

N° d'ordre: 3628

THÈSE

Présentée devant

devant l'Université de Rennes 1

pour obtenir

le grade de : DOCTEUR DE L'UNIVERSITÉ DE RENNES 1
Mention INFORMATIQUE

par

Sabri BAYOUDH

Équipe d'accueil : CORDIAL - IRISA

École Doctorale : Matisse

Composante universitaire : ENSSAT

Titre de la thèse :

*Apprentissage par
Proportion Analogique*

soutenue le 14 Novembre 2007 devant la commission d'examen

M. :			Président
MM. :	Stan	MATWIN	Rapporteurs
	François	YVON	
MM. :	Amedeo	NAPOLI	Examineurs
	Jacques	NICOLAS	
	Arnaud	DELHAY	
	Laurent	MICLET	

Table des matières

Remerciements	7
Introduction	9
I Présentation et situation du travail	13
1 Le raisonnement par analogie	15
1.1 Histoire et définition du concept d'analogie	15
1.1.1 Histoire du concept	15
1.1.1.1 Antiquité : Les Grecs	16
1.1.1.2 Moyen âge et Renaissance	17
1.1.1.3 Contemporains	18
1.1.2 Différentes définitions de l'analogie	19
1.1.3 Raisonnement par analogie	19
1.1.4 Conclusion	20
1.2 Analogie dans différents domaines	20
1.2.1 Architecture	21
1.2.2 Traitement d'image	23
1.2.3 Droit	23
1.3 Linguistique	25
1.4 Traitement automatique des langues	26
2 La proportion analogique et l'apprentissage	27
2.1 La proportion analogique	27
2.2 Apprentissage par proportion analogique	28
2.2.1 La proportion analogique en apprentissage artificiel	28
2.2.2 Exemple	29
2.3 La proportion analogique en Intelligence Artificielle	30
2.3.1 Récapitulatif	32

II	Théorie et Algorithmes	35
3	Proportion analogique exacte	37
3.1	Les axiomes de l'analogie	37
3.2	Distance cohérente avec l'analogie	38
3.3	Équations Analogiques	39
3.4	Proportion analogique sur les ensembles	40
3.4.1	Définition d'un ensemble fini cyclique	40
3.4.1.1	Propriétés de l'opérateur $+$ suivant l'analogie	40
3.4.2	Définition d'un ensemble fini par des attributs binaires	40
3.4.3	Transitivité de l'analogie dans les ensembles finis	42
3.4.4	Une distance dans un ensemble binaire	42
3.5	Proportion analogique dans un espace vectoriel	42
3.5.1	Une proportion analogique dans un espace vectoriel	42
3.5.2	La transitivité de l'analogie dans l'espace vectoriel	43
3.5.3	Un ensemble de distances cohérentes	43
3.6	Proportion analogique entre séquences	44
3.6.1	Notations	44
3.6.2	Une première définition	44
3.6.3	Une définition plus générale	45
3.6.3.1	Motivation	45
3.6.3.2	Une proportion analogique entre séquences	46
3.6.3.3	Relation entre les deux définitions	48
4	Résolution d'équations analogiques	49
4.1	Définition	49
4.2	Résolution d'équations analogiques dans les ensembles finis	50
4.2.1	Groupes finis cyclique	50
4.2.2	Ensembles finis binaires	50
4.3	Résolution d'équations analogiques dans un espace vectoriel	51
4.4	Résolution d'équations analogiques dans les séquences	51
4.4.1	Résolution selon Lepage	51
4.4.2	Résolution selon Yvon et Stroppa	52
4.4.3	Notre approche	53
5	La dissemblance analogique	55
5.1	Motivation	56
5.2	Dissemblance analogique dans les ensembles finis binaires	56
5.2.1	Définition	56
5.2.2	Exemple	57
5.2.3	Propriétés	58
5.3	Dissemblance analogique dans un espace vectoriel	58
5.4	Dissemblance analogique dans un espace métrique	60
5.5	Dissemblance analogique approchée entre séquences	61

5.5.1	La relation “ <i>est à</i> ”	61
5.5.2	La relation “ <i>comme</i> ”	62
5.6	Dissemblance analogique entre séquences	65
5.6.1	Définition	65
5.6.2	SEQUANA4	66
5.6.3	Remarques	67
5.6.3.1	Inégalité triangulaire	67
5.6.3.2	Comment ajouter - à l’alphabet ?	68
5.7	Dissemblance Analogique exacte vs. Approchée	68
5.8	Résolution généralisée des équations analogiques	69
5.8.1	Présentation	69
5.8.2	Résolution généralisée dans les ensembles finis	70
5.8.3	Résolution généralisée dans un espace vectoriel	70
5.8.4	Résolution généralisée dans les séquences	70
5.8.4.1	L’algorithme SOLVANA	70
5.8.4.2	Exemple	71
5.8.4.3	Recherche des k meilleures solutions	73
5.9	Expérimentations	73
5.9.1	Dispersion des triplets dans un espace vectoriel	73
5.9.2	Classification d’éléments de un espace vectoriel	74
5.9.3	Dissemblances analogiques entre séquences	74
5.9.3.1	Constitution de la base de données.	78
5.9.3.2	Protocole expérimental.	79
5.9.3.3	Résultats et premières conclusions.	80
III	Applications	83
6	Apprentissage d’une règle de classification par analogie	85
6.1	Description du problème	86
6.1.1	Le codage des données nominales	87
6.1.2	Le problème à deux classes	87
6.1.3	Problème multi-classes et valeurs manquantes	88
6.1.4	Exemple	89
6.2	Pondération des attributs	90
6.2.1	Matrice de pondération analogique	90
6.2.2	Apprentissage des poids	91
6.2.3	Algorithme	92
6.3	Recherche Rapide	93
6.3.1	Solution naïve.	93
6.3.2	Transformation de la dissemblance analogique	96
6.3.3	Les kd-trees	97
6.3.4	Algorithme <i>Branch and Bound</i>	98
6.3.5	Recherche rapide du plus proche voisin : algorithme AESA	98

6.3.5.1	Principe	98
6.3.5.2	Élimination	99
6.3.5.3	Sélection	99
6.3.5.4	Réduction du pré-calcul : <i>LAESA</i>	100
6.3.6	“FADANA” : FAsT search of the least Dissimilar ANAlogy	100
6.3.6.1	Calcul préliminaire	102
6.3.6.2	Principe de l’algorithme	102
6.3.6.3	Notations	102
6.3.6.4	Initialisation	102
6.3.6.5	Sélection	102
6.3.6.6	Élimination	103
6.3.6.7	Sélection des prototypes bases dans FADANA	103
6.3.7	Résultats de FADANA et FADANA*	105
6.4	Expérimentations et Résultats	106
6.4.1	Protocole expérimental	106
6.4.2	Résultats	110
6.4.3	Variation du paramètre k	110
6.4.4	Discussion et Perspectives	110
7	Expériences en génération de séquences.	113
7.1	Description du signal d’écriture manuscrite en ligne	114
7.1.1	Processus d’acquisition de données	114
7.1.2	Points d’ancrage	115
7.1.3	Vue d’ensemble	116
7.2	Génération basée sur l’analogie	116
7.2.1	Dissemblance analogique entre objets	117
7.2.2	Dissemblance analogique entre séquences	117
7.2.3	Matrice de pondération	118
7.2.4	Exemple	118
7.3	Génération à base de connaissances	118
7.3.1	Génération par distorsions d’images : échelle et pente	118
7.3.2	Génération par distorsions en ligne : vitesse et pente	119
7.3.2.1	Variation de la vitesse	120
7.3.2.2	Modification de la courbure	120
7.4	Expérimentations	120
7.4.1	Protocole expérimental	120
7.4.2	Exemples de caractères synthétiques d’écriture manuscrite	123
7.4.3	Résultats	123
7.4.4	Validité et significativité des tests	123
7.4.4.1	t-test	125
7.4.4.2	Sign test	126
7.4.4.3	Résultats	126
7.5	Conclusion	126

<i>Table des matières</i>	5
Conclusion	129
Bibliographie	135
Table des figures	137
Table des Tableaux	139
Table des Algorithmes	141

Remerciements

Si le travail de recherche présenté ici est un travail individuel et quelquefois solitaire, sa réalisation est indissociable des contributions de nombreuses personnes. L'occasion m'est ici offerte de les en remercier.

Je commence par les chefs, de toute façon je n'ai pas le choix.

Laurent Miclet, je te remercie le plus chaudement du monde pour ta disponibilité et la diligence de tes réponses, pour m'avoir aiguillé et m'avoir fait bénéficier de l'étendue de tes connaissances.

J'adresse aussi mes remerciements à Arnaud Delhay pour le soutien constant qu'il a apporté tout au long de ma thèse et pour son encadrement judicieux.

Je remercie sincèrement les membres du jury pour leurs commentaires sur ce mémoire, qui m'auront permis de l'améliorer. Je remercie en particulier François Yvon pour m'avoir fait l'honneur de rapporter et présider mon jury, Stan Matwin pour avoir accepté la lourde tâche de rapporter ce travail. Ma reconnaissance va également à Amedeo Napoli et Jacques Nicolas pour avoir accepté d'être membres de mon jury.

Une pensée va à Patrick Vannoorenberghe qui m'a initié à la recherche pendant le DEA.

Je voudrais aussi remercier mes collègues du projet CORDIAL qui par leurs conseils, leur aide et les nombreuses discussions que nous avons pu avoir, ont contribué à l'aboutissement de ce travail. Je remercie tout particulièrement : Ali, Pierre, Damien, Sylvie, Nelly, Joëlle qui ont activement œuvré à la bonne ambiance qui règne dans le labo.

Un grand merci s'adresse aux personnes avec qui j'ai eu la chance de travailler à travers des collaborations et notamment Harold, Goulven, José, Franck.

Je remercie sincèrement l'IRISA d'avoir financé ma thèse par une bourse INRIA.

À mes parents, Mohamed et Kadija, votre présence pour tout et toujours, votre amour contribue à mon Bonheur jour après jour,

À ma sœur Wafa, sans qui je ne serais pas tout à fait moi,

À ma grande famille, Jalloul Nejet et Haïthem, qui par leur soutien inconditionnel m'ont poussé vers le haut,

À mes amis, Adnene, Daly, Florence, Imen, Khaled, Majed, Neder, Riadh, qui par leurs actions tant sur le plan information que sur le plan humain m'ont beaucoup apporté,

je vous dédie cette thèse

Merci également à ceux que je ne cite pas ici mais à qui je dois beaucoup et qui sauront sans nul doute se reconnaître.

Introduction

*“A native talent for perceiving analogies is ...
the leading fact in genius of every order”*
William James [Jam90] [HJO⁺01]

Présentation

L’analogie est un mode de raisonnement qui a été étudié à travers l’histoire de la philosophie et qui a été largement utilisé dans le domaine de l’intelligence artificielle et de la linguistique. Nous nous restreignons dans ce mémoire au concept mathématique de “*proportion analogique*”, qui est une relation du type “*A est à B comme C est à D*”. La restriction porte sur le fait que A, B, C et D sont quatre objets appartenant au même univers. Nous ne traitons donc pas la notion de transfert ou de transport d’information entre univers différents. Cependant, nous partageons avec ces travaux l’idée fondamentale de l’analogie : il peut y avoir des relations similaires entre deux couples d’objets structurés même si les objets sont apparemment très différents.

Selon la nature des objets, une proportion analogique peut avoir plusieurs significations : sémantique, comme dans “*jument est à poulain comme vache est à veau*” ; morphologique comme dans “*abcd est à ecd comme abfab est à efab*”, etc. De cette variété résulte la difficulté et l’ambiguïté du travail sur l’analogie. Nous avons donc choisi, à travers ce document, de limiter le type d’analogie que nous traitons et de définir rigoureusement les axiomes, les propriétés et les définitions qu’elle vérifie. Nous nous intéressons en particulier aux proportions analogiques entre des séquences écrites sur un alphabet sur lequel est définie une proportion analogique.

Nous nous sommes aussi intéressés à définir un processus d’apprentissage supervisé ([Mit97] [CM02]), du type apprentissage paresseux (*lazy learning*). Ce qui veut dire que nous ne cherchons pas à extraire un modèle de l’ensemble d’apprentissage, mais à classer un nouvel objet par le seul examen des données d’apprentissage. La méthode de base en apprentissage paresseux est celle des plus proches voisins, qui nécessite une notion de distance. Nous introduisons dans ce travail une nouvelle mesure, appelée la dissemblance analogique, pour quantifier ce qu’il manque à quatre objets (y compris des séquences) pour être en proportion analogique exacte.

L’apprentissage par analogie sur les séquences a déjà été étudié, de manière un peu plus restreinte, et appliqué à des données linguistiques ([Yvo97] [Yvo99] [IH97a], etc). L’apprentissage par analogie a montré son utilité dans des applications comme

la conversion graphème-phonème ou la traduction automatique. Nous nous sommes intéressés dans ce travail, en ce qui concerne les séquences, à une autre application : les caractères manuscrits saisis en ligne.

Cadre du travail

Ce travail de thèse s'est déroulé au sein du projet Cordial de l'Irisa-Rennes, situé dans les locaux de l'ENSSAT, à Lannion. Le projet Cordial s'intéresse au Dialogue Oral Homme-Machine : Communication homme-machine, Apprentissage automatique, et Technologie vocale (reconnaissance et synthèse de la parole). Le travail réalisé au cours de cette thèse s'intègre dans l'apprentissage automatique pour des séquences et doit contribuer en particulier à la création de prosodie en synthèse de la parole.

Plan de la thèse

Ce mémoire de thèse est organisé en trois grandes parties. Les deux premiers chapitres fixent le contexte de notre travail en rappelant brièvement l'histoire du raisonnement par analogie en philosophie et en intelligence artificielle et en présentant quelques applications dans des domaines variés. Les trois chapitres qui suivent composent la seconde partie : ils traitent de la partie théorique de cette thèse, notamment des nouvelles notions que nous avons introduites comme la dissemblance analogique. La dernière partie, constituée des deux derniers chapitres, montre deux applications de nos travaux sur la proportion analogique, que se soit en apprentissage pour la classification ou en résolution pour la génération de caractères manuscrits. Nous détaillons ci-dessous le contenu de chacun de ces chapitres.

Chapitre 1 : Ce chapitre est consacré à une présentation de l'analogie et de son emploi. Il présente brièvement l'évolution du concept d'analogie de l'Antiquité au Moyen Age, jusqu'à l'époque contemporaine. La deuxième partie de ce chapitre introduit quelques applications de l'analogie dans différents domaines, en particulier la linguistique.

Chapitre 2 : Ce chapitre traite de manière plus précise de l'analogie ou de la proportion analogique telle que nous l'utilisons. Il aborde en premier lieu l'utilisation de la proportion analogique en apprentissage artificiel ; en second lieu, il confronte ses utilisations dans l'intelligence artificielle et ses rapports avec le raisonnement à partir de cas.

Chapitre 3 : Ce chapitre introduit les notions de base de la proportion analogique exacte. Nous y décrivons la manière dont se définit l'analogie dans différents domaines, dont les ensembles et les séquences.

Chapitre 4 : Dans ce chapitre, nous définissons une équation analogique et nous en décrivons les méthodes de résolution. Ce chapitre n'est consacré qu'à la résolution exacte pour des objets et des séquences.

Chapitre 5 : Dans ce chapitre, nous abordons le problème des équations analogiques approchées, qui par définition ne se résolvent pas exactement. Nous introduisons la notion de dissemblance analogique sur des ensembles structurés et les différentes propriétés que doit vérifier cette mesure. Nous développons en particulier la résolution généralisée d'une équation analogique dans les séquences.

Chapitre 6 : Ce chapitre accorde une attention particulière à l'apport du raisonnement par proportion analogique à un problème classique d'apprentissage de règles de classification. Il décrit la méthode d'apprentissage d'une règle de classification par analogie, dont le principe de base est semblable à celui de la méthode du plus proche voisin. La classification par analogie présentée dans ce chapitre ne traite que des objets binaires et nominaux.

Chapitre 7 : Dans ce chapitre, nous présentons la deuxième application de la proportion analogique qui est la génération de séquence. Ainsi, à partir de trois exemples de séquences d'écriture manuscrite, nous montrons comment générer des séquences synthétiques pour un meilleur apprentissage d'un classificateur d'écriture manuscrite en mono-scripteur.

