

Utilisation de mécanismes inférentiels dans le processus d'explication automatique de la métaphore à une inconnue

Jérémie Roux¹ Hani Guenoune¹ Mathieu Lafourcade¹ Richard Moot¹

(1) LIRMM, Univ Montpellier, CNRS, Montpellier, France

prenom.nom@lirmm.fr

RÉSUMÉ

Considérons la métaphore comme une analogie à une inconnue. L'expliquer revient à résoudre l'unique variable du carré analogique qui en résulte et dont les trois autres termes sont fixés. Nous proposons ici une méthode détaillée pour arriver à cet objectif en utilisant la base de connaissances *JeuxDeMots*. Nous procédons par reconnaissance de schémas de relations préalablement identifiés et qui permettent d'évaluer la force de la similarité relationnelle et celles des deux similarités attributionnelles pour en déduire celle de l'analogie dans sa globalité. Le terme candidat qui permet d'obtenir la meilleure force d'analogie entre les quatre termes de l'analogie à trou ainsi complétée est élu. Enfin, on cherche à démontrer que l'utilisation d'inférences dans ce processus permet d'aboutir à de meilleurs résultats, c'est-à-dire augmenter le nombre de fois où un bon candidat est élu.

ABSTRACT

Using inference mechanisms for automatically explaining metaphors with one unknown

We consider metaphors to be analogies with one unknown. Explaining it means solving the variable of the resulting analogical square, the other three terms being fixed. We propose a detailed method to achieve this goal using the knowledge base *JeuxDeMots*. We proceed by recognizing previously identified relational patterns that allow us to evaluate the strength of the relational similarity and those of the two attributional similarities to deduce that of the analogy as a whole. The candidate term that allows us to obtain the best strength of analogy between the four terms of the analogy with the gap thus completed is elected. Finally, we seek to demonstrate that the use of inferences in this process allows us to achieve better results, that is, to increase the number of times a good candidate is elected.

MOTS-CLÉS : inférence, métaphore, analogie, contenu figuré, traitement automatique du langage naturel, base de connaissances.

KEYWORDS: inference, analogy, metaphor, figurative language, natural language processing, knowledge base.

Contribution originale

1 Introduction

La “question classique de la figuration dans les métaphores lexicales, (*Paul est un bulldozer*) et verbales (*Le déficit de la Sécurité sociale a explosé du fait du coronavirus*)” (Kleiber, 2023) mène à des “divergences, analyses contradictoires et descriptions inappropriées de telle sorte qu’il est nécessaire de voir de plus près quelle est la réelle portée et quelle est l’utilité exacte du vecteur métaphorique” (Conenna & Kleiber, 2002).

L’analyse automatique de textes est un processus complexe issu d’une double approche, linguistique et informatique, à l’œuvre dans le traitement automatique du langage naturel. Parfois, les textes à analyser par ordinateur contiennent des contenus figurés tels que la métaphore, ce qui complexifie l’examen et la tâche de les paraphraser (Shutova, 2010). Après analyse syntaxique et syntagmatique, identifier des éléments de ces textes qui semblent être des contresens permet de soupçonner la présence de contenus figurés.

Dans notre démarche d’explication du processus, nous essayons de nous affranchir de la boîte noire de l’approche par intelligence artificielle neuronale mais plutôt profiter des “mérites relatifs des approches logiques et fondées sur le raisonnement par similarité sur les analogies” (Russell, 1988) afin que l’ensemble du processus soit beaucoup plus contrôlable et transparent. En considérant les métaphores et les comparaisons comme des analogies à trou (Roux *et al.*, 2024), comprendre ou expliquer les relations de similarité constitutives de l’analogie entre les termes impliqués dans ce qu’on considère alors comme des anomalies, revient à résoudre cette hypothétique analogie. On peut alors confirmer la présence de contenus figurés ou l’infirmier. En effet, dans ce deuxième cas, il se peut que ces contresens en soient réellement et qu’il ne s’agisse alors pas de contenus figurés mais d’un texte ou d’un fragment de texte qui n’a pas de sens en soit, qui contient des erreurs grammaticales par exemple et qui est donc mal formé. Il se peut aussi qu’on arrive à trouver des liens de similarité entre ces termes alors qu’il n’y a pas de sens figuré mais seulement un sens propre.

L’analogie aristotélicienne *A est à B ce que C est à D* qui “dicte le choix de nos mots et notre compréhension des situations les plus quotidiennes” (Hofstadter & Sander, 2013) se transforme donc en une métaphore à une inconnue lorsque l’un de ses 4 termes fixé devient une variable. On cache ainsi ce terme et on essaye de le retrouver en utilisant un processus d’élection que nous détaillerons.

L’analogie est un “mécanisme cognitif puissant que les gens utilisent pour faire des inférences et apprendre de nouvelles abstractions” (Gentner & Holyoak, 1997) mais pourquoi ne pas alors utiliser ces mêmes inférences pour essayer d’améliorer l’élection d’un bon candidat pour une analogie à trou ? En utilisant le réseau lexico-sémantique *JeuxDeMots* (présenté en section 2), nous proposons de dérouler le processus de résolution de la métaphore (section 3), puis nous verrons comment l’utilisation d’inférences (section 4) influence positivement l’élection du bon candidat de l’analogie à une inconnue (section 5). La “résolution de l’analogie entre chaînes de symboles [que] nous cherchons à atteindre, ce ne sont ni des règles, ni des lois, mais la formalisation d’une opération que tout le monde reconnaît être à l’œuvre dans la langue” (Lepage, 2003).

2 *JeuxDeMots* comme base de connaissances

JeuxDeMots (abrégé *JDM*) est une base de connaissances principalement en langue française développée et hébergée au LIRMM depuis 2007 (Lafourcade, 2007) sous licence *Creative Common Zero*. Il s’agit d’un réseau lexico-sémantique sous la forme d’un graphe orienté. Il est composé de termes représentés par les nœuds de ce graphe et d’un type de relation r_t sur chaque arc. Une relation relie

deux nœuds¹, elle est pondérée (entier positif ou négatif) et potentiellement annotée.

Un fragment de ce vaste réseau est présenté en FIGURE 1. Début 2025, la base était composée d'environ 14 millions de termes (dont plus de 160 000 raffinements²) liés par environ 915 millions de relations de différents types. La base s'améliore au fur et à mesure que les utilisateurs jouent à des jeux sérieux³ (modèle *Game With A Purpose*⁴) sur la [plateforme web JDM](#) [↗](#).

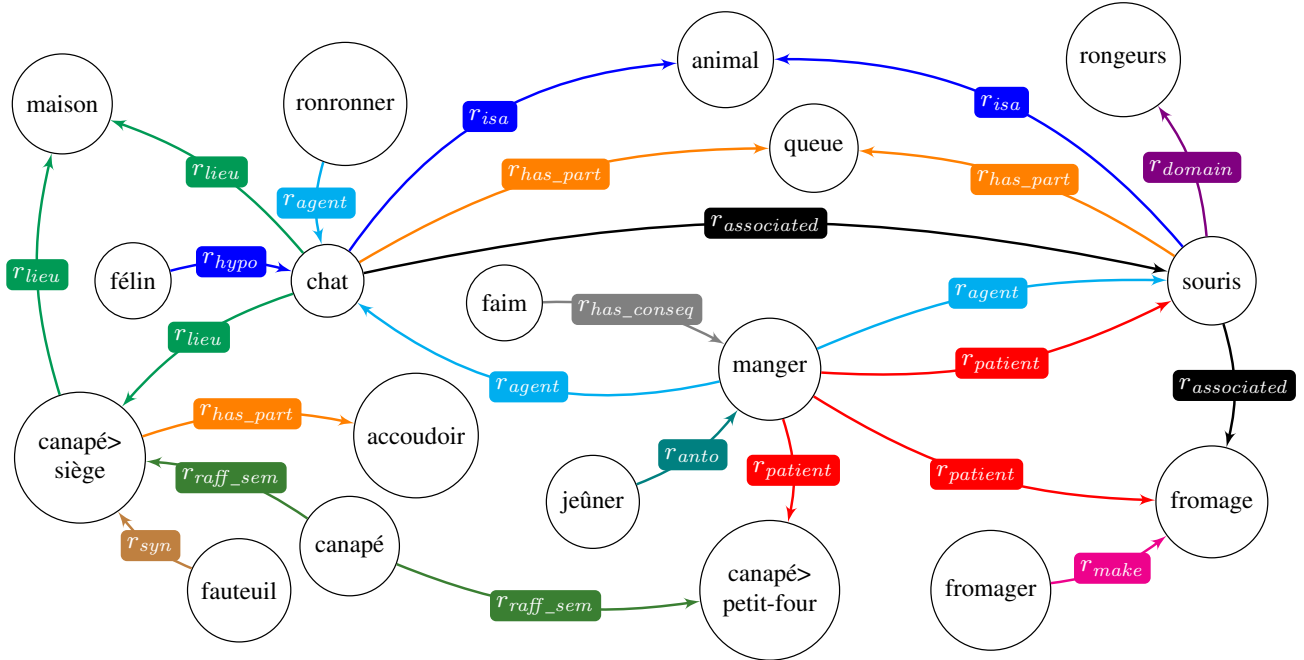


FIGURE 1 – Sous-graphe du réseau lexico-sémantique *JDM*

2.1 Relations

Parmi les 172 types de relations disponibles dans *JDM*, voici en FIGURE 2 les principales. La plupart de celles que nous allons utiliser ont un caractère sémantique, qu'on considère alors "non pas comme des primitives non analysables, mais en fonction des propriétés relationnelles qui les distinguent" (Chaffin & Herrmann, 1984). Les relations de mêmes couleurs sont dites conversives, c'est-à-dire que l'on a : $x \xrightarrow{r_t} y \Leftrightarrow y \xrightarrow{r_{t-1}} x$. À noter que l'on peut avoir des boucles ($x \xrightarrow{r_t} x$) et que certaines relations sont symétriques ($r_t = r_{t-1}$) comme r_{syn} .

