Noyaux de réécriture de phrases munis de types lexico-sémantiques

Martin Gleize[1,2] Brigitte Grau[1]

[1] LIMSI-CNRS, Orsay, [2] Université Paris-Sud, Orsay

TALN 2015, 23 juin 2015, Caen

Plan de la présentation

Réécriture de phrases

Problème : décider si une phrase se réécrit en une autre.

- Reconnaître l'implication textuelle (RTE)
  Réécriture dans 1 sens
  T : Scott Island is 370 metres long and 180 m wide, with the highest elevation being Haggits Pillar at 63 m.
  $\rightarrow$ *H : Haggits Pillar is located in Scott Island.*
- Reconnaissance de paraphrases
  Sur une paire de phrases : réécriture dans les 2 sens
- Réponse à des questions
  Question -> candidat de réponse ?

Problème de classification binaire

Fonctions noyau (kernels)

- Méthodes d'apprentissage supervisé
- Mesurent similarité entre 2 éléments
- Bonne fonction noyau :
  - élevée pour 2 éléments de même étiquette
  - basse pour 2 éléments d'étiquettes différentes
- Algorithme d'apprentissage : machines à vecteurs de support (Vapnik 2000)

Noyau de réécriture de phrases

Principe : dénombre les réécritures communes entre deux couples de phrases vues comme séquences de leurs mots.
Formellement, si phrases = éléments de Σ^*, les séquences de mots de Σ :

$$K : (\Sigma^* \times \Sigma^*) \times (\Sigma^* \times \Sigma^*) \to \mathbb{R}$$

$$K((s_1, t_1), (s_2, t_2)) = \langle \Phi(s_1, t_1), \Phi(s_2, t_2) \rangle$$

$\Phi(s, t) = (\phi_r(s, t))_{r \in R}$ avec les caractéristiques $\phi_r(s, t) = n$ le nombre de fois où r se déclenche dans (s, t).
R est l'ensemble des règles de réécriture.
Intérêt : pas de calcul explicite de Φ

Noyaux en TAL

- *string kernel* : catégorisation de texte (Lodhi et al. 2002)
- *tree kernel* : extraction de relation (Culotta et Sorensen 2004)

Noyaux de réécriture de phrases :

- Opèrent sur 2 couples de phrases (4 phrases au total)
- ~~Similarité entre 2 phrases~~
- Similarité entre les façons dont les phrases des 2 couples se réécrivent l'une en l'autre.
- Noyau de paires d'arbres syntaxiques (Zanzotto et al. 2007, 2011)
- *String rewriting kernel* (Bu et al. 2012) : pas d'analyse syntaxique préalable

Exemple

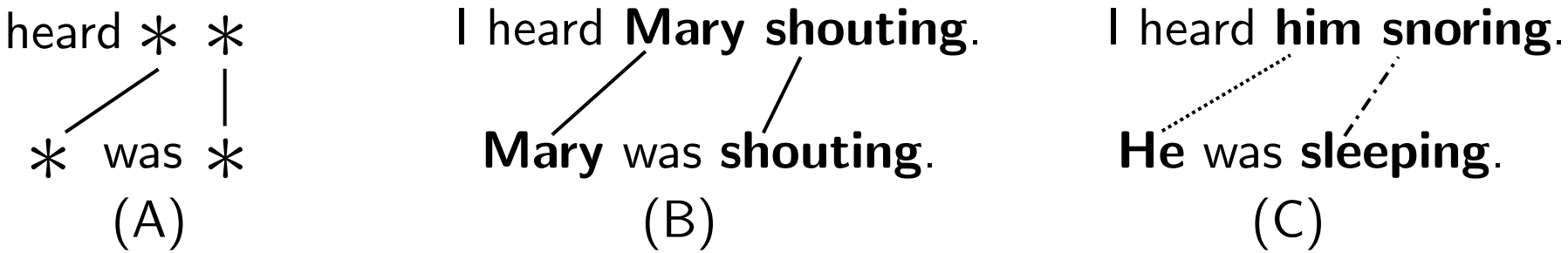


Figure: (A) se déclenche dans (B) mais pas dans (C).