Première partie

Présentation et situation du travail

Chapitre 1

Le raisonnement par analogie

Sommaire

1.1 Histoire et définition du concept d'analogie	15
1.1.1 Histoire du concept	15
1.1.2 Différentes définitions de l'analogie	19
1.1.3 Raisonnement par analogie	19
1.1.4 Conclusion	20
1.2 Analogie dans différents domaines	20
1.2.1 Architecture	21
1.2.2 Traitement d'image	23
1.2.3 Droit	23
1.3 Linguistique	25
1.4 Traitement automatique des langues	26

Contenu du chapitre

Ce chapitre traite de généralité concernant l'analogie. Nous parcourons d'abord la notion d'analogie à travers l'histoire. En effet, ce concept, analysé par les Grecs, a ensuite été repris au Moyen-âge et est encore de nos jours l'objet de débats philosophiques. Nous verrons par la suite différents domaines qui utilisent de manière opérationnelle l'analogie et le raisonnement par analogie.

1.1 Histoire et définition du concept d'analogie

1.1.1 Histoire du concept

Nous étudierons dans cette partie l'utilisation de l'analogie dans l'histoire et les différents sens qu'elle prend et qui parfois sont contradictoires, à travers quelques philosophes.

1.1.1.1 Antiquité : Les Grecs

Euclide, mathématicien grec (325 av.J.-C 265 av.J.-C), dans ses écrits rapportés par *Henrion* [Lep03], décrit l'analogie comme suit “*Tout ainsi que la comparaison de deux quantités entre elles, est dite raison, ainsi la comparaison et ressemblance de deux ou plusieurs raisons entre elles, est dite proportion : comme si la raison de A à B est semblable à la raison de C à D, l'habitude entre ces raisons sera dite proportion. Et c'est ce que les Grecs appellent analogie, et quelques Latins proportionnalité*”, notons que *raison* définit ici une mesure quantifiée et donc, d'après la définition précédente, l'analogie est une similitude de raisons. Sur un autre écrit, *Euclide* définit l'analogie comme une égalité de proportion entre grandeurs mathématiques “- *Des grandeurs sont dites être dans le même rapport, une première relativement à une deuxième et une troisième relativement à une quatrième quand des équi-multiples de la première et de la troisième ou simultanément dépassent ou sont simultanément égaux, ou simultanément inférieurs à des équi-multiples de la deuxième et de la quatrième, selon n'importe quelle multiplication, chacun à chacun, et pris de manière correspondante. - Et que les grandeurs qui ont le même rapport soient dites en proportion.*”¹.

Analogie entre figures géométriques : *Olympiodore* [Lep03] rapporte que *Platon*, philosophe grec disciple de *Socrate* (427 av.J.-C 348 av.J.-C), utilisait l'analogie en soutenant que “*ce que la médecine est à la cuisine, la justice l'est à la sophistique (rhétorique)*”². La comparaison entre médecine, cuisine, justice et sophistique est difficilement mesurable. On peut lire aussi que *Platon* utilise l'analogie pour définir le bien “*Ce que le bien est dans le domaine de l'intelligible à l'égard de la pensée et de ses objets, le soleil l'est dans le domaine du visible à l'égard de la vue et de ses objets*”³.

Aristote, philosophe grec (384 av.J.-C 322 av.J.-C), quant à lui définit l'analogie dans deux de ses écrits de deux manières différentes [Lep03].

1. la première, issue de la *Poétique*, introduit l'analogie comme une métaphore : “*J'entends par rapport d'analogie tous les cas où le second terme est au premier comme le quatrième au troisième, car le poète emploiera le quatrième au lieu du second, ou le second au lieu du quatrième ; et quelquefois aussi, on ajoute le terme auquel se rapporte le mot remplacé par la métaphore. Pour m'expliquer par des exemples, il y a le même rapport entre la coupe et Dionysos [dieu des jonctions des opposés et des ambiguïtés] qu'entre le bouclier et Arès [dieu de la guerre] ; le poète dira donc de la coupe qu'elle est le bouclier de Dionysos et du bouclier qu'il est la coupe d'Arès. De même, il y a le même rapport entre la vieillesse et la vie qu'entre le soir et le jour ; le poète dira donc du soir, avec Empédocle, que c'est la vieillesse du jour, de la vieillesse que c'est **le soir de la vie** ou le couchant de la vie*”. Rappelons au passage que le mot métaphore est constitué de **meta** (entre ou parmi en grec) et **pherien** (porter) qui signifie un transfert de quelque chose à un autre. Et que le mot analogie est constitué de **ana** (de nouveau, de bas en haut) et **logia** (rapport, proportion) qui signifie même relation, rapport égal ;

¹Euclide, Éléments de géométrie, V, traduction B. Vitrac

²Olympiodore, Prolégomènes à la philosophie de Platon, 1990, XI, 27, 10, p 43

³livre VI de La République

2. la seconde, issue de *l'Éthique à Nicomaque*, définit l'analogie avec une mesure de quantité : “*La proportion étant une égalité des rapports et supposant quatre termes au moins.*
 - *Que la proportion discontinue implique quatre termes cela est évident, mais il en est de même aussi pour la proportion continue, puisqu'elle emploie un seul terme comme s'il y en avait deux et qu'elle le mentionne deux fois ; par exemple, ce que la ligne A est à la ligne B, la ligne B l'est à la ligne C, la ligne B est donc mentionnée deux fois, de sorte que si l'on pose B deux fois, il y aura quatre termes proportionnels.*
 - *Et le juste, donc, implique quatre termes au moins et le rapport est le même, car la division s'effectue d'une manière semblable entre les personnes et les choses”.*

Certains pensent que l'analogie est un mode de raisonnement, qu'il nous aide à déduire des propositions qui ne sont d'ailleurs pas toujours vraies, et même quand elles le sont l'analogie ne forme pas une preuve suffisante. L'analogie est donc un outil d'explication, d'éclaircissement mais aucunement un outil de démonstration ou de justification. D'autres diront que lorsque l'analogie se trompe, cela s'appelle une anomalie (absence de régularité) et que ce n'est donc pas un argument qui pêche contre l'analogie. Pour Varron, écrivain et savant romain (116 av.J.-C 27 av.J.-C) [Lep03], analogie et anomalie ne peuvent aller l'un sans l'autre : “... on ne saurait rejeter ni l'analogie ni l'anomalie. Ceci pour la même raison que l'homme n'est pas qu'une âme, mais à la fois un corps et une âme.”⁴.

1.1.1.2 Moyen âge et Renaissance

La formulation médiévale de l'analogie porte sur l'analogie de l'être qui traite de la problématique de l'unité du sujet de la métaphysique.

D'après [SB02]⁵, Albert le Grand, philosophe, théologien, chimiste et naturaliste germanique (≈1200 - 1280), définit au XIII^{ème} siècle l'analogie comme le mode selon lequel l'être créé entre dans la substance des choses, tout en existant en certaines de façon primordiale et en d'autres de façon dérivée. L'analogie théologique se voit seule confier la tâche d'exprimer le rapport du créé et de l'incrée : le créé est dit être “analogiquement” ce que Dieu est selon son essence ou substance en tant qu' “il y participe en l'atteignant autant qu'il le peut”. En d'autres termes, le rapport des créatures à Dieu est analogique au sens strict où chaque créature a en elle, selon sa propre vertu, une analogie, une similitude de ce qui est en Dieu.

On peut voir aussi cette même idée et cette même relation qui lie Dieu et l'utilisation de l'analogie dans les pensées de Thomas d'Aquin [Rob05], théologien et philosophe italien (≈1225 - 1274), quand il essaye d'analyser des expressions de type “Dieu est bon” ou encore “Dieu est sage” en plaçant l'analogie entre l'équivocité et l'univocité : “il reste donc que ce qui est dit de Dieu et des autres choses n'est prédiqué ni de manière univoque ni de manière équivoque, mais de manière analogique, c'est-à-dire d'après un

⁴De lingua latina, 1954, livre IX, paragraphe 1

⁵Chapitre 8 ,pp.213-236

ordre ou un rapport à quelque chose d'unique"⁶.

Dans le même sens, *Kant*, philosophe allemand ($\approx 1724 - 1804$) [Rob05], approprie à l'analogie un pouvoir de produire un résultat inédit en s'appuyant toujours sur cette relation entre l'être et son créateur : "L'analogie ne signifie pas, comme on l'entend d'ordinaire, une ressemblance imparfaite entre deux choses, mais une ressemblance parfaite de deux rapports entre des choses tout à fait dissemblables. Ainsi, il y a une analogie entre le rapport juridique d'actions humaines et le rapport mécanique des forces motrices : je ne puis en aucun cas faire quelque chose contre autrui sans lui donner un droit d'en faire autant contre moi dans les mêmes conditions ; exactement comme un corps ne peut agir sur un autre avec sa force motrice, sans être, par là même, cause que l'autre réagisse sur lui dans la même mesure. Dans ce cas, droit et force motrice sont choses entièrement dissemblables, mais il y a pourtant une parfaite ressemblance dans leur rapport. Grâce à une analogie de ce genre, je puis donner un concept du rapport qu'entretiennent des choses qui me sont absolument inconnues. Par exemples ce que la proportion du bonheur des enfants = a , est à l'amour que les parents leur portent = b , la prospérité du genre humain = c , l'est à cette inconnue = x en Dieu que nous appelons amour divin ; non que ce dernier ait la moindre ressemblance avec une inclination humaine quelconque, mais nous pouvons poser son rapport au monde comme semblable à celui que les choses du monde entretiennent entre elles"⁷, *Kant* fait d'ailleurs de l'analogie l'une des opérations essentielles et indispensables de l'entendement, en la présentant comme l'une des deux espèces de raisonnement de la faculté de juger.

1.1.1.3 Contemporains

Le philosophe *Edgar Morin* [Sca03], définit l'analogie de la façon suivante "*Livrée à elle-même, l'analogie erre, vagabonde, voyage, traverse sans entraves frontières, espaces et temps. Elle porte en elle, potentiellement erreur, délire, folie, raisonnement, invention, poésie. Elle a besoin, dès qu'elle s'applique à la pratique, d'être testée, vérifiée, réfléchie, et doit entrer en dialogique avec les procédures analytiques/logiques/empiriques de la pensée rationnelle. La rationalité véritable ne réprime pas l'analogie, elle s'en nourrit tout en la contrôlant. Il peut y avoir dérèglement de la navette analogique-logique, l'excès analogique et l'atrophie logique conduisent au délire ; mais l'hypertrophie logique et l'atrophie analogique conduisent à la stérilité de la pensée*"⁸.

Michel Conan [Sca03], philosophe, pense de l'analogie que c'est un mode de raisonnement qui nous permet de clarifier et de simplifier des procédés complexes, elle permet en outre d'avancer sur une démarche restructurée par son intégration dans un nouveau contexte intellectuel grâce au raisonnement par analogie. "*Dans tous les cas l'analogie s'est avérée fructueuse, qu'il s'agisse des arts ou des sciences, la forme originale a été profondément transformée par son intégration dans un nouveau contexte intellectuel. Au contraire, lorsque l'analogie est maintenue strictement et aboutit au déplacement d'une forme d'un domaine intellectuel dans un autre, elle devient stérile et bloque tout proces-*

⁶Thomas d'Aquin, Somme contre les gentils, 34, trad. C. Michon, p. 227-229

⁷Kant, Prolegomènes à toute métaphysique future, 58

⁸Troisième volume de La méthode, La connaissance de la connaissance, 1986

sus de création intellectuelle et donc de progrès de la connaissance [...] Pour vaincre la difficulté principale des problèmes de design qui consiste à transformer des problèmes très complexes en problèmes simples, le recours à des analogies permet un recodage et une restructuration des représentations que l'on a du problème, et présente en outre l'avantage de suggérer une démarche familière qui permet d'approcher ce qui paraissait totalement étranger au premier abord en faisant apparaître sur de nouveaux objets des aspects inattendus du terme familier de l'analogie."⁹. Ainsi Conan voit le raisonnement par analogie comme une relation entre deux domaines source et cible.

Alors que *Philibert Secretan* [Sca03], voit l'analogie comme une structure de pensée composée de trois éléments essentiels "*pour montrer que l'analogie n'est pas un mixte hybride d'images et de raisonnements, mais une structure de pensée et d'être, il convient de se réinterroger sur le statut tripartite de l'analogie : proportion - ressemblance - transgression*"¹⁰. Il considère donc ce que d'autres appellent anomalie comme une partie intégrante de l'analogie et refuse de réduire sa définition à une simple copie de formes, de l'ordre d'un mimétisme.

1.1.2 Différentes définitions de l'analogie

Les concepts qui découlent de l'histoire de l'analogie peuvent être résumés ci-dessous :

- C'est une ressemblance perçue comme non fortuite entre deux éléments [Wik05].
- C'est un rapport d'ordre physique, intellectuel ou moral qui existe à certains égards entre deux ou plusieurs choses différentes [AP32].
- C'est un rapport de ressemblance, d'identité partielle, entre des réalités différentes préalablement soumises à comparaison (trait(s) commun(s) aux réalités ainsi comparées, ressemblance bien établie, correspondance) [ATI].
- Il y a analogie (ou proportion) lorsque le second nom est au premier comme le quatrième est au troisième. Car on dira le quatrième à la place du second et le second à la place du quatrième¹¹ [Lep03].
- L'analogie prédictive est celle qui permet la résolution de problèmes en transférant les connaissances tirées de problèmes suffisamment similaire déjà résolus [JGCM83].

1.1.3 Raisonnement par analogie

Les différents points de vue du raisonnement par analogie dans l'histoire sont énoncés dans ce qui suit :

- Mode de raisonnement permettant de comprendre et d'interpréter de nouvelles situations à partir de situations analogues. Le raisonnement analogique consiste à utiliser les ressemblances et les dissemblances entre deux domaines, appelés la source et la cible, pour transférer les résultats connus de la source vers la cible. Le raisonnement analogique est très souvent confondu avec une autre forme,

⁹Conan, 1990, p.19

¹⁰Secretan, 1984, p.121

¹¹Aristote, la Poétique

semblable, développée par les chercheurs en IA : le raisonnement par cas. Dans le raisonnement par cas, les similarités entre la source et la cible sont notées, puis les résultats connus dans la source sont appliqués, à la similarité près, à la cible. Dans le raisonnement analogique, on doit remarquer en plus l'existence de relations de dépendance interne au sein de la source, appelées relations causales, et transférer aussi ces relations causales dans la cible [qdllf06].

- On raisonne par analogie lorsque l'on se sert des ressemblances entre problèmes ou entre données pour trouver des solutions. Le raisonnement typique est : A est à B ce que C est à D . Ce mode de raisonnement est certainement très employé par l'homme. Il est productif, mais non dépourvu d'ambiguïté : un est à deux ce que trois est à quatre (son prédécesseur) ou ce que trois est à six (sa moitié). L'intelligence artificielle a mis en évidence la nécessité de l'existence simultanée de relations de causalité, et définit l'analogie plutôt comme un raisonnement qui combine similarité et causalité [qdllf06].
- Méthode de raisonnement qui consiste à passer d'une relation donnée à une relation partiellement semblable et partiellement différente (GOBLOT 1920). Au sens large : opération mentale consistant à conclure d'une ressemblance à une autre ressemblance (FOULQ.-ST-JEAN 1962) [ATI].
- Raisonner par analogie, former un raisonnement fondé sur les ressemblances ou les rapports d'une chose avec une autre. On dit de même conclure, juger par analogie. Être guidé par l'analogie. Se dit particulièrement, en termes de grammaire, du rapport qu'ont entre eux certains mots ou certaines lettres. Il y a de l'analogie entre le B et le P, consonnes labiales, le D et le T, consonnes dentales, etc. Les mots nouveaux ne s'introduisent d'ordinaire qu'à l'aide de l'analogie [AP32].

1.1.4 Conclusion

D'après ces différentes définitions d'analogie et du raisonnement par analogie, on peut d'ores et déjà dire que ces termes ont été utilisés dans plusieurs domaines et que le sens n'est pas le même à chaque fois et peut même être contradictoire. Ainsi, par exemple, l'analogie mathématique signifie le rapport exact et rigoureux (deux est à trois ce que six est à neuf). En revanche, l'analogie philosophique se dit des rapports plus ou moins éloignés, ou même de la similitude (l'analogie du fer avec l'aimant) [AP98].