Certaines relations sont vagues comme $r_{associated}$ car il s'agit d'une association entre 2 termes sans qu'on en sache davantage. Les autres sont plus précises, certaines sont sémantiques comme r_{raff_sem} (raffinement sémantique) ou bien r_{syn} et r_{syn_strict} (qui étudient respectivement la synonymie classique et celle de termes strictement substituables). D'autres sont syntaxiques comme r_{pos} qui s'intéresse à la catégorie grammaticale du terme. r_{lemma} désigne la forme canonique du terme de départ, c'est-à-dire la forme à l'infinitif pour un verbe, celle au singulier pour un nom et celle au masculin singulier pour un adjectif.

1. Il peut parfois y avoir des boucles, le nœud de départ est le même que celui d'arrivée comme pour certains r_{lemma} .
2. Un raffinement sémantique noté $x \xrightarrow{r}$ d'un terme polysémique x (liés grâce à la relation $x \xrightarrow{r_{raff_sem}} x \xrightarrow{r}$) est un sens ou une signification possible du terme x . Par exemple, le terme *glace* a les raffinements sémantiques *glace>eau gelée*, *glace>crème glacée* et *glace>miroir* qu'il faut différencier.
3. Activité qui combine une intention pédagogique avec des ressorts ludiques.
4. Traduction littérale : "Jeu Avec Un But"

ID	Nom	Description	Exemple
0	<i>r_{associated}</i>	est associé à	souris <i>r_{associated}</i> fromage
1	<i>r_{raff_sem}</i>	a pour raffinement sémantique	glace <i>r_{raff_sem}</i> glace>miroir
3	<i>r_{domain}</i>	a pour domaine	corner <i>r_{domain}</i> sport
4	<i>r_{pos}</i>	a pour catégorie grammaticale	poireau <i>r_{pos}</i> Nom :
5	<i>r_{syn}</i>	a pour synonyme	canapé <i>r_{syn}</i> divan
6	<i>r_{isa}</i>	a pour hyperonyme	chat <i>r_{isa}</i> animal
7	<i>r_{anto}</i>	a pour antonyme	chaud <i>r_{anto}</i> froid
8	<i>r_{hypo}</i>	a pour hyponyme	véhicule <i>r_{hypo}</i> autobus
9	<i>r_{has_part}</i>	a pour méronyme	bras <i>r_{has_part}</i> main
10	<i>r_{holo}</i>	a pour holonyme	roue <i>r_{holo}</i> voiture
13	<i>r_{agent}</i>	a pour agent	miauler <i>r_{agent}</i> chat
14	<i>r_{patient}</i>	a pour patient	manger <i>r_{patient}</i> croissant
15	<i>r_{lieu}</i>	a pour lieu	oiseau <i>r_{lieu}</i> nid
16	<i>r_{instr}</i>	a pour instrument	couper <i>r_{instr}</i> couteau
17	<i>r_{carac}</i>	a pour caractéristique	eau <i>r_{carac}</i> liquide
19	<i>r_{lemma}</i>	a pour lemme	avaient <i>r_{lemma}</i> avoir
28	<i>r_{lieu_1}</i>	est un lieu pour	niche <i>r_{lieu_1}</i> chien
30	<i>r_{lieu_action}</i>	est un lieu pour l'action	bibliothèque <i>r_{lieu_action}</i> lire
31	<i>r_{action_lieu}</i>	est une action dans le lieu	nager <i>r_{action_lieu}</i> piscine
41	<i>r_{has_conseq}</i>	a pour conséquence	tomber <i>r_{has_conseq}</i> se blesser
42	<i>r_{has_causatif}</i>	a pour cause	vol <i>r_{has_causatif}</i> pauvreté
53	<i>r_{make}</i>	produit	abeille <i>r_{make}</i> miel
54	<i>r_{product_of}</i>	est le produit de	électricité <i>r_{product_of}</i> éolienne
64	<i>r_{has_instance}</i>	a pour instance	montagne <i>r_{has_instance}</i> Everest
71	<i>r_{variante}</i>	a pour variante	clé <i>r_{variante}</i> clef
72	<i>r_{syn_strict}</i>	a pour synonyme strict	auto <i>r_{syn_strict}</i> automobile
78	<i>r_{cohypo}</i>	est co-hyponyme de	ordinateur <i>r_{cohypo}</i> tablette
130	<i>r_{is_instance_of}</i>	est une instance de	Titanic <i>r_{is_instance_of}</i> bateau
155	<i>r_{make_use_of}</i>	fait usage de	agriculteur <i>r_{make_use_of}</i> tracteur
156	<i>r_{is_used_by}</i>	est utilisé par	essence <i>r_{is_used_by}</i> voiture

FIGURE 2 – Types de relations utiles pour notre problème et présentes dans JDM

2.2 Annotations

Dans la structure de données choisie, une annotation porte sur une relation par l'intermédiaire d'un nœud de relation, il existe alors une relation de type *r_{annotation}* depuis le nœud de relation vers l'annotation (voir FIGURE 3). Les types d'annotations de JDM sont de nature différente, voici la liste telle que présentée dans (Ramadier et al., 2014) :

- annotations **fréquentielles** : très rare, rare, possible, fréquent, toujours vrai, probable
- annotations d'**usage** : souvent cru vrai, abus de langage
- annotations **quantitatives** : un nombre (1, 2, 4, ...), beaucoup, peu, etc
- annotations d'**exception** : exception
- annotations **qualitatives** : pertinent, non pertinent, inférable

Il nous faut prendre en compte ces annotations dans notre algorithme afin d'éviter d'inférer à partir de relations non pertinentes et rares mais plutôt s'en tenir en priorité à celles qui sont toujours vraies, pertinentes et/ou inférables. Par exemple, la relation *corbeau* *r_{carac}* *blanc* est annotée comme étant *rare* car il existe des corbeaux albinos. Il est clair que la plupart des corbeaux sont noirs d'où la rareté de la caractéristique blanche pour un corbeau. La relation *corbeau* *r_{carac}* *noir* est quant à elle annotée comme étant *pertinente*.

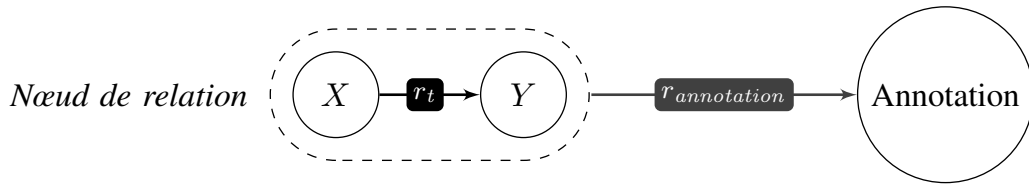


FIGURE 3 – Annotation par réification d’une relation sous forme de nœud

2.3 Des jeux pour constituer la base de connaissances et pondérer les relations

La base *JDM* est constituée à l’aide du jeu sérieux éponyme, les utilisateurs jouant librement et gratuitement sur la [plateforme web JDM](#). Le but, pour un terme x et une relation r_t , est de lister les meilleurs termes y selon le joueur tels que $x \ r_t \ y$.

L’intérêt du jeu réside cependant dans le fait de se comparer à un autre joueur (voir FIGURE 4) qui aura lui aussi renseigné les meilleurs y selon lui pour ces mêmes x et r_t donnés tels que $x \ r_t \ y$. On notera les y du joueur a les y_a et ceux du joueur b les y_b . Ainsi, on obtient 2 listes y_a et y_b dont l’intersection constitue les y communs. On note cet ensemble $y_\cap = y_a \cap y_b$. L’ensemble des termes $y_\cap$ apportent des points pour les 2 joueurs en fonction de leur relation dans le graphe *JDM*. Du point de vue de la base de connaissances, ces $y_\cap$ communs impliquent une augmentation des poids des relations $x \ r_t \ y_\cap$. Le jeu a donc un impact direct sur le contenu de la base *JDM*.