Règle de réécriture :

- 2 motifs
- variables liées, remplacées par le même mot

Une règle se déclenche dans (s, t) si on retrouve un couple de sous-séquences de (s, t) en substituant les variables par des mots. Ici, les variables ne coincident pas dans (C).

Exemple 2

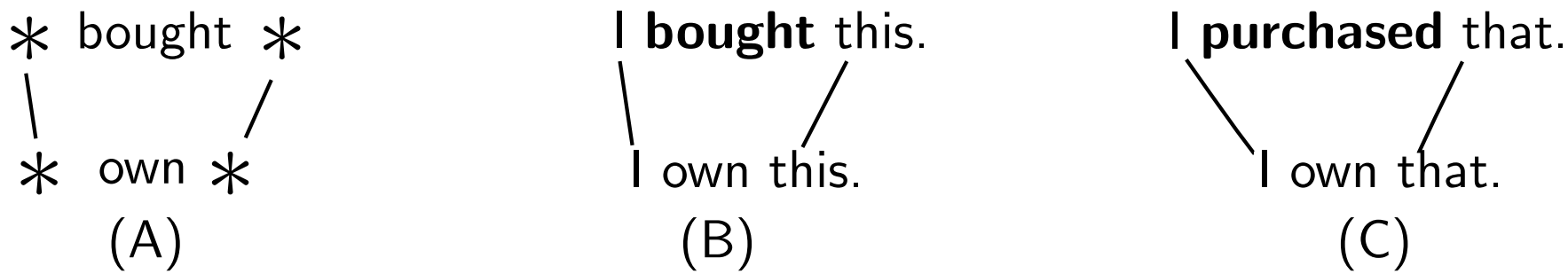
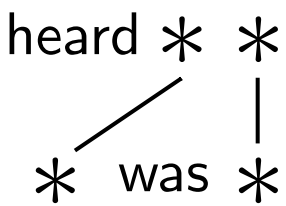


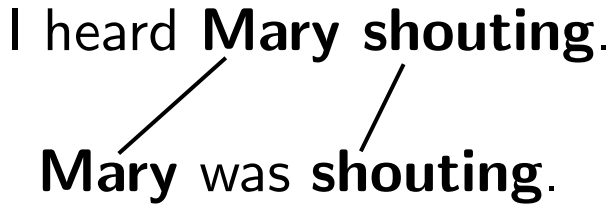
Figure: (A) se déclenche dans (B) mais pas dans (C).

Les motifs ne coincident pas dans (C).

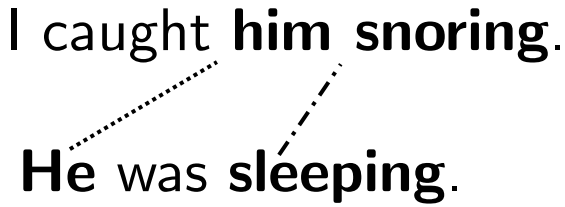
Motivation



(A)



(B)



(C)

- Autoriser plus de flexibilité dans les réécritures
  - motifs : *caught* $\approx$ *heard*
  - variables : *him* $\approx$ *he*
- Ajouter une couche sémantique
  snoring implique *sleeping*

