

JSynATS : un analyseur syntaxique pour la reconnaissance automatique de la parole

Christophe Cerisara et Claire Gardent
LORIA, Nancy

2009-09-11

Plan de l'exposé

- 1 Reconnaissance automatique de la parole pour l'analyse syntaxique
 - Limites et erreurs des modèles de langage en reconnaissance
 - Défis spécifiques de l'analyse de l'oral transcrit
 - Erreurs de reconnaissance et disfluences
- 2 Développement d'un analyseur syntaxique de l'oral transcrit
 - Justification des choix
 - Présentation JSynATS
 - Guide d'annotation
- 3 Évaluations de l'analyseur
 - Evaluation sur ESTER
 - Evaluation sur PASSAGE
 - Pistes pour intégrer la syntaxe dans la reconnaissance

Introduction

Objectifs :

- 1 Analyse syntaxique des transcriptions automatiques de l'oral
- 2 Améliorer les performances des systèmes de transcription automatique de la parole :
 - Compenser les limites des modèles de langage actuels (n-grams)
 - La qualité linguistique de l'analyse est souhaitée, mais non indispensable...

Défis :

- Comment analyser un flux de parole transcrit automatiquement ?
- Comment exploiter cette analyse pour identifier les erreurs de transcription ?
- Comment exploiter cette analyse pour corriger les erreurs de transcription ?

Limites de la modélisation du langage en reconnaissance

Illustration : système de transcription automatique ANTS (temps-réel)
issu de l'évaluation ESTER1

Limites de la modélisation du langage en reconnaissance

Le modèle n-gram de base :

- Le modèle 3-gram : $P(w_t | w_{t-1}, w_{t-2})$
- Repli sur $P(w_t | w_{t-1})$ voire $P(w_t)$ cause de nombreuses erreurs

Extensions :

- Variable n-gram, triggers, skipping models, ...
- [Estève02] : intègre n-grams et automates pour modéliser des phénomènes spécifiques
- [Chelba&Jelinek] : n-grams de têtes

→ Intégration de la syntaxe en reconnaissance reste un domaine ouvert

Défis spécifiques à l'analyse de l'oral transcrit

Difficultés spécifiques :

- Syntaxe de l'oral vs. syntaxe de l'écrit (disfluences, structures spécifiques)
- Absence de ponctuation (segmentation en groupes de souffle)
- Erreurs de transcription (au mieux, 1 mot sur 10) : insertion, substitution, omission

Atouts :

- Mesure de confiance acoustique : utile à l'analyseur syntaxique ?

Étude des erreurs de reconnaissance

Étude des erreurs de reconnaissance : impact sur l'analyseur ? Erreurs corrigibles grâce à la syntaxe ?

2h France-Info, 778 groupes de souffle :

- 35% des groupes de souffle ont des erreurs à leurs limites
→ Importance de la segmentation
- 9% des groupes de souffle ont des erreurs liées aux disfluences
→ Traitement des disfluences non urgent
- 6% des groupes de souffle ont des erreurs corrigibles par la thématique
→ Inefficacité de la cohésion lexicale en intégration directe
- 43% des groupes de souffle ont des erreurs corrigibles par la syntaxe + sémantique
→ Importance de la syntaxe

Disfluences

Pré-traiter les disfluences ?

- Sortie reconnaissance : tous les mots (dont “euh”, répétitions)
- Analyseur de l' *oral transcrit*, qui doit rester fidèle à la parole
- Risque d'éliminer des répétitions volontaires
- Impact des disfluences limitées sur les erreurs
- Disfluences portent de l'information :
 - [Stolcke96] suggère que les “euh” contiennent de l'information pour prédire les mots voisins (ex : marque parfois le début d'un segment linguistique)
 - [Shriberg96] suggère que les mots suivants “euh” sont plus difficiles à prédire

Plan de travail : vers un analyseur syntaxique de l'oral transcrit

Construction de JSynATS :

- ① Choix d'un corpus et d'un outil d'analyse syntaxique
- ② Implémentation d'un logiciel d'annotation manuelle
- ③ Définition d'un guide d'annotation
- ④ Itérations :
 - Phase d'annotation/correction manuelle du corpus
 - Apprentissage de l'analyseur
 - Annotation automatique du reste du corpus d'apprentissage
- ⑤ Evaluation sur le corpus ESTER
- ⑥ Evaluation sur le corpus PASSAGE
- ⑦ Evaluation pour identifier les erreurs de transcription

Choix du corpus

Cadre applicatif : transcription des émissions radiophoniques francophones (projet ESTER)