	Joueur a (y_a)	Joueur b (y_b)
	monnaie	blé
	thune	espèces
	billet	pognon
	liquidité	monnaie
	pèze	fortune
	fonds	sous
	pognon	thune
		bifton
		grisbi

Soit $x \ r_t \ y$ avec $x = argent$ et $r_t = r_{syn}$

FIGURE 4 – Exemple de partie de *JDM*

Dans le cas de notre exemple, les joueurs a et b cherchent les meilleurs synonymes du terme *argent*. Ici, les termes *monnaie*, *thune* et *pognon* sont communs aux deux joueurs ($y_\cap$)⁵. En conséquence, à l’issue de la partie, on va augmenter le poids des relations suivantes :

- *argent* r_{syn} *monnaie* (280 $\rightarrow$ 285⁶)
- *argent* r_{syn} *thune* (140 $\rightarrow$ 145)
- *argent* r_{syn} *pognon* (146 $\rightarrow$ 151)

5. Les joueurs perdront des points si $y_\cap$ est vide, en gagneront d’autant plus qu’il y a de $y_\cap$ et qu’ils sont pertinents. Ils pourront même capturer le terme s’ils effectuent le meilleur score de tous les joueurs pour ce terme.

6. Le choix d’augmenter le poids de +5 vient de la nécessité de le distinguer de certains robots qui jouent au jeu et ne font augmenter que de +1 le poids de la relation.

2.4 Base de données *JeuxDeMots*

Les relations et les nœuds (termes) de *JDM* sont rassemblés dans une base de données dont le modèle entité-association est décrit en FIGURE 5. Chaque relation a un *id* unique, un nœud de départ et d'arrivée, un type et un poids. Chaque nœud a lui aussi un *id* unique ainsi qu'un nom (chaîne de caractère correspondant au mot ou au groupe de mots).

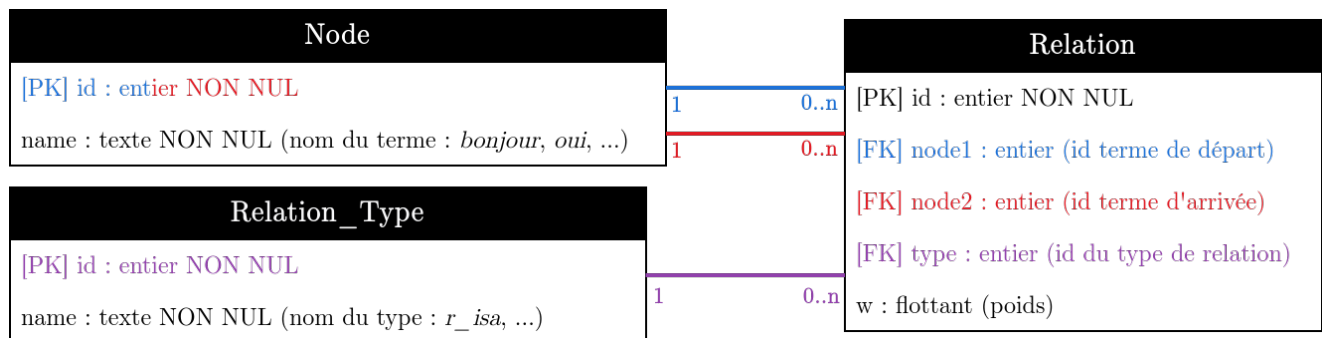


FIGURE 5 – Modèle entité-association simplifié de la base de données *JDM* (*PK* désigne une clé primaire, *FK* une clé étrangère)

3 Analogie et résolution de métaphore

On considère la métaphore comme une analogie à trou. Étudions dans un premier temps ce qu'est l'analogie et de manière plus détaillée le carré analogique formé par les termes qui la compose. Voyons ensuite en quoi la métaphore en est un type particulier et comment faire pour la résoudre.

3.1 Carré analogique

Une analogie (ou carré analogique) est un ensemble de 4 termes liés par des relations de similarités. La force d'une analogie procède de la similarité des termes qui la composent, son explicabilité se fonde sur les multiples relations entre ces termes (FIGURE 6).

On a la citation suivante :

« Ce qu'un terme est à un second, un troisième l'est à un quatrième (par exemple, ce que la science est à son objet, la sensation l'est au sensible) » (Aristote, *Topiques*, IV^e siècle av. J.-C.)

et on peut alors noter :

A est à B ce que C est à D (en langage naturel) $\iff A : B :: C : D$ (de manière plus concise)

Voici quelques exemples d'analogies

- que l'on peut trouver dans la littérature :
 - **vieillesse : vie :: soir : jour** (Aristote, *Poétique*, IV^e siècle av. J.-C.)
 - **sagesse : âge :: santé : corps** (La Rochefoucauld, *Maximes et réflexions morales*, 1664)
 - **charpentier : bois :: maçon : pierre** (Turney & Pantel, 2010)
 - **nid : oiseau(x) :: maison : humain(s)** (abris et êtres vivants) (Hilgert, 2016)
- inventés :
 - **Atomium : Bruxelles :: Tour Eiffel : Paris** (monuments et villes)
 - **renard : poule :: chat : souris** (prédateurs et proies)
 - **pomme : pommier :: poire : poirier** (arbres et fruits)

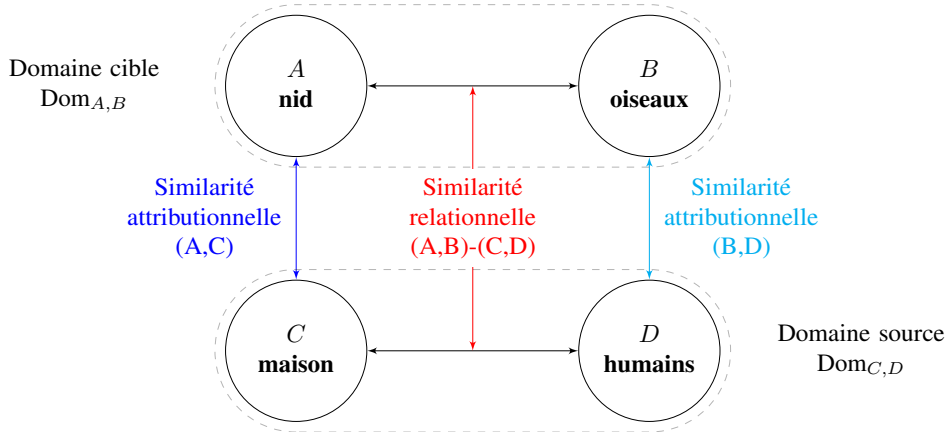


FIGURE 6 – Carré analogique et relations de similarité (**vieillesse : vie :: soir : jour**)

3.2 Analogie des relations

Dans l'explication de l'analogie, comme présenté en FIGURE 6 et détaillé en FIGURE 7, on distingue les relations impliquées dans la similarité relationnelle $(A, B) - (C, D)$ et celles impliquées dans l'une ou l'autre des similarités attributionnelles (A, C) et (B, D) .

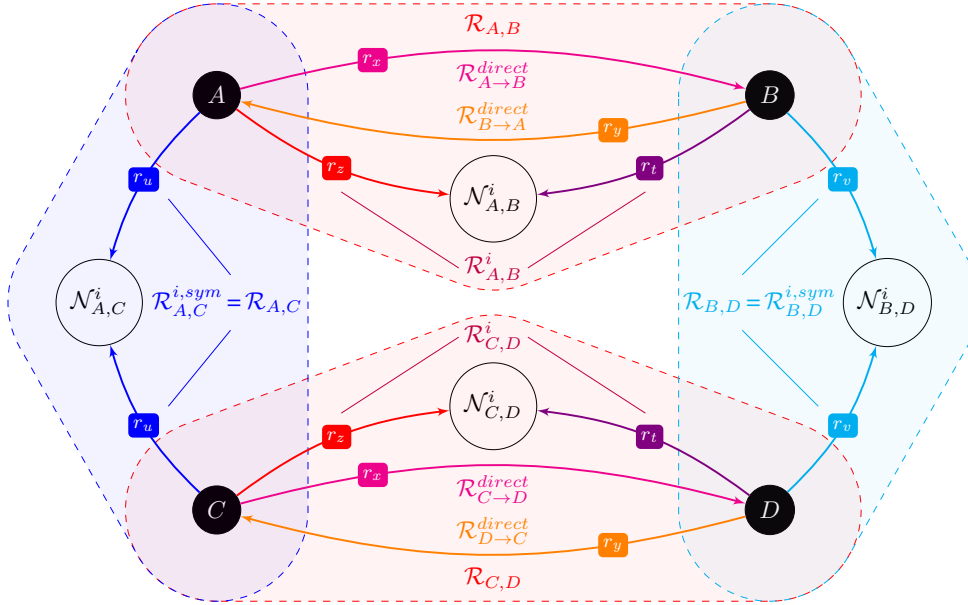


FIGURE 7 – Schémas de relations symétriques, directes et indirectes expliquant l'analogie (explications détaillées dans les sous-sections suivantes)

3.2.1 Similarité relationnelle

La similarité relationnelle $(A, B) - (C, D)$ s'explique et se quantifie par la présence de relations de même type (en plus ou moins grand nombre) entre les 2 éléments du domaine cible et ceux du domaine source.

