précis en fonction de certains facteurs (par exemple, la précision, la disponibilité, etc.). De plus, dans le chapitre précédent, nous avons soulevé un problème lié à l'utilisation inappropriée des pauses dans la plupart des systèmes TTS disponibles, qui semblent s'appuyer uniquement sur la fonction de lecture de la langue arabe.

En outre, la construction d'un module de phonétisation nécessite une étude approfondie des règles phonologiques les plus pertinentes. Dans la littérature, différentes règles sont utilisées, y compris celles liées à l'arabe classique (par exemple, les règles du *Tajweed*). Cependant, dans ce travail, nous n'abordons que les règles les plus importantes qui sont conformes au style d'expression du MSA. Le dernier module implémenté de notre système TTS frontal effectue l'extraction d'informations linguistiques et prosodiques, que nous utiliserons pour construire le vecteur de caractéristiques de chaque unité phonétique.

3.2.2. *Diacritisation*

Avant d'étudier le choix d'un système de diacritisation, nous devons d'abord définir les critères à prendre en compte pour faire ce choix. Le paramètre le plus crucial d'un outil de diacritisation dans une perspective TTS est le niveau de diacritisation, qui varie entre complet, demi ou partiel selon le nombre de marques diacritiques présentées sur un mot (Mohamed Attia Mohamed Elaraby, 2000).

Le premier mode implique que toutes les lettres se voient attribuer des diacritiques appropriés. Dans le mode de *demi-diacritisation*, seules les lettres indépendantes de la morphologie sont diacritisées, tandis que les diacritiques des lettres qui dépendent de l'analyse syntaxique sont supprimés. Par exemple, avec la demi-diacritisation, le mot " ? xDh" èYg @ (signifiant "il l'a pris") est diacritisé en " ? axaDh" èYg @. Dans le dernier mode, une seule lettre ou un sous-ensemble de lettres est diacritisé. Par exemple, le Sokun ou les voyelles qui suivent les voyelles longues avec un son similaire, comme dans " ? xDu:h" èðYg @ qui est partiellement diacritisé comme " ? axaDu:hu" èð Yg @. @, puisque l'omission des diacritiques dans de tels cas n'affecte pas la prononciation du mot, nous supposons que les modes complet et partiel sont les plus appropriés pour une application TTS.

Avec ces critères à l'esprit, nous avons filtré certains diacritistes à partir de nombreuses enquêtes menées, qui ont guidé notre décision. Dans (Hamed et Zesch,

3017) et (Fadel et al., 2019), des études empiriques comparatives ont été menées sur différents systèmes disponibles, parmi lesquels les outils relativement puissants MADAMIRA (Pasha et al., 2014), Farasa (Abdelali et al., 2016) et Mishkal (Zerrouki, 2014). Les deux travaux s'accordent à dire que Farasa est plus performant que MADAMIRA. Mishkal, cependant, a montré un taux d'erreur plus élevé (en utilisant les mesures du taux d'erreur diacritique (DER) et du taux d'erreur de mot (WER)) dans les deux études. Par ailleurs, d'autres outils de diacritisation sont également disponibles, par exemple, ArabicDiacritizer1 et Ali-Soft2.

Pourtant, ils n'ont pas été pris en compte pour cette décision en raison du nombre excessif de problèmes (par exemple, la taille limitée du texte d'entrée, les incompatibilités d'intégration).

Pour les raisons exposées ci-dessus, nous avons décidé d'adopter l'outil de diacritisation Farasa, qui permet une intégration simple en tant que bibliothèque dans les applications basées sur Java ou par le biais d'une API RESTful mise en œuvre pour différents langages de programmation.

3.2.2 *Pause de la prédiction*

Dans le discours naturel, outre les pauses imposées par les signes de ponctuation, les locuteurs ont naturellement tendance à intégrer des pauses dans un énoncé. Ce phénomène est de plus en plus perçu pour les phrases longues, qui conduisent généralement le locuteur à s'essouffler. Par ailleurs, le locuteur peut introduire des pauses pour exprimer une hésitation ou mettre en valeur un segment de la phrase. Tous ces scénarios caractérisent le discours naturel qui, lorsqu'ils sont ignorés, peuvent conduire à un discours monotone ou à une ambiguïté dans le sens de la phrase.

Dans le contexte du TTS, l'insertion automatique des pauses n'est pas moins importante que les autres caractéristiques prosodiques telles que la hauteur, l'intensité et la modélisation de la durée. Les pauses, ainsi que les autres caractéristiques, contribuent à la prosodie, qui est fortement corrélée avec le naturel de la parole synthétisée.

Globalement, les pauses insérées par le système TTS sont liées à deux types de pauses : les pauses basées sur la ponctuation, également appelées groupes de souffle, et les limites de phrases ou petits groupes de souffle. Dans cette thèse, les deux types sont abordés.

3.2.2.1*Prédiction des pauses basée sur la ponctuation*

Prenons l'exemple suivant :

Il a dit que Bilal était célibataire.
La phrase ci-dessus accepte deux significations possibles :

1. Lui, dit Bilal, a un célibataire.

2. Il a dit : Bilal a un célibataire.

Pour ces exemples, nous pouvons percevoir le rôle crucial des pauses prosodiques indiquées par la ponctuation. Dans un système typique, la ponctuation du texte est l'indicateur le plus fiable de l'emplacement de la pause (Huang et al., 2001).

Il est à noter qu'à ce stade, nous supposons qu'une phase de normalisation a été préalablement effectuée pour résoudre les abréviations et les symboles spéciaux qui pourraient être confondus avec un signe de ponctuation (par exemple, les points et les virgules dans le nombre 20.000 ou 01,12).

La première tâche effectuée par notre module de prédiction des pauses est le remplacement de chaque signe de ponctuation par son étiquette correspondante. Dans cette optique, nous avons défini un ensemble de balises de pause communes en fonction de la durée moyenne de silence qu'elles peuvent nécessiter. Cette classification est basée sur une étude statistique que nous avons menée sur un corpus de discours transcrit pour localiser tous les signes de ponctuation et calculer la durée moyenne de chacun d'entre eux.

L'échantillon de corpus de discours utilisé pour cette étude a été extrait du corpus construit dans cette thèse (chapitre 5). L'échantillon comprend 300 phrases délimitées par un point, un point d'interrogation ou un point d'exclamation, parmi lesquelles 188 déclaratives et 112 interrogatives avec une taille moyenne de 12 mots par phrase. Les phrases ont été lues par le locuteur à un rythme moyen de 13 phonèmes/seconde. L'étude a permis d'analyser un total de 1 013 signes de ponctuation. Les résultats sont présentés dans le tableau 3.1.

Sur la base de ces résultats, nous avons cherché à définir une balise unique pour les signes de ponctuation dont la durée est relativement proche. Par conséquent, deux étiquettes de pause ont été définies comme suit :

<*SP*> pour une courte pause représentée par les signes de ponctuation : virgules, parenthèses, deux points et toutes sortes de guillemets. <*LP*> pour une longue pause représentée par les signes de ponctuation : points, points d'ellipse, points-virgules et points d'exclamation.

En outre, l'étiquette <*QS*> est attribuée aux points d'interrogation car ils transmettent des informations essentielles sur la prosodie de la phrase, qui ne doivent pas être confondues avec d'autres signes de ponctuation.

En outre, sur la base d'une analyse approfondie des propriétés spectrales et de la forme de la *F0* des différents types de phrases, il semble essentiel de considérer le type de questions qui acceptent des réponses "Oui" ou "Non" comme un type distinct de phrases interrogatives puisqu'elles sont caractérisées par une *F0* croissante à la fin de la phrase, comme le montre la figure 3.1. Pour cela, l'étiquette *<QSYN>* est particulièrement attribuée aux phrases de cette classe.

3.2.2.2 *Prédiction des limites de phrases*

Outre les pauses marquées par des signes de ponctuation, un type de pause supplémentaire peut se produire à l'intérieur d'un groupe de souffle, c'est-à-dire entre deux signes de ponctuation, pour délimiter un groupe de mots grammaticalement compatibles. Le processus par lequel ces limites de groupes sont définies est appelé "phraséologie". Dans la littérature, ce processus est généralement abordé à l'aide de techniques basées sur des règles.