1.2 Analogie dans différents domaines

L'analogie et le raisonnement par analogie ont été beaucoup utilisés dans différents domaines de la recherche (Math [TLFU05], Droit, Gestion [SB02], IA [KF03], Traitement d'image, Traitement Automatique des Langues, Linguistique, Architecture ...). Dans cette section, nous allons présenter les différentes utilisations de l'analogie ou du raisonnement par analogie en se restreignant à quelques domaines d'applications.



FIG. 1.1 – Le Parthénon à Athènes

 (image issue de <http://serge.mehl.free.fr/chrono/Fibonacci.html> le 30/08/2007)

1.2.1 Architecture

On trouve déjà dans la période de la Grèce antique l'utilisation de l'analogie dans l'architecture. Ainsi, différentes parties du fronton des temples grecs sont conçues pour produire une analogie harmonieuse de mesures 1.1, car ils pensaient que la proportion analogique était porteuse d'une harmonie et d'une perfection propre. Cette analogie harmonieuse de mesures est assurée par ce qui a été appelé "segment doré". Le segment doré est une manière de découper un segment de longueur arbitraire a , de telle sorte que le plus grand segment ainsi construit, de longueur b , soit au petit segment délimité ce que le segment total est au grand segment [Rob05] ($\frac{a}{b} = \frac{b}{a-b}$). La solution positive de l'équation précédente est $\frac{a}{b} = \frac{1+\sqrt{5}}{2}$, appelée aussi nombre d'or. On peut donc voir sur la figure 1.1 que le temple semble respecter, dans sa façade en particulier, fronton compris, la fameuse divine proportion, représentée ici par le rectangle intérieur rouge.

On retrouve aussi ce fameux nombre d'or dans le temple d'*Andros* découvert sous la mer des Bahamas¹², ou encore dans la pyramide de *Khéops* qui a des dimensions qui mettent en évidence l'importance que son architecte attachait au nombre d'or. En effet, le rapport de la hauteur de la pyramide de *Khéops* par sa demi-base est le nombre d'or. On retrouve aussi le sculpteur grec *Phidias* qui utilisait le nombre d'or pour décorer le *Parthénon* à Athènes, en particulier pour sculpter la statue d'*Athéna Parthénos* ...

De nos jours, l'analogie et l'architecture sont restées liées. Ainsi, raisonner par analogie est la démarche de transfert de connaissances d'éléments sources connus et familiers à des éléments cibles. Prenons l'exemple de l'Institut du Monde Arabe (IMA) à Paris [Sca03] sur lequel on peut voir une démarche analogique. On peut voir sur la figure 1.2 l'analogie entre le bâtiment de l'IMA d'un côté et l'appareil photo ou l'œil d'un autre côté. On trouve aussi le moucharabieh, un dispositif analogue, de part son aspect et son fonctionnement, au diaphragme d'un appareil photo et d'un iris, qui permet de filtrer la lumière et de voir l'extérieur sans être vu de l'intérieur du bâtiment.

¹²voir http://trucsmaths.free.fr/nombre_d_or.htm pour plus de détail, site visité le 07/09/2007

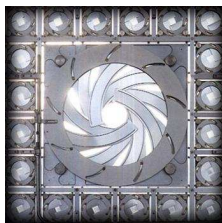


FIG. 1.2 – L’Institut du Monde Arabe à Paris de Jean Nouvel (1981-1987) [[Sca03](#)]

1.2.2 Traitement d'image

Dans le travail auquel nous faisons référence ici, le but est de créer à partir d'une image cible non filtrée une autre image cible filtrée, en utilisant deux images sources qui représentent respectivement un même objet avant et après filtrage [HJO⁺01]. Cela permet donc d'obtenir une image cible filtrée de façon analogue aux images sources. En choisissant donc différents types d'images sources, nous obtenons plusieurs images cibles filtrées par différents types de filtres tels que :

- *les filtres d'image traditionnels* : comme le flou et l'emboutissage (fait d'arrondir).
- *la synthèse de texture améliorée* : dans laquelle la texture est synthétisée avec une plus grande qualité qu'avec les autres approches.
- *la super résolution* : dans laquelle les images à haute résolution sont inférées à partir d'images à basse résolution.
- *le transfert de texture, les filtres artistiques, ...*

Nous voulons donc trouver une image B' qui est reliée à l'image B de la même façon que l'image A' est reliée à l'image A . L'un des avantages d'utiliser l'analogie est qu'au lieu de choisir une multitude de filtres avec ces paramètres respectifs, l'utilisateur n'a qu'à prendre un exemple de transformation illustré sur une image et d'appliquer la même transformation à l'image cible.

Nous n'allons pas détailler davantage cette méthode de traitement d'image par analogie (pour de plus ample information voir [HJO⁺01]) et nous nous limiterons à quelques exemples pour illustrer l'apport de l'analogie dans ce type d'applications, voir Figure 1.3.

1.2.3 Droit

L'analogie joue un rôle central dans le raisonnement légal. Ainsi, un juge va s'appuyer sur un jugement précédent pour donner le jugement du cas que la loi ne traite pas directement. Mais cela n'est pas toujours évident, prenant le cas suivant traité dans [Hun04] : Un plaignant s'est fait voler ses bagages de sa chambre à bord d'un ferry. Supposons qu'il n'y ait aucune déclaration statutaire sur le sujet. Le juge a deux analogies possibles

1. L'avocat du plaignant utilisera l'analogie du ferry avec un hôtel (les cabines sont au ferry ce que les chambres sont à l'hôtel, le restaurant, le fait de dormir à bord ...). Il dira donc que la compagnie du ferry est responsable du vol dans les chambres comme c'est le cas dans un hôtel.
2. La défense utilisera l'analogie du ferry avec un train car l'intention première du passager est de voyager et non de passer la nuit, de plus les trains aussi ont des restaurants ... Donc la compagnie du ferry n'est pas responsable de la perte du bagage du plaignant comme c'est le cas dans un train.

Le raisonnement analogique est celui qui répond par excellence à l'hypothèse du silence de la loi et permet de ce fait son adaptation à des situations nouvelles (*V. supra* n 47). L'encyclopédie polonaise *WIEM* donne la définition suivante du mot analogie : “*méthode de raisonnement juridique - établissement des conséquences juridiques d'un*

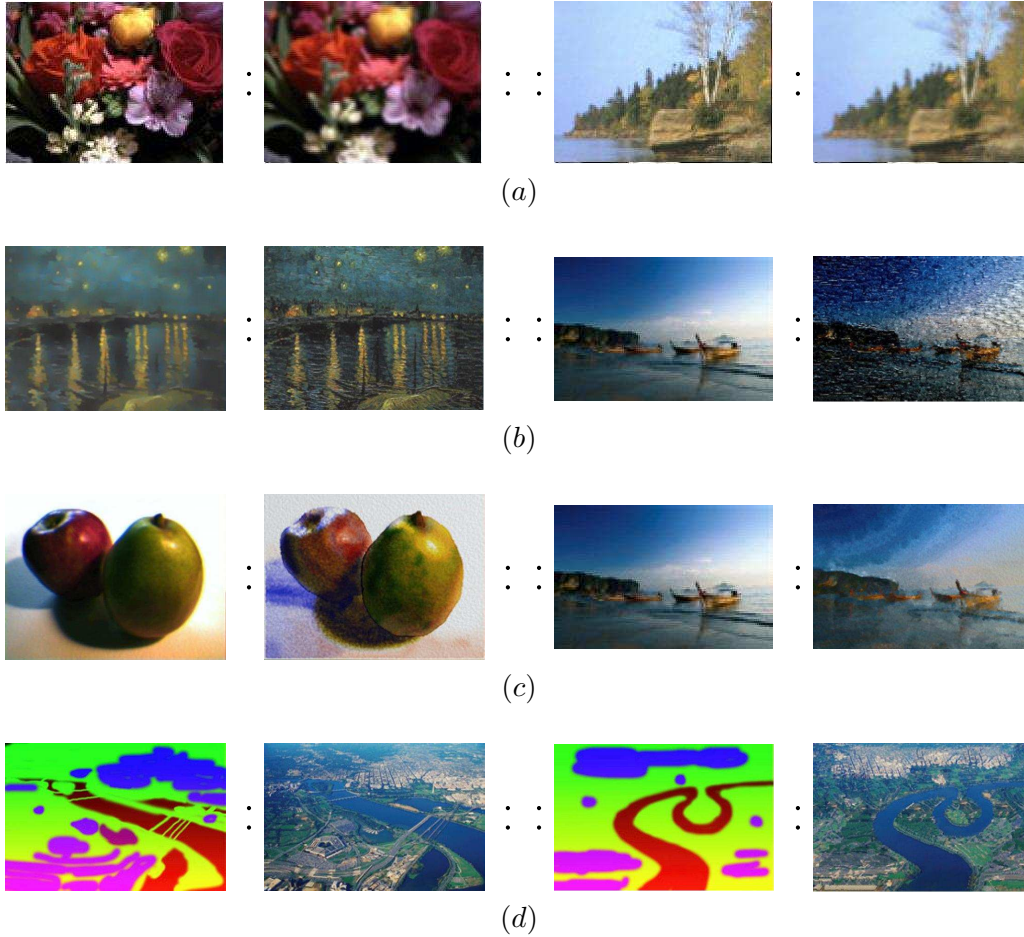


FIG. 1.3 – Traitement d’images par analogie [HJO⁺01] :

- (a) $\left\{ \begin{array}{l} \text{l'image 2 est le résultat du filtrage de l'image 1 par un filtre flou.} \\ \text{l'image 4 est l'image obtenue par analogie.} \end{array} \right.$
- (b) $\left\{ \begin{array}{l} \text{l'image 1 représente le filtrage par "flou intelligent" de l'image 2.} \\ \text{l'image 2 est le tableau de Van Gogh, "starry Night above the Rhône".} \\ \text{l'image 4 est l'image transformée avec le style de Van Gogh.} \end{array} \right.$
- (c) $\left\{ \begin{array}{l} \text{l'image 2 est l'image filtrée par un filtre "watercolor".} \\ \text{l'image 4 est l'image transformée par analogie avec le style "watercolor".} \end{array} \right.$
- (d) $\left\{ \begin{array}{l} \text{l'image 1 est la reconstruction par filtrage de l'image 2.} \\ \text{l'image 2 est l'image aérienne d'une ville.} \\ \text{l'image 4 est la reconstruction de la texture de l'image 3 par analogie.} \end{array} \right.$

fait non réglé par le droit (lacune du droit) sur la base d'une norme établissant les conséquences juridiques de faits semblables sous des rapports réels ou bien sur la base de principes généraux ou selon des valeurs attribuées au droit faisant obligation. Dans nombre de pays, l'analogie n'est pas reconnue dans le droit pénal". L'analogie est ainsi apte à tendre vers une certaine plénitude du système juridique sans pour autant exiger une multiplication des textes ou leur actualisation incessante. En outre, le raisonnement analogique est profondément créatif non seulement parce que le juge aura tendance à établir des similitudes entre des situations, s'il désire en réalité faire application d'un texte qui lui paraît convenable, mais encore parce que le juge va souvent inventer le principe au regard duquel la similitude est établie. Pour reprendre encore l'exemple, il s'agira d'imposer la protection du faible comme fondement de la législation en matière d'incapacité. L'analogie est donc, dans tous les sens du terme, un raisonnement de principe et dans le même temps ce que certains¹³ désignent comme "le point extrême de l'audace".

Ainsi, l'article 4 du Code civil permet au juge, afin d'éviter un déni de justice, de faire application d'une loi qui est objectivement insuffisante en ce qu'elle ne prévoit pas le cas qu'il doit trancher. Mais cela ne veut pas dire que le raisonnement par analogie soit toujours disponible.

1.3 Linguistique

De nombreux travaux et ouvrages ont abordé l'analogie dans la linguistique et ce depuis l'antiquité. En se limitant à quelques ouvrages récents qui nous semblent bien résumer les différentes analyses, on peut citer Yvon [Yvo], Lepage [Lep03], Itkonen et Haukioja [IH97b] et Lavie [Lav03].

L'analogie a notamment contribué à l'introduction de nouveaux mots dans les langues comme le montrent les citations présentées ci-dessous (elles sont tirées respectivement de la première et la sixième édition du Dictionnaire de l'Académie [AP98][AP32]) : "Le mot *ambitionné* est formé par analogie d'*ambition*, comme *passionné* est formé de *passion*". "Les mots nouveaux ne peuvent guère s'introduire qu'à l'aide de l'analogie". De manière similaire, on peut noter l'usage des suffixes, en particulier "phile" (amateur de) ou "phage" (qui dévore) pour créer de nombreux nouveaux mots tels que : cinéphile, téléphage. On peut aussi citer l'exemple très connu de Saussure [dS16] sur le couple de mots latin "honor / honorem" qui dans un état ancien du latin donnait "honos / honosem". Le couple "honos / honosem" a changé par rhotacisme en "honos / honorem" : le "s" intervocalique de "honosem" s'est transformé en "r". Bien que le "s" de "honos" ne soit pas intervocalique, donc non concerné par le rhotacisme, on trouve dans le latin classique le couple "honor / honorem", ce qui est d'après Saussure le fruit de l'analogie.

Mitchell [Mit93] introduit l'analogie dans la linguistique comme catégorisation et comme moyen d'apprentissage ; en voici quelques extraits [Par98] :

¹³François Gény, Méthode d'interprétation et sources en droit privé positif, t. 1, n 16, p. 35, LGDJ 1919

- Un article de journal a dépeint *Denis Thatcher*, mari de *Margaret*, comme la “première dame de Grande Bretagne”.
- Jean dit à Simon “J’appelle mes parents une fois par semaine.” Simon répond “Moi aussi.” ; cela ne signifie pas, bien sûr, qu’il appelle les parents de Jean une fois par semaine, mais bien ses propres parents.

Ces exemples montrent bien la portée de la capacité humaine dans l’utilisation de l’analogie dans la linguistique.

1.4 Traitement automatique des langues

Le traitement automatique des langues qui chevauche le domaine de l’intelligence artificielle et de la linguistique a lui aussi profité de l’analogie. En effet, l’application la plus ambitieuse est celle de la traduction. Deux méthodes se distinguent pour ce faire. D’un côté, on présente les deux premières phrases de l’analogie dans une langue et la troisième phrase (traduction de la première) dans une autre langue pour avoir la quatrième phrase par résolution, comme le fait le système *Copycat* par exemple [Hof84, MH]. D’un autre côté, on utilise deux équations analogiques, chacune dans une langue [Lep03] pour la résolution (voir figure 2.3). Le même raisonnement peut aussi s’appliquer à la prononciation où, à partir d’une forme orthographique, on cherche, par exemple, à obtenir une représentation de sa prononciation dans l’alphabet phonétique international.

Chapitre 2

La proportion analogique et l'apprentissage

Sommaire

2.1	La proportion analogique	27
2.2	Apprentissage par proportion analogique	28
2.2.1	La proportion analogique en apprentissage artificiel	28
2.2.2	Exemple	29
2.3	La proportion analogique en Intelligence Artificielle	30
2.3.1	Récapitulatif	32

Contenu du chapitre

Ce chapitre traite de manière plus précise de l'analogie ou de la proportion analogique telle que nous l'utilisons. Il aborde en premier lieu l'utilisation de la proportion analogique en apprentissage artificiel ; en second lieu, il discute son utilisation dans l'intelligence artificielle et ses rapports avec le raisonnement à partir de cas.

2.1 La proportion analogique

Une relation de proportion analogique relie quatre objets A , B , C et D appartenant tous les quatre à un même espace. Le fait que quatre objets soient en proportion analogique s'énonce de la manière suivante :

$$\begin{aligned} & \text{“}A \text{ est à } B \text{ comme } C \text{ est à } D\text{”} \\ & \text{ou} \\ & \text{“}A \text{ est à } B \text{ ce que } C \text{ est à } D\text{”} \end{aligned}$$

Notons ici que l'ordre dans lequel on met les objets est important. On verra plus tard qu'en modifiant cet ordre suivant certaines règles, on obtient des analogies équivalentes.