Dans un premier temps, on étudie les relations directes et dans les 2 sens :

- $A \xrightarrow{r_x} B$ (domaine cible) et $C \xrightarrow{r_x} D$ (domaine source)
- $B \xrightarrow{r_y} A$ (domaine cible) et $D \xrightarrow{r_y} C$ (domaine source)

Du point de vue de la base de connaissances, on peut obtenir ces types de relations directes analogues dans chaque domaine via une requête SQL comme présentée en FIGURE 8. On ajoute quelques contraintes à celles évoquées précédemment telles que :

- ne considérer que les relations ayant un poids supérieur ou égal à 20 (pour filtrer les relations négatives et celles qui sont faibles)
- exclure certains types de relations (T_EXCLU) comme $r_{associated}$ qui est trop vague, r_{pos} qui est morpho-syntaxique et qui ne nous intéresse pas dans la résolution de ce problème ou d'autres relations de JDM qui n'ont pas leur utilité ici.

```

SELECT r_ab.type AS type
FROM Relation r_ab, Relation r_cd
WHERE r_ab.type=r_cd.type
AND r_ab.node1={A} AND r_ab.node2={B} AND r_ab.w>=20
AND r_cd.node1={C} AND r_cd.node2={D} AND r_cd.w>=20 -- A,B,C,D sont des variables
AND r_ab.type NOT IN {T_EXCLU} AND r_cd.type NOT IN {T_EXCLU};

```

FIGURE 8 – Requête SQL pour obtenir les types relations communes directes de A vers B et de C vers D (similarité relationnelle, relations directes, sens 1)

On étudie également les relations indirectes avec la présence de 2 nœuds intermédiaires :

- $A \xrightarrow{r_z} \mathcal{N}_{A,B}^i$ et $B \xrightarrow{r_t} \mathcal{N}_{A,B}^i$ (cible) ainsi que $C \xrightarrow{r_z} \mathcal{N}_{C,D}^i$ et $D \xrightarrow{r_t} \mathcal{N}_{C,D}^i$ (source)

Du point de vue de la base de connaissance, cela donne la requête présentée en FIGURE 9. De même que précédemment, on exclut des types de relations étant donné leur type et leur faible poids. De plus, on considère que chacun des nœuds intermédiaires doit être différent des 4 termes fixés.

```

SELECT r_ai.type AS type1, r_bi.type AS type2,
aib.name AS aib, cid.name AS cid
FROM Relation r_ai, Relation r_bi, Relation r_ci, Relation r_di,
Node aib, Node cid
WHERE r_ai.type=r_ci.type AND r_bi.type=r_di.type
AND r_ai.node2=r_bi.node2 AND r_ci.node2=r_di.node2
AND r_ai.node1={A} AND r_bi.node1={B} AND r_ai.node2=aib.id
AND r_ci.node1={C} AND r_di.node1={D} AND r_ci.node2=cid.id
AND aib.id NOT IN {(A,B,C,D)} AND cid.id NOT IN {(A,B,C,D)} -- A,B,C,D sont des variables
AND r_ai.w>=20 AND r_bi.w>=20 AND r_ci.w>=20 AND r_di.w>=20
AND r_ai.type not in {T_EXCLU} AND r_bi.type not in {T_EXCLU};

```

FIGURE 9 – Requête SQL pour obtenir les couples de types de relations communes indirectes entre A et B ainsi qu'entre C et D et les nœuds intermédiaires impliqués (similarité relationnelle)

Des cas particuliers peuvent se produire pour ces relations indirectes puisque $\mathcal{N}_{A,B}^i$ et $\mathcal{N}_{C,D}^i$ peuvent être communs formant un nœud intermédiaire commun que l'on formalisera $\mathcal{N}^i$, d'autre part, il peut arriver que $r_z = r_t$. On a alors les 3 cas particuliers possibles comme étant compris dans la première formalisation⁷ :

- $A \xrightarrow{r_z} \mathcal{N}^i$ et $B \xrightarrow{r_t} \mathcal{N}^i$ (cible) ainsi que $C \xrightarrow{r_z} \mathcal{N}^i$ et $D \xrightarrow{r_t} \mathcal{N}^i$ (source)
- $A \xrightarrow{r_z} \mathcal{N}^i$ et $B \xrightarrow{r_z} \mathcal{N}^i$ (cible) ainsi que $C \xrightarrow{r_z} \mathcal{N}^i$ et $D \xrightarrow{r_z} \mathcal{N}^i$ (source)
- $A \xrightarrow{r_z} \mathcal{N}_{A,B}^i$ et $B \xrightarrow{r_z} \mathcal{N}_{A,B}^i$ (cible) ainsi que $C \xrightarrow{r_z} \mathcal{N}_{C,D}^i$ et $D \xrightarrow{r_z} \mathcal{N}_{C,D}^i$ (source)

3.2.2 Similarité attributionnelle

Les similarités attributionnelles (A, C) et (B, D) peuvent quant à elles s'expliquer plus facilement (et avec une complexité plus faible) puisque l'on cherche des schémas de relation tels que :

- $A \xrightarrow{r_u} \mathcal{N}_{A,C}^i$ et $C \xrightarrow{r_u} \mathcal{N}_{A,C}^i$ pour la similarité attributionnelle (A, C)
- $B \xrightarrow{r_v} \mathcal{N}_{B,D}^i$ et $D \xrightarrow{r_v} \mathcal{N}_{B,D}^i$ pour la similarité attributionnelle (B, D)

À noter que les nœuds intermédiaires peuvent être confondus comme vu précédemment, y compris ceux de similarité attributionnelle, entre eux et avec ceux de similarité relationnelle. Côté base de

7. Pour le moment on ne considère pas que ces cas de nœuds intermédiaires communs et d'égalités des types de relations constituent une meilleure explication de la similarité relationnelle même si cela est parfois possible.

connaissances, on récupère les informations liées à l'explication de la similarité attributionnelle comme décrit en FIGURE 10. Celle-ci se concentre sur la similarité attributionnelle (A, C) mais le raisonnement est analogue pour la similarité attributionnelle (B, D). On filtre également dans ce cas les poids et les types de relations ainsi que la différenciation des nœuds intermédiaires de similarité attributionnelle vis-à-vis de A, B, C et D .

```

SELECT r_ai.type AS type, aic.name AS aic
FROM Relation r_ai, Relation r_ci, Node aic
WHERE r_ai.type=r_ci.type AND r_ai.node2=r_ci.node2
AND r_ai.node2=aic.id AND aic.id NOT IN {(A,B,C,D)} -- A,B,C,D sont des variables
AND r_ai.node1={A} AND r_ci.node1={C}
AND r_ai.w>=20 AND r_ci.w>=20 AND r_ai.type not in {T_EXCLU};

```

FIGURE 10 – Requête SQL pour obtenir les types de relations communes indirectes symétriques entre A et C et les nœuds intermédiaires impliqués (similarité attributionnelle (A, C))

Dans l'explication de la similarité attributionnelle, on étudie donc les relations intermédiaires symétriques de type r_t (voir FIGURE 11). Parmi celles-ci, il se peut que $r_t = r_{isa}$ et que les nœuds intermédiaires soient donc des hyperonymes communs et A et C sont alors co-hyponymes. On considère que la similarité attributionnelle est d'autant plus forte, par la présence en grand nombre de ces hyperonymes communs, d'autant plus si une relation r_{cohypo} est déjà présente dans JDM entre les 2 termes. Enfin, la présence d'une relation d'association directe ($r_{associated}$) entre les deux termes susceptibles d'être similaires attributionnellement amplifie également leur proximité dans une moindre mesure, tout en renforçant par la même l'analogie globale.

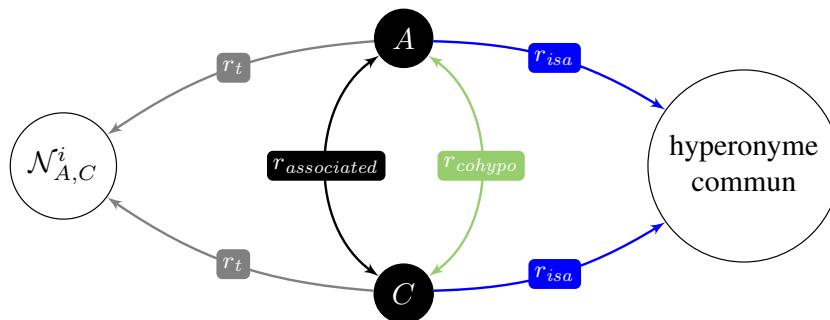


FIGURE 11 – Schémas de relations de cohyponymie, d'association et nœuds intermédiaires liés par des relations symétriques expliquant la similarité attributionnelle

3.3 La métaphore expliquée, une analogie à un trou

La classification proposée par (Ferrari, 1996) consiste à représenter "la mise en relation de concepts, sur des constructions syntaxiques récurrentes" que sont les métaphores et comparaisons. Cela se fait en utilisant "des attributs source, cible et tension indiquant, lorsque cela est possible, quels éléments de la structure sont porteurs respectivement de la source, de la cible et de la nature de la relation". Dans une comparaison, la cible et la source sont proches. Dans le cas d'une métaphore, "il s'agit de [résoudre le] lien entre la source et la cible".