Corpus d'apprentissage : transcriptions manuelles de France-Inter (1999)

- Répétitions, “euh” annotées
- Mots incomplets, bruits, ... supprimés
- La ponctuation est supprimée

“ce ce texte du projet de résolution qui deviendrait alors euh obligatoire euh s'il y a pas arrêt des bombardements”

Choix d'un outil d'analyse syntaxique

Besoins, objectifs :

- Scores (probabilité) d'analyse
- Solutions multiples
- Facilement adaptable : libre-source, conçu modulairement
- Portable : java
- Rapidité de mise en œuvre : proche état de l'art
- Dépendances

→ Malt parser (Joakim Nivre)

Principes de l'analyseur Malt

Algorithme de Nivre-Eager :

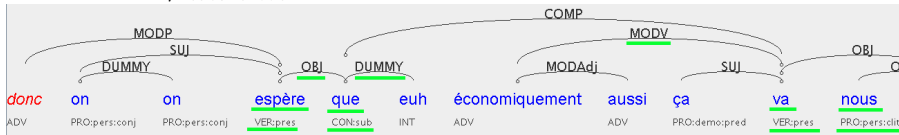
- Algorithme déterministe adapté du shift-reduce pour les grammaires hors-contexte
- Utilise 2 piles : mots déjà traités — mots restant
- Parse de gauche à droite :
 - SHIFT : transfert du mot en tête de pile droite
 - REDUCE : suppression du mot en tête de pile gauche
 - LEFT-ARC : dépendance de gauche à droite
 - RIGHT-ARC : dépendance de droite à gauche

- Complexité : linéaire en nombre de mots
- Contrainte 1 : structure arborescente
- Contrainte 2 : pas de non-projectivité*

Principes de l'analyseur Malt

Choix de l'action : classifieur SVM

```
<featuremodel name="nivreeager">  
  <feature>InputColumn(FORM, Stack[0])</feature>  
  <feature>InputColumn(FORM, Input [0])</feature>  
  <feature>InputColumn(POSTAG, Stack[0])</feature>  
  <feature>InputColumn(POSTAG, Input [0])</feature>  
  <feature>InputColumn(POSTAG, Input [1])</feature>  
  <feature>InputColumn(POSTAG, Input [2])</feature>  
  <feature>InputColumn(POSTAG, Input [3])</feature>  
  <feature>InputColumn(POSTAG, Stack [1])</feature>  
  <feature>OutputColumn(DEPREL, Stack[0])</feature>  
  <feature>OutputColumn(DEPREL, ldep(Stack[0]))</feature>  
  <feature>OutputColumn(DEPREL, rdep(Stack[0]))</feature>  
  <feature>OutputColumn(DEPREL, ldep(Input [0]))</feature>  
  <feature>InputColumn(FORM, Input [1])</feature>  
  <feature>InputColumn(FORM, head(Stack[0]))</feature>  
  <feature>InputColumn(LEMMA, Input [0])</feature>  
  <feature>InputColumn(LEMMA, Stack[0])</feature>  
</featuremodel>
```



Implémentation d'un logiciel d'annotation manuelle

Annotation de corpus par le logiciel JSynATS. Fonctionnalités :

- portable (100% JAVA), libre-source
- mode navigation : choix phrase + mot, découpage / jointure de phrases
- mode édition : choix tête + relation
- Import/Export : CONLL, XML(Syntex), EASY, PASSAGE
- Export : Latex, JPG

Procédure d'annotation par itération :

- Phase d'annotation automatique
- Phase de correction manuelle des annotations

Implémentation d'un logiciel d'annotation manuelle

The screenshot shows a software window titled "File Edit Preferences Help" containing three sentences with their syntactic annotations. Each sentence is numbered and includes a "NaN" label. The annotations consist of syntactic tree structures with labels like MOD-V, SUJ-V, AUX-V, COORD, CPL-V, GRP, and MOD-N, and a list of grammatical categories for each word.