Selon la nature des objets, la signification de cette relation d'analogie varie. Par exemple, une analogie sur la sémantique de mots français est :

“jument est à poulain comme vache est à veau”

En revanche, la relation analogique suivante est purement morphologique :

“abcd est à ecd comme abfab est à efab”

Elle ne porte que sur la transformation des lettres dans des séquences :

Pour transformer **abcd** en **ecd**, exactement comme pour transformer **abfab** en **efab**, il faut remplacer en début de séquence **ab** par **e**.

C'est à ce genre d'analogie dans les séquences que nous nous intéressons. Les autres d'analogies du type :

“ab est à abab comme abfg est à abfgabfg”

ou

“abc est à abd comme aababc est à aababcd”

ne sont pas traités ici, bien qu'elles aient toutes les deux de bonnes raisons cognitives d'être considérées comme correctes. L'argument est purement opérationnel : nous utilisons une notion de ressemblance entre séquences qui ne traite pas naturellement ce type de relations. Notre approche est donc limitée du point de vue de la simulation du comportement cognitif.

2.2 Apprentissage par proportion analogique

2.2.1 Utilisation de la proportion analogique en apprentissage artificiel

Soit \mathcal{S} un ensemble d'apprentissage $\mathcal{S} = \{(x, u(x))\}$, avec x la description d'un exemple (x peut être une séquence ou un vecteur dans \mathbb{R}^n , par exemple) et $u(x)$ son étiquette dans un ensemble fini. Ayant la description y d'un nouvel exemple, nous voudrions assigner une étiquette $u(y)$ à ce dernier uniquement à partir des connaissances de \mathcal{S} . Ce problème est celui de l'apprentissage inductif d'une règle de classification à partir d'exemples ([Mit97]), qui consiste à trouver la valeur de u au point y . La méthode du plus proche voisin, qui est la technique la plus populaire dans le domaine de l'apprentissage paresseux (*lazy learning*), revient à trouver dans \mathcal{S} la description x^* qui minimise une certaine distance de y et assigne ensuite $u(x^*)$, l'étiquette de x^* , comme étiquette de y .

Pour aller encore plus loin, l'apprentissage par proportion analogique consiste à rechercher dans \mathcal{S} un triplet (x^*, z^*, t^*) tel que “ x^* est à z^* comme t^* est à y^* ” et prédire pour y l'étiquette $\hat{u}(y)$ qui est une solution de l'équation “ $u(x^*)$ est à $u(z^*)$ comme $u(t^*)$ est à $\hat{u}(y)$ ”. Si plus d'un triplet est trouvé, une procédure de vote est utilisée. Cette technique d'apprentissage est basée sur la résolution d'équations analogiques. [PY99] traite de la pertinence d'une telle procédure d'apprentissage pour différentes tâches d'analyse linguistique. Il est important de noter que y et $u(y)$ peuvent appartenir à deux domaines

différents : par exemple, dans le cas simple d'apprentissage de règle de classification, y peut être une séquence alors que u est une étiquette de classe.

La prochaine étape dans l'apprentissage par proportion analogique est de trouver dans \mathcal{S} un triplet (x^*, z^*, t^*) tel que " x^* est à z^* comme t^* est à y^* " soit vraie, ou dans le cas où une mesure de proximité est définie, le triplet qui est le plus proche de y en termes de proportion analogique.

2.2.2 Exemple

Soit *matou* l'objet à classer dans l'une des classes suivantes : *Félin* ou *Ruminant*. Supposons qu'il n'y ait que trois exemples dans l'ensemble d'apprentissage : *veau*, *taureau* et *chaton*. Les animaux sont décrits par quatre attributs binaires (1 pour *VRAI* et 0 pour *FAUX*) qui sont *a des griffes (AG)*, *est un adulte (EA)*, *est un mâle (EM)*, *boit du lait (BL)*.

Animaux	<i>AG</i>	<i>EA</i>	<i>EM</i>	<i>BL</i>	classe
<i>veau</i>	0	0	0	1	<i>Ruminant</i>
<i>taureau</i>	0	1	1	0	<i>Ruminant</i>
<i>chaton</i>	1	0	0	1	<i>Félin</i>
<i>matou</i>	1	1	1	0	?

Pour chaque attribut, nous remarquons qu'il y a une relation d'analogie entre les trois objets de l'ensemble d'apprentissage et l'objet *matou*, car les quatre attributs binaires sont dans une des configurations analogiques suivantes (nous verrons plus tard quelles sont toutes les analogies possibles entre les valeurs binaires) :

$$\begin{array}{llll}
 0 & \text{est à} & 0 & \text{ce que} & 1 & \text{est à} & 1 & (AG) \\
 0 & \text{est à} & 1 & \text{ce que} & 0 & \text{est à} & 1 & (EA \text{ et } EM) \\
 1 & \text{est à} & 0 & \text{ce que} & 1 & \text{est à} & 0 & (BL)
 \end{array}$$

Ainsi, les quatre objets dans cet ordre sont en analogie puisque tous leurs attributs binaires sont en analogie :

$$\textit{veau} \text{ est à } \textit{taureau} \text{ ce que } \textit{chaton} \text{ est à } \textit{matou}$$

La résolution de l'équation analogique sur les classes nous donne la classe du *matou* :

$$\textit{Ruminant} \text{ est à } \textit{Ruminant} \text{ ce que } \textit{Félin} \text{ est à } \textit{classe(matou)}$$

Ce qui nous donne : $\textit{classe(matou)} = \textit{Félin}$.

Notons que la technique du 1-plus proche voisin, avec la distance de Hamming, conclut dans cet exemple que $\textit{classe(matou)} = \textit{Ruminant}$.

Nous voyons sur l'exemple ci-dessus comment l'utilisation d'un classifieur analogique permet d'avoir une vision différente de la classification par rapport au plus proches voisins. La force du classifieur par analogie réside dans le fait qu'il peut utiliser des objets dans l'ensemble d'apprentissage appartenant à une certaine classe ω_1 pour conclure que l'objet inconnu appartient à la classe ω_2 . Dans un classifieur analogique, tous les objets,

indépendamment de leur classe, peuvent avoir la même importance pour classer le nouvel objet.

En termes de distance, un classificateur par analogie peut également donner de l'importance à des objets qui sont loin de l'objet à classer. Il fait une utilisation de l'information dans l'ensemble d'apprentissage différente de celle des arbres de décision ou des classificateurs métriques, parce qu'il emploie une certaine forme (simple) de raisonnement par analogie [GHK01]. Mais l'analogie est-elle un concept réellement approprié pour la classification ? Nous tenterons de répondre à cette question par la suite.

2.3 La proportion analogique en Intelligence Artificielle

D'après Plaza et Aamodt[AA94], le raisonnement par analogie (analogy based reasoning) revient à utiliser des anciennes expériences appelées aussi *cas*, sachant qu'une expérience est constituée d'un problème et de sa solution dans un domaine, pour la résolution du nouveau problème, qui n'est pas nécessairement dans le même domaine. Ce genre de raisonnement est appelé analogie inter-domaines. A l'opposé, dans le raisonnement à partir de cas, le problème et les expériences appartiennent tous deux au même domaine (analogie intra-domaine). De plus, le raisonnement par analogie, toujours d'après Plaza et Aamodt[AA94], procède de deux manières différentes. La première (voir figure 2.1) consiste à adapter la solution du problème analogue de l'expérience au nouveau problème appelé aussi problème cible. Cette méthode de résolution est appelée la réutilisation transformationnelle (*transformational reuse*).

La seconde méthode, appelée la réutilisation dérivationnelle (*derivational reuse*), consiste à adapter la transformation (\mathcal{T}) qui a permis le passage du problème source à la solution source. Ainsi nous n'adaptions pas la solution source mais nous reconstruisons la solution cible à partir du problème cible et de l'adaptation de la transformation de \mathcal{T} (voir figure 2.2).

Un autre point de vue portant sur le raisonnement par analogie et le raisonnement à partir de cas est qu'il n'y a pas de différence majeure entre ces deux méthodes. Surtout que, d'après Leake [Lea96], on peut considérer l'analogie comme étant une étape du raisonnement à partir de cas, étant donné que l'analogie est une façon de mettre en relation un cas passé connaissant sa solution avec le problème actuel dont on veut connaître la solution. Ainsi [Mit93] répartit le raisonnement à partir de cas (RàPC) en quatre étapes :

$$\text{RàPC} = \text{récupération} + \text{analogie} + \text{adaptation} + \text{apprentissage}.$$

Lepage[Lep03] quant à lui, a utilisé l'analogie dans le traitement automatique des langues, plus précisément dans la conjugaison et la traduction. Ainsi, si les phrases (a), (b), (c) et (d) vérifient une analogie et si les analyses de (a), (b) et (c) sont connues, alors résoudre l'équation analogique permet d'obtenir l'analyse de (d) (voir figure 2.3).

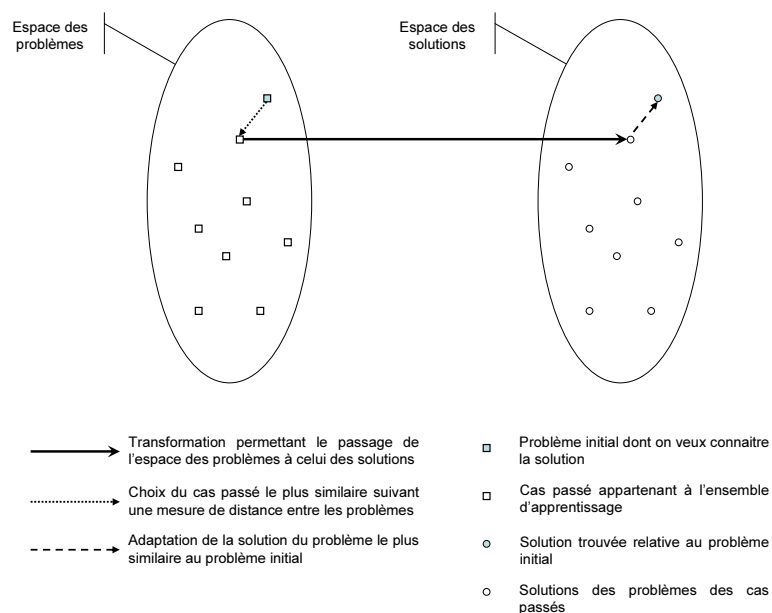


FIG. 2.1 – Réutilisation transformationnelle

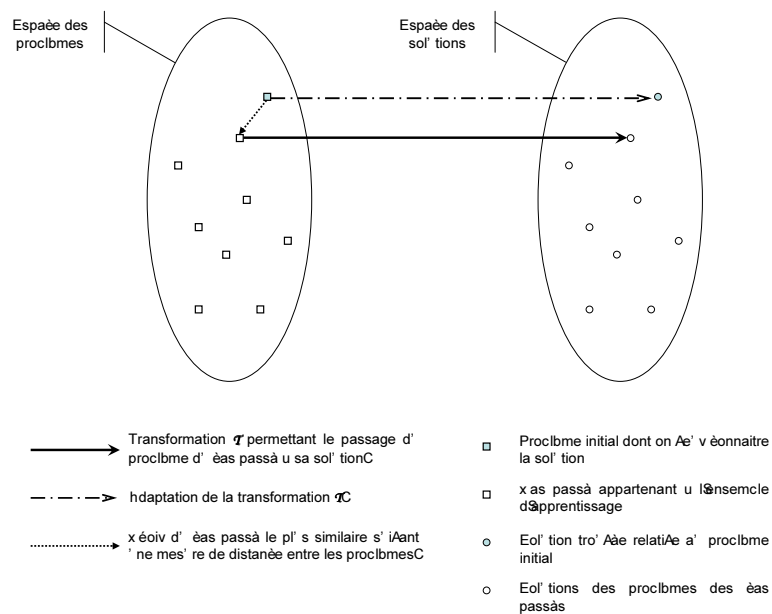


FIG. 2.2 – Réutilisation dérivationnelle

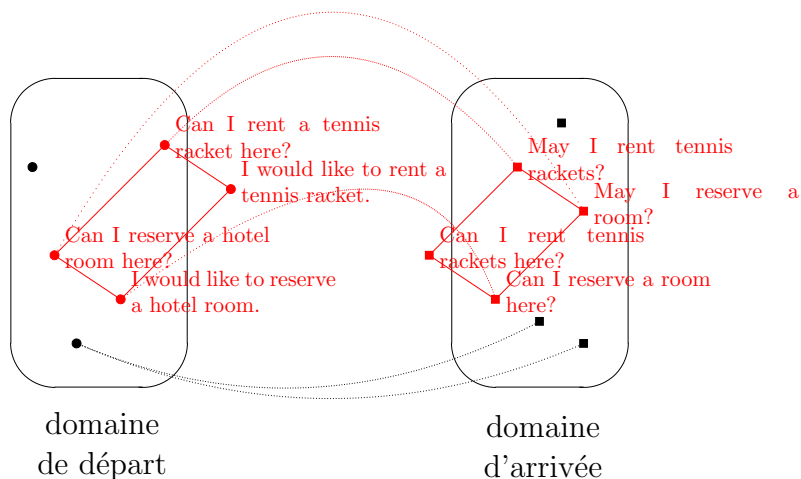


FIG. 2.3 – Modèle analogique pour la conjugaison

2.3.1 Récapitulatif

Dans le raisonnement à partir de cas, le raisonnement par analogie ne représente qu'une étape du RàPC qui permet en première phase de choisir un cas passé de la base d'apprentissage qui est analogue en utilisant une certaine mesure de distance, comme la similarité. La deuxième phase consiste à utiliser ce cas passé et sa solution pour la résolution du problème actuel (voir figure 2.4) sachant que la résolution peut être l'adaptation de la solution ou sa transformation ou la simple copie de cette solution dans le cas de classification par exemple.

Le terme raisonnement par analogie est aussi utilisé dans le domaine de l'IA pour représenter une méthode d'apprentissage et de résolution. Ainsi, le raisonnement par analogie revient en première phase à choisir un triplet de cas de la base d'apprentissage dont les problèmes forment une analogie avec le problème cible. En deuxième phase, par résolution du quatrième terme de l'équation analogique formée par les solutions du triplet des cas, on trouve la solution du problème cible (voir figure 2.5).

Pour relier les deux définitions, on peut dire que cette dernière définition du raisonnement par analogie peut être vue comme une analogie intra-domaine associée à une analogie inter-domaine dans le raisonnement à partir de cas.