Nous estimons que le positionnement des inconnues dans le carré analogique comme le montre la FIGURE 12 exprime les liens entre les 4 termes source et cible. On suppose alors que métaphores et comparaisons sont en fait des analogies dont certains termes deviennent des variables, variables qu'il faut résoudre si on veut fournir le carré complet et donc l'explication des métaphores et comparaisons dont il résulte. La rotation de la mise en inconnue d'un des termes de l'analogie explicite l'ensemble des possibilités pour créer des métaphores *in praesentia* (ou expliquées) en ayant 3 autres termes fixés (lignes 6,7,8,9 en FIGURE 12).

	A	:	B	::	C	:	D	Phrase	Type
1	a		b		c		d	a est à b ce que c est à d	Analogie
2	a				c			a est comme c	Comparaison
3			b				d	b est comme d	
4	a						d	a de d d de a	Métaphore
5			b		c			b de c c de b	
6	a		b		c			c de b est a	Métaphore <i>in praesentia</i>
7	a		b				d	d de a est b	
8	a				c		d	a de d est c	
9			b		c		d	b de c est d	

FIGURE 12 – Catégorisation de l'analogie à trou selon le positionnement des inconnues

3.4 Élection du meilleur candidat de la métaphore expliquée

Pour sélectionner les candidats possibles dans le cas de l'analogie à une inconnue, on regarde tout d'abord où se positionne l'inconnue. Le domaine opposé à celui où se trouve l'inconnue, c'est-à-dire celui complet, permettra de déterminer les relations que l'on doit retrouver afin d'élire les candidats.

Dans le cas de la FIGURE 13, on voit que le domaine cible est complet, c'est-à-dire sans terme inconnu, on étudie donc dans un premier temps les relations directes $A \xrightarrow{r_x} B$ puisque l'inconnue se situe en D , c'est-à-dire le nœud potentiellement similaire attributionnellement à B . On détermine l'ensemble des types r_x et on recherche ensuite les relations dans JDM telles que $C \xrightarrow{r_x} d$, la liste de d constitue alors les candidats de la similarité relationnelle directe. Si cette liste est vide, on étudie ensuite les couples de relations indirectes (r_z, r_t) tels que $A \xrightarrow{r_z} \mathcal{N}_{A,B}^i$ et $B \xrightarrow{r_t} \mathcal{N}_{A,B}^i$ côté cible puis on détermine la liste des d du côté source telle que $C \xrightarrow{r_z} \mathcal{N}_{A,B}^i$ et $d \xrightarrow{r_t} \mathcal{N}_{A,B}^i$. On obtient pour finir l'ensemble des candidats de similarité relationnelle pour le nœud inconnu.

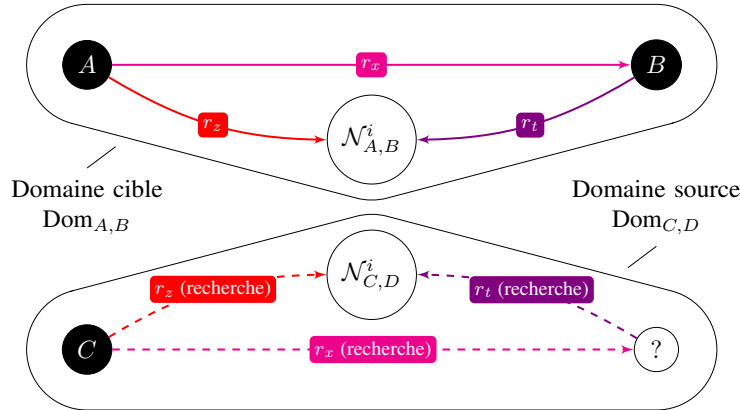


FIGURE 13 – Schémas de relations permettant de choisir les candidats dans le cas de la similarité relationnelle

Une fois ces candidats obtenus, il s'agit maintenant de déterminer un ordre de préférence et d'élire le meilleur en attribuant une $Force_d$ pour chaque candidat d . Cette force est en réalité celle de l'analogie qui complète la métaphore à trou avec d . Elle est exprimée en fonction de la force de la similarité relationnelle ($Force_{(A,B)-(C,d)}^{SimRel}$) mais aussi de celle de similarité attributionnelle dans laquelle d est impliquée ($Force_{(B,d)}^{SimAttr}$) :

$$\begin{aligned}
Force_d &= Force_{(A,B)-(C,d)}^{SimRel} \times Force_{(B,d)}^{SimAttr} \\
&= Force_{(A,B)-(C,d)}^{Rel} \times Force_{(B,d)}^{Attr} \times Force_{(B,d)}^{hyperonymes} \\
&= \frac{nbRel_{(A,B)-(C,d)}}{\forall d \max[nbRel_{(A,B)-(C,d)}]} \times \frac{nbAttr_{(B,d)}}{\forall d \max[nbAttr_{(B,d)}]} \times \frac{nbHyperCommun_{(B,d)}}{\forall d \max[nbHyperCommun_{(B,d)}]}
\end{aligned}$$

Le processus de calcul de la force du candidat pour l'analogie à trou a été choisi à force de tâtonnements. Il forme cependant une mesure plutôt cohérente car il parvient à classer convenablement les bons candidats (nous verrons à quel point cela est convenable dans la section 5). Il conviendra dans la suite de nos recherches de déterminer un calcul de force optimal par apprentissage. À noter que l'on veut également privilégier un candidat associé ($r_{associated}$) avec son similaire attributionnel et même davantage s'il est noté comme étant co-hyponyme ($r_{cohyponym}$). Voici en FIGURE 14 la liste des candidats par ordre de préférence, c'est-à-dire par force décroissante, pour les 4 analogies à une inconnue issues de l'analogie complète *charpentier : bois : : maçon : pierre*.

<i>? : bois : : maçon : pierre</i>				
candidat	F	F^{Rel}	F^{Attr}	F^{hyper}
charpentier ^a	.708	1	.654	.983
ébéniste	.619	1	.675	.917
menuisier	.618	1	.618	1
...				

<i>charpentier : ? : : maçon : pierre</i>				
candidat	F	F^{Rel}	F^{Attr}	F^{hyper}
bois ^{c a}	.324	1	.426	.161
marteau	.274	.5	.547	1
scie	.210	.5	.542	.774
niveau	.021	.5	.426	.097
équerre	.008	.5	.5	.032
scie à onglet	.008	.5	.5	.032

<i>charpentier : bois : : ? : pierre</i>				
candidat	F	F^{Rel}	F^{Attr}	F^{hyper}
maçon ^a	.716	1	.651	1
tailleur de pierre	.706	1	.706	1
...				

<i>charpentier : bois : : maçon : ?</i>				
candidat	F	F^{Rel}	F^{Attr}	F^{hyper}
pierre ^{c a}	.305	.5	.426	.833
ciment	.182	.5	.438	.833
brique	.181	.5	.362	1
plâtre	.176	.5	.422	.833
...				

FIGURE 14 – Résolution de l'unique inconnue dans les différentes configurations issues de l'analogie *charpentier : bois : : maçon : pierre* et classement des candidats par force décroissante (^c lorsque le candidat est lié par la relation de co-hyponymie avec son similaire attributionnel, ^a s'il y est associé)

4 Des inférences pour améliorer les performances

Dans la base de connaissances, inférer consiste à calculer une nouvelle relation à partir d'autres relations déjà existantes. On pourrait combler les relations manquantes dans l'ensemble de la base de connaissances mais on va plutôt inférer de manière locale. Dès qu'une relation est absente alors que sa présence ferait sens dans l'explication d'un carré analogique, on regarde s'il est possible de l'inférer avec un degré de confiance aussi grand que possible, auquel cas on la crée virtuellement (elle n'est pas mise dans la base) et on l'utilise dans l'explication de l'analogie. On pourrait par la suite la rajouter dans *JDM* car elle est visiblement manquante et qu'il sera plus rapide d'y accéder plutôt que l'inférer à nouveau dans une situation similaire.

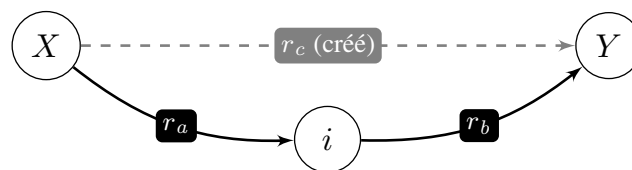


FIGURE 15 – Triangle inférentiel

Du point de vue de l'inférence logique, à partir de la présence de deux relations, $X \xrightarrow{r_a} i$ et $i \xrightarrow{r_b} Y$, on peut inférer que $X \xrightarrow{r_c} Y$. Cette relation r_c créée par inférence est ce que l'on appelle la base du

triangle inférentiel (voir FIGURE 15). On utilise uniquement certains types de relations pour r_a (voir FIGURE 16), ainsi on infère de manière plus précise en garantissant que les propriétés de r_b sont *a priori* vraies pour r_a .