Pour la langue arabe, très peu de travaux ont abordé le phrasé en TTS. Par exemple, dans (Baloul et Baudry, 2003), l'auteur a formalisé un ensemble de règles qui prédisent la position des pauses au sein d'un groupe grammatical en se basant sur un corpus de test de 168 phrases. Grâce à ce corpus, l'auteur a étudié la corrélation entre les types de pauses et les limites des groupes grammaticaux. L'étude a abouti à des règles guidées par le type et la nature du groupe grammatical (verbal, sujet, direct, indirect) tout en tenant compte du nombre min et max de syllabes depuis ou avant la dernière ou la prochaine pause. Néanmoins, ce modèle est loin d'être généraliste compte tenu de la taille et du type du corpus d'analyse. La même conclusion a été formulée par l'auteur de ce travail.

Selon (Huang et al., 2001), ces systèmes basés sur des arbres de décision peuvent atteindre jusqu'à 81% de précision des pauses. D'autre part, une plus grande précision a été obtenue avec des techniques orientées données basées sur une grande quantité de données annotées à différents niveaux (POS, ponctuation, etc.)(Braunschweiler et Chen, 2013 ; Sarkar et Rao, [2015]).

Dans ce travail, avant de penser à procéder à la préparation d'un corpus de test, qui est généralement une tâche coûteuse, pour mener une étude guidée par les données, nous avons lancé plusieurs tests avec des outils d'analyse syntaxique disponibles tels que MADAMIRA, avec l'intention de les utiliser pour obtenir un premier niveau d'analyse syntaxique. Cependant, ce dernier était considéré comme très minimal (c'est-à-dire une analyse syntaxique au niveau du mot), nécessitant l'inclusion d'autres facteurs pour regrouper les sous-chaînes sous un seul segment syntaxique.

En outre, le RGB arabe, un analyseur syntaxique de dépendance, a été évalué dans le même but. Cet outil est développé par le groupe Natural Language Processing du Massachusetts Institute of Technology (MIT). Il a été adapté à la langue arabe par les

mêmes auteurs en collaboration avec le QCRI et en utilisant des données entraînées avec la boîte à outils Farasa. L'outil est distribué dans une version opensource java [4]ou via la page de démonstration du site web de [5]Farasa . L'approche implémentée est basée sur la décomposition tensorielle et le décodage glouton, (Zhang et al.,2014).

Les résultats de cet outil ont été comparés à ceux de l'analyseur de dépendances de Stanford pour la langue arabe. Il en est ressorti que le premier semblait correspondre davantage aux emplacements des frontières prosodiques (pauses). Il a donc été adopté pour servir de premier niveau d'analyse syntaxique. La figure 3.2 montre un exemple d'analyse syntaxique de dépendance utilisant l'outil RVB arabe fourni par Farasa.

Les dépendances sont identifiées par les *ID de tête* entre le mot courant et le mot dont il dépend. Dans notre module de prédiction de pause, la sortie de l'analyseur de dépendances est traitée élément par élément (les éléments se réfèrent ici aux mots de la séquence de sortie). Sur la base de l'*ID de tête* de chaque item, le module décide si une limite de phrase sera placée ou non avant l'item. En effet, une balise de pause *<IP>* , pour une Pause Interne, n'est insérée que si l'item courant n'est pas dépendant de l'item précédent ni de l'item suivant, c'est-à-dire si l'ID de l'item courant est différent de son *Head ID +1* et de son *HeadID -1*.

Par conséquent, en utilisant le même exemple présenté à la figure 3.2, la sortie du module de prédiction de pause est présentée à la figure 3.3.

De plus, sur la base de notre analyse d'échantillons provenant de livres audio (un total d'environ 1,20 heures de discours), nous avons défini un ensemble de *mots d'analyse* après ou avant lesquels le locuteur introduit généralement une courte pause. Des exemples de ces mots sont présentés dans le tableau 3.2. Les mots post-rupture font référence aux mots qui se produisent après une limite de phrase, et les mots pré-rupture représentent ceux qui se produisent avant une limite de phrase. Certains mots sont redondants dans les deux catégories puisqu'ils apparaissent avant et après une limite de phrase. La liste complète des mots est présentée à l'annexe II.

Compte tenu de ces nouvelles règles, la décision concernant le placement de la balise *<IP>* sera conditionnée par l'ensemble des mots définis, comme le montre la figure 3.4.

Après avoir effectué plusieurs tests avec le module de prédiction des limites de phrases implémenté, nous avons remarqué que l'insertion de certaines balises *<IP>* était forcée par les mots d'analyse syntaxique. Ces derniers semblent être performants lorsque la ponctuation est manquée ou lorsque la sortie de l'analyseur de dépendances amène le modèle à ignorer une pause potentielle. Un exemple de tels cas est présenté dans la figure 3.5. Les balises <IP> rouges sont déduites par l'analyseur de dépendance RGB. La balise bleue est forcée par l'analyseur

4 https://github.com/qcri/ArabicRBGParser

5 https://farasa-api.qcri.org/dependency/

3.2.3 *Grapheme-To-Phone*

Comme nous l'avons mentionné au chapitre 2, le processus G2P est généralement réalisé soit par une mise en correspondance directe, soit par une consultation de dictionnaire pour les langues non régulières telles que l'anglais et le français, où les irrégularités de prononciation ne peuvent être contrôlées par un ensemble fini de règles. En revanche, pour la langue arabe, étant une langue phonémique, le défi des prononciations irrégulières est moins aigu et est généralement traité par un dictionnaire d'exception d'un nombre fini de mots. Cependant, un autre défi est soulevé dans la langue arabe : la prononciation d'un phonème peut varier en fonction du contexte dans lequel il se produit. Celles-ci sont principalement liées aux effets de co-articulation entre les sons d'un même mot ou entre les mots d'une même phrase. Heureusement, les phonéticiens et les linguistes ont déjà contourné ce problème dans des travaux antérieurs. Cette approche est désignée dans la littérature comme la transcription phonétique à base de règles.

Dans ce travail, nous avons adopté une approche basée sur les règles pour construire le module G2P qui sera incorporé dans le frontal de notre système TTS. En conséquence, nous avons rassemblé un ensemble de règles phonologiques provenant de différents travaux pour rendre l'ensemble de règles aussi complet que possible, ainsi qu'un dictionnaire d'exceptions. Ces règles sont décrites en détail dans cette section.

3.2.3.1 *Transcription phonétique*

Les règles présentées dans cette section sont principalement issues d'un certain nombre d'ouvrages linguistiques de référence (Al-ghamdi, Al-Muhtasib et Elshafei, 2004 ; Alghmadi et Abdulaziz, [2003] ; El-Imam, [1989] ; Watson, [2007]). Cependant, comme nous l'avons dit précédemment, l'ensemble des règles mises en œuvre n'est en aucun cas exhaustif, car nous n'avons sélectionné que les règles les plus pertinentes qui sont couramment employées dans le style d'expression orale des MSA.

1. **Transcription des mots irréguliers**

   La première étape de la conversion lettre-son est le traitement des mots irréguliers. Ceux-ci sont définis comme des mots dont la prononciation ne peut être prédite à partir d'une correspondance directe entre lettres et sons, ni à partir des règles phonologiques.

   L'approche standard pour traiter ces exceptions dans MSA est de construire un lexique des mots de cette classe avec leurs prononciations. Nous avons donc rassemblé une liste de mots irréguliers provenant de différentes sources afin de la rendre aussi exhaustive que possible. Ils sont listés dans le tableau 3.3.

2. **Conversion graphème-phonème** Cette étape vise à effectuer la conversion graphème-phonème qui n'implique pas l'utilisation de règles phonologiques. Les graphèmes sont plutôt inspectés un par un et mis en correspondance avec les phonèmes correspondants, en tenant compte de leur contexte. Les règles graphème-phonème sont mises en œuvre dans l'ordre indiqué ci-dessous : - Remplacement de la marque de nunation arabe appelée *Tanween* par son phonème correspondant.