1 NaN
 Longtemps j'ai été comme eux et j'ai souffert du même malaise
 NOM PRO:pers:conj VER:aux:pres VER:ppe CON:sub PRO:pers:disj CON:coo PRO:pers:conj VER:aux:pres VER:ppe PRE:det ADV ADJ

2 NaN
 Je leur offre le remède qui m'a guéri .
 NOM PRO:pers:clit VER:pres DET:def NOM PRO:rela PRO:pers:clit VER:aux:pres VER:ppe PON:sep

3 NaN
 Je fus élevé chrétiennement et , après ma première communion , j'ai
 NOM VER:aux:simp VER:ppe ADV CON:coo PON:comma PRE PRO:poss ADJ:num:ord NOM PON:comma PRO:pers:conj VER:aux:pr

Définition d'un guide d'annotation

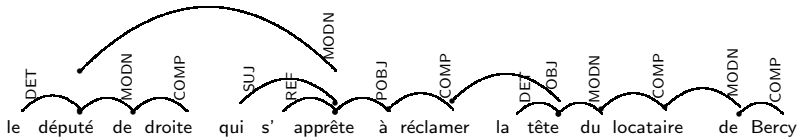
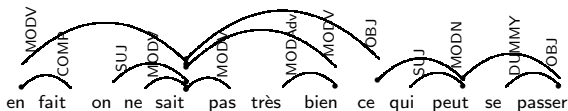
- Inspiré des schémas EASY, ALPAGE et Paris 7
- Contraintes de Malt
 - structure en dépendances complète (relation DET) en arbre,
 - éviter la non-projectivité
 - lier **tous** les mots
 - Gouverneur unique
- Contrainte du reconnaiseur : pas de ponctuation

Définition d'un guide d'annotation

RAPSODY (18)	ALPAGE (19)	P7 (8)	EASY (14)
suj	suj	SUJ	SUJ_V
obj	obj	OBJ	COD_V
pobj	p_obj	P-OBJ	CPL_V
	de_obj	DE-OBJ	CPL_V
	a_obj	A-OBJ	
	dep		
atts	ats	ATS	ATB_SO
atto	ato	ATO	
modV	mod	MOD	MOD_V
ref			
dummy	aff		
aux	aux_pass		
	aux_caus		
det	det		
modN	mod		MOD_N
comp	arg_cons, arg_comp, obj, p_obj		COMP
cc	coord, arg_coord		COORD
multimots			
	punct		
modA			MOD_A
modADV			MOD_R
			MOD_P
appos			APP
juxt			JUXT

Définition d'un guide d'annotation

Exemples d'annotation :



Itérations pour l'annotation

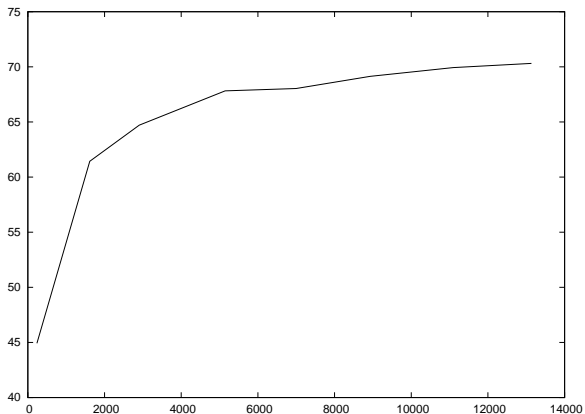
- Corpus \mathcal{C}^0 annoté par une linguiste experte : 458 mots
- Apprentissage de Malt sur \mathcal{C}^0
- Première série d'itérations annotations automatiques + corrections manuelles + vérification des corrections :
un mois, 5 itérations : corpus \mathcal{C}^1 ($\mathcal{C}^0 \subset \mathcal{C}^1$) 5305 mots
- Deuxième série d'itérations annotations automatiques + corrections manuelles :
un mois, 4 itérations : corpus \mathcal{C}^2 21515 mots

Evaluation :

- corpus \mathcal{C}^1 : évaluation
- corpus \mathcal{C}^2 : apprentissage

Evaluation sur ESTER

Scripts d'évaluation de CONLL : Labeled attachment score



Comparaison : CONLL'2008 (anglais) : entre 72% et 89%

Evaluation sur Passage

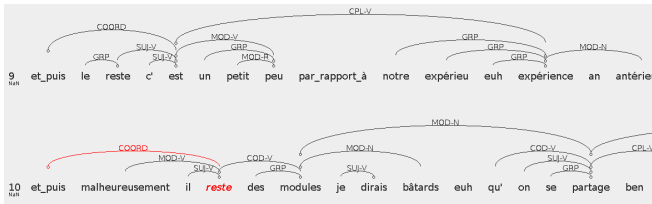
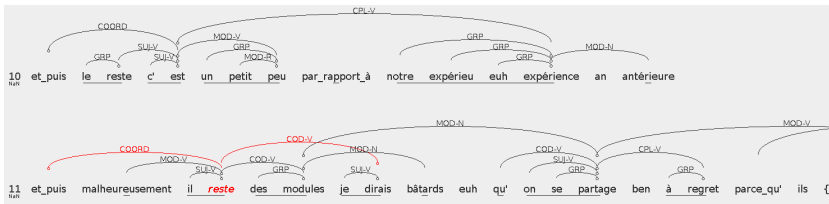
Ré-apprentissage de JSynATS sur corpus de développement de Passage
(84000 mots)