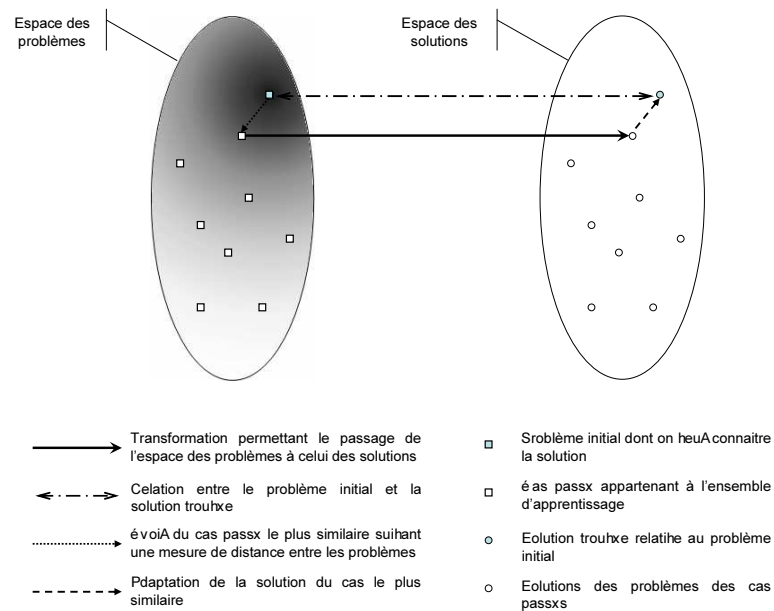


FIG. 2.4 – Raisonnement à Partir de Cas

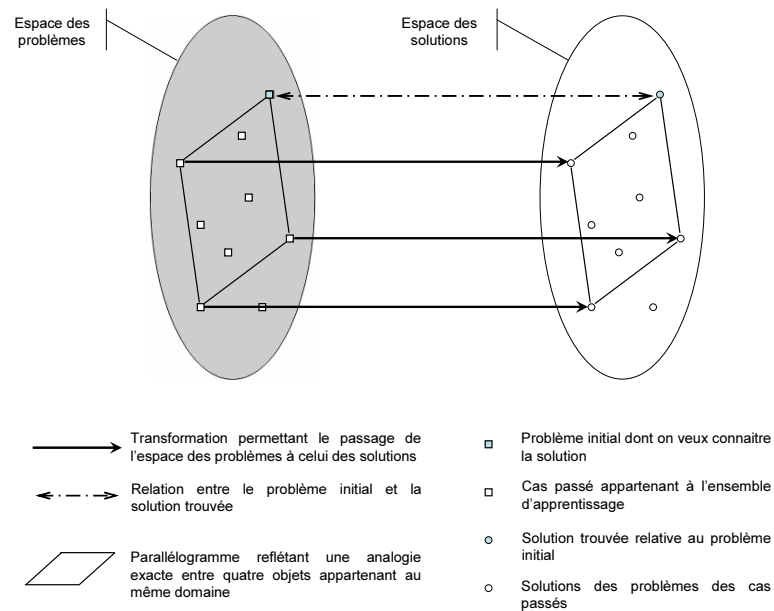


FIG. 2.5 – Analogie

Deuxième partie

Théorie et Algorithmes

Chapitre 3

Proportion analogique exacte

Sommaire

3.1	Les axiomes de l'analogie	37
3.2	Distance cohérente avec l'analogie	38
3.3	Équations Analogiques	39
3.4	Proportion analogique sur les ensembles	40
3.4.1	Définition d'un ensemble fini cyclique	40
3.4.2	Définition d'un ensemble fini par des attributs binaires	40
3.4.3	Transitivité de l'analogie dans les ensembles finis	42
3.4.4	Une distance dans un ensemble binaire	42
3.5	Proportion analogique dans un espace vectoriel	42
3.5.1	Une proportion analogique dans un espace vectoriel	42
3.5.2	La transitivité de l'analogie dans l'espace vectoriel	43
3.5.3	Un ensemble de distances cohérentes	43
3.6	Proportion analogique entre séquences	44
3.6.1	Notations	44
3.6.2	Une première définition	44
3.6.3	Une définition plus générale	45

Contenu du chapitre

Ce chapitre introduit les notions de base de la proportion analogique exacte. On y donne une description de la manière dont se définit l'analogie dans différents domaines : les ensembles définis par des traits, les groupes cycliques, les espaces euclidiens et les séquences d'objets. Il s'agit ici de proportion analogique exacte.

3.1 Les axiomes de l'analogie

Il n'y pas de définition universelle de la proportion analogique entre quatre objets appartenant à un ensemble X "A est à B comme C est à D", puisque les relations "est à"

et “ce que” dépendent de la nature de X . Cependant, selon la signification habituelle du mot “analogie” en philosophie et en linguistique, trois axiomes fondamentaux sont généralement requis ([LA96]) :

Définition 3.1.1 (Proportion analogique.) *Une proportion analogique (en abrégé : une analogie) sur un ensemble X est une relation sur X^4 , c’est à dire un sous-ensemble $\mathcal{A} \subset X^4$ qui respecte trois axiomes. Soit $(A, B, C, D) \in \mathcal{A}$. Les quatre éléments A , B , C et D sont dits en proportion analogique. On note : “la proportion analogique $A : B :: C : D$ est vraie”, ou simplement $A : B :: C : D$, qui se lit “ A est à B ce que C est à D ”.*

Les deux premiers axiomes sont :

$$\begin{aligned} \text{Symétrie de la relation “est à”} : & \quad A : B :: C : D \Leftrightarrow C : D :: A : B \\ \text{Échange des moyens} : & \quad A : B :: C : D \Leftrightarrow A : C :: B : D \end{aligned}$$

Le troisième axiome (déterminisme) exige que l’une des deux implications suivantes soit vraie (l’autre étant une conséquence) :

$$\begin{aligned} A : A :: B : x & \Rightarrow x = B \\ A : B :: A : x & \Rightarrow x = B \end{aligned}$$

Au regard de ces axiomes, il s’avère que cinq autres formulations sont équivalentes à $A : B :: C : D$:

$$\begin{array}{ccc} B : A :: D : C & D : B :: C : A & C : A :: D : B \\ D : C :: B : A & \text{et} & B : D :: A : C \end{array}$$

Donc, au total, huit proportions analogiques sont axiomatiquement équivalentes. En conséquence, il y a seulement trois analogies non équivalentes possibles entre quatre objets, ayant les formes canoniques :

$$A : B :: C : D \quad A : C :: D : B \quad A : D :: B : C$$

3.2 Distance cohérente avec l’analogie

X est un *ensemble métrique* s’il existe une distance δ entre les éléments de X , répondant à la définition classique suivante [DM04].

Définition 3.2.1 (Distance dans un ensemble X .) *On définit une distance δ dans un ensemble X comme une application de $X \times X$ dans \mathbb{R} qui vérifie les propriétés suivantes :*

Réflexivité. $\forall x, y \in X, \delta(x, y) = 0 \Leftrightarrow x = y$

Positivité stricte. $\forall x, y \in X, x \neq y \Rightarrow \delta(x, y) > 0$

Symétrie. $\forall x, y \in X, \delta(x, y) = \delta(y, x)$

Inégalité triangulaire. $\forall x, y, z \in X, \delta(x, y) \leq \delta(x, z) + \delta(z, y)$

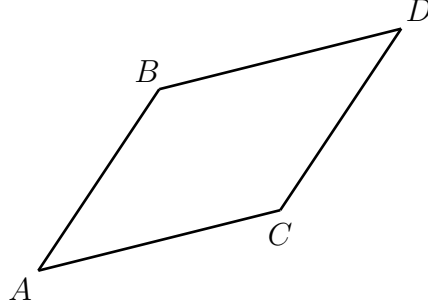


FIG. 3.1 – L’analogie et une distance cohérente dans \mathbb{R}^n . Quand l’analogie $A : B :: C : D$ est vraie, les quatre éléments forment un parallélogramme et la distance euclidienne δ_2 a les propriétés suivantes : $\delta_2(u, v) = \delta_2(w, x)$ et $\delta_2(u, w) = \delta_2(v, x)$

S’il existe une analogie dans un espace métrique X , une relation entre la distance et l’analogie dans X peut être définie. L’intérêt d’une telle relation se révélera ultérieurement, spécialement lors de la définition de l’analogie sur les séquences (section 4.4) et de la dissemblance analogique (section 5.2 et 5.6).

Définition 3.2.2 (Distance cohérente avec l’analogie.) Une distance δ est dite cohérente avec l’analogie si pour chaque quadruplet A, B, C et D en proportion analogique ($A : B :: C : D$), δ a les propriétés suivantes :

$$\delta(A, B) = \delta(C, D) \quad \text{et} \quad \delta(A, C) = \delta(B, D)$$

C’est clairement le cas, par exemple, dans $X = \mathbb{R}^n$, où quatre éléments sont définis en proportion analogique s’ils forment un parallélogramme (voir la figure 3.2) et si δ est la distance euclidienne. Cet exemple sera plus amplement développé dans la section 3.5.

3.3 Équations Analogiques

Résoudre une équation analogique consiste à trouver le quatrième terme d’une proportion analogique, les trois premiers termes étant connus.

Définition 3.3.1 (Équation Analogique.) t est une solution de l’équation analogique : $x : y :: z : ?$ si et seulement si $x : y :: z : t$.

Selon la nature des objets et la définition de l’analogie, une équation analogique peut ne pas avoir de solution, avoir une seule solution¹ ou plusieurs solutions.

Nous étudierons dans la suite comment résoudre des équations analogiques dans les différents ensembles que nous avons introduits. Ensuite, nous donnons deux définitions de la résolution d’équation analogique dans les séquences, la seconde étant originale et plus générale que la première.

¹C’est d’ailleurs le seul cas où l’analogie est transitive, voir section 3.4.3, 3.5.2.

\oplus	a	b	\boxed{c}	d	e	\boxed{f}
\boxed{a}	a	b	c	\boxed{d}	e	f
b	b	c	d	e	f	a
c	c	d	e	f	a	b
d	d	e	f	a	b	c
\boxed{e}	e	f	a	b	c	\boxed{d}
f	f	a	b	c	d	e

TAB. 3.1 – Table de l'opérateur \oplus sur un alphabet cyclique de 6 éléments vu comme un groupe cyclique additive \mathcal{G}_6 .

3.4 Proportion analogique sur les ensembles

3.4.1 Définition d'un ensemble fini cyclique

Dans cette section, nous reprenons la définition de l'analogie sur les groupes cycliques finis [DM04]. Soit (X, \oplus) un ensemble avec un opérateur interne. Soit a, b, c et d quatre éléments de X . On relie l'analogie à l'opérateur \oplus sur X par la propriété suivante :

$$(a \oplus d = b \oplus c) \Leftrightarrow (a : b :: c : d)$$

3.4.1.1 Propriétés de l'opérateur \oplus suivant l'analogie

Pour chaque axiome de l'analogie déjà présenté dans 3.1, nous déduisons une propriété algébrique pour l'opérateur \oplus .

Symétrie : $a : b :: c : d \Rightarrow c : d :: a : b$, qui est : $a \oplus d = b \oplus c \Rightarrow c \oplus b = d \oplus a$

Échange des moyens : $a : b :: c : d \Rightarrow a : c :: b : d$ qui est $a \oplus d = b \oplus c \Rightarrow a \oplus d = c \oplus b$

A partir de là, nous concluons que l'opérateur \oplus doit être commutatif, vu que $c \oplus b = b \oplus c$ si $a : b :: c : d$.

Déterminisme : $a : a :: c : x \Rightarrow x = c$ et $a : b :: a : x \Rightarrow x = b$ qui est $a \oplus x = a \oplus c \Rightarrow x = c$ et $a \oplus x = b \oplus a \Rightarrow x = b$

Une méthode simple de trouver les analogies dans un groupe cyclique, par exemple $X = \{a, b, c, d, e\}$, est d'utiliser le tableau de l'opérateur \oplus (voir table 3.1). Ainsi pour une équation analogique de type $a : b :: c : d$, elle se transforme en $\underbrace{b \oplus c}_d = \underbrace{a \oplus d}_d$.

3.4.2 Définition d'un ensemble fini par des attributs binaires

Soit X un ensemble fini ou un *alphabet*, composé d'éléments que nous appellerons *objets*. Soit un ensemble \mathcal{F} , de cardinal n , d'attributs binaires tel que chaque objet $x \in X$ peut être défini par un unique vecteur binaire $(f_1(x), \dots, f_n(x))^\top$. Pour chaque x et chaque $i \in [1, n]$, $f_i(x) = 1$ (respectivement $f_i(x) = 0$) signifie que l'attribut binaire

prend la valeur *VRAI* ou 1 (respectivement *FAUX* ou 0) pour l'objet x . Nous appelons un tel ensemble X *un ensemble fini défini par des attributs binaires*.

De manière équivalente, un objet $x \in X$ peut être vu comme un sous ensemble de \mathcal{F} , composé des éléments f_i telle que $f_i(x) = 1$. Par conséquent, l'étude de l'analogie entre quatre objets dans un alphabet défini par des attributs binaires est équivalente à l'étude de l'analogie entre quatre ensembles.

Une première définition. Quand la relation “est à” est l'égalité entre ensembles, Lepage [Lep03] a donné une définition de la proportion analogique entre ensembles qui respecte les axiomes.

Définition 3.4.1 (Analogie entre ensembles finis (1).) *Quatre ensembles finis A , B , C et D sont en proportion analogique $A : B :: C : D$ si et seulement si A peut être transformé en B et C en D en ajoutant ou soustrayant les mêmes éléments à A et C .*

C'est le cas, par exemple, des quatre ensembles : $A = \{t_1, t_2, t_3, t_4\}$, $B = \{t_1, t_2, t_3, t_5\}$ et $C = \{t_1, t_4, t_6, t_7\}$, $D = \{t_1, t_5, t_6, t_7\}$, où t_4 a été retiré de A et C , et t_5 a été ajouté à A et C , pour composer B et D .

A partir de cette définition, Lepage ([Lep03]) a montré une double condition nécessaire sur quatre ensembles pour qu'ils puissent être en proportion analogique :

$$A \subseteq B \cup C \quad \text{et} \quad A \supset B \cap C \quad (3.1)$$

Dans la section 4.2.2, nous verrons, sous ces conditions, qu'une solution unique D peut être donnée à l'équation $A : B :: C : x$ en respectant les axiomes de l'analogie :

$$x = ((B \cup C) \setminus A) \cup (B \cap C)$$

Une deuxième définition équivalente. Stroppa et Yvon [SY05] ont donné une autre définition de l'analogie entre quatre ensembles finis, qui s'avère être équivalente à celle de Lepage.

Définition 3.4.2 (Analogie entre ensembles finis (2).) *Quatre ensembles A , B , C et D sont en proportion analogique $A : B :: C : D$ si et seulement si il existe quatre ensembles E , F , G et H tels que :*

$$\begin{aligned} A &= E \cup F \\ B &= E \cup G \\ C &= H \cup F \\ D &= H \cup G \end{aligned}$$

Il a été donné dans [MD04] la preuve complète que les deux définitions sont équivalentes. La démonstration est simple : les conditions d'inclusion de l'équation 3.1 impliquent que, parmi les 16 ensembles disjoints créés par l'intersection de A , B , C et D , seul 5 sont non-vides. Ils peuvent être combinés par union selon soit la première définition soit la deuxième.

3.4.3 Transitivité de l'analogie dans les ensembles finis

Dans les ensembles, la proportion analogique possède la propriété de transitivité [Lep03] :

Propriété 3.4.1 (Transitivité de l'analogie dans les ensembles finis.) *Soient A , B , C , D , E et F six ensembles finis. Alors l'implication suivante est vraie :*
 $(A : B :: C : D \text{ et } C : D :: E : F) \Rightarrow A : B :: E : F$

3.4.4 Une distance dans un ensemble défini par des attributs binaires cohérente avec l'analogie.

Soit X un ensemble défini par des attributs binaires. Nous pouvons voir un élément A de X soit comme un ensemble d'attributs qui sont *VERAI* dans A soit comme un vecteur binaire de taille n , où n est le nombre total d'attributs (le cardinal de l'ensemble \mathcal{F}). Dans le premier cas, nous définissons la distance $\delta(A, B)$ entre deux éléments A et B de X comme le cardinal de la différence symétrique entre les deux ensembles. Dans le deuxième cas, $\delta(A, B)$ est la distance de Hamming entre les deux vecteurs binaires. Évidemment, les deux définitions sont équivalentes. Nous avons les propriétés suivantes :

Propriété 3.4.2 (Cohérence de la différence symétrique.) *Soit $\delta(A, B)$ le cardinal de la différence symétrique entre les deux ensembles A et B . Alors δ est cohérente avec l'analogie dans les ensembles.*

La preuve est directe à partir des définitions.

3.5 Proportion analogique dans l'espace vectoriel \mathbb{R}^n

Nous avons montré, dans les sections précédentes, comment construire des analogies et des distances cohérentes dans un alphabet défini par des attributs binaires. Nous nous intéressons maintenant au cas où X est l'espace vectoriel \mathbb{R}^n . Une analogie entre quatre objets, ou vecteurs, dans \mathbb{R}^n est généralement (voir [SY05]) définie de façon informelle comme suit : a , b , c , et d sont quatre vecteurs en proportion analogique si et seulement si ils forment un parallélogramme dans \mathbb{R}^n .

3.5.1 Une proportion analogique dans \mathbb{R}^n

Soit O l'origine de l'espace vectoriel. Soit $a = (a_1, a_2, \dots, a_n)^\top$ un vecteur de \mathbb{R}^n , défini par ses n coordonnées. Soit a , b , c et d quatre vecteurs de \mathbb{R}^n . L'interprétation

d'une analogie $a : b :: c : d$ est généralement que a, b, c, d sont les sommets d'un parallélogramme, a et d étant des sommets opposés.

Définition 3.5.1 (Analogie dans \mathbb{R}^n .) *Quatre vecteurs de \mathbb{R}^n sont en proportion analogique $a : b :: c : d$ si et seulement si ils forment un parallélogramme d'équation :*

$$\overrightarrow{Oa} - \overrightarrow{Ob} = \overrightarrow{Oc} - \overrightarrow{Od} \quad \text{ou} \quad \overrightarrow{ab} = \overrightarrow{cd}$$

ou de façon équivalente

$$\overrightarrow{ac} = \overrightarrow{bd} \quad \text{ou} \quad \overrightarrow{ab} = \overrightarrow{cd}$$

Il est évident que les axiomes de l'analogie, donnés dans la section 3.1, sont vérifiés par cette définition.