#	Type d'inférence par ordre de préférence	r_a	r_b	r_c (créé)
1.	variante	$r_{variante}$	r_t	r_t
2.	synonymie stricte	r_{syn_strict}	r_t	r_t
3.	instanciation	$r_{is_instance_of}$	r_t	r_t
4.	lemmatisation	r_{lemma}	r_t	r_t
5.	déduction	r_{isa}	r_t	r_t
6.	raffinement	r_{raff_sem}	r_t	r_t
7.	synonymie	r_{syn}	r_t	r_t
8.	transitivité	r_t	r_t	r_t

#	X	r_a	i	r_b	Y	→	X	r_c (créé)	Y
1.	clé	$r_{variante}$	clef	r_{carac}	perdue		clé	r_{carac}	perdue
2.	auto	r_{syn_strict}	automobile	r_{agent_t}	rouler		auto	r_{agent_t}	rouler
3.	Tour Eiffel	$r_{is_instance_of}$	monument	$r_{patient_t}$	visiter		Tour Eiffel	$r_{patient_t}$	visiter
4.	avions	r_{lemma}	avion	r_{lieu}	aéroport		avions	r_{lieu}	aéroport
5.	mouche	r_{isa}	insecte	r_{has_part}	aile		mouche	r_{has_part}	aile
6.	corps	r_{raff_sem}	corps>anatomie	r_{make}	sueur		corps	r_{make}	sueur
7.	patauger	r_{syn}	nager	r_{action_lieu}	piscine		patauger	r_{action_lieu}	piscine
8.	merlan	r_{isa}	poisson	r_{isa}	vertébré		merlan	r_{isa}	vertébré

FIGURE 16 – Schémas d'inférence utilisés et exemples associés

Chaque triangle inférentiel peut subir une rotation en jouant avec la conversivité des relations. Les schémas inférentiels en FIGURE 17 sont donc équivalents.

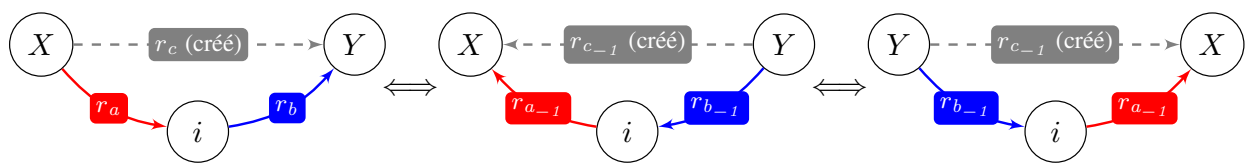


FIGURE 17 – Équivalence des schémas inférentiels

4.1 Déduction et induction

La déduction (ou inférence déductive) est un schéma d'inférence dont le raisonnement se fonde sur les propriétés du générique du terme étudié. Par exemple, si la mouche est un insecte (*mouche* r_{isa} *insecte*) et que les insectes ont des ailes (*insecte* r_{has_part} *ailes*) alors la mouche a des ailes (*mouche* r_{has_part} *ailes*). Ce raisonnement utilise la relation r_{isa} ou sa converse r_{hypo} dans le cas de la rotation du triangle inférentiel (voir FIGURE 18).

L'induction (chercher un spécifique, voir FIGURE 19) est quant à elle beaucoup moins certaine puisque le raisonnement consiste à prendre un cas particulier pour le généraliser à un hyperonyme. Pour prendre le même exemple que celui de la déduction, c'est comme si on disait que si la mouche a des ailes (*ailes* r_{holo} *mouche*) et que la mouche est un insecte (*mouche* r_{isa} *insecte*) alors les

insectes ont des ailes (*ailes* r_{holo} *insecte*). On s'en tiendra donc à la déduction puisque l'induction peut produire assez souvent des généralisations abusives. En effet, un hyperonyme peut plus facilement se substituer à un de ses hyponymes que l'inverse qui propage les propriétés de hyponyme à l'hyperonyme avec les risques d'erreur que cela comporte.

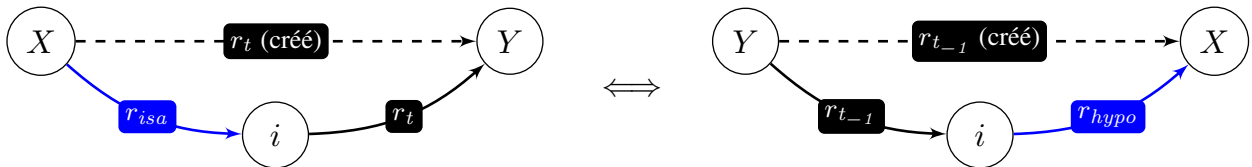


FIGURE 18 – Déduction et ses deux rotations sous la forme de triangle inférentiel

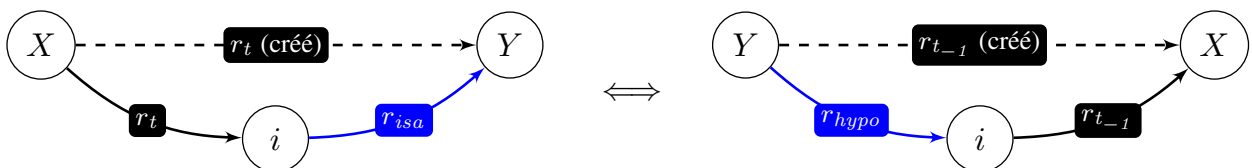


FIGURE 19 – Induction et ses deux rotations sous la forme de triangle inférentiel

4.2 Transitivité et annotations

La transitivité est une inférence où $r_a = r_b = r_c$. En traitant le cas des inférences 1. à 7. (FIGURE 16), on traite également les transitivités sur ces types r_a . Pour augmenter encore le nombre d'inférences effectuées, on pourra ajouter certains autres types de relations dont la transitivité est évidente comme r_{lieu} , r_{has_part} , r_{has_conseq} et leurs conversives. La préférence dépend du type de transitivité utilisé mais nous ne l'avons pas encore défini pour le moment.

Pour toute inférence effectuée, on doit s'assurer que les conditions sont réunies au niveau du typage des relations. Il faut aussi vérifier qu'aucune des relations impliquées, c'est-à-dire que r_a et r_b , ne soient ni *peu pertinente*, *non pertinente*, annotée comme une *exception* ou *contrastive*⁸. Il est également nécessaire que les poids de r_a et r_b soient à minima positifs voire supérieurs à 20.

Par exemple, l'autruche est un oiseau et les oiseaux peuvent voler (dans le sens déplacement aérien). On pourrait en déduire que l'autruche peut voler (*autruche* $r_{agent-1}$ *voler*) or on sait déjà qu'il existe les relations et annotations suivantes dans JDM :

- *autruche* $r_{agent-1}$ *voler* est négative et annotée comme étant *contrastive*
- *autruche* r_{isa} *oiseau* est *non spécifique*
- *oiseau* $r_{agent-1}$ *voler* est *probable* et *possible*

4.3 Types d'inférences préférées

Les relations $r_{variante}$ et r_{syn_strict} (ainsi que r_{syn}) sont symétriques ($r_t = r_{t-1}$). Ainsi, lorsqu'on procède à la rotation du triangle inférentiel, on obtient la situation présentée en FIGURE 20 où l'on peut intervertir la relation symétrique menant à l'inférence indifféremment en position r_a ou r_b (voir FIGURE 16). Il existe donc 2 manières d'inférer (tout en étant confiant sur la précision de l'inférence) avec des relations symétriques. De plus, les relations $r_{variante}$ et r_{syn_strict} sont très fortes puisque qu'elles lient des variantes orthographiques de termes ou des mots quasi identiques

8. Le fait qu'une relation soit contrastive signifie qu'il existe au moins un de ses raffinements pour lequel elle est fausse et au moins un pour lequel elle est vraie, ce qui fait douter de sa véracité pour le terme non-raffiné.

ce qui permet d'inférer avec un très fort degré de fiabilité. Avec r_{lemma} ⁹ ces types de relations constituent des inférences fiables et donc préférées à r_{isa} et $r_{is_instance_of}$.

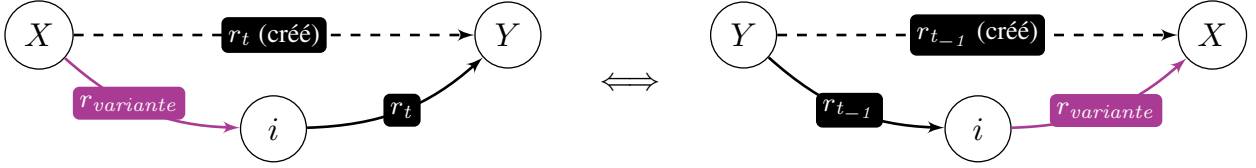


FIGURE 20 – Rotations du triangle inférentiel pour une relation symétrique

5 Influence du mécanisme inférentiel sur l'élection du candidat

On souhaite à présent évaluer si l'utilisation d'inférences conduit à élire le bon candidat plus souvent que dans les cas où on se passe d'inférence.