Classification : projection de Φ dans le plan

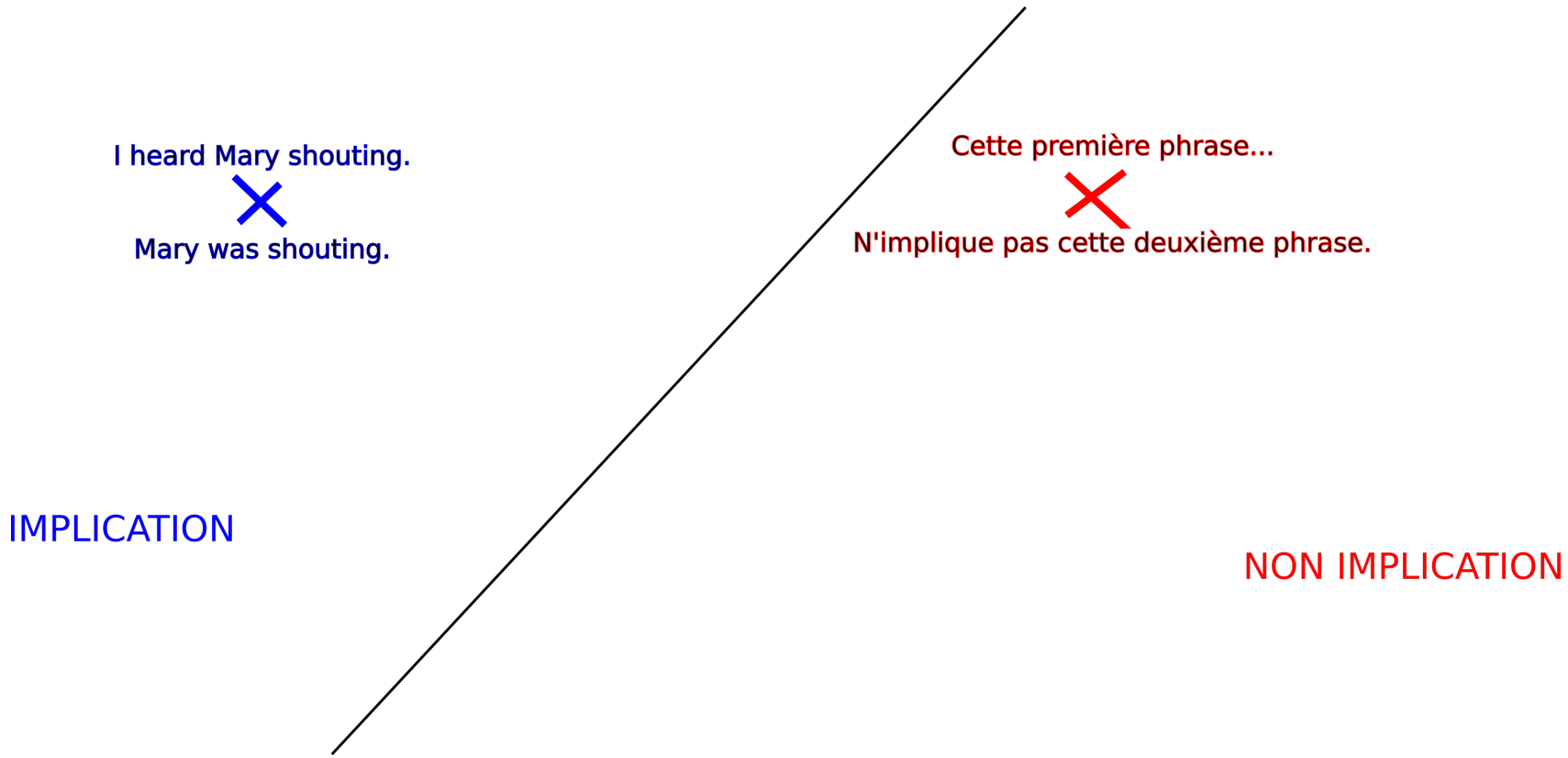


Figure: Exemples d'entraînement

Classification : projection de Φ dans le plan