3.5.2 La transitivité de l'analogie dans l'espace vectoriel

Dans les espaces vectoriels, l'analogie a aussi la propriété de transitivité :

Propriété 3.5.1 (Transitivité de l'analogie dans \mathbb{R}^n .) *Soit a, b, c, d, e et f six vecteurs de \mathbb{R}^n . Alors l'implication suivante est vraie :*

$$(a : b :: c : d \quad \text{et} \quad c : d :: e : f) \Rightarrow a : b :: e : f$$

La preuve provient de la définition 3.5.1.

3.5.3 Un ensemble de distances cohérentes

Rappelons qu'une distance δ est dite *cohérente* avec une analogie si pour chaque 4-uplet a, b, c et d qui sont en proportion analogique :

$$a : b :: c : d$$

La distance δ a les propriétés :

$$\delta(a, b) = \delta(c, d) \quad \text{et} \quad \delta(a, c) = \delta(b, d)$$

Dans \mathbb{R}^n , toute distance δ_p définie à partir de la norme $\| \cdot \|_p$

$$\delta_p(a, b) = \|a - b\|_p = \left(\sum_{i=1}^n |a_i - b_i|^p \right)^{1/p}$$

est cohérente avec l'analogie définie par $\overrightarrow{ab} = \overrightarrow{cd}$.

Ceci se prouve directement à partir de propriétés classiques d'un espace euclidien : pour chaque distance δ_p dans \mathbb{R}^n , $\overrightarrow{ab} = \overrightarrow{cd}$ implique $\delta_p(a, b) = \delta_p(c, d)$.

3.6 Proportion analogique entre séquences

Dans cette section, nous présentons deux façons différentes de définir une proportion analogique entre quatre séquences d'objets. Après avoir donné quelques notations classiques sur les séquences, nous allons en premier lieu donner la définition de Yvon, qui raffine et consolide celle de Lepage. Ensuite, nous présentons notre définition, que nous montrons être plus générale. Les objets qui forment les séquences peuvent être des éléments d'un alphabet fini (dans notre cas, défini par les attributs binaires) ou des vecteurs d'un espace vectoriel.

3.6.1 Notations

Une *séquence*² est une suite dénombrable de symboles de l'alphabet Σ . On note Σ^* l'ensemble de toutes les séquences. Pour x, y dans Σ^* , xy est la concaténation de x et y . On note aussi $|x|$ la longueur de x , et $x = x_1 \dots x_{|x|}$ ou $x = x[1] \dots x[n]$, avec x_i ou $x[i] \in \Sigma$ et $n = |x|$. Nous notons ϵ le mot vide, de longueur nulle, et $\Sigma^+ = \Sigma^* \setminus \{\epsilon\}$.

Finalement, nous notons $\mathcal{L}(x)$ le sous ensemble de Σ dans lequel sont pris les lettres du mot x et $\overline{\mathcal{L}}(x)$ le sous-ensemble de Σ composé de lettres qui n'apparaissent pas dans x .

Un *facteur* (ou sous-mot) f d'une séquence x est une séquence de Σ^* telle qu'il existe deux séquences u et v dans Σ^* avec $x = ufv$. Par exemple, *abb*, *bbac* et *abbacbababa* sont des facteurs de *abbacbbaba*.

Une *sous-séquence* d'une séquence $x = x_1 \dots x_{|x|}$ est composée de lettres de x ayant les indices $i_1 \dots i_k$, tels que $i_1 < i_2 < \dots < i_k$. Par exemple, *ca* et *aaa* sont deux sous-séquences de *abbacbbaba*.

3.6.2 Une première définition

Yvon ([Yvo03]) donne la définition suivante d'une proportion analogique entre séquences.

Définition 3.6.1 (Proportion analogique entre séquences.)

$(x, y, z, t) \in \Sigma^+$ sont en proportion analogique, notée $x : y :: z : t$, si et seulement si il existe un entier strictement positif n et deux ensembles de séquences $(\alpha_i)_{i \in [1, n]}$ et $(\beta_i)_{i \in [1, n]}$ dans Σ^* tels que :

$$\begin{aligned} x &= \alpha_1 \dots \alpha_n, \quad t = \beta_1 \dots \beta_n, \quad y = \alpha_1 \beta_2 \alpha_3 \dots \alpha_n, \quad z = \beta_1 \alpha_2 \beta_3 \dots \beta_n \\ \text{ou} \\ x &= \alpha_1 \dots \alpha_n, \quad t = \beta_1 \dots \beta_n, \quad y = \beta_1 \alpha_2 \beta_3 \dots \alpha_n, \quad z = \alpha_1 \beta_2 \alpha_3 \dots \beta_n \end{aligned}$$

et $\forall i, \alpha_i \beta_i \neq \epsilon$.

Le plus petit entier n pour lequel la propriété reste vraie est appelé *degré de la proportion analogique*.

²Plus classiquement dans la théorie des langages, un *mot* ou une *phrase*.

Par exemple $reception : refaction :: deceptive : defective$ est une analogie entre séquences, avec $n = 3$ avec les facteurs : $\alpha_1 = re$, $\alpha_2 = cept$, $\alpha_3 = ion$, $\beta_1 = de$, $\beta_2 = fect$, $\beta_3 = ive$.

Nous aurions aussi pu choisir, avec le même degré (minimum, en l'occurrence) les facteurs suivants :

$\alpha_1 = r$, $\alpha_2 = ecept$, $\alpha_3 = ion$ et par conséquent $\beta_1 = d$, $\beta_2 = efect$ et $\beta_3 = ive$.

Le degré d'une proportion analogique peut être vu comme une mesure de sa complexité : plus petit est le degré, meilleure est l'analogie. Ce qui va dans le sens de l'intuition que les bonnes analogies devraient préserver des larges portions des mots originaux. Les analogies triviales s'écrivent avec des mots deux à deux identiques.

Ayant cette définition, les propriétés suivantes restent vraies (voir aussi [Lep01]) :

$$\forall x \in \Sigma^+, x : x :: x : x \quad (3.2)$$

$$\forall x, y \in \Sigma^+ : x : x :: y : y \quad (3.3)$$

$$\forall x, y, z, t \in \Sigma^+ : x : y :: z : t \Rightarrow z : t :: x : y \quad (3.4)$$

$$\forall x, y, z, t \in \Sigma^+ : x : y :: z : t \Rightarrow x : z :: y : t \quad (3.5)$$

Ce qui prouve que cette définition de l'analogie est consistante vis à vis des axiomes donnés dans la section 3.1.

Lepage et Yvon [Yvo03] ont prouvé que les deux conditions suivantes sont nécessaires pour que les séquences x , y , z et t soient en analogie :

– fermeture des symboles :

$$\overline{\mathcal{L}(x)} \cup \overline{\mathcal{L}(t)} = \overline{\mathcal{L}(y)} \cup \overline{\mathcal{L}(z)} \quad (3.6)$$

– similarité : $|x| + |t| = |y| + |z|$.

Yvon ([Yvo03]) a montré que l'ensemble des solutions d'une proportion analogique sur les mots, selon la définition 3.6.2, peut être exprimé à partir de deux opérations de base sur les mots et les langages, le *produit de mélange*³ et la *complémentation*. Ils ont aussi donné un algorithme pour la construction de l'ensemble de toutes les solutions, qui peut être modifié pour produire seulement les solutions de degré minimal. Il est fondé sur la construction de transducteurs à états finis et l'utilisation d'algorithmes de programmation dynamique [YSMD04].

3.6.3 Une définition plus générale utilisant l'analogie sur les alphabets

3.6.3.1 Motivation

La définition de l'analogie entre séquences présentée dans la section précédente est relativement stricte dans le sens où le quatrième terme est construit avec des lettres qui sont apparues dans les trois autres termes, en raison de la propriété de fermeture. De plus, les lettres sont considérées comme des objets indépendants. En particulier, s'il y a

³En anglais : *shuffle*.

une certaine proportion analogique sur l'alphabet, elle ne peut pas être transmise aux séquences. Nous nous sommes intéressés à cette lacune.

Par exemple, supposons un alphabet $\Sigma = \{a, b, \alpha, \beta, B, C\}$ dans lequel il existe les proportions analogiques suivantes : $a : b :: A : B$, $a : \alpha :: b : \beta$, $A : \alpha :: B : \beta$. Soit l'équation analogique suivante dans Σ^* : $aaBAB : \alpha\alpha bab :: bbBAB : x$. Nous concluons de façon certaine, en progressant lettre par lettre, que $x = \beta\beta bab$. Mais cette solution ne peut pas être obtenue à partir de ce qui a été présenté dans la section précédente, puisque la lettre β n'apparaît pas dans les trois premiers termes de l'équation.

Ainsi, nous voulons étendre la définition de l'analogie entre séquences à de tels cas. Nous voulons aussi traiter les analogies sur des séquences de longueurs différentes, comme le fait la définition précédente. L'étude de l'analogie sur les ensembles, donnée plus haut, va être utile : ces ensembles seront les alphabets sur les séquences.

3.6.3.2 Une proportion analogique entre séquences avec la notion d'alignement

Nous présentons dans ce qui suit une définition plus générale de la proportion analogique entre quatre séquences et nous montrons que cette définition satisfait les axiomes donnés dans la section 3.1.

Soit Σ un alphabet. Nous rajoutons une nouvelle lettre à Σ , notée \smile , pour obtenir un alphabet augmenté Σ' . L'interprétation de cette nouvelle lettre est simplement celle d'un symbole "vide", qui sera nécessaire pour calcul de l'alignement entre séquences dans les prochaines sections.

Définition 3.6.2 (Équivalence sémantique.) *Soit x une séquence de Σ^* et y une séquence de Σ'^* . Les séquences x et y sont sémantiquement équivalentes si la sous-séquence de y composée de lettres de Σ est x . Nous notons cette relation par $x \equiv y$.*

Par exemple, $ab \smile \smile a \smile a \equiv abaa$.

Supposons qu'il y ait une analogie dans Σ' , c'est à dire que pour chaque quadruplet a, b, c, d de lettres de Σ' , la relation $a : b :: c : d$ est définie comme étant soit *VRAIE* ou *FAUSSE*.

Définition 3.6.3 (Alignement entre deux séquences.) *Un alignement entre deux séquences $x, y \in \Sigma^*$, de longueurs m et n , est un mot z de l'alphabet $(\Sigma') \times (\Sigma') \setminus \{(\smile, \smile)\}$ dont la projection sur la première dimension est sémantiquement équivalente à x et dont la projection sur la seconde dimension est sémantiquement équivalente à y .*

De façon informelle, un alignement représente un appariement lettre à lettre entre les deux séquences, dans lequel une ou plusieurs lettres \smile peuvent être insérées. L'appariement (\smile, \smile) n'est pas permis. Un alignement peut être présenté comme un tableau à deux lignes, l'une relative à x et l'autre relative à y , chaque mot étant complété par un certain nombre de \smile pour donner deux autres mots appartenant à Σ' qui ont la même longueur.

Par exemple, voici un alignement entre $x = abgef$ et $y = acde$:

$$\begin{array}{ccccccc} x' & = & a & b & \smile & g & e & f \\ & & | & | & | & | & | & | \\ y' & = & a & c & d & \smile & e & \smile \end{array}$$

La notion d'alignement entre deux séquences est utilisée depuis longtemps pour le calcul de distances entre séquences, en particulier depuis l'article de référence [WF74]. Elle est facile à étendre à un plus grand nombre de séquences, ce qui a été fait en particulier pour le problème de la recherche des plus longues sous-séquences communes à plusieurs séquences [Jus99]. Nous nous intéressons ici à la proportion analogique entre quatre séquences et nous utilisons la définition suivante de l'alignement entre quatre séquences.

Définition 3.6.4 (Alignement entre quatre séquences.) *Un alignement entre quatre séquences $u, v, w, x \in \Sigma^*$, est un mot z de l'alphabet $(\Sigma \cup \{\smile\})^4 \setminus \{(\smile, \smile, \smile, \smile)\}$ dont la projection*

- sur la première dimension est sémantiquement équivalente à u ;
- sur la deuxième dimension est sémantiquement équivalente à v ;
- sur la troisième dimension est sémantiquement équivalente à w ;
- sur la quatrième dimension est sémantiquement équivalente à x .

La définition suivante utilise l'alignement entre quatre séquences.

Définition 3.6.5 (Proportion analogique entre séquences.) *Soit u, v, w et x quatre séquences de Σ^* , sur lesquelles une analogie est définie. Nous disons que u, v, w et x sont en proportion analogique dans Σ^* s'il existe quatre séquences u', v', w' et x' de même longueur n dans Σ' qui vérifient les propriétés suivantes :*

1. $u' \equiv u, v' \equiv v, w' \equiv w$ et $x' \equiv x$;
2. $\forall i \in [1, n]$ les analogies $u'_i : v'_i :: w'_i : x'_i$ sont vraies dans Σ' .

Nous prouvons maintenant que cette définition vérifie les axiomes de l'analogie.

Ces axiomes restent vrais pour chaque quadruplet u'_i, v'_i, w'_i et x'_i , i.e. :

1. $w'_i : x'_i :: u'_i : v'_i$;
2. $u'_i : w'_i :: v'_i : x'_i$;
3. $u'_i = v'_i \Rightarrow w'_i = x'_i$.

Donc, par concaténation des n termes, nous avons dans Σ'^* :

1. $w' : x' :: u' : v'$;
2. $u' : w' :: v' : x'$;
3. $u' = v' \Rightarrow w' = x'$.

et, par équivalence sémantique, nous avons dans Σ^* :

1. $w : x :: u : v$;
2. $u : w :: v : x$;

3. $u = v \Rightarrow w = x$.

ce qui nous assure que les axiomes de l'analogie sont vérifiés pour la définition 3.6.5.

Par exemple, soit $\Sigma' = \{a, b, \alpha, \beta, B, C, \sim\}$ avec les analogies $a : b :: A : B$, $a : \alpha :: b : \beta$ et $A : \alpha :: B : \beta$. L'alignement suivant entre les quatre séquences aBA , $abBA$, ba et βba est une analogie sur Σ^* :

a	\sim	B	A
α	b	B	A
b	\sim	a	\sim
β	b	a	\sim

3.6.3.3 Relation entre les deux définitions

Nous établissons maintenant la propriété suivante :

Propriété 3.6.1 *L'analogie entre séquences basée sur les alignements que nous avons donnée dans la définition 3.6.5 est strictement plus générale que celle définie précédemment (Définition 3.6.1).*

La démonstration se base sur la redéfinition de la première analogie en termes d'alignements. C'est d'autant plus naturel que les travaux des auteurs utilisent des transducteurs et que l'extension de l'alphabet à la lettre \sim correspond à l'adjonction de boucles sur leurs états. Il est donc simple de constater que l'alignement correspondant à leur définition est un sous-ensemble de tous les alignements possibles. En se restreignant à la première définition, chaque colonne aurait une des deux formes particulières suivantes, pour $a \in \Sigma$

a		a
a		\sim
	ou	
\sim		a
\sim		\sim

Chapitre 4

Résolution d'équations analogiques

Sommaire

4.1 Définition	49
4.2 Résolution d'équations analogiques dans les ensembles finis	50
4.2.1 Groupes finis cyclique	50
4.2.2 Ensembles finis binaires	50
4.3 Résolution d'équations analogiques dans un espace vectoriel	51
4.4 Résolution d'équations analogiques dans les séquences	51
4.4.1 Résolution selon Lepage	51
4.4.2 Résolution selon Yvon et Stroppa	52
4.4.3 Notre approche	53

Contenu du chapitre

Résoudre une équation analogique consiste, connaissant trois objets, à calculer un quatrième qui soit en proportion analogique exacte avec les trois premiers.

Le problème a été abordé assez récemment ce qui concerne les séquences, en particulier par Lepage de manière empirique [Lep03] et par Yvon et Stroppa de manière plus formelle [SY05]. Nous décrirons notre approche originale dans le chapitre suivant, après avoir défini les concepts nécessaires.