5.1 Positionnement des triangles inférentiels dans le carré analogique

Ces éventuelles inférences sont réalisées au moment de l'analyse des types de relations directes identiques r_x des deux côtés de la similarité relationnelle (voir FIGURE 21). D'autres inférences sont effectuées sur les r_y pour en apprendre davantage sur la similarité attributionnelle du côté de laquelle le candidat est impliqué.

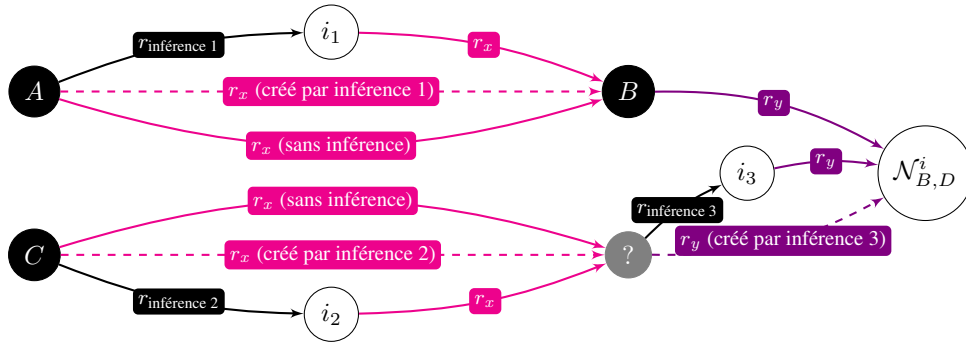


FIGURE 21 – Schémas d'inférence aidant à combler les manques dans la base de connaissance lors de la recherche de candidats de similarité relationnelle

On regarde dans un premier temps les $A \ r_x \ B$ qui existent, ce qui correspond au processus sans inférence jusque là. Ensuite, on complète ces r_x si on en trouve d'autres par inférence, c'est-à-dire si on trouve des relations telles que $A \ r_{inférence\ 1} \ i_1 \ r_x \ B$ ce qui nous autorise dans ce cas à créer la base du triangle inférentiel $A \ r_x \ B$, et à ajouter le type dans la liste s'il n'y était pas déjà. Comme vu en FIGURE 16, la relation r_a (nommée $r_{inférence}$ dans notre cas) peut prendre plusieurs types : $r_{variante} \succ r_{syn_strict} \succ r_{is_instance_of} \succ r_{lemma} \succ r_{isa} \succ r_{raff_sem} \succ r_{syn}$. De plus, si le type r_x le permet, on peut également inférer par transitivité, c'est-à-dire que $A \ r_x \ i_1 \ r_x \ B$ donne $A \ r_x \ B$.

Le but est ensuite de déterminer les candidats d possibles tels que $C \ r_x \ d$ pour l'ensemble des r_x déterminés précédemment (y compris ceux obtenus par inférence). On procède tout d'abord de manière directe comme en FIGURE 13. Puis on peut enrichir cette liste de candidats en procédant là aussi par inférence en trouvant des schémas de relations tel que $C \ r_{inférence\ 2} \ i_2 \ r_x \ d$ (pour les mêmes r_a trouvés à l'étape précédente, avec et sans inférence) ce qui nous autorise dans ce cas à créer la base du triangle inférentiel $C \ r_x \ d$ et ajouter les d en question dans la liste des candidats.

9. Si elle est unique dans le cas d'un terme polysémique

On peut également inférer sur les relations impliquées dans les similarités attributionnelles, en l'occurrence celles entre B et chacun des candidats d . Lorsque l'on a $B \text{ } r_y \text{ } \mathcal{N}_{B,D}^i$ et que l'on a pas $d \text{ } r_y \text{ } \mathcal{N}_{B,D}^i$, on peut tenter d'inférer grâce à $d \text{ } r_{\text{inférence 3}} \text{ } i_3 \text{ } r_y \text{ } \mathcal{N}_{B,D}^i$.

5.2 Analyse comparative des performances de l'élection sans et avec inférence

Après des tests effectués sur plus de 140 analogies (avec presque à chaque fois la rotation de l'inconnue sur chacun de ses 4 termes), on rassemble les données des 577 analogies à une inconnue testées en FIGURE 22.

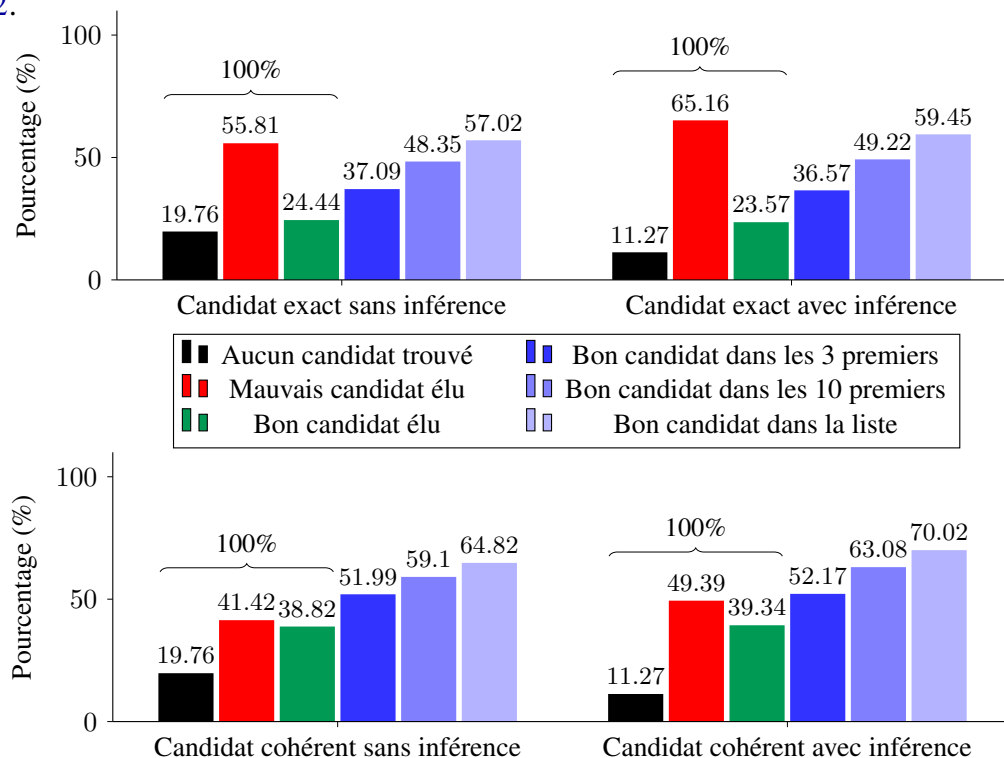


FIGURE 22 – Viabilité du candidat élu et positionnement dans la liste d'un bon candidat (ou candidat valide) pour la métaphore à une inconnue (pour 577 analogies à une inconnue testées)

On obtient la liste des candidats ordonnés par préférence sans utilisation d'inférence (diagrammes de gauche) et avec inférences (diagrammes de droite). On peut déterminer si la liste est vide (noir), si un bon candidat n'est pas dans la liste (rouge), s'il est dans la liste en première position et donc élu (vert) ou s'il est ailleurs dans la liste (nuances de bleu). On distingue le fait de trouver exactement le bon candidat (diagrammes du haut, effectué automatiquement) et trouver le bon ou un autre dont la substitution semble cohérente (diagrammes du bas, effectué à la main).

On constate tout d'abord que le fait de considérer qu'un candidat est cohérent au lieu qu'il soit exactement celui attendu permet d'augmenter la présence et l'élection d'un bon candidat. Cependant, il est nécessaire de regarder une par une les 577 analogies à trou pour vérifier : on pourra automatiser ce processus dans le futur en regardant la proximité sémantique du candidat théorique avec ceux proposés. L'utilisation d'inférences permet de faire très légèrement augmenter le nombre de fois où un candidat cohérent est élu (de 38.82% à 39.34%). Il est beaucoup plus notable cependant que la présence et le classement dans la liste d'un bon candidat est amélioré par l'utilisation de mécanismes inférentiels. Un candidat cohérent est présent dans la liste des candidats dans 70.02% des métaphores ainsi résolues (64.82% sans inférence) et dans 52.17% des cas il est dans les 3 premiers par ordre de préférence (51.99% sans inférence). L'utilisation d'inférence permet aussi de diminuer le nombre de fois où aucun candidat n'est proposé, en faisant passer ces cas de 19.76% à 11.27%.