   •

3. **Conversion des phonèmes en phones** La sortie de l'étape précédente, qui est une séquence de phonèmes, est utilisée comme entrée pour ce module qui effectue la mise en correspondance des phonèmes en phones. Comme indiqué précédemment dans la section 2.3.1.2, ce module utilise un ensemble de règles phonologiques recueillies dans la littérature (Al-ghamdi, Al-Muhtasib, et Elshafei, 2004 ; Alghmadi et Abdulaziz, [2003] ; Watson, [2007]) pour prédire la prononciation réelle d'un phonème influencée par son contexte.

   En MSA, l'influence des sons les uns sur les autres se traduit par des allophones différents des mêmes phonèmes. Les cas concernés par cette transformation sont classés en MSA en deux phénomènes : L'assimilation et les règles de Hamza. Sur la base d'un certain nombre de travaux linguistiques et phonologiques(Al-ghamdi, Al-
   Muhtasib, et Elshafei, [2004] ; El-Imam, [2004]), nous avons collecté et implémenté ces règles. En outre, nous avons inclus un ensemble supplémentaire de règles qui ont démontré une amélioration notable de la qualité de la synthèse.

   - L'assimilation :

   En MSA, l'assimilation se produit dans différents cas (par exemple, dans un contexte nasal, avant les consonnes alvéolaires stop, etc.) Cependant, la plupart des règles d'assimilation en arabe ne sont pas obligatoires en MSA ; elles sont généralement employées en arabe classique et sont appelées règles *Tajweed*. Par conséquent, en ce qui nous concerne, nous avons simplement mis en œuvre la règle d'assimilation (È@), qui est obligatoire en MSA. Néanmoins, nous n'ignorons pas complètement les autres cas d'assimilation ; ils sont implicitement pris en compte par l'algorithme de sélection des unités implémenté, qui sélectionne les unités vocales cibles en fonction de leur contexte phonétique (diphones précédents et suivants). Nous avons implémenté les règles d'assimilation comme suit :

- Si l'article défini est suivi d'une lettre alvéolaire ou dentaire, également désignée en phonologie arabe par le terme de *consonne solaire* (SC) (voir tableau 3.4), elle est initiale dans la phrase :

/?al/ + SC → [?a] + SC

Par exemple, Ò Ë@ / ? aSSamsu/ → [SSamsu] (signifiant "le soleil").

- Si l'article défini est suivi d'un SC et qu'il s'agit d'une phrase médiane ou finale :

/?al/ + SC → SC

Par exemple, ÒË@ áÓ /mina ? aSSamsi/ → [mina SSamsi] (signifiant " du soleil ").

- Si l'article défini est suivi d'une *consonne lunaire* (MC) (voir le tableau 3.4) et est à l'origine de la phrase :

/?al/ + MC → [?al] + MC

Par exemple, ÈA® ÜÏ@ / ? alomaqa:l/ → [? almaqa:l] (signifiant " l'article ").

- Si l'article défini est suivi d'un MC et qu'il s'agit d'une phrase médiane ou finale :

/?al/ + MC → [l] + MC

Par exemple, ÈA® ÜÏ@áÓ /mina ? alomaqa:l/ → [mina lmaqa:l] (signifiant " de l'article ").

- **Hamza règne** :

/ ? i/) est généralement abordée dans la littérature avec quelques règles limitéesTraiter les différentes variantes de prononciation de Hamza (@ ou @ qui ne traitent pas de tous les aspects de ce phonème. Ainsi, pour rendre ces règles aussi complètes que possible, nous avons défini des règles

supplémentaires basées sur l'analyse des prononciations de différents locuteurs. Ainsi, nous avons identifié deux cas principaux :

a) Hamza se prononce s'il est à l'initiale de la phrase (indiqué dans le texte suivant). La règle du bas avec ˆ) ou précédé d'une consonne :

$$\hat{} \ / ? i/ \rightarrow [? i]$$
$$C + / ? i/ \rightarrow C + [? i]$$

Par exemple, le mot XAJ¯@ / ? iqtis'a:d/ se prononce [? iqtis'a:d] lorsqu'il apparaît au début d'une phrase. De même, le mot XAJ¯B@ / ? al?iqtis'a:d/ se prononce [? al?iqtis'a:d] car il est précédé de la consonne /l/.

b) Hamza est supprimé s'il est précédé d'une voyelle :

$$V + / ? i/ \rightarrow V$$

Par exemple, le mot XAJ¯AK. /bi?iqtis'a:d/ se prononce [bi?qtis'a:d] car il est précédé de la voyelle /i/.

Notez que la dernière règle inclut les cas où le mot est préfixé (par exemple /fa/, /ka/, /bi/, et /li/).

- **La confluence de deux lettres non-voyées.**

En phonologie arabe, si un mot se termine par un *sakin* qui fait référence soit à l'une des voyelles longues /a:/, /i:/, ou /u:/, soit à une consonne non-voyée (*Sokun*), et est suivi d'un mot commençant par un *sakin*, généralement l'article défini /?al/ ou Hamza / ? i/, la première lettre muette est omise. Bien que cette règle soit largement connue et incluse dans de nombreux ouvrages, elle n'est pas correctement mise en œuvre, et donc de nombreux cas de cette règle ne sont généralement pas couverts. Par conséquent, nous avons défini un ensemble de règles à travers lesquelles nous avons tenté de couvrir tous les scénarios possibles du phénomène des lettres muettes.

- Une voyelle longue (LV) est raccourcie si elle se trouve à la fin d'un mot et est suivie d'un mot commençant soit par Sokun, soit par une consonne géminée (CC) (résultant généralement de l'assimilation de l'article défini /?al/). Cette règle est mise en œuvre comme suit :

LV + CC → SV (CC est soit une consonne géminée, soit un Sokun).
consonne)

(Par exemple, le mot signifiant "dans la maison "l). J.Ë@ú ¯/fi : lbajti/ se prononce [fi lbajti].

- La voyelle courte *Fatha* est insérée après un /n/ non-voyellé si ce dernier se trouve à la fin d'une préposition et est suivi d'un mot commençant par Sokun ou une consonne géminée. Cette règle est mise en œuvre comme suit :

  Prép/n/ + CC → [na] + CC

  Par exemple, le mot l J.Ë@áÓ /min lbajti/ se prononce [mina lbajti] (ce qui signifie "de la maison").

- La voyelle courte *Damma* [u] est insérée après un /m/ non-voyellé si ce dernier se trouve à la fin d'un mot pluriel (pronom ou verbe) et est suivi d'un mot commençant par Sokun ou une consonne géminée. Cette règle est mise en œuvre comme suit :

  Pluriel/m/ + CC → [mu] + CC

  Par exemple, le mot øQåJ.Ë@ÑêË /lahum lbuSra:/ se prononce [lahumu lbuSra :] (ce qui signifie "Pour eux sont de bonnes nouvelles").

- Si aucune des règles précédentes ne s'applique, et qu'une consonne (C) se trouve à la fin d'un mot suivi d'un mot commençant par Sokun ou une consonne géminée, la voyelle courte Kasra [i] est insérée à la fin du premier mot. Cette règle est mise en œuvre comme suit :

  C + CC → C + [i] + CC

  Par exemple, le mot H.AJºË@Yg /xuD lkita:ba/ se prononce [xuDi lkita:ba] (ce qui signifie "prends le livre").

Il est important de noter que ces règles sont compilées dans l'ordre mentionné.

3.2.4. *Étiquetage*

Dans un système typique basé sur la sélection d'unités, la sortie du frontal d'un système TTS à sélection d'unités est une séquence de vecteurs de caractéristiques linguistiques et prosodiques qui seront transmis comme entrée au bloc dorsal, qui utilise ces paramètres pour calculer le coût de jonction entre chaque unité cible et son unité candidate correspondante.

L'extraction de ce vecteur de caractéristiques intervient généralement après une phase de syllabation (dans le cas d'unités de syllabes) ou de segmentation, donnant lieu à une séquence de tokens, dont chacun sera ensuite étiqueté avec son vecteur de caractéristiques linguistiques et prosodiques correspondant. Dans cette thèse, nous faisons référence à ce concept comme étant l'*étiquetage*.