4.1 Définition

Quand l'un des quatre éléments est inconnu, une proportion analogique devient une équation analogique. Par exemple, résoudre l'équation sur les séquences de lettres : “**wolf is to leaf as wolves is to x** ”, consiste à calculer l'ensemble des séquences x qui satisfont la proportion analogique, cet ensemble pouvant être vide. La séquence **leaves** est une solution exacte vis à vis d'une analogie linguistique ou morphologique. Mais il n'est pas trivial d'écrire un algorithme capable de résoudre une telle équation, plus particulièrement si on cherche aussi une ou plusieurs solutions approchées.

Résoudre des équations analogiques sur les séquences est particulièrement utile pour les tâches de linguistique computationnelle. La résolution d'équations analogiques a été appliquée d'abord principalement dans des tâches d'analyse lexicale, avec des techniques empiriques ou dans des cas simples. De manière plus élaborée, Yvon([Yvo97], [Yvo99]) a présenté une approche analogique pour la conversion graphème-phonème des noms propres dans une application de synthèse de la parole à partir du texte. Plus généralement, la résolution d'équations analogiques peut aussi être vue comme une composante élémentaire des systèmes d'*apprentissage par analogie*, qui font partie des techniques de *lazy learning* [Dae96].

4.2 Résolution d'équations analogiques dans les ensembles finis

4.2.1 Résolution d'équations analogiques dans les groupes finis cyclique

Comme on a pu le voir dans la section 3.4.1, l'opérateur \oplus permet une réécriture d'une équation analogique sous la forme :

$$(a \oplus d = b \oplus c) \Leftrightarrow (a : b :: c : d)$$

Donc pour une résolution d'une équation $a : b :: e : ?$, on récrit cette dernière sous la forme $b \oplus e = a \oplus ?$

or d'après le tableau 3.1 $b \oplus e = f$

donc $a \oplus ? = f$ ce qui fait que

$$? = f$$

4.2.2 Résolution d'équations analogiques dans les ensembles finis définis par des attributs binaires

En ce qui concerne l'analogie sur les ensembles, Lepage ([Lep03]) a montré les théorèmes suivants, en respectant les axiomes de l'analogie (section 3.1) :

Théorème 4.2.1 (Solution d'une équation analogique dans les ensembles.) *Soit A , B et C trois ensembles. L'équation analogique $A : B :: C : D$, où D est inconnu, a une solution si et seulement si les conditions suivantes sont vraies :*

$$A \subseteq B \cup C \text{ et } A \supseteq B \cap C$$

La solution est alors unique :

$$D = ((B \cup C) \setminus A) \cup (B \cap C)$$

Si nous supposons que A , B et C sont des sous-ensembles d'un ensemble \mathcal{P} , la solution peut alors être aussi écrite, en notant \bar{A} le complément de A dans \mathcal{P} , comme l'union de trois ensembles disjoints :

$$D = (B \cap \bar{A}) \cup (C \cap \bar{A}) \cup (A \cap B \cap C)$$

Ce théorème s'applique aussi à la résolution d'équations analogiques pour les ensembles X définis par des attributs binaires. Rappelons que pour chaque $x \in X$ et chaque $i \in [1, n]$, $f_i(x) = 1$ (respectivement $f_i(x) = 0$) signifie que l'attribut binaire f_i prend la valeur *VERAI* (respectivement *FAUX*) sur l'objet x .

Soit $A : B :: C : D$ une équation analogique où D est l'inconnue. Pour chaque attribut f_i , il y a seulement huit possibilités différentes pour les valeurs dans A , B et C . À partir du théorème précédent, nous pouvons déduire D en utilisant les deux principes suivants :

- chaque attribut $f_i(D)$ peut être calculé indépendamment ;
- le tableau suivant donne la solution $f_i(D)$:

$f_i(A)$	0	0	0	0	1	1	1	1
$f_i(B)$	0	0	1	1	0	0	1	1
$f_i(C)$	0	1	0	1	0	1	0	1
$f_i(D)$	0	1	1	?	?	0	0	1

Dans deux cas parmi les huit, $f_i(D)$ n'a pas de solution ou n'existe pas. Cela dérive de la définition de X par des attributs binaires, qui est équivalent à définir X comme un ensemble d'attributs. Le théorème 4.2.1 impose des conditions sur la résolution d'équations analogiques sur les ensembles finis, qui implique que ces deux équations analogiques binaires ($0 : 1 :: 1 : ?$ et $1 : 0 :: 0 : ?$) n'ont pas de solution¹.

4.3 Résolution d'équations analogiques dans \mathbb{R}^n

Résoudre l'équation analogique $u : v :: w : x$, où u , v et w sont des vecteurs de \mathbb{R}^n et x l'inconnue, dérive directement de la définition de l'analogie dans un espace vectoriel : les quatre vecteurs doivent former un parallélogramme. Il y a toujours une solution unique, donnée par l'équation :

$$\overrightarrow{Ox} = \overrightarrow{Ov} + \overrightarrow{Ow} - \overrightarrow{Ou}$$

4.4 Résolution d'équations analogiques dans les séquences

4.4.1 Résolution selon Lepage

Lepage[Lep98] présente une méthode de résolution d'équation analogique sur les séquences. Pour ce faire, il introduit une pseudo-distance "*pdist*" qui ressemble à une distance d'édition entre séquence et qui se calcule de la même façon que cette dernière à la différence près que le coût de l'insertion est égal à zéro. On a donc $pdist(like, unlike) = 0$, en revanche $pdist(unlike, like) = 2$ alors qu'une distance d'édition aurait donné $DistEdition(like, unlike) = 2$, $DistEdition(unlike, like) = 2$.

Notons qu'on cherche à trouver D dans une équation analogique sur les séquences $(A : B :: C : D)$. Afin d'y parvenir, on va en premier lieu trouver toutes les sous-

¹Certains auteurs proposent de donner la solution 0 à la première, 1 à la seconde. Ces propositions ne vérifient pas les axiomes de l'analogie que nous avons choisi de suivre (voir le paragraphe 3.1).

séquences communes entre A et B et entre A et C et entre A , B et C . En second lieu, on va construire la séquence D en y mettant :

- les sous-séquences communes à A , B et C ;
- les sous-séquences qui sont dans B et qui ne figurent pas dans A ;
- les sous-séquences qui sont dans C et qui ne figurent pas dans A .

L'insertion des sous-séquences qui vont former la séquence D se fait dans leur ordre d'apparition dans les séquences originales.

On notera que si la séquence A contient des sous-séquences qui ne figurent ni dans B ni dans C la résolution d'après cet algorithme est impossible. En revanche une sous-séquence de B ou de C peut ne pas figurer dans A et c'est d'ailleurs la raison pour laquelle on ne compte pas dans la pseudo-distance le coût d'une insertion. La condition nécessaire et suffisante pour que la résolution puisse se faire d'après [Lep98], est que :

$$|A| = pdist(A, B) + pdist(A, C) + com(A, B, C)$$

avec $com(A, B, C)$ le nombre de caractère en commun entre les séquences A , B et C et $|A|$ la longueur de la séquence A .

On remarque que dans cette version de l'algorithme de résolution on ne traite que des analogies du type $x : x :: y : y$ ou $x : y :: x : y$ avec $x, y \in \Sigma \cup \{\sim\}$, mais on ne traite pas d'analogies du type $x : y :: z : t$ même si cette dernière est vraie. Pour remédier donc à cette lacune, Lepage, dans [Lep01], introduit une amélioration de sa méthode pour prendre en compte cette dernière analogie dans la résolution en définissant ce qu'il appelle un *langage de chaînes analogiques*.

4.4.2 Résolution selon Yvon et Stroppa

Dans ce paragraphe on va introduire quelques définitions introduites par [Yvo03][YSMD04] nécessaire à la compréhension du déroulement de la résolution d'équation analogique dans les séquences selon ces travaux.

Définition 4.4.1 (Shuffle [Yvo03].) Soient u, v deux mots appartenant à Σ^* et u', v' leur équivalent sémantique (voir définition 3.6.2), alors :

$$u \bullet v = \{u'_1 v'_1 u'_2 v'_2 \dots u'_n v'_n \mid \forall u', v'\}$$

Les propriétés du shuffle sont les suivantes :

$$\begin{aligned} u \bullet \sim &= \{u\} \\ u \bullet v &= v \bullet u \text{ (commutativité)} \\ (u \bullet v) \bullet w &= u \bullet (v \bullet w) \text{ (associativité)} \\ u(v \bullet w) &\subset (uv) \bullet w \end{aligned}$$

Si $u = abc$, $v = ef$, alors $u \bullet v$ contient les mots suivants : $abcef$, $abefc$, $aebcf$... mais ne contient pas le mot $adfce$ car la lettre e dans le mot v figure avant la lettre c ce qui n'est pas le cas dans $adfce$.

Définition 4.4.2 (Mot complémentaire [Yvo03].) Soit $u, v \in \Sigma^*$ tel que $u \subset v$, alors le complémentaire de u dans v , noté $v \setminus u$, est l'ensemble des mots v auquel on a retiré de façon ordonnée les lettres présents dans u .

Si $u = ce$ et $v = acbecde$ alors $v \setminus u$ contient les mots suivants : $abcde, acbed \dots$ mais ne contient pas le mot suivant $acbde$ car on a retiré ec et non ce .

On peut dire donc que [YSMD04]

$$w \in (u \bullet v) \Leftrightarrow u \in w \setminus v$$

Il s'en suit que, toujours selon [YSMD04], pour qu'il y ait analogie entre quatre éléments, la condition suivante est nécessaire et suffisante :

$$x : y :: z : t \Leftrightarrow x \bullet t \cap y \bullet z \neq \emptyset$$

Par conséquent, une définition formelle d'une solution d'une équation analogique est d'après [Yvo03] :

$$t \text{ est une solution de l'équation } x : y :: z : ? \Leftrightarrow t \in y \bullet z \setminus x$$

Ensuite pour résoudre réellement l'équation analogique, [YSMD04] utilise un solveur d'analogie à états finis et démontre donc que la solution t définit un ensemble rationnel qui peut donc être calculé par un automate, sachant que les opérations de shuffle et de mot complémentaire peuvent se représenter par un transducteur (un transducteur à état finis est un automate à état finis qui a une entrée et une sortie).

Un exemple de la sortie du transducteur est donné pour la résolution de l'équation suivante ([YSMD04]) : “**wolf est à leaf comme wolves est à ?**”

x	$=$	w	o	l	\smile	\smile	\smile	f	\smile	\smile	\smile
y	$=$	\smile	\smile	\smile	l	e	a	f	\smile	\smile	\smile
z	$=$	w	o	l	\smile	\smile	\smile	\smile	v	e	s
t	$=$	\smile	\smile	\smile	l	e	a	\smile	v	e	s

Ce dernier automate traite uniquement de ce qu'on appelle analogie triviale du type $x : x :: y : y$ ou $x : y :: x : y$ avec $x, y \in \Sigma \cup \{\smile\}$. Une amélioration traitant des analogies non triviales, du type $x : y :: z : t$, est proposée dans un deuxième temps dans [YSMD04] et qui utilise de façon similaire les automates pour la résolution généralisée.

4.4.3 Notre approche

Notre approche est similaire à celle de Yvon et Stroppa et traite donc de tous les types d'analogie, de plus elle traite même des analogies non exactes. C'est le cas où on n'a pas de solution sur les objets, c'est pourquoi on a besoin d'introduire avant la notion de dissemblance analogique dans les séquences pour présenter notre approche, chose qui sera faite dans la section 5.6. La méthode de résolution est ensuite décrite dans la section 5.8.4.

Chapitre 5

La dissemblance analogique

Sommaire

5.1	Motivation	56
5.2	Dissemblance analogique dans les ensembles finis binaires	56
5.2.1	Définition	56
5.2.2	Exemple	57
5.2.3	Propriétés	58
5.3	Dissemblance analogique dans un espace vectoriel	58
5.4	Dissemblance analogique dans un espace métrique	60
5.5	Dissemblance analogique approchée entre séquences	61
5.5.1	La relation “ <i>est à</i> ”.	61
5.5.2	La relation “ <i>comme</i> ”.	62
5.6	Dissemblance analogique entre séquences	65
5.6.1	Définition	65
5.6.2	SEQUANA4	66
5.6.3	Remarques	67
5.7	Dissemblance Analogique exacte vs. Approchée	68
5.8	Résolution généralisée des équations analogiques	69
5.8.1	Présentation	69
5.8.2	Résolution généralisée dans les ensembles finis	70
5.8.3	Résolution généralisée dans un espace vectoriel	70
5.8.4	Résolution généralisée dans les séquences	70
5.9	Expérimentations	73
5.9.1	Dispersion des triplets dans un espace vectoriel	73
5.9.2	Classification d’éléments de un espace vectoriel	74
5.9.3	Dissemblances analogiques entre séquences	74

Contenu du chapitre

Dans ce chapitre, nous abordons le problème de l’approximation d’une proportion analogique. Nous introduisons la notion de dissemblance analogique sur des ensembles

structurés et les différentes propriétés que doit vérifier cette mesure. Nous développons en particulier, grâce à cet outil, une méthode généralisée de résolution d’une équation analogique dans les séquences.

5.1 Motivation

Dans cette section, nous définissons une proportion analogique approximative, que pourrait traduire l’expression linguistique “*a est à b à peu près ce que c est à d*”. Pour rester cohérent avec les définitions précédentes, nous mesurons le terme “à peu près” par une certaine valeur réelle positive, égale à 0 quand l’analogie est exacte, et croissante au fur et à mesure que les quatre objets sont de moins en moins en proportion. Nous voulons aussi que cette valeur, que nous appelons “dissemblance analogique” (*DA*), ait des bonnes propriétés vis-à-vis de l’analogie. Il est souhaitable qu’elle soit symétrique, qu’elle reste inchangée lors de permutation des termes moyens de l’analogie et enfin qu’elle vérifie l’inégalité triangulaire. Ces propriétés nous permettront, dans le chapitre 6, de présenter un algorithme d’apprentissage dont le principe est d’extraire, d’un ensemble d’apprentissage, le triplet d’objets qui a la moindre dissemblance analogique avec un autre objet inconnu. Cette technique de “lazy learning” sera donc une généralisation de la méthode classique des plus proches voisins. Les propriétés de la *DA* permettront de généraliser un algorithme classique de recherche rapide de plus proche voisin avec lequel nous espérons rendre opératoire cet apprentissage.

En premier lieu, nous définissons la dissemblance analogique dans les mêmes ensembles structurés que ceux de la section précédente. En deuxième lieu nous étendons la notion de dissemblance analogique aux séquences.

5.2 Dissemblance analogique dans les ensembles finis définis par des attributs binaires

5.2.1 Définition

Nous savons depuis la section 4.2.2 que quatre éléments d’un ensemble fini X défini par des attributs binaires sont en proportion analogique $u : v :: w : x$ si est seulement si, pour chaque attribut, les quatre valeurs forment une des six colonnes de la table suivante :

$f_i(u)$	0	0	0	1	1	1
$f_i(v)$	0	0	1	0	1	1
$f_i(w)$	0	1	0	1	0	1
$f_i(x)$	0	1	1	0	0	1

La dissemblance analogique entre quatre objets doit refléter en quelque sorte de combien ces objets ratent la proportion analogique. Nous définissons tout d’abord le cas où les objets sont définis sur un seul attribut, ensuite nous le ferons pour tout nombre d’attributs.

Définition 5.2.1 (Dissemblance analogique entre objets binaires.)

La dissemblance analogique entre quatre valeurs binaires est nulle si on est en analogie exacte et à un sinon. On peut aussi affecter la valeur deux aux analogies où il faut modifier deux valeurs pour avoir une analogie exacte.

Le tableau suivant résume bien la définition précédente :

u	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
v	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
w	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
x	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
$AD(u, v, w, t)$	0	1	1	0	1	0	2	1	1	2	0	1	0	1	1	0
							1			1						

Cette définition assure d'abord que la dissemblance est nulle dans les six cas où les quatre éléments sont en proportion analogique. Ensuite, elle attribue la valeur "1" à huit cas simples, où seul le quatrième attribut peut être changé pour retrouver une proportion analogique exacte. Il y a enfin deux cas spéciaux où deux valeurs non nulles ("2" et "1") sont indiquées comme possibles. Attribuer la valeur "1" a l'avantage de garder la dissemblance entre quatre valeurs binaires comme une valeur binaire. Choisir la valeur "2" à la place reflète le fait que cette dissemblance est plus grande au regard des autres, puisque que la solution de l'équation correspondante n'existe pas et ne peut pas être retrouvée à partir de la valeur du quatrième terme. Ce choix n'a pas de conséquence sur les propriétés de la dissemblance analogique dans les ensembles définis par des attributs binaires. Dans la suite de ce travail, nous ferons toujours ce second choix. Par conséquent, nous posons par définition que : $DA(0, 1, 1, 0) = DA(1, 0, 0, 1) = 2$.