Analogie à trou $A : B :: C : D$	Liste des candidats par ordre de préférence	
	Sans inférence	Avec inférence
1. hache : ? :: pioche : mineur	bûcheron > pompier > exécuter > ...	bûcheron > pompier > exécuter > ...
2. ? : industrie :: faucille : agriculture	marteau > tabac > mine > ...	marteau > tabac > mine > ...
3. riz : saké :: sucre : ?	rhum	rhum
4. Madrid : ? :: Rome : Italie	ville > commune > lieu > ...	Espagne > ville > commune > ...
5. lire : lecture :: ? : nourriture	nourrissant > nourrir > nourri > ...	ferme > poire > préparer > ...
6. tailleur : ? :: actrice : joue	∅	travailler > regarder > couper > ...
7. scorpion : ? :: chien : morsure	∅	∅

FIGURE 23 – Exemples de métaphores issues du corpus de 577 analogies à trou, comparaison de la résolution de l’inconnue sans et avec inférence sur les 3 premiers éléments des listes de préférence

Voici en FIGURE 23 des exemples de listes de candidats par ordre de préférence sur des analogies à un trou. On est face à plusieurs situations :

- **Métaphore 1.** : On trouve parfois les mêmes candidats dans le même ordre sans et avec inférence dont certains sont plus ou moins convaincants. *bûcheron* paraît être un très bon candidat tout comme *pompier* même s’il est moins manuel que le similaire attributionnellement *mineur*. Ensuite, *exécuter* puis *chevalier* et *bourreau* arrivent dans cet ordre. On se pose alors la question de savoir s’il s’agit de métiers dont l’outil de travail est la *hache*. On pourrait dire que oui même s’ils sont moins pertinents que le *bûcheron* ou le *pompier*.
- **Métaphore 2.** : Dans certains cas, le premier candidat est très convaincant comme *marteau*, mais les candidats suivants ne conviennent pas. Dans ce cas, *tabac* et *mine* ne sont pas des outils comme la *faucille* et ne sont visiblement pas de bons candidats.
- **Métaphore 3.** : La liste contient parfois un unique candidat identique avec et sans inférence. Ici, *rhum* est tout à fait valable et paraît d’ailleurs le meilleur.
- **Métaphore 4.** : La plupart du temps, l’utilisation d’inférences permet d’élire un bon candidat. C’est le cas d’*Espagne* dont la capitale est *Madrid* tout comme *Rome* est la capitale de l’*Italie*. Sans inférence, on avait uniquement les relations $r_{is_instance_of}$, r_{domain} , r_{lieu} depuis *Rome* vers *Italie* mais l’utilisation d’inférences permet d’apprendre que *Rome* r_{holo} *Italie*. De plus, les inférences sur la similarité attributionnelle entre *Rome* et le candidat *Espagne* permettent de faire d’augmenter le nombre d’hyperonymes communs, comme *capitale* et *ville* par exemple. Il y en a 30 dans le cas du candidat *Espagne* ce qui favorise son élection.
- **Métaphore 5.** : Il peut arriver qu’un bon candidat soit élu ou alors bien positionné dans la liste sans inférence mais que l’utilisation d’inférences fasse disparaître plus loin dans la liste un tel candidat voire n’en donne que des mauvais. On peut alors soupçonner un sur-apprentissage sans doute dû au fait que trop d’inférences ont été faites et que les bons candidats se sont noyés dans les mauvais. Il conviendra alors de regarder pourquoi l’inférence provoque cela et y ajouter des contraintes.
- **Métaphore 6.** : Les inférences permettent dans certains cas d’avoir une liste de candidats dont certains et même parfois le premier peuvent être considérés comme bons alors que la non-utilisation d’inférence n’a pas permis d’en trouver.
- **Métaphore 7.** : Il arrive que l’utilisation d’inférences conduise tout de même à une liste de candidat vide, dans ce cas on peut soupçonner un manque dans la base.

6 Conclusion

Le fait de trouver un candidat viable pour la métaphore avec l'utilisation d'inférences alors que le même algorithme sans inférence ne permet pas d'en trouver permet de mettre en évidence le manque de relations dans la base de connaissances. Il conviendra après vérification de les rajouter dans *JDM* puisqu'elles semblent pertinentes. À l'inverse, lorsqu'un mauvais candidat est proposé après l'utilisation d'une ou plusieurs inférences, c'est soit que le schéma d'inférence est faux, soit que les relations impliquées dans ces inférences peuvent être fausses ou incorrectement annotées. Notre système sert donc à expliquer des métaphores mais également à corriger de potentielles erreurs dans la base de connaissances et à l'enrichir. Nous pourrions améliorer notre système d'inférence dans le futur en cherchant de nouveaux schémas mais aussi en définissant mieux leurs priorités. À noter que ce système peut aussi servir à détecter des relations incohérentes non impliquées dans des inférences.

Nous projetons d'effectuer des résolutions d'analogies en utilisant d'autres techniques comme celles basées sur les vecteurs, des *embeddings* (plongements lexicaux) pour trouver le terme manquant de l'analogie à trou. Cela permettrait de quantifier le compromis (s'il faut faire un compromis) entre le choix d'utiliser une méthode explicite à forte précision qui s'appuie sur la base de connaissances et celui d'une méthode "boîte noire" à fort rappel. Cette méthode là serait plutôt évaluée grâce à des métriques plus traditionnelles de F-score. Il reste à évaluer dans quelle mesure il est pertinent de comparer des méthodes rationnelles (règles et connaissances) avec des méthodes intuitives (réseaux de neurones).

Il arrive que l'analogie soit absurde, notre programme s'acharne à chercher un candidat même si ça ne fait aucun sens, mais impossible de le savoir *a priori*. D'autre part, si l'analogie peut sembler absurde mais que l'analyse par le système permet de mettre au jour des relations pertinentes alors elle n'est plus si absurde finalement. Enfin, si l'analogie est basée sur des relations négatives comme dans le cas de *couscous : porc :: curry : bœuf* (le couscous ne contient pas de porc et le curry ne contient pas de bœuf), il n'est pas possible pour l'instant de trouver un candidat. Une solution possible serait de passer par des concepts négatifs ("ne pas contenir" dans l'exemple) activés par des relations positives.

Références

- CHAFFIN R. & HERRMANN D. J. (1984). The similarity and diversity of semantic relations. *Memory & Cognition*, **12**(2), 134–141. DOI : [10.3758/BF03198427](https://doi.org/10.3758/BF03198427).
- CONENNA M. & KLEIBER G. (2002). De La Métaphore Dans Les Proverbes. *Langue Française*, **134**(134), 58–77. Publisher : Armand Colin.
- FERRARI S. (1996). Traitement automatique des métaphores : une approche par marquage textuel. In *Actes du Deuxieme Colloque Jeunes Chercheurs en Sciences Cognitives*.
- GENTNER D. & HOLYOAK K. J. (1997). Reasoning and learning by analogy : Introduction. *American psychologist*, **52**(1), 32. Publisher : American Psychological Association.
- HILGERT E. (2016). L'analogie est-elle plus explicite que la métaphore ? *Langue française*, **189**(1), 67–86. Publisher : Armand Colin.
- HOFSTADTER D. & SANDER E. (2013). *L' Analogie, cœur de la pensée*. Odile Jacob. Google-Books-ID : S7x1DwAAQBAJ.

- KLEIBER G. (2023). Allées et venues au pays des métaphores nominales et verbales» ou «C’est la figure qui empêche de voir les traits. *Verbum : revue de linguistique*, **45**(1), 99–119. Publisher : Presses Universitaires de Nancy.
- LAFOURCADE M. (2007). Making people play for Lexical Acquisition with the JeuxDeMots prototype. In *SNLP’07 : 7th international symposium on natural language processing*, p.7.
- LEPAGE Y. (2003). *De l’analogie rendant compte de la commutation en linguistique*. PhD Thesis, Université Joseph-Fourier-Grenoble I.
- RAMADIER L., ZARROUK M., LAFOURCADE M. & MICHEAU A. (2014). Annotations et inférences de relations dans un réseau lexico-sémantique : application à la radiologie. In *21e Conférence sur le Traitement Automatique des Langues Naturelles*, Marseille, France.
- ROUX J., GUENOUNE H., LAFOURCADE M. & MOOT R. (2024). Explication de métaphores via la résolution d’analogies à l’aide d’un graphe de connaissances. In *JADT 2024 - 17es Journées internationales d’Analyse statistique des Données Textuelles*, volume 2, p. 813.
- RUSSELL S. (1988). Analogy by Similarity. In D. H. HELMAN, Éd., *Analogical Reasoning : Perspectives of Artificial Intelligence, Cognitive Science, and Philosophy*, p. 251–269. Dordrecht : Springer Netherlands. DOI : [10.1007/978-94-015-7811-0_12](https://doi.org/10.1007/978-94-015-7811-0_12).
- SHUTOVA E. (2010). Automatic metaphor interpretation as a paraphrasing task. In *Human Language Technologies : The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, p. 1029–1037.
- TURNERY P. D. & PANTEL P. (2010). From frequency to meaning : Vector space models of semantics. *Journal of artificial intelligence research*, **37**, 141–188.