3.2.4.1 *Segmentation phonétique*

La segmentation d'un énoncé phonétique consiste à diviser le texte en une séquence d'unités phonétiques qui correspondent aux unités disponibles dans la base de données vocales. En principe, cette étape devrait être effectuée après la définition des unités de base. Néanmoins, puisque cette décision sera discutée dans le bloc DSP (Section 3.3.2), il semble nécessaire de décrire brièvement les unités de base adoptées dans ce travail, sur lesquelles la segmentation sera effectuée. Dans ce travail, deux types de segments sont privilégiés :

- L'**unité de type lemme** fait référence à la séquence de phones correspondant à un lemme dépouillé de sa dernière voyelle courte/longue (par exemple, [Dahaba] → [Dahab].

(graphème bic I.ëX signifiant "il est allé")) ou le phonème final /h/ qui correspond à l'ara-
è en cas de noms féminins (par exemple, [? imka:niyyah] → [? imka:niyy]

(éJ KA¾Ó@ , signifiant "la capacité")). Cette modification du lemme arabe de base

vise à favoriser la couverture de plusieurs mots, y compris ceux qui sont dans des formes fléchies.

- Le **segment de transition** est simplement la séquence de téléphones allant du dernier téléphone d'une unité de type lemme au premier téléphone de l'unité de type lemme suivante.

Fondamentalement, la réalisation d'une segmentation basée sur les lemmes implique l'extraction des lemmes correspondant à chaque mot du texte d'entrée à l'aide d'un outil de lemmatisation.

Pour cela, nous avons adopté le lemmatiseur AlKhalil2 (Boudchiche et al., 2017). Cette décision a été motivée par les caractéristiques attrayantes de cet outil par rapport aux autres systèmes disponibles. Nous notons surtout sa disponibilité et sa précision acceptable. Une autre caractéristique importante qui a motivé notre choix est la sortie diacritisée fournie par cet outil qui fait défaut dans les autres systèmes évalués.

La segmentation basée sur les lemmes est réalisée comme suit :

- Le lemmatiseur AlKhalil prend le texte d'entrée (en tant que séquence de graphèmes) et produit un tableau de mots et leurs étiquettes de lemme diacritisées uniques correspondantes dans les graphèmes arabes et le schéma Buckwalter. Si le système ne parvient pas à trouver le lemme d'un mot, il renvoie un null pour ce mot dans la matière.

- La sortie du lemmatiseur est d'abord convertie de Buckwalter en alphabet SAMPA. La séquence de phonèmes SAMPA qui en résulte est ensuite convertie en une séquence de téléphones sensibles au contexte à l'aide de notre module de phonétisation (section 3).

- En fonction du type de l'unité de base, la séquence phonétique de chaque lemme résultant de l'étape précédente est dépouillée de sa voyelle courte finale ou de son téléphone final /h/.

- La dernière étape avant la segmentation consiste à vérifier si le lemmalike résultant de l'étape précédente est une sous-chaîne du mot auquel il est affecté. Si c'est le cas, le module de segmentation scinde le mot en question en un lemma-like et des clitiques (par exemple, [fa?imka:niyya:tuhu] → [fa] [? imka:niyy] [a:tuhu] (éJ KA¾ÓA ¯ , qui signifie " sa capacité ")). Sinon, si le lemme n'est pas une sous-chaîne du mot, il sera ignoré par le module de segmentation.

- En outre, les mots pour lesquels le lemmatiseur n'a pas retourné de lemme (valeur nulle) sont également ignorés par le module de segmentation.

- Une fois les limites des lemma-like définies, la dernière étape de la segmentation consiste à concaténer les séquences de phonèmes s'étendant du dernier phonème d'un lemma-like au premier phonème de l'unité lemma-like suivante. Cette étape vise à définir les limites du segment de transition.

En outre, pour éviter la concaténation des unités aux points de transition entre les phones, ce qui entraîne généralement des artefacts audibles aux points de jonction, le module de segmentation phonétique divise le dernier phone de l'unité de type lemme en deux sousphones et renvoie la seconde moitié du dernier phone à l'unité suivante. Il en va de même pour le premier phonème, qui est divisé en deux sous-phones, dont la première moitié est jointe à l'unité précédente. Par exemple, la séquence de mots [xaraZati lfata:tu] (èAJ® Ë@Ik.Qk La séquence de mots [xaraZati lfata:tu] (èAJ®, signifiant "la fille est sortie") sera segmentée en [xaraZ] [Zati-lf] [fata:t] [tu].

Les différentes étapes décrites ci-dessus sont résumées dans la figure 3.6.

3.2.4.2*Extraction des vecteurs de caractéristiques*

Une fois l'énoncé segmenté, le module d'étiquetage traite chaque unité et lui attribue un vecteur caractéristique qui sera ensuite fourni en entrée du bloc DSP.

Différentes caractéristiques sont utilisées dans la littérature pour décrire la séquence d'unités cibles. Les descripteurs extraits sont généralement liés au contexte gauche et droit, au stress syllabique, à la position dans la phrase, au POS, à la modalité de la phrase, etc.

Néanmoins, d'autres travaux considèrent un ensemble de caractéristiques acoustiques et prosodiques prédites telles que le contour de la hauteur, la durée phonémique, l'enveloppe spectrale (par exemple, MFCC), etc. Ces caractéristiques sont généralement prédites par des modèles statistiques entraînés (basés sur des arbres ou des DNN). Par exemple, dans (Alías, Formiga, et Llorà, 2011), les auteurs extraient la hauteur normalisée, l'énergie et la durée. Dans le même ordre d'idées, un travail antérieur mené par Hifny et Rashwan, 2002, a présenté un réseau neuronal construit pour prédire la durée des phones arabes sur la base de la représentation phonologique de la séquence cible.

Malgré la popularité de cette méthode, appelée synthèse hybride, elle souffre encore de plusieurs limitations, et l'erreur de prédiction reste élevée, ce qui entraîne une dégradation de la qualité de la synthèse.

Dans notre implémentation, nous enrichissons les descripteurs linguistiques et phonologiques avec des caractéristiques linguistiques et prosodiques pertinentes et plus complètes. Pour chaque unité phonétique constituant l'énoncé cible, le module d'étiquetage construit un vecteur de caractéristiques en répondant aux questions présentées dans le tableau 3.5.

De plus, dans ce travail, nous ne prédisons aucune caractéristique prosodique. Celles-ci sont remplacées par une prédiction implicite basée sur les caractéristiques linguistiques capturées. En outre, nous supposons que la continuité de *F0 sera* assurée par le coût de jonction. Notre défi consiste maintenant à rechercher les

caractéristiques linguistiques qui capturent avec précision les informations contextuelles pertinentes pour la prosodie.

En effet, selon (Baloul et Baudry, 2003), il existe une forte corrélation entre le type de phrase (déclaratif, interrogatif et exclamatif) et l'intonation. Au niveau spectral, l'intonation est déterminée par le contour *F0*,
qui varie en fonction du type de peine.

Par conséquent, dans cette thèse, au lieu d'étudier la prédiction du contour *F0*, la prédiction de la prosodie est basée sur le type de phrase, en plus de la position du mot dans la phrase et de la position de l'unité dans le mot, qui s'avèrent être en corrélation avec la prosodie du discours.

En ce qui concerne la caractéristique de stress lexical, il est communément admis que le stress dépend principalement de la longueur de la syllabe et de sa position dans le mot. En ce qui nous concerne, nous faisons l'hypothèse que la prédiction de cette information n'aura pas un intérêt considérable puisque nos unités de base sont constituées de grandes unités au-delà de la longueur de la syllabe. Au contraire, l'annotation des unités avec des informations contextuelles (contexte gauche et droit) et la position de l'unité jouerait un rôle similaire à celui des caractéristiques de stress. Cette hypothèse reste néanmoins à valider empiriquement dans une étude future.

Un exemple concret de vecteur caractéristique unitaire est illustré à la figure 3.7.

3.3 traitement du signal numérique (back-end)

Dans la section précédente, nous avons présenté les différents composants de notre front-end TTS. Cette section se concentre sur le deuxième bloc du pipeline TTS : le back-end TTS.