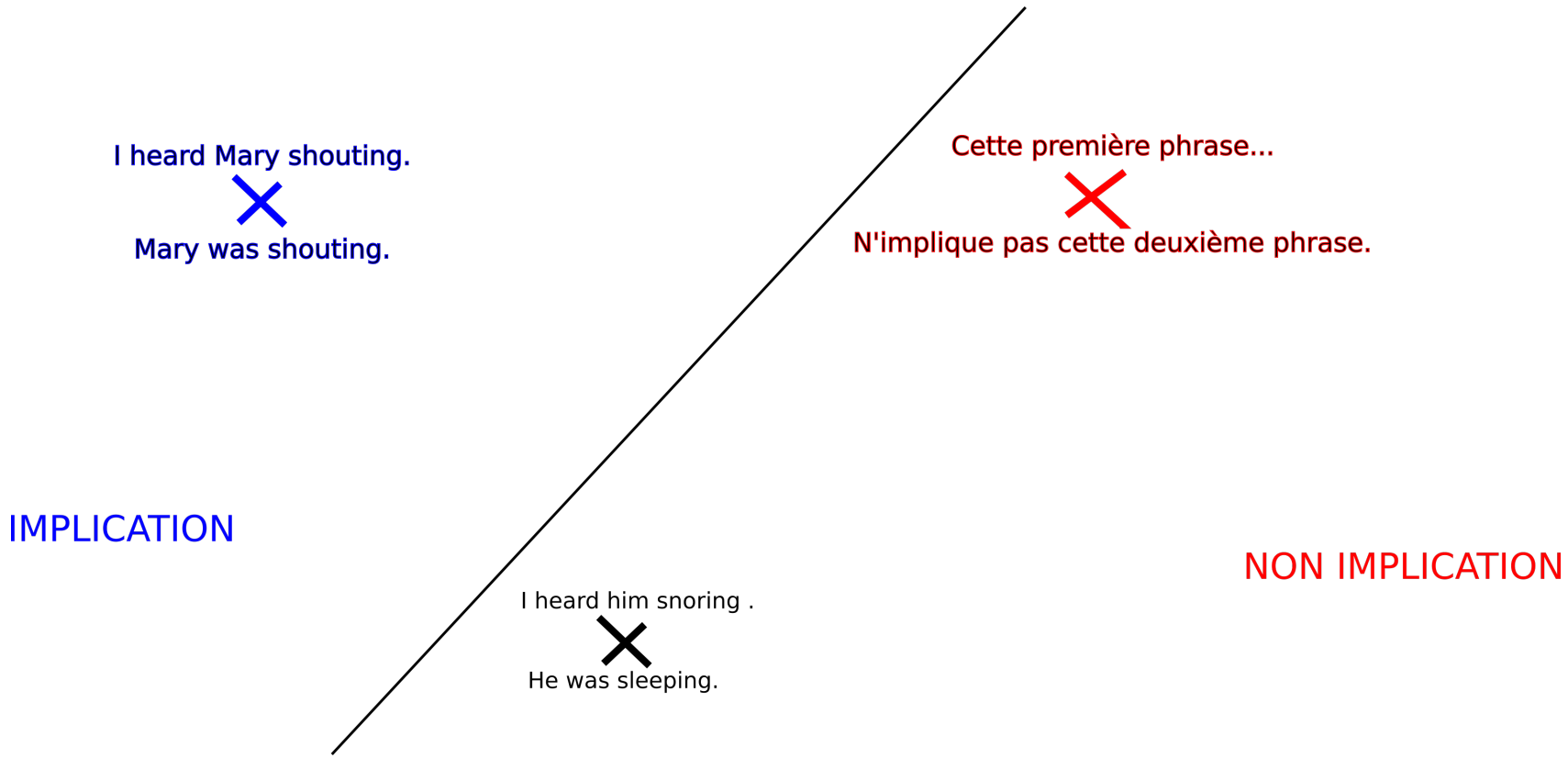


Figure: Classification d'un exemple de test

Classification : projection de Φ dans le plan

