Interface homme machine

Améliorer l'interface homme machine à travers la reconnaissance vocale

Réalise par

Yahya Akli

INDEX

A. Représentation générale de l'étude

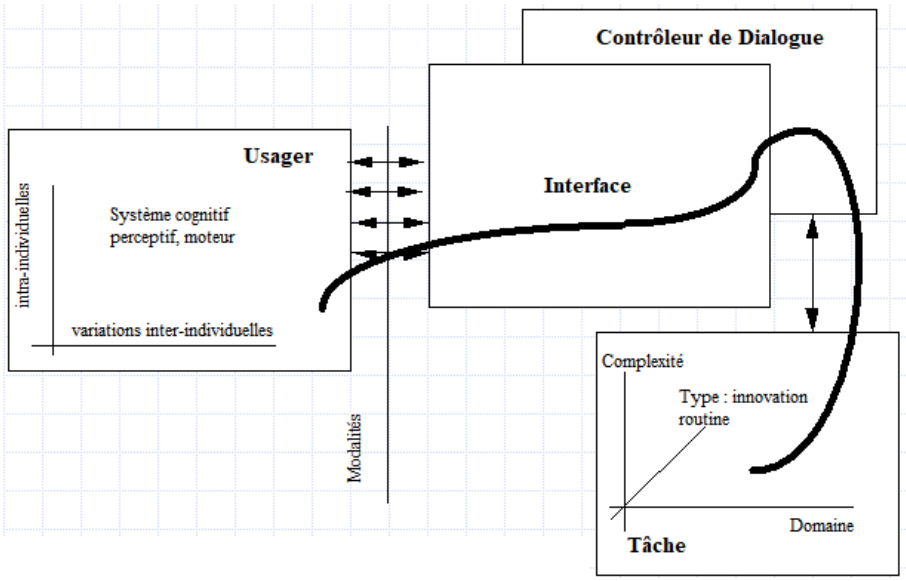
Aussi vielle que l'informatique, l'histoire de l'interaction homme-machine est marquée d'une péripétie d'évènements résultants de la volonté d'optimiser et faciliter l'utilisation de ses interfaces.

En effet, le domaine d'interactions homme-machine implique la conception et le développement des systèmes interactifs, médiatrices de la communication entre l'homme et la machine par le biais de différents modes d'interaction :

Mode parlé : commandes vocales, guides vocaux…
Mode écrit : entrées par le clavier et la tablette graphique, affichage du texte sur l'écran…
Mode gestuel : désignation 2D ou 3D (***souris***, ***gants de données*** , *écran tactile),* retour d'effort…
Mode visuel : graphiques, images, animations ….

[1]

Fig. 1 - Mode de fonctionnement de l'IHM



B. Problématique

Ce travail s'inscrit dans le domaine de l'ingénierie de l'interaction homme-machine (IHM). Il s'intéresse à l'adaptation des interfaces homme-machine à l'usage. Le développement de ces dernières n'est plus en correspondance avec l'évolution à cause de la complexité et les faiblesses émanant de l'usage de quelques modes d'interaction (Ex: mode d'écriture). Certains modes sont plus adaptables au contexte d'usage que d'autres ne le sont. L'exploitation du mode parlé et de l'outil de la reconnaissance vocale est susceptible de mieux participer au développement de cette interaction. Chose qui met en évidence la question de recherche suivante : comment accompagner l'amélioration du mode parlé pour maximiser le bénéfice de l'adaptation de l'usage pour une interaction homme-machine plus facile (naturelle) ainsi que poursuivre le développement ?

C – La reconnaissance automatique de la parole

Rassemblant les efforts de recherches de spécialistes de nombreuses disciplines (linguistes, informaticiens, logiciens, psychologues, traducteurs…), le traitement automatique des langues est un vaste domaine et champs de manœuvre économiquement porteur, dont les applications sont multiples dans divers secteurs

tels que la bureautique, l'aide aux handicapés, l'enseignement, la domotique, la traduction, l'aide à la navigation, la documentation et bien d'autres … Le traitement automatique des langues repose sur son outil majeur qui est la reconnaissance automatique de la parole, sujet de notre recherche. La reconnaissance automatique de la parole (souvent improprement appelée reconnaissance vocale) est une technique informatique qui permet d'analyser la voix humaine captée au moyen d'un microphone pour la transcrire sous la forme d'un texte exploitable par une machine. Cet outil accompagné de la synthèse de la parole ainsi que de l'identification du locuteur ou la vérification du locuteur, sont comptés parmi les techniques de traitement de la parole. Ces techniques permettent notamment, la réalisation des interfaces homme-machine (IHM) où d'une partie de l'interaction se fait à la voix : « interfaces vocales ».
[2]

L'activité de recherche sur la reconnaissance vocale a débuté en 1950, avec le développement des Laboratoires Bell basés sur les progrès qu'a connu le domaine de la phonétique. Le VODER est un exemple de ces évolutions. Encore, en 1952, un système qui pouvait reconnaître des chiffres uniques à partir d'un seul locuteur a fait son apparition. D'autres inventions plus performantes et mieux avancées ont suivi, provenant d'institutions telles que l'Institut de technologie du Massachusetts (MIT), les Laboratoires RCA ou l'Université de Kyoto, qui ont conçu des systèmes reconnaissant des unités basiques du discours jusqu'au plus complet des travaux [3]

Selon le BCC Research, la situation actuelle du marché de la reconnaissance vocale a atteint 47 milliards de dollars en 2011, 53 milliards de dollars en 2012 et 113 milliards de dollars en 2017, soit un taux de croissance annuel cumulé de 16,2%. La répartition entre les sections de consommateurs, d'entreprises et de soins de santé est exprimée par la figure 1, d'où il est clair que le marché de la consommation est en nette voie de développement, du temps qu'aucun changement significatif n'est prévu dans le domaine médical. En effet, ceci reflète l'intérêt croissant des utilisateurs finaux à l'égard de l'utilisation des applications vocales dans leurs appareils intelligents, chose qui favorise le développement de tels programmes à la fois pour les consommateurs et les entreprises. [4]

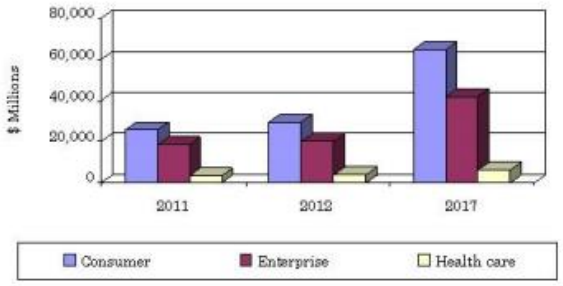


Fig.2 – la distribution du marché de la reconnaissance vocale en 2011, 2012 et 2017

D – Etude des modes de fonctionnement de la reconnaissance automatique de la parole

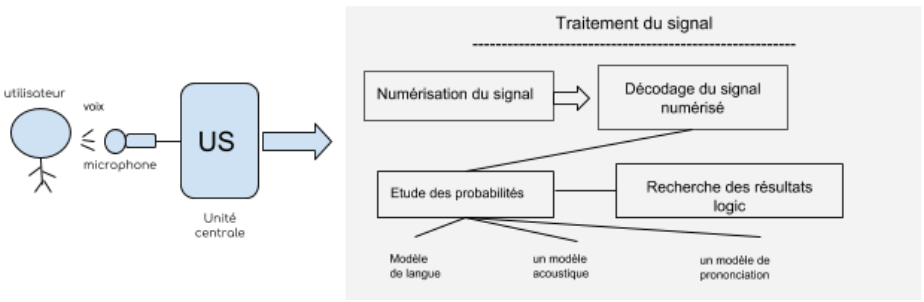


Fig.3 chaine de fonctionnement du système de reconnaissance vocale

La reconnaissance de la parole, vue comme un problème de la théorie de la communication, a pour but de reconstruire un message m à partir d'une séquence d'observations y. Le message de parole passe de la forme acoustique à la forme électrique par l'emploi d'un microphone, pour être digitalisé et sauvegardé sur un support informatique noté par s. Ensuite, s est subdivisé en unités de temps (typiquement de 10ms chacune). De ces unités de temps, on extrait un certain nombre de paramètres (au moyen de techniques de traitement du signal)

L'étape de la reconnaissance consiste donc à déterminer la suite de mots m qui maximise le produit des deux termes P(m) et P(y/m). Le premier terme représente la probabilité a priori d'observer la suite de mots m indépendamment du signal. Cette probabilité est déterminée par le modèle de langage. Le deuxième terme indique la probabilité d'observer la séquence de vecteurs acoustiques y sachant une séquence de mots spécifiques. Cette probabilité est estimée par le modèle acoustique.

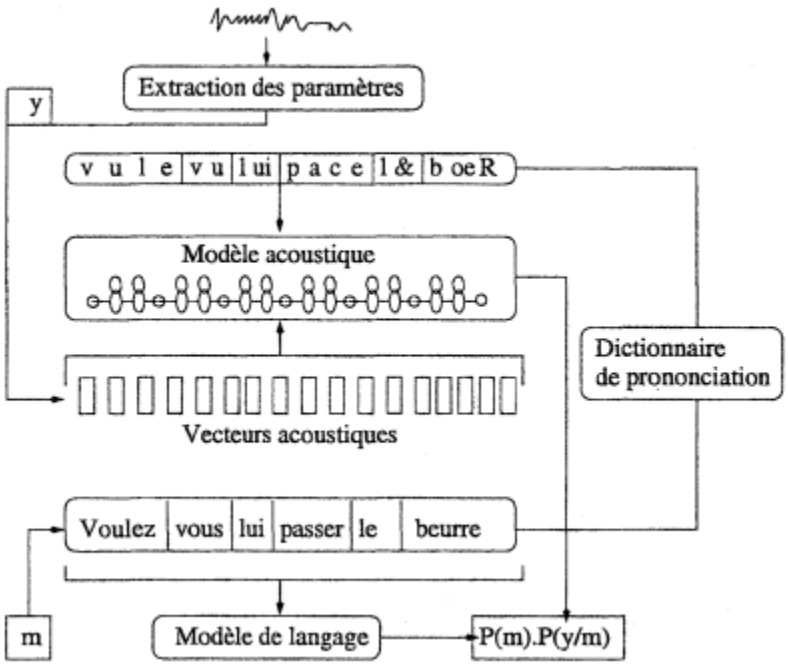


Fig.4 - L'approche probabiliste de la RAP pour l'hypothèse (Voulez-vous lui passer le beurre)

Les modèle de langue

L'objectif des modèles de langage probabilistes est d'estimer la probabilité $P(W_1^T)$ d'une séquence de mots $W_1^T = \omega_1, \omega_2, \ldots, \omega_T$.

Les modèles n-grammes

Ils reposent sur l'application de la règle des probabilités conditionnelles, définissent la vraisemblance d'une suite de mots $W_1^T = \omega_1, \omega_2, \ldots, \omega_T$ comme suit:

$$P(w_1, w_2, \ldots w_N) = \prod_{i=1}^{N} p(w_i / w_1, \ldots w_{i-1})$$

où $p(w_i/w_1, \ldots w_{i-1})$ est la probabilité du $i^{ème}$ mot de la suite W_1^N, sachant tous les mots précédemment émis.

Dans la version la plus simple du modèle, la probabilité du mot Wi sachant son contexte $\omega_{i-n+1}, \ldots \omega_{i-1}$ est estimée suivant le principe du maximum de vraisemblance sur un corpus d'apprentissage W représentant la langue:

$$p(w_i / w_{i-n+1}, \ldots w_{i-1}) = \frac{N(w_{i-n+1}, \ldots w_{i-1}, w_i)}{\sum_{j \in W} N(w_{i-n+1}, \ldots w_{i-1}, w_j)}$$

où $N(.)$ est le nombre d'occurrences de la suite de mots en argument dans le corpus W

Les modèles n-classes

L'espace de paramètres requis par les modèles n-grammes peut être sensiblement réduit, et par conséquent la fiabilité des évaluations peut être améliorée, en regroupant les mots dans des classes.

La probabilité d'apparition d'un mot ω_i à la suite d'un historique dépend de la classe C(ω_i) à la quelle appartient ce mot ω_i, et de la probabilité d'apparition de cette classe C(ω_i) à la suite de l'historique de mots(exprimé par la suite des classes auxquelles appartiennent les mots de l'historique) [5 ,6]. Si n suppose qu'un mot ne peut appartenir qu'à une seule classe, la probabilité d'un mot sachant son contexte est définie comme suit

$$p(w_i/w_{i-n+1},\dots w_{i-1}) = p(w_i/C(w_i))p(C(w_i)/C(w_{i-n+1}),\dots C(w_{i-1}))$$

où $C(.)$ est la classe à laquelle appartient le mot en argument.

La probabilité d'appartenance d'un mot à une classe, p($\omega_i/C(\omega_i)$), est calculée comme suit :

La probabilité d'appartenance d'un mot à une classe, $p(w_i/C(w_i))$, est calculée comme suit :

$$p(w_i/C(w_i)) = \frac{N(w_i)}{N(C(w_i))}$$

où $N(.)$ est le nombre d'occurrences de l'argument dans le corpus.

Les modèles POS

Ces modèles, qui se fondent sur une partition de parole ("Part of Speech"), ont été introduits pour les langages fortement flexionnels, comme le français, l'allemand ou l'italien. Les modèles POS8 sont inspirés de la connaissance syntaxique [7]. Ainsi, un mot donné peut appartenir à différentes classes (POS) à des instants différents, par exemple, le mot "part" peut être un nom ou un verbe. Par définition, chaque occurrence d'un mot doit avoir une seule classe à un instant donné. En pratique, il n'est pas aisé d'extraire la vraie classe d'un mot dans un contexte, parmi l'ensemble des classes associées à ce mot. Ainsi, pour estimer la probabilité d'un mot ω_t dans un contexte h, le modèle POS calcule la somme des probabilités n-classes sur toutes les classes associées au mot à prédire ω_t. Un modèle POS tri grammes (3-grammes), estimant la probabilité d'un mot ω_t sachant un historique de classes, est généralement défini comme suit :

$$p(w_t/g_{t-2},g_{t-1}) \approx \sum_{g_t \in G_{w_t}} p(w_t/g_t)p(g_t/g_{t-2},g_{t-1})$$

où $g(w_t) = g_t$ est la classe (POS) du mot w_t au temps t et G_{w_t} est l'ensemble de toutes les classes associées au mot w_t, $p(w_t/g_t)$ est la probabilité d'appartenance du mot w_t à la classe g_t (peut être estimée par la formule 3.5) et $p(g_t/g_{t-2},g_{t-1})$ est la probabilité de la suite de classes g_{t-2},g_{t-1},g_t qui peut être estimée par la formule

$$p(w_i/w_{i-n+1},\dots w_{i-1}) = \frac{N(w_{i-n+1},\dots w_{i-1},w_i)}{\sum_{j\in W} N(w_{i-n+1},\dots w_{i-1},w_j)}$$

Une variante de ce modèle présentée dans [151] suppose que l'historique de classes est non fixe. Par conséquent, le modèle fait la somme sur les probabilités n-classes et sur tous les historiques de classes possibles. En gardant les mêmes notations utilisées dans la formule 3.6, un modèle POS bigrammes (2-grammes) estime donc la probabilité d'un mot ω_t sachant un historique ω_{t-1} comme suit :

$$\begin{aligned} p(w_t/w_{t-1}) &= \sum_{g_t \in G_{w_t}} p(w_t/g_t)p(g_t/w_{t-1}) \\ &= \sum_{g_t \in G_{w_t}} p(w_t/g_t) \sum_{g_{t-1} \in G_{w_{t-1}}} p(g_t/g_{t-1})p(g_{t-1}/w_{t-1}) \end{aligned}$$

avec

$$p(g_{t-1}/w_{t-1}) = \frac{N(w_{t-1},g_{t-1})}{N(w_{t-1})},$$

où $N(.)$ est l'occurrence de la suite de mots en argument.

Afin de modéliser cette approche probabiliste, on utilise le langage de programmation *python* pour modéliser le fonctionnement, dans le domaine pratique, de la RAP.
pour cette modélisation, on utilise **Google Web Speech API,** et le module *python* **speech_recognition**

```
import speech_recognition as sr

# create a recognizer
r = sr.Recognizer()
mic = sr.Microphone()

L=''
while L=='':
    with mic as source:
        audio = r.listen(source)
    try:
        L=r.recognize_google(audio)
    except sr.UnknownValueError:
        print('Sorry, could you repeat your message?')
print(L)
```

On utilise par suite, la commande **r.recognize_google(audio, show_all=True)** qui renvoie les possibilité prise par le système afin d'aboutir au message finale.
On lance le programme python, puis on dit la phrase suivante :

Input >> '*one of the joys of being a geologist is fieldwork*'

Le résultat de cette expérience :

Output >> '*one of the joys of being a geologist is fieldwork*'

On appelle la commande **r.recognize_google(audio, show_all=True),** le système renvoie

```
>>> r.recognize_google(audio, show_all=True)

{'alternative': [{'transcript': 'one of the joys of being a geologist is fieldwork', 'confidence': 0.87437057}, {'transcript': 'one of the joys of being a geologist his field work'}, {'transcript': 'one of the joys of being a geologist his fieldwork'}, {'transcript': 'one of the joys of being a geologist is field work'}, {'transcript': 'one of the joys of being a geologist use fieldwork'}], 'final': True}
```

Le système choisit le résultat le plus probable 'confidence': 0.87437057, puis l'affiche sur écran.

E – Modélisation et simulation

Description

Pour modéliser l'efficacité de la communication avec la machine à travers la reconnaissance vocale, j'ai programmé le système ROBODOC qui interagit avec l'utilisateur d'une manière délicate, afin de repérer sa maladie probable, en lui présentant une liste de symptômes. L'utilisateur choisit une possibilité à chaque essai, et avec chaque réponse le système fait ses calcules pour trouver la maladie la plus probable.

Etude du système

(Ce programme est basic, il contient des informations sur 7 maladies seulement, il sert que pour modéliser la facilité de la communication avec la machine à travers la reconnaissance vocale)

Le système ROBODOC fonctionne suivant le processus suivant :
Le système repère les données sur les maladies d'après une base de données SQL ' **Myhealth.db'**

```
import sqlite3
conn = sqlite3.connect('Myhealth.db')
c = conn.cursor()
```

la base de donnée est constitué d'une table ' **disease** ' qui contient les symptômes des maladies suivantes

```
disease=['cancer','migraine','depression','mauxestomac','allergie','colonirritable','maladiecardiovasculaire']
```

TABLE DISEASE

	cancer	migraine	depression	nauxestomac	allergie	colonirritable	maladiecardiovasculaire
	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	fatigue	fatigue	tristesse	vomissem...	oeil rouge	constipation	malaise dans la poitrine
2	vomissement	mal à la tête	perte d'intérêt	fatigue	vomissement	crampes au ventre	vertiges
3	difficulté à rés...	vomissement	baisse de libido	ballonnem...	larmoiement	diarrhée	vomissements
4	troubles de la...	sensibilté à la...	fatigue	crampes a...	difficultés à respirer	fatigue	difficulté à réspirer
5	douleur	sueurs froides	insomnie	crampes a...	douleurs abdominal...	besoin urgent d'aller...	manque d'énergie
6	perte d'appétit	troubles de vi...	troubles digestif	douleur au...	peau rouge	des flatulences	rythme cardiaque plus rapide
7	constipation	sensibilité au ...	perte d'appétit	brûlures d'...	peau séche	trouble digestif	palpitation cardiaque
8	trouble digestif	sensation d'él...	trouble de co...	troubles di...	constipation	douleur	fatigue

Pour classifier les données du tableau dans des listes python, on utilise les instructions suivantes

```
def transformer(tup):
        return [tup[i][0] for i in range(len(tup))]

def frombase(ch):
        c.execute('SELECT '+ch+' FROM disease')
        tab=c.fetchall()
        pad=transformer(tab)
        return pad
cancer = frombase('cancer')
migraine = frombase('migraine')
depression = frombase('depression')
maux_d_estomac = frombase('mauxestomac')
allergie = frombase('allergie')
colon_irritable = frombase('colonirritable')
maladie_cardiovasculaire = frombase('maladiecardiovasculaire')
```

La fonction **'transformer'** sert à transformer une liste de tuples en une liste simple
La fonction **'frombase'** sert à repérer la colonne du symptôme choisit

On combine ses deux fonctions pour extraire les données sur chaque maladie de la table **'disease'**
On utilise le module python ***speech_recognition*** pour recevoir les messages de l'utilisateur, ainsi que **Google Web Speech API**

```
import speech_recognition as sr

r = sr.Recognizer()
mic = sr.Microphone()
```

Le fonctionnement de ce système consiste à proposer successivement 3 symptôme à l'utilisateur, dont il choisira un. A chaque choix, le système rajoute +1 au compteur de la maladie correspondante. Finalement le système propose une maladie en se basant sur le compteur le plus grand à la fin de l'essai.
Pour des propositions aléatoires on utilise le module python **random**

Et pour envoyer des messages vocaux aux utilisateurs, on utilise les modules *pygame* et *gtts*

```
import random
import time
from pygame import mixer
from gtts import gTTS

symptome=list()


L = cancer + migraine + depression + maux_d_estomac + allergie + colon_irritable +
maladie_cardiovasculaire
for i in range(len(L)):
        if L[i] not in symptome:
                symptome+= [L[i]]
# create counter
acan = 0
bmig = 0
cdep = 0
dmaux = 0
eall = 0
fcol = 0
gcard = 0

while acan < 4 and bmig < 4 and cdep < 4 and dmaux < 4 and eall < 4 and fcol < 4 and gcard < 4:
        S = symptome
        ab = random.choice(S)
        S=S[:S.index(ab)] + S[S.index(ab)+1:]
        bb = random.choice(S)
        S=S[:S.index(bb)] + S[S.index(bb)+1:]
        cb = random.choice(S)
        print ('============')
        print (' ressens-tu? ')
        print(ab)
        myobj = gTTS(text=ab, lang=language, slow=False)
        myobj.save("obj.mp3")
        mixer.init()
        mixer.music.load('obj.mp3')
        mixer.music.play()
        time.sleep(2)
        print(bb)
        myobj = gTTS(text=bb, lang=language, slow=False)
        myobj.save("obj.mp3")
        mixer.init()
        mixer.music.load('obj.mp3')
        mixer.music.play()
        time.sleep(2)
        print(cb)
        myobj = gTTS(text=cb, lang=language, slow=False)
        myobj.save("obj.mp3")
        mixer.init()
```

```
            mixer.music.load('obj.mp3')
            mixer.music.play()
            time.sleep(2)
            print('aucune des proposition')
            L=''
            time.sleep(2)
            while L=='':
                        print('Microphone on)
                        with mic as source:
                                    r.adjust_for_ambient_noise(source)
                                    audio = r.listen(source)
                        try:
                                    L=r.recognize_google(audio,language='fr-FR')
                        except sr.UnknownValueError:
                                    print('pouvez vous répéter votre message?')

            if L in cancer: acan += 1
            if L in migraine: bmig += 1
            if L in depression: cdep += 1
            if L in maux_d_estomac: dmaux += 1
            if L in allergie: eall += 1
            if L in colon_irritable: fcol +=1
            if L in maladie_cardiovasculaire: gcard += 1
if acan >= 4: print('il est possible que vous avez un cancer')
if bmig >= 4: print('il est possible que vous avez une migraine')
if cdep >= 4: print('il est possible que vous avez une depression')
if dmaux >= 4: print('il est possible que vous avez un maux d estomac')
if eall >= 4: print('il est possible que vous avez une allergie')
if fcol >= 4: print('il est possible que vous avez une colon irritable')
if gcard >= 4: print('il est possible que vous avez un e maladie cardio vasculaire')
```

Le programme retourne trois propositions, et reçoi le message vocal de l'utilisateur

Résultats de l'expérience :

Sources :

[1] https://fr.wikipedia.org/wiki/Interactions_homme-machine
[2] https://fr.wikipedia.org/wiki/Reconnaissance_automatique_de_la_parole
[3] Fang Chen and Kristiina Jokinen, Speech Technology, Theory and Applications. New York, USA: Springer, 2010.
[4] BCC Research. Global Voice Recognition Market To Reach $113 Billion In 2017. [Online]. http://www.bccresearch.com/pressroom/ift/global-voice-recognition-market-reach-$113- billion-2017
[5] Jardino (M.) et Adda (G.). - Language modeling for csr of large corpus using automatic classification of words. In: Proceeding of the European Conference On Speech Communication and Technologie, pp. 1191-1194. - September 1993.
[6] Witschel (P.). - Constructing lingcistic oriented language models for large vocabulary speech recognition. In: Proceeding of the European Conference On Speech Communication and Technologie, pp. 1199-1202. - September 1993.
[7] Cerf-Danon (H.) et El-Bèze (M.). - Three different probabilistic language models: Comparison and combination. In: Proceeding of the International Conference on Acoustics, Speech and Signal Processing, pp. 297-300. - Toronto, Canada, 1991.