Comme discuté précédemment, dans cette thèse, nous avons décidé d'adopter l'approche de synthèse par sélection d'unités pour construire notre bloc back-end. Ce dernier prend la séquence de vecteurs de caractéristiques - appelée ci-après séquence cible - correspondant au texte à synthétiser et déduite par le bloc frontal (bloc NLP). L'objectif est de trouver la meilleure séquence d'unités vocales correspondante - appelée **séquence candidate** - à partir d'une base de données de paroles annotées.

En général, le bloc de sélection des unités accepte deux entrées : les unités acoustiques du corpus de parole et la séquence de caractéristiques linguistiques et prosodiques fournie par le frontal TTS. Néanmoins, le corpus doit être considéré comme une brique totalement interchangeable. C'est pourquoi un corpus de test temporaire a d'abord été utilisé pour effectuer des expérimentations préliminaires pendant le développement du système TTS.

Cette décision a été motivée par le fait que la conception d'un corpus vocal de haute qualité devrait être définie sur la base des exigences de l'approche de sélection d'unités proposée. Par conséquent, il serait approprié d'utiliser un corpus de test comme première étape pour mener des expériences fonctionnelles du système développé. Par conséquent, sur la base des tests préliminaires effectués sur le système TTS développé en utilisant le corpus de test, nous définirons les exigences de notre approche de synthèse, sur lesquelles le corpus de parole final sera construit.

3.3.1 *Le corpus de test*

Construire un large inventaire d'unités de parole est une tâche laborieuse qui nécessite une procédure d'enregistrement stricte et de longues heures de sessions d'enregistrement. Par conséquent, tant que le système TTS n'est pas encore construit, un corpus définitif ne doit pas être étudié. Par conséquent, nous avons décidé d'utiliser un *livre audio* préenregistré.

Masmoo3[6] est une collection de livres audio expressifs enregistrés en prosodie naturelle par un locuteur masculin professionnel. Le *livre audio* sélectionné a un F0 moyen de près de 88 Hz sur les segments vocaux. Le corpus construit à partir du *livre audio* contient 1 148 phrases, principalement des déclarations et quelques phrases interrogatives. La durée totale de l'enregistrement est de 4 heures et 20 minutes de parole propre échantillonnée à 44kHz, en mono (1 canal). Les fichiers vocaux ont été segmentés en 98 fichiers d'une longueur moyenne de 2,5 minutes. La transcription orthographique de chaque fichier texte a été diacritisée et vérifiée par rapport à la prononciation du locuteur. Ensuite, nous avons effectué une normalisation manuelle du texte sur l'ensemble du fichier de transcription en développant les chiffres, les signes et les abréviations dans la forme textuelle. De plus, comme nous avons repéré certaines fautes d'orthographe, nous avons procédé à une vérification manuelle de l'orthographe afin d'assurer une correspondance correcte entre les enregistrements et l'orthographe du texte.

Conformément à notre approche de sélection des unités, les fichiers vocaux ont été segmentés et annotés au niveau des lemmes et des diphones. La segmentation et l'annotation ont été effectuées manuellement dans le logiciel Praat (Boersma et Weenink, 2000) ; les limites des unités de type lemme ont été placées du centre du premier phonème du lemme au centre du dernier phonème du même lemme. Le tableau 3.6 présente les principales statistiques du corpus de *livres audio* utilisé.

3.3.2. *L'unité de base*

L'un des principaux défis de la synthèse concaténative est la définition de l'unité de base. La nature et la taille de cette dernière ont un effet déterminant sur le naturel de la parole synthétisée.

Dans le chapitre 2, nous avons discuté de ce point, en montrant que les unités utilisées dans la littérature varient de petites unités (demi-phones, téléphones et diphones) à des unités plus grandes comme les triphones, les demi-syllabes, les syllabes, les mots et parfois des phrases entières pour des applications spécifiques à un domaine.

6 http://www.masmoo3.com/

En général, deux faits contradictoires guident le choix de l'unité acoustique : Un nombre limité d'unités est préférable pour réduire l'espace de recherche pendant le processus de sélection des unités ; et, d'autre part, les unités doivent être suffisamment grandes pour modéliser la dynamique de la co-articulation (Peterson, Wang, et Sivertsen, [1958]).

Ce compromis est couramment abordé avec les diphones qui permettent la couverture des effets de co-articulation avec une taille d'inventaire limitée (moins de deux heures) ; ils sont donc les plus abordables pour les langues à faibles ressources. Néanmoins, malgré l'avantage de nécessiter de petites empreintes, le diphone peut toujours hériter des inconvénients des unités courtes, qui ne permettent de couvrir que les variations locales de co-articulation. De plus, l'utilisation de diphones comme unité de base peut entraîner de nombreux points de concaténation, provoquant ainsi des artefacts audibles.

Il semble donc évident que nous ne pouvons pas faire face au problème de la discontinuité spectrale et des artefacts de concaténation à moins d'adopter des unités plus grandes. Cependant, un nombre important d'unités est nécessaire pour obtenir la couverture requise,
ce qui nécessiterait des efforts et des ressources considérables pour construire de grands corpus vocaux (jusqu'à des dizaines d'heures), en particulier pour les TTS non spécifiques à un domaine. Par conséquent, le choix de l'unité de base peut être considéré comme un compromis entre empreinte et performance.

Dans cette thèse, nous avons choisi de considérer ce compromis en utilisant des unités plus grandes et en traitant le problème de la couverture. Pour cela, nous avons étudié de manière approfondie le choix d'une grande unité qui hérite des avantages des mots en tant qu'unités vocales - qui sont connus pour donner une grande naturalité de la parole - sans exiger une quantité excessive de données. Il s'est avéré que l'unité vocale de **type lemme** répondait à cet objectif. De plus, les unités qui s'étendent d'une unité de type lemme à l'unité de type lemme suivante, à moins qu'une pause ne soit rencontrée, sont également considérées comme des unités privilégiées car elles couvrent les transitions entre les mots adjacents. Cette unité est appelée **segment de transition**.

3.3.2.1*Aborder la question de la performance*

Dans ce travail, l'unité de type lemme fait référence au lemme arabe dépouillé de sa dernière voyelle courte/longue ou de sa dernière consonne (pour les mots féminins se terminant par TaeMarbuta). Dans la langue arabe, le lemme désigne la représentation canonique réduite d'un mot avant de subir toute inflexion. Le lemme des verbes est la forme masculine singulière sans clitique. Pour les noms et les adjectifs, il correspond au

nominatif singulier masculin (ou féminin si le mot n'accepte pas le masculin) sans clitique. Dans le cas des particules, il s'agit de la particule dépouillée de ses clitiques.

Notre principale motivation pour l'utilisation de cette unité est basée sur l'hypothèse qu'elle représente le plus grand segment partagé par un large éventail de mots arabes. Par conséquent, nous supposons que l'adoption du lemma-like comme unité de base réduirait considérablement le nombre de points de concaténation nécessaires pour générer une séquence de mots donnée.

Pour approfondir cette hypothèse, nous avons mené une expérience statistique visant à obtenir le pourcentage de mots que les $10\ 000$ lemmes les plus fréquents pourraient couvrir dans un texte arabe aléatoire. Dans cette étude, tous les mots composés d'un lemme plus les clitiques sont comptés comme des instances de leur lemme.

Par conséquent, une liste de bandes de fréquences de lemmes et leur couverture dans les corpus de test de référence (≈ 4 millions de mots) a été compilée. Le processus de génération du dictionnaire de fréquences de lemmes sera décrit en détail au chapitre 5. Comme le montre le tableau 3.7, l'étude a révélé un taux de couverture élevé avec une quantité relativement faible de lemmes ; 95,54 % de la couverture des mots arabes peut être obtenue avec les $10\ 000$ lemmes les plus fréquents, et $78,86$ % avec seulement $1\ 000$ lemmes. En d'autres termes, une base de données vocales contenant seulement 1 000 lemmes peut être utilisée pour synthétiser plus de 78 % des mots arabes. Nous sommes cependant conscients que ce chiffre devrait être multiplié par au moins dix si nous prévoyons une multi-représentation des unités acoustiques dans divers contextes.