Définition 5.2.2 (DA dans les ensembles à attributs binaires.) La dissemblance analogique $DA(u, v, w, t)$ entre quatre objets u, v, w et t d'un ensemble fini X défini par des attributs binaires est la somme des valeurs de la dissemblance analogique entre les attributs.

5.2.2 Exemple

Soit quatre animaux : *crow*, *raven*, *cat* et *cheetah*, décrits par les attributs suivants :

- f_1 Est un animal
- f_2 A un nom commençant par la lettre "c"
- f_3 A une grande taille
- f_4 Est un animal domestique
- f_5 Est un mammifère
- f_6 Est un poisson
- f_7 Peut voler
- f_8 A une lettre "a" dans la seconde position de son nom
- f_9 pourrait être noir

Ce qui nous donne le tableau suivant :

	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9
crow	1	1	0	0	0	0	1	0	1
raven	1	0	1	0	0	0	1	1	1
cat	1	1	0	1	1	0	0	1	1
cheetah	1	1	1	0	1	0	0	0	0
DA	0	1	0	1	0	0	0	2	1

Dans cet exemple : $DA(\text{crow}, \text{raven}, \text{cat}, \text{cheetah}) = \sum_{i=1}^9 DA(f_i(\text{crow}), f_i(\text{raven}), f_i(\text{cat}), f_i(\text{cheetah})) = 5$

5.2.3 Propriétés

Avec la définition 5.2.1, la dissemblance analogique a les propriétés suivantes :

Propriété 5.2.1 (Propriétés de la DA dans les ensembles d'attributs.)

Cohérence avec l'analogie.

$$\forall u, v, w, x \in X, DA(u, v, w, x) = 0 \Leftrightarrow u : v :: w : x$$

Symétrie de la relation “ce que”. $\forall u, v, w, x \in X, DA(u, v, w, x) = DA(w, x, u, v)$

Échange des moyens. $\forall u, v, w, x \in X, DA(u, v, w, x) = DA(u, w, v, x)$

Inégalité triangulaire. $\forall u, \dots, t \in X, DA(u, v, z, t) \leq DA(u, v, w, x) + DA(w, x, z, t)$

Non symétrie de la relation “est à”. $\exists (u, v, w, x) \in X^4 :$

$$DA(u, v, w, x) \neq DA(v, u, w, x)$$

À partir des deux premières propriétés, nous déduisons que :

$$DA(u, v, w, x) = DA(w, x, u, v) = DA(v, u, x, w) = DA(u, w, v, x) =$$

$$DA(x, v, u, w) = DA(x, w, v, u) = DA(v, x, u, w) = DA(w, u, x, v)$$

La démonstration de la première propriété provient de façon directe de la définition.

La démonstration de la troisième propriété est aussi assez simple. Si la propriété

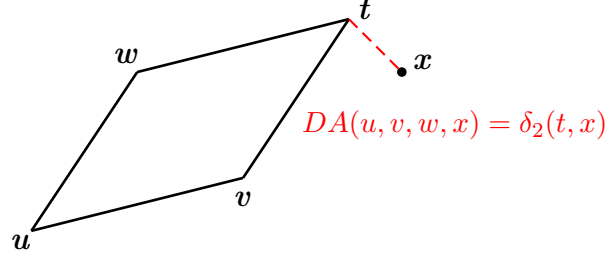
$$DA(f_i(u), f_i(v), f_i(z), f_i(t)) \leq DA(f_i(u), f_i(v), f_i(w), f_i(x)) + DA(f_i(w), f_i(x), f_i(z), f_i(t))$$

est vraie pour chaque 6-uplet d'éléments et chaque attribut f_i , alors la propriété 5.2.1 est vraie.

La démonstration se fait en examinant tout les cas possibles : il est impossible de trouver 6 attributs binaires a, b, c, d, e, f tels que $DA(a, b, e, f) = 2$ et $DA(a, b, c, d) + DA(c, d, e, f) < 2$. Plus précisément, si $DA(a, b, e, f) = 2$, $DA(a, b, c, d) + DA(c, d, e, f)$ est aussi égal à 2 pour toutes les quatre valeurs que (c, d) peut prendre.

5.3 Dissemblance analogique dans \mathbb{R}^n

La dissemblance analogique entre quatre vecteurs doit refléter en quelque sorte de combien ces vecteurs manquent la construction d'un parallélogramme. Quand quatre vecteurs u, v, w et x sont en proportion analogique, ils forment un parallélogramme

FIG. 5.1 – Dissemblance analogique dans un espace vectoriel avec une distance δ_2 .

avec comme cotés opposés \overrightarrow{uv} et \overrightarrow{wx} ce qui équivaut à $\overrightarrow{Ou} + \overrightarrow{Ox} = \overrightarrow{Ov} + \overrightarrow{Ow}$, ou encore, à $u + x = v + w$. Nous avons choisi la définition suivante (voir la figure 5.1) :

Définition 5.3.1 (Dissemblance analogique entre vecteurs.) *La dissemblance analogique entre quatre vecteurs u, v, w et x , de \mathbb{R}^n dans lequel est défini la norme $\|\cdot\|_p$ et la distance correspondante δ_p est donnée par la valeur réelle positive $DA(u, v, w, x) = \delta_p(u + x, v + w) = \|(u + x) - (v + w)\|_p$. La dissemblance analogique est égale aussi à $\delta_p(t, x)$, avec t la solution de l'équation analogique $u : v :: w : t$.*

Propriété 5.3.1 (Propriétés de la DA entre vecteurs.) *Cette définition de la dissemblance analogique dans \mathbb{R}^n garantit que les propriétés suivantes restent vraies : cohérence avec l'analogie, symétrie de la relation “ce que”, échange des moyens, inégalité triangulaire et non-symétrie de la relation “est à”.*

Les deux premières propriétés sont assez évidentes à partir de la définition, contrairement à la troisième qui mérite une démonstration.

Nous voulons prouver que :

$$DA(u, x, v, w) + DA(w, t, x, z) \geq DA(u, t, v, z)$$

qui s'écrit

$$\delta_p(u + x, v + w) + \delta_p(w + t, x + z) \geq \delta_p(u + t, v + z)$$

Les normes $\|\cdot\|_p$ sont par exemple les suivantes :

- $L_1 : \|x\|_1 = \sum_{i=1}^n |x_i|$;
- $L_k : \|x\|_k = \sqrt[k]{\sum_{i=1}^n |x_i|^k}$ (norme euclidienne pour $k = 2$) ;
- $L_\infty : \|x\|_\infty = \max_{i=1}^n |x_i|$.

En réécrivant le troisième terme avec la propriété de la translation :

$$\begin{aligned} \delta_p(u + t, v + z) &= \|(u + t) - (v + z)\|_p = \|(u + t + w + x) - (v + z + w + x)\|_p \\ &= \|(u + x) - (v + w) + (w + t) - (x + z)\|_p \end{aligned}$$

Afin de simplifier, notons :

- $a = (u + x) - (v + w)$;
- $b = (w + t) - (x + z)$.

Et nous avons donc :

$$DA(u, v, z, t) = \|a + b\|_p$$

Or comme $\| \cdot \|_p$ est une norme, elle respecte l'inégalité triangulaire :

$$\|a + b\|_p \leq \|a\|_p + \|b\|_p$$

En redonnant leurs valeurs à a et b , nous obtenons :

$$\|(u + x) - (w + v) + (w + t) - (x + z)\|_p \leq \|(u + x) - (v + w)\|_p + \|(w + t) - (x + z)\|_p$$

Et finalement :

$$\|(u + t) - (v + z)\|_p \leq \|(u + x) - (v + w)\|_p + \|(w + t) - (x + z)\|_p$$

$$DA(u, v, z, t) \leq DA(u, v, w, x) + DA(w, x, z, t)$$

ce qu'il fallait démontrer.

5.4 Commentaire sur la définition de la dissemblance analogique dans un espace métrique

Nous n'avons pas réussi à définir une dissemblance analogique avec toutes les bonnes propriétés dans un espace métrique, *i.e.* un espace où une distance est définie, mais sans autre structure.

Nous avons prouvé que les tentatives de définitions suivantes de la DA :

$$DA(a, b, c, d) = |\delta(a, b) - \delta(c, d)| + |\delta(a, c) - \delta(b, d)|$$

et

$$DA(a, b, c, d) = ((\delta(a, b) - \delta(c, d))^2 + (\delta(a, c) - \delta(b, d))^2)^{\frac{1}{2}}$$

rendent vraie la propriété $a : b :: c : d \Rightarrow DA(a, b, c, d) = 0$, mais ne vérifient pas en général l'inégalité triangulaire.

D'un autre côté, définir la DA de la façon suivante :

$$DA(a, b, c, d) = \min \begin{cases} \delta(a, b) + \delta(c, d) \\ \delta(a, c) + \delta(b, d) \end{cases}$$

assure l'inégalité triangulaire

$$\Delta(S_{a \rightarrow b}, S_{c \rightarrow d}) + \Delta(S_{c \rightarrow d}, S_{e \rightarrow f}) \geq \Delta(S_{a \rightarrow b}, S_{e \rightarrow f})$$

Mais la propriété de la cohérence avec l'analogie n'est plus assurée ; on a seulement : $\Delta(S_{a \rightarrow b}, S_{c \rightarrow d}) = 0 \Rightarrow a : b :: c : d$.

Ceci n'est pas très important du moment que l'apprentissage supervisé par analogie n'est pas concerné. En revanche, les algorithmes rapides ne peuvent être appliqués à la DA que lorsque toutes les propriétés sont vérifiées. Ceci va être développé dans le chapitre 6.

5.5 Dissemblance analogique approchée entre séquences

Le calcul de la dissemblance analogique approchée procède en deux étapes. Nous verrons une définition d'un calcul de la dissemblance analogique (dite exacte) qui procède en une seule étape (section 5.6).

5.5.1 La relation “*est à*”.

Nous avons choisi de définir le terme $u : v$ d'une analogie entre séquences comme leur alignement optimal, c'est à dire comme la séquence optimale de transformations calculée par l'algorithme de Wagner et Fisher. Ce choix permet de décrire précisément comment u se transforme en v : il modélise donc bien une relation “*est à*”.

Par exemple, pour la table de distances :

δ	a	b	c	\smile
a	0	1.5	1.5	1
b	1.5	0	1.3	1
c	1.5	1.3	0	1
\smile	1	1	1	1

la transformation optimale (ici unique) entre $u = abbcc$ et $v = bbcc$ est la séquence

$$\mathcal{S}(u, v) = S_{a \rightarrow \smile} S_{b \rightarrow b} S_{b \rightarrow b} S_{c \rightarrow b} S_{c \rightarrow c}$$

de coût

$$\delta(a, \smile) + \delta(b, b) + \delta(b, b) + \delta(c, b) + \delta(c, c) = 1 + 0 + 0 + 1.3 + 0 = 2.3$$

Nous posons donc que $\mathcal{S}(u, v) = S_{a \rightarrow \smile} S_{b \rightarrow b} S_{b \rightarrow b} S_{c \rightarrow b} S_{c \rightarrow c}$ définit ce que “ u est à v ”.

Nous notons que cette transformation optimale peut ne pas être unique. Par exemple, en prenant une autre distance δ , telle que $\forall a, b \in (\Sigma \cup \{\smile\}), a \neq b, \delta(a, b) = 1$, et $\forall a \in \Sigma, \delta(a, a) = 0$, les transformations entre $u = abbcc$ et $v = bbcc$

$$S_{a \rightarrow \smile} S_{b \rightarrow b} S_{b \rightarrow b} S_{c \rightarrow b} S_{c \rightarrow c}$$

de coût

$$\delta(a, \smile) + \delta(b, b) + \delta(b, b) + \delta(c, b) + \delta(c, c) = 1 + 0 + 0 + 1 + 0 = 2$$

et

$$S_{a \rightarrow b} S_{b \rightarrow b} S_{b \rightarrow b} S_{c \rightarrow c} S_{c \rightarrow \smile}$$

de coût

$$\delta(a, b) + \delta(b, b) + \delta(b, b) + \delta(c, c) + \delta(c, \smile) = 1 + 0 + 0 + 1 + 0 = 2$$

sont toutes les deux optimales. Nous reviendrons plus loin sur cette difficulté.

5.5.2 La relation “comme”.

Soit l'analogie $u : v :: w : x$. Nous connaissons désormais le terme $u : v$, qui est la séquence optimale $\mathcal{S}(u, v)$, supposée pour le moment unique, de transformations élémentaires entre u et v et nous connaissons aussi $w : x$ qui est la séquence $\mathcal{S}(w, x)$.

La relation “exactement comme”. Examinons d'abord un exemple où la relation “comme” est l'identité, avant de la généraliser. Soient les séquences : $u = abbcccb$, $v = bbbcccc$, $w = abbccaab$ et $bbbcaac$. Le calcul, sur la table des distances donnée pour l'exemple du paragraphe 5.5.1, donne :

$$\mathcal{S}(u, v) = S_{a \rightarrow \sim} S_{b \rightarrow b} S_{b \rightarrow b} S_{c \rightarrow b} S_{c \rightarrow c} S_{c \rightarrow c} S_{c \rightarrow c} S_{b \rightarrow c}$$

$$\mathcal{S}(w, x) = S_{a \rightarrow \sim} S_{b \rightarrow b} S_{b \rightarrow b} S_{c \rightarrow b} S_{c \rightarrow c} S_{a \rightarrow a} S_{a \rightarrow a} S_{b \rightarrow c}$$

Il est naturel de mettre toutes les transformations élémentaires d'égalité, qui doivent être de coût nul, dans une même classe d'équivalence¹ notée S_e . Nous obtenons alors :

$$\mathcal{S}(u, v) = S_{a \rightarrow \sim} S_e S_e S_{c \rightarrow b} S_e S_e S_e S_{b \rightarrow c}$$

$$\mathcal{S}(w, x) = S_{a \rightarrow \sim} S_e S_e S_{c \rightarrow b} S_e S_e S_e S_{b \rightarrow c}$$

Dans cet exemple, puisque $\mathcal{S}(u, v) = \mathcal{S}(w, x)$, la relation “comme” de l'équation analogique $abbccaab : bbbcaac :: abbcaabb : bbbcaac$ est donc une identité. Afin de généraliser ce cas particulier, il nous faut définir complètement une distance Δ entre transformations élémentaires. Nous pourrions alors poser la seconde partie de notre construction :

La valeur de la relation « comme » est donnée par la distance d'édition, calculée à partir de Δ , entre $\mathcal{S}(u, v)$ et $\mathcal{S}(w, x)$.

Nous cherchons donc maintenant à définir une distance Δ entre transformations élémentaires sur un alphabet $(\Sigma \cup \{\sim\})$, lui-même muni d'une distance δ .

Des contraintes sur Δ . Pour être cohérent avec les axiomes de l'analogie et la remarque du paragraphe précédent, il faut imposer les conditions suivantes :

- la distance entre deux opérations de transformation identiques est nulle ;

$$\forall a, b \in (\Sigma \cup \{\sim\}), \Delta(S_{a \rightarrow b}, S_{a \rightarrow b}) = 0$$

- insérer ou supprimer des opérations d'égalités de lettres se fait à coût nul ;

$$\forall a \in (\Sigma \cup \{\sim\}), \Delta(-, S_{a \rightarrow a}) = \Delta(S_{a \rightarrow a}, -) = 0$$

- la distance entre deux transformations d'égalité est nulle.

$$\forall a, b \in (\Sigma \cup \{\sim\}), \Delta(S_{a \rightarrow a}, S_{b \rightarrow b}) = 0$$

En effet, si l'une de ces propriétés n'était pas respectée, la distance d'édition obtenue à partir de Δ sur quatre phrases en analogie exacte ne serait pas nulle.

¹Ceci se justifie par le fait que, pour tout a et b , l'équation $a : a :: b : b$