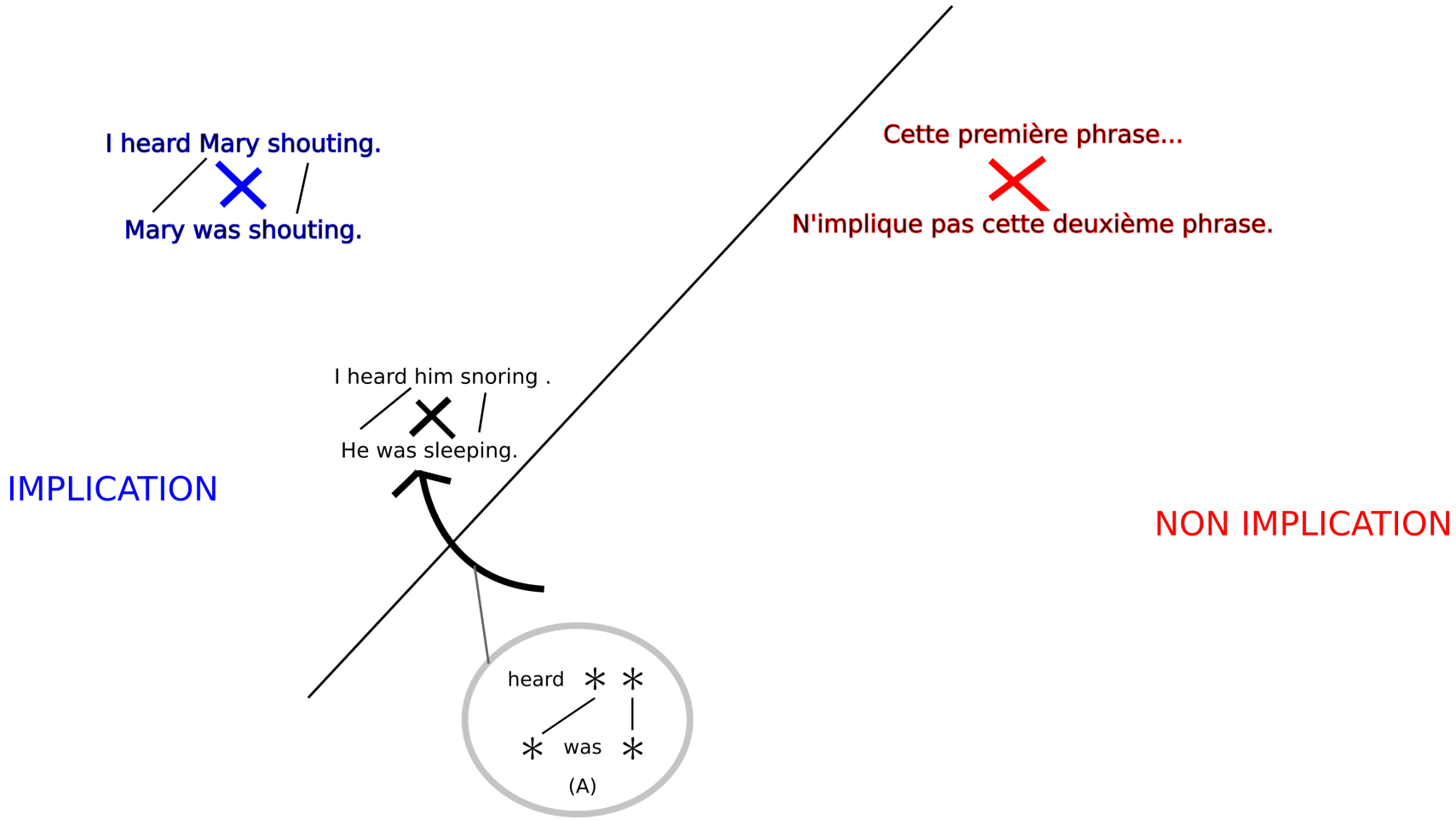


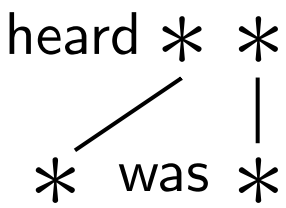
Figure: La règle (A) se déclenche et rapproche deux couples de phrases