Cette conclusion tirée de cette étude a été confirmée dans un autre travail (Mohamed Attia Mohamed Elaraby, 2000), où il a été démontré que moins de mots donnent une plus grande couverture lorsqu'ils sont comptés comme lemmes. Cela suggère que la langue arabe est fortement infléchie et dérivée, et que les règles de génération de nouveaux mots à partir de mots de base (c'est-à-dire de lemmes) sont appliquées de manière extensive.

En ce qui concerne la représentation acoustique des unités de base, les différentes unités de la base de données acoustiques sont divisées du centre de leur premier téléphone au centre de leur dernier. En conséquence, l'unité de type lemme et le segment de transition sont étendus de la seconde moitié du premier phonème à la première moitié du dernier phonème de la même unité. Ceci afin de permettre une concaténation douce aux parties de l'état stable des limites des unités.

La figure 3.8 montre un exemple de séquence d'unités de type lemme **(2)** et de segments de transition **(1)** utilisée pour synthétiser la phrase présentée dans **(3)**.

3.3.2.2 *Traiter les empreintes*

Assurer la couverture complète des lemmes n'est pas une tâche triviale, voire impossible, étant donné leur nombre extrêmement important dans la langue arabe. Selon (Namly et al., 2020), le nombre de lemmes en arabe est de plus de $^{164\ 272}$, auxquels il faut ajouter les différentes variantes de chaque lemme qui diffèrent en termes de caractéristiques phonétiques et prosodiques, telles que la position dans la phrase et les paramètres acoustiques et phonétiques, entre autres. Par conséquent, nous avons décidé de cibler uniquement les lemmes arabes les plus fréquents. Cette décision implique systématiquement de fixer un seuil de fréquence (c'est-à-dire le nombre de lemmes fréquents à couvrir) à considérer pour la couverture. Cependant, cette tâche sera reportée à la phase de conception du corpus(5).

En outre, comme il n'y a aucune garantie qu'un lemme donné de la séquence cible sera disponible dans la base de données vocales, nous avons mis en place une stratégie de backoff pour compenser ces unités manquantes en utilisant des unités plus courtes. Cela peut également concerner les unités de taille non lemme si l'algorithme ne parvient pas à trouver la séquence téléphonique entière.

Le back-off sera désigné dans la section suivante comme la relaxation du filtre.

Ce processus est décrit en détail dans la section 7.3.3.

qui correspondent à l'unité phonétique et qui sont représentés dans la base de données de la parole avec plus d'une instance (A Alabbad, 2019).

De plus, la plupart des implémentations existantes qui utilisent des filtres de présélection n'utilisent pas de coût cible, qui est généralement fixé à 0. Cette décision est principalement basée sur le fait que les filtres linguistiques et phonétiques peuvent remplacer efficacement le coût cible. Néanmoins, la sélection du meilleur chemin des unités candidates sera basée uniquement sur le coût de jonction, ce qui entraînera probablement une pénurie d'unités pour l'algorithme de sélection, rendant la fonction de coût pratiquement inefficace. Par exemple, (Guennec, 2016) applique un ensemble de filtres de présélection qui incluent des caractéristiques prédites telles que le contour *F0*, en plus d'autres caractéristiques linguistiques (position dans la syllabe, position dans la phrase, etc.). Cependant, la précision des caractéristiques prédites (à savoir le contour de la F0) n'est pas garantie, ce qui entraîne le rejet d'unités qui auraient pu avoir une meilleure correspondance de la *F0* que celles sélectionnées par le filtre. De tels cas, lorsqu'ils se produisent, peuvent conduire à une dégradation drastique de la parole synthétisée.

Par conséquent, nous faisons l'hypothèse que le choix des unités en fonction de ces paramètres devrait être confié à la fonction de coût, qui pénalisera les unités en fonction des coûts de concaténation et des coûts cibles.

Dans notre système, nous avons implémenté un module de présélection qui applique des filtres pour accepter ou rejeter une unité candidate en comparant ses caractéristiques à celles de l'unité cible avant d'ajouter le nœud correspondant au

---

7. 3.3. *Présélection*

Dans une implémentation typique de sélection d'unités, au moment de la synthèse, l'algorithme de sélection récupère toutes les unités disponibles dans la base de données vocales qui correspondent à l'identité phonétique de la séquence d'unités cible. Par conséquent, la fonction de coût sera calculée pour toutes les unités avant de sélectionner le meilleur chemin. Ce processus est coûteux en termes de calcul. La complexité de la recherche augmente de façon exponentielle avec une base de données vocales contenant des unités de taille variable, ce qui rend l'espace de recherche plus grand.

Par conséquent, les tables de hachage et les filtres de présélection sont couramment mis en œuvre pour optimiser la taille du graphe de sélection et donc le temps de sélection (Beutnagel, Mohri et Riley, 1999). Une autre approche consiste à présélectionner les unités

graphe de recherche. En fait, nous considérons qu'il n'est pas nécessaire d'ajouter les nœuds qui n'ont pas un ajustement minimal.
avec la séquence cible dans le graphe.

Dans cette optique, nous avons défini quatre filtres de présélection supposés être obligatoires dans chaque unité candidate, faute de quoi elle ne sera pas intégrée au graphe. En revanche, les caractéristiques jugées plus flexibles sont laissées pour le calcul du coût cible.

Les filtres de présélection sont définis comme suit :

1. Le segment est-il un NSU ?
2. Quelle est l'identité phonétique de l'unité ?
3. Quel est le diphone suivant ?
4. Quel est le diphone précédent ?

Les filtres (1), (3) et (4) sont les plus sélectifs, c'est-à-dire que si une unité candidate de la base de données présente des valeurs différentes pour ces caractéristiques par rapport aux caractéristiques de l'unité cible, l'unité candidate est rejetée. Les filtres (3) et (4), étant les plus importants dans ce travail, seront appelés ci-après filtres *dépendants du contexte* (CD).

Pour chaque unité de la séquence cible, la fonction de présélection crée une liste de valeurs de filtres représentées par des chaînes de caractères et des valeurs binaires. Ensuite, l'algorithme recherche les valeurs des filtres correspondants dans une table de hachage contenant les unités du corpus.

Supposons qu'aucun élément n'est renvoyé, ce qui signifie qu'aucun des segments disponibles ne correspond aux filtres de l'unité. Dans ce cas, le filtre d'identité phonétique (2) est relaxé en supprimant le dernier diphone et en le rattachant à l'unité suivante. Si le diphone à supprimer contient une transition de mot marquée par "-", l'algorithme supprime le segment entier à partir du téléphone précédant la transition jusqu'à la fin de l'unité.

Par exemple, pour la séquence de l'unité cible :

si nous supposons que la deuxième unité phonétique n'est pas disponible dans le corpus vocal et qu'il faudra donc revenir à des segments plus courts, la relaxation du deuxième filtre (c'est-à-dire l'identité phonétique) aboutirait à la nouvelle séquence d'unités suivante :

[? *iStar*][**rat**][**t-D**][*Dahab*][*ban*]]

Cette décision était basée sur le fait qu'une transition de mot représentée par "-" est considérée comme un segment atomique ; elle ne doit donc pas être segmentée.

L'exemple ci-dessus montre que le processus de relaxation déclenche une mise à jour de la séquence cible avec la nouvelle séquence d'unités et les étiquettes correspondantes (caractéristiques concernées). La mise à jour affecte principalement l'unité courante soumise à la relaxation et les deux unités suivantes. Toutes les étiquettes attachées à chaque unité sont mises à jour, à l'exception des caractéristiques de niveau phrase. L'algorithme réinitialise les filtres de présélection des unités sur la base des étiquettes mises à jour avant de relancer la recherche de la nouvelle unité phonétique compte tenu de ses filtres.

Ce processus de relaxation suivi de la recherche est répété jusqu'à ce que le nombre de candidats récupérés pour chaque unité cible soit supérieur à 15, ou si la longueur de l'unité phonétique courante soumise à la relaxation est égale à 2 (diphone).

L'algorithme de présélection insère les unités candidates correspondantes dans le graphe de recherche, que l'algorithme de sélection parcourra pour trouver le meilleur chemin en fonction des fonctions de coût.

Ce mécanisme permet de trouver un chemin dans tous les cas avec un espace de recherche réduit et un temps de sélection réduit.

3.[8].4 *Algorithme de sélection*

Après que la fonction de présélection ait effectué une première sélection d'unités candidates sur la base de leur identité phonétique et de leur contexte, donnant lieu à plusieurs unités candidates pour chaque unité cible, la tâche suivante consiste à trouver le chemin le moins cher représentant la meilleure séquence d'unités correspondante parmi les candidats initialement sélectionnés.

Comme indiqué dans l'état de l'art (chapitre 2), le calcul de la meilleure séquence est classé comme un problème de recherche de chemin dans un graphe dirigé pondéré (Guennec et Lolive, 2014). Par conséquent, l'algorithme de Viterbi (Viterbi, 1967) (et ses

---

8. [3 5]. *Coût cible*

Alors que la fonction de présélection sélectionne les unités correspondantes dans la base de données sur la base d'un ensemble minimal de filtres, le coût cible doit utiliser des caractéristiques plus raffinées pour noter les nœuds (unités candidates) en fonction de leur niveau de similarité avec les caractéristiques de l'unité cible.

dérivés), Bellman-Ford, Dijkstra et *A ?* (Hart, Nilsson et Raphael, 1968) sont généralement adaptés à cette tâche.

Le premier est l'algorithme le plus largement utilisé. Cependant, tant que nous incluons une phase de présélection conduisant à un espace de recherche réduit, le choix de l'algorithme n'aura pas un effet considérable sur les performances du système. Dans ce but, l'algorithme de Viterbi a été implémenté.

Les unités candidates sont organisées dans un graphe $G = (V ; A ; C)$ dirigé et ordonné dans le même ordre que la séquence phonétique cible. V représente les nœuds correspondant à l'ensemble des unités candidates liées par un ensemble d'arcs A. Chaque arc représente une concaténation possible entre deux unités. L'inadéquation entre chaque paire d'unités est quantifiée par le coût *C* lié à chaque arc, appelé dans ce travail le coût de jonction. Un coût de jonction élevé peut être considéré comme un risque plus élevé de créer des artefacts audibles si la paire d'unités est concaténée.

Chaque nœud du graphe est associé à un vecteur de caractéristiques correspondant à une unité candidate. Tous les vecteurs de caractéristiques sont pré-calculés pendant la construction du corpus et stockés dans un fichier texte. Deux classes de caractéristiques sont définies : acoustique et linguistique. La dernière est utilisée pour calculer le coût cible de chaque unité candidate par rapport à l'unité cible. Tandis que la première est utilisée pour calculer les coûts de jonction. Il est à noter que le vecteur de caractéristiques linguistiques associé à chaque unité candidate doit avoir le même format que le vecteur de caractéristiques attribué aux unités cibles. Par conséquent, les caractéristiques linguistiques décrites au point 3.2.4.2 sont les mêmes que celles utilisées pour annoter les unités vocales du corpus. Les détails concernant les caractéristiques des unités du corpus seront décrits au chapitre 5.

Rappelons que les unités candidates en la matière sont celles sélectionnées par la fonction de présélection - c'est-à-dire que la recherche ne sera effectuée que sur un graphe d'unités présélectionnées sans avoir à récupérer d'autres unités dans la base de données.

Comme décrit au chapitre 2, tout au long de la recherche, des sous-couts cibles sont attribués à chaque nœud. Un sous-coût de *zéro* lié à une caractéristique *j* donnée signifie que l'unité candidate et l'unité cible ont exactement la même valeur pour *j*.

Les caractéristiques linguistiques et acoustiques extraites ou prédites en amont sont généralement utilisées pour calculer les sous-couts cibles (section 3.2.4.2).

Néanmoins, l'ensemble complet des caractéristiques est rarement fourni dans la littérature, et la pertinence de ces caractéristiques pour le calcul du coût cible est à peine abordée, en particulier pour l'arabe, car nous supposons que ces caractéristiques

sont indépendantes de la langue. Par exemple, (Black et Campbell, 1995) ont noté que le coût cible est calculé à l'aide de 30 sous-couts sans fournir de détails.

Dans ce travail, les sous-couts cibles sont calculés sur la base des caractéristiques définies en 3.2.4.2 après avoir exclu celles utilisées par la fonction de présélection, à savoir l'identité phonétique, la NSU, et les diphones gauche et droit. La décision d'utiliser l'ensemble des caractéristiques définies s'est appuyée sur une batterie de tests. En effet, pour n'inclure que les descripteurs les plus pertinents qui ont une contribution considérable à la qualité de la parole synthétisée, nous avons commencé par un ensemble complet de caractéristiques collectées dans la littérature, et nous avons progressivement retiré les caractéristiques une par une de la fonction de coût cible. À chaque étape, nous avons évalué la qualité de la parole résultante, ce qui nous a permis de décider si une caractéristique donnée devait être conservée ou rejetée. L'ensemble des caractéristiques que nous avons jugées pertinentes pour notre système sont celles présentées dans le tableau 3.5.

Pour chaque nœud du graphe, les fonctions de distance $C^{(t)}_j(_{ti}, ui)$ sont calculées entre chaque caractéristique j d'une unité candidate ui et l'unité cible $_{ti}$. La somme pondérée des sous-couts donne le coût unitaire cible $C^{(t)}(_{ti}, ui)$:

$$C^{(t)}(_{ti}, ui) = \sum_{j=1}^{J} w_j^{(t)} C_j^{(t)}(_{ti}, ui) \tag{3.1}$$

Une autre question couramment abordée lors de la définition des coûts cibles est la pondération des caractéristiques. En effet, certaines caractéristiques ont un effet plus fort que d'autres sur la qualité de la synthèse. Par conséquent, un ensemble de poids, formés ou ajustés manuellement à l'aide de connaissances d'experts, est utilisé pour donner la priorité à certaines caractéristiques par rapport à d'autres. Ces poids sont principalement utilisés avec la formulation des caractéristiques indépendantes (IFF) telle que définie dans le Festival Multisyn (Clark, Richmond et King, 2007), qui calcule le coût cible en additionnant les sous-couts binaires d'une unité candidate selon que les caractéristiques candidates correspondent ou non exactement aux caractéristiques de l'unité cible. Puisque cette formulation la plus simple n'inclut pas de concept de "correspondance proche", il est nécessaire d'utiliser des poids qui quantifient la pertinence de chaque caractéristique.

Néanmoins, dans notre travail, comme le coût cible est calculé sur la base des distances numériques entre les caractéristiques, tous les poids ($w_{(jt)}$ dans l'équation 3.1)

sont fixés à 1. De plus, nous supposons que toutes les caractéristiques utilisées pour le calcul du coût cible ont la même pertinence.

3.3.6. *Coût de l'adhésion*

Le coût de jonction vise à empêcher la concaténation d'unités susceptibles de provoquer des artefacts audibles aux points de concaténation en attribuant des coûts élevés aux unités candidates présentant un décalage important avec l'unité cible. Ce coût est ajouté au coût cible sur la base duquel le choix de la séquence d'unités la moins coûteuse est effectué.

De nombreux travaux ont démontré que l'utilisation des seuls coûts de jonction sans calcul des coûts cibles permet généralement d'obtenir une qualité de parole acceptable. En revanche, la sélection basée sur les coûts cibles sans tenir compte des coûts de jonction conduit souvent à une parole inintelligible.

Par conséquent, la définition des informations pertinentes pour le calcul du coût de jonction a fait l'objet d'une grande attention dans la littérature. Comme indiqué dans de nombreux travaux, la distance la plus importante est *F0* (Black et Campbell, 1995). Outre la distance euclidienne des MFCC autour du point de jonction, les distances cepstrales, en particulier le LPC (Linear Prediction Coefficient), sont également étudiées dans de nombreux travaux (Alías, Formiga et Llorà, 2011).

Pour la synthèse arabe, le *F0*, l'intensité et le MFCC sont les plus couramment étudiés A Alabbad, 2019 ; Abdelmalek et Mnasri, 2016.

Dans notre travail, nous avons adopté quatre sous-coûts :

- F0 moyen,
- F0 aux limites de l'unité,
- L'amplitude,
- La durée du demi-téléphone aux limites de l'unité.

Ces caractéristiques ont été choisies car elles sont bien évaluées dans la littérature. Nous supposons que ces caractéristiques sont précalculées pour toutes les unités du corpus. Par conséquent, nous calculons la distance entre chaque unité et les sous-couts des unités suivantes. En utilisant la formulation introduite au chapitre 2, le coût de jointure assigné à chaque arc joignant deux unités candidates ultérieures est formulé comme suit :

$$C^{(j)}(ui\text{-}1, ui) = Camp(ui\text{-}1, ui)+\mathrm{CMF0}(ui\text{-}1, ui)+\mathrm{CBF0}(ui\text{-}1, ui)+\mathrm{CPHdur}(ui\text{-}1, ui),$$

(3.2)

où $C_{amp}(ui\text{-}1, ui)$, $C_{MF0}(ui\text{-}1, ui)$, $C_{BF0}(ui\text{-}1, ui)$, et $CMPHdur(ui\text{-}1, ui)$ correspond respectivement à l'amplitude, à la F0 moyenne, à la F0 aux limites de l'unité et à la durée du demi-téléphone.

Néanmoins, ceux-ci sont encore loin d'être représentatifs des paramètres perceptifs de la parole, et des artefacts audibles ont encore été remarqués après avoir effectué des tests préliminaires avec les coûts définis. Par conséquent, pour tenter de localiser et d'identifier la source des artefacts, nous avons analysé un certain nombre de phrases qui comportaient des artefacts audibles. Les résultats ont montré que la plupart des artefacts se produisent aux points de concaténation entre une paire de demi-voyelles, en particulier la voyelle courte /a/, qui était fortement influencée par les phones voisins. Ces derniers sont principalement responsables de la modification de l'accentuation et de la durée de la voyelle.

Sur la base de ces faits, il semble que la capture du contexte phonétique droit et gauche de la voyelle ne soit pas suffisante. De nombreux auteurs ont débattu de cette question concernant la propagation de l'accentuation, menant à la conclusion que nous ne pouvons pas minimiser la distorsion entre deux demi-voyelles concaténées à moins qu'elles ne soient extraites de la même séquence (constituée d'au moins un mot). Ceci est cependant peu pratique, voire impossible.

Une autre solution plus optimale consiste à pénaliser les unités en fonction de la nature du dernier phonème (Guennec, 2016). Par conséquent, nous avons empiriquement fixé une pénalité élevée pour les demi-voyelles /a/ et une pénalité moins importante pour les autres demi-voyelles /u/ et /i/. Quant aux autres phones, leurs pénalités sont toutes fixées à zéro. Par conséquent, la nouvelle fonction de coût de jonction sera la suivante

$$C0^{(j)}(ui\text{-}1, ui) = C^{(j)}(ui\text{-}1, ui) + P(ph), \quad (9.3)$$

où $C^{(j)}(ui\text{-}1, ui)$ est le coût de jonction formulé avec l'équation 3.2 et P(ph) est la pénalité du dernier demi-téléphone de la première unité *ui-1*, qui est la même que le demi-téléphone au début de l'unité *ui*.

Le problème consistant à trouver la meilleure séquence d'unités qui minimise la somme des coûts totaux des unités candidates peut être formulé comme suit :

$$U = \underset{u=u1,\ldots,un}{\arg\min}\left(\sum_{i=1}^{n} C^{(t)}(_{ti}, ui) + \sum_{i=2}^{n} C0^{(j)}(ui\text{-}1, ui)\right) \quad (3.4)$$

Il est à noter que le calcul du coût de jonction $C0^{(}j)$ est calculé uniquement pour les unités de *u2* à $_{un}$.

La figure 3.9 montre un exemple de graphe généré pour la séquence cible *[? iStarat]*, qui est formée par le lemme *[? iStar]* et le segment de transition *[rat]*. Les unités plus courtes sont récupérées par le module de présélection, qui recule vers des segments alternatifs si le seuil n'est pas atteint. Le symbole "#" indique une étiquette de silence. Nous définissons un nœud de début "Init" et un nœud de fin "End" pour éviter les choix arbitraires des premières et dernières unités. Le nœud de départ est lié à toutes les unités candidates qui correspondent au début de la séquence cible. Il en va de même pour le nœud de fin, qui est lié aux unités candidates qui correspondent à la fin de la séquence cible.

3.3.8 *Concaténation d'unités*

La sortie de la fonction de sélection des unités est le chemin de moindre coût correspondant à une séquence d'unités candidates qui correspond le mieux à une séquence cible délimitée par deux limites de pause <LP>.

---

9. 3.7 *Coût total*

Comme indiqué au point 2, le coût total de chaque nœud est la somme de son coût de jonction $C0^{(j)}(ui\text{-}1, ui)$ et du coût cible $C^{(t)}(_{ti}, ui)$. $_{ti}$ étant la ième unité de la séquence cible $T = t1, \ldots, _{ti}$.

Dans la mesure où nous considérons le corpus vocal comme une bande de parole entière plutôt que comme un dictionnaire d'unités de parole pré-segmentées, la séquence des unités sélectionnées n'est extraite qu'au moment de la synthèse.

Par conséquent, la dernière tâche avant la concaténation des signaux consiste à récupérer les heures de début et de fin de chaque unité dans la base de données vocales en fonction de son identifiant unique. Ce dernier est indexé dans un fichier d'annotation où toutes les unités de la base de données sont répertoriées avec leurs informations linguistiques et acoustiques. La description complète du format du corpus sera présentée au chapitre 5.

La fonction de concaténation utilise les temps de début et de fin pour extraire l'unité d'un long fichier WAV en utilisant la boîte à outils Praat. Pour les unités consécutives (ayant des heures de début et de fin), la fonction de concaténation ne récupère que l'heure de début de la première unité et l'heure de fin de la dernière unité contiguë, évitant ainsi les concaténations inutiles entre unités adjacentes.

En outre, certains travaux incluent une phase de post-traitement du signal qui effectue un ensemble de modifications prosodiques sur la chaîne résultante (Eide et al., 2003) ou le lissage des joints entre les unités concaténées extraites de différents contextes. Il est cependant avancé que ce traitement du signal conduit généralement à une dégradation significative de la parole résultante, comme indiqué dans l'état de l'art. Par conséquent, après avoir effectué des tests d'écoute préliminaires sur la parole synthétisée produite par notre système TTS, nous avons décidé de sauter la phase de prétraitement du signal car la parole résultante sans aucune modification du signal est déjà jugée acceptable.

3.4conclusion

Dans ce chapitre, nous avons présenté les différents composants de notre système TTS.
Nous avons présenté le bloc NLP, à travers lequel nous avons discuté de la pertinence des outils de diacritisation les plus populaires en tenant compte des exigences de l'arabe TTS. En outre, nous avons suggéré une approche pour prédire l'emplacement des groupes de souffle sur la base d'un analyseur syntaxique de dépendance, ainsi qu'un ensemble complet de règles phonologiques qui ont été discutées en détail en fonction de leur importance. Les règles sélectionnées ont été utilisées pour mettre en œuvre un convertisseur G2P. Pour préparer la séquence cible pour le bloc DSP, nous avons défini un ensemble de caractéristiques pertinentes utilisées pour étiqueter la séquence phonétique cible avec des informations linguistiques et prosodiques implicites.

Dans le bloc DSP, nous avons proposé une nouvelle unité de base pour la sélection des unités, que nous avons définie comme "lemma-like". Cette unité a été adoptée car elle représente le compromis idéal entre les performances et les empreintes, puisque nous ne ciblons que les plus fréquentes, qui se sont avérées couvrir un large éventail de mots arabes.

Un mécanisme de retour en arrière a également été décrit pour traiter les unités manquantes dans la base de données.

3.4 conclusion

Figure 3.10 : Vue du flux de travail du système TTS développé.

Comme la recherche d'unités de taille variable dans un grand corpus est coûteuse en calcul, nous avons défini des filtres de présélection pertinents qui élaguent les unités candidates en fonction de leur correspondance minimale mais essentielle avec le contexte de la séquence cible (diphones environnants) avant de les ajouter au graphe de recherche. De plus, les fonctions de coût de la cible et de la jointure ont été décrites. Un accent particulier a été mis sur la pénalisation des phonèmes qui sont plus susceptibles de produire des artefacts de concaténation.