Problèmes :

- Groupes → arbres de dépendances :
 - Identification tête du groupe (heuristiques)
 - Nouvelle relation GRP
 - Dépendances entrant/sortant du groupe
- Relations ternaires COORD :
 - 1 rel. COORD → 2 relations CC
 - Heuristique pour recomposer les relations COORD

Evaluation sur Passage

- Principale difficulté : annotations manquantes, élimination des têtes multiples



Evaluation sur Passage

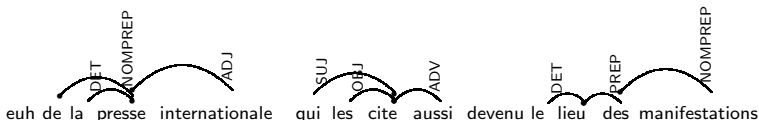
Test sur fichier "littéraire_1" :

```
EVAL SUBCORPUS_ALL ALL_RELATIONS p=0.694905 r=0.493461 f=0.577109
EVAL SUBCORPUS_ALL SUJ_V p=0.90301 r=0.65534 f=0.759494
EVAL SUBCORPUS_ALL AUX_V p=0.928571 r=0.902778 f=0.915493
EVAL SUBCORPUS_ALL COD_V p=0.625 r=0.419847 f=0.502283
EVAL SUBCORPUS_ALL CPL_V p=0.621005 r=0.387464 f=0.477193
EVAL SUBCORPUS_ALL MOD_V p=0.720721 r=0.462428 f=0.56338
EVAL SUBCORPUS_ALL COMP p=0.173077 r=0.147541 f=0.159292
EVAL SUBCORPUS_ALL ATB_SO p=0.530612 r=0.40625 f=0.460177
EVAL SUBCORPUS_ALL MOD_N p=0.75 r=0.626549 f=0.682739
EVAL SUBCORPUS_ALL MOD_A p=0.727273 r=0.489796 f=0.585366
EVAL SUBCORPUS_ALL MOD_R p=0.5 r=0.4 f=0.444444
EVAL SUBCORPUS_ALL MOD_P p=0 r=0 f=0
EVAL SUBCORPUS_ALL COORD p=0.378788 r=0.306748 f=0.338983
EVAL SUBCORPUS_ALL APPOS p=0 r=0 f=0
EVAL SUBCORPUS_ALL JUXT p=0 r=0 f=0
```

- Déficit de relations
- Problèmes spécifiques : COORD, APPOS, JUXT, MODP, COMP

Evaluation pour localiser les erreurs de transcription

Exemple d'analyse avec erreurs de reconnaissance (Syntax) :



Mesure de confiance topologique :

- Extraire des indices syntaxiques du graphe d'analyseur
- Apprendre un classifieur pour identifier les mots faux
- Mesure de confiance = score du classifieur

Evaluation pour localiser les erreurs de transcription

Classifieur : MLP. Indices syntaxiques :

- 1 Taille du sous-arbre contenant un mot
- 2 Profondeur du mot dans cet arbre
- 3 Présence ou non d'un gouverneur du mot
- 4 Nombre de dépendants directs
- 5 Taille du sous-arbre dépendant du mot
- 6 Classe morpho-syntaxique du mot
- 7 Mesure de confiance "acoustique" du mot

Features used	Equal Error Rate (%)
Acoustic posterior (baseline)	15.5
Syntactic features	30.0
All features combined	13.8

Perspectives

- Nouvel analyseur 2ème passe : ajouter dépendances entre constituants, ajouter têtes multiples
- Prise en compte des erreurs de reconnaissance : algorithme non-déterministe + classifieur bayésien
- Extension à l'étiquetage sémantique des arguments verbaux

Merci !

Références

- A. Stolcke and E. Shriberg, "Statistical language modeling for speech disfluency," in Proc. Int. Conf. Acoustics, Speech, Signal Processing, 1996, pp. 405–408.
- E. Shriberg and A. Stolcke, "Word predictability after hesitations : A corpus-based study," in Proc. Conf. Spoken Language Processing, 1996, pp. 1868–1871.