Règles de réécriture enrichies de types

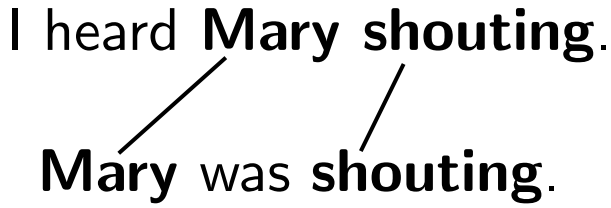
- Une règle de réécriture peut se déclencher aux types près
- *Types motif* : définissent les classes de mots coincidant avec les motifs
- *Types variable* : définissent les classes de mots pouvant substituer la même variable liée

Un type de base : *identité*, mots doivent être identiques dans les motifs ou pour substituer une variable liée.
Possibilité de choisir les ensembles de types motif et types variable.

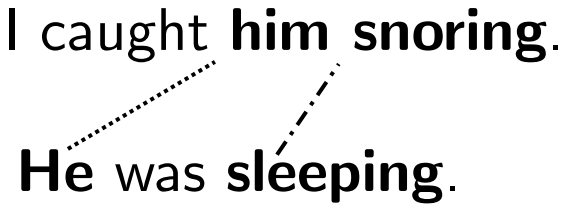
Type de base uniquement



(A)



(B)

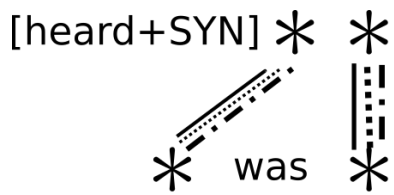


(C)

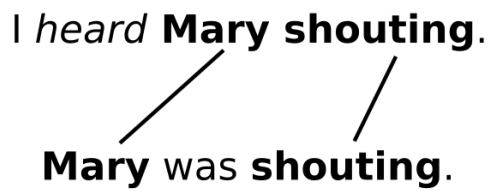
- Types motif : {identité}
- Types variable : {identité}

→ La règle (A) ajoute 0 au calcul du noyau de (B) et (C)

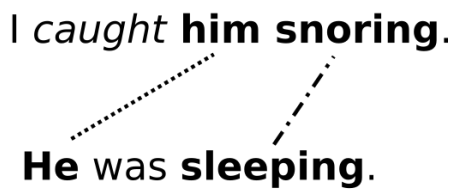
Types avec ressources lexico-sémantiques



(A)



(B)



(C)

- Types motif : {identité, synonym_WN}
- Types variable : {identité, même_pronom, entailment_WN}

→ La règle (A) ajoute 1 au calcul du noyau de (B) et (C)

Nécessité d'un algorithme efficace pour le calcul du noyau

Dans nos expériences, les motifs des règles sont restreints à des k-grammes.

$$K_k((s_1,t_1),(s_2,t_2)) = \sum_{\substack{\alpha_{s_1}\in k-grams(s_1)\\ \alpha_{t_1}\in k-grams(t_1)}} \sum_{\substack{\alpha_{s_2}\in k-grams(s_2)\\ \alpha_{t_2}\in k-grams(t_2)}} \bar{K}_k((\alpha_{s_1},\alpha_{t_1}),(\alpha_{s_2},\alpha_{t_2}))$$

$$\bar{K}_k((\alpha_{s_1},\alpha_{t_1}),(\alpha_{s_2},\alpha_{t_2})) = \sum_{r\in R} \mathbb{1}_r(\alpha_{s_1},\alpha_{t_1})\mathbb{1}_r(\alpha_{s_2},\alpha_{t_2})$$

avec $\mathbb{1}_r = 1$ si r se déclenche dans le couple de k-grammes considéré, 0 sinon.

$\rightarrow$ Calcul naïf coûteux en temps.

Solutions aux sources d'explosion combinatoire

$$\sum_{r \in R} \mathbb{1}_r(\alpha_{s_1}, \alpha_{t_1}) \mathbb{1}_r(\alpha_{s_2}, \alpha_{t_2}) \rightarrow$$

- Générer uniquement les règles communes à $(\alpha_{s_1}, \alpha_{t_1})$ et $(\alpha_{s_2}, \alpha_{t_2})$
- Se ramène à : problème de dénombrement de matchings de graphes bipartis

$$\sum_{\substack{\alpha_{s_1} \in k-grams(s_1) \\ \alpha_{t_1} \in k-grams(t_1)}} \sum_{\substack{\alpha_{s_2} \in k-grams(s_2) \\ \alpha_{t_2} \in k-grams(t_2)}} \ldots \rightarrow$$

Sommer uniquement sur un sous-ensemble des $((\alpha_{s_1}, \alpha_{t_1}), (\alpha_{s_2}, \alpha_{t_2}))$

- Rapide à calculer, de petite taille
- Tel qu'il contient tous les éléments de noyau non nul

Expérimentations

- Tâches : reconnaissance de paraphrases et d'implication textuelle, sur l'anglais
- Pré-traitement des paires de phrases : OpenNLP, algorithme de Porter
- Apprentissage : LIBSVM
- Noyaux : SRK (type de base seul), TESRK (types lexico-sémantiques), PR (précision-rappel d'unigrammes)
- Somme des noyaux pour différentes fenêtres de k-grammes (jusqu'à $k = 4$)

Types

Type	Relation de type entre les mots (a, b)
id	mots ayant la même forme de surface et même POS
idMinusTag	mots ayant la même forme de surface
lemma	mots ayant le même lemme
stem	mots ayant la même racine
synonym, antonym	mots ayant la relation [type] dans WordNet
hypernym, hyponym	b est [type] de a dans WordNet
entailment, holonym	
lvhsn	mots ayant une distance d'édition de 1

Table: Types

Reconnaissance de paraphrases

MSR Paraphrase Corpus, 4076 couples de phrases d'entraînement et 1725 couples de test

Exemple : *Licensing revenue slid 21 percent to $107.6 million.* $\Longleftrightarrow$ *License sales fell 21 percent to $107.6 million.*

Types de TESRK : $\{stem, synonym\}$ pour les types motif et les types variables

Système Paraphrase	Accuracy	F-score
All paraphrase	66.5	79.9
Madnani et al. (2012)	**77.4**	**84.1**
PR	73.5	82.1
SRK + PR (baseline)	76.2	83.6
TESRK	76.6	83.7
TESRK + PR	**77.2**	**84.0**

Reconnaissance d'implication textuelle

RTE1+2+3dev pour entraîner (3767), RTE3test pour tester (800)
Exemple : *Black bears are dangerous. They can and do kill people.* $\Longrightarrow$ *Black bear attacks people.*
$\{stem, synonym\}$ pour les types motif
$\{stem, synonym, hypernym, hyponym, entailment, holonym\}$ pour les types variables

Système RTE	Accuracy
Zanzotto et al. (2007)	65.8
Hickl et al. (2006)	**80.0**
PR	61.8
TESRK (All)	62.1
SRK + PR (baseline)	63.8
TESRK (Syn) + PR	64.1
TESRK (All) + PR	**66.1**

Conclusion

- Extension de types pour une classe de noyaux de réécriture de phrases
- Intègre l'apprentissage de structure, de variations lexico-syntaxiques, combinés à l'apport de sémantique par ressources externes
- Méthode de calcul efficace du noyau
- Performances état-de-l'art sur la reconnaissance de paraphrase
- Sur RTE, performances comparables aux systèmes n'utilisant pas d'autre ressource lexico-sémantique que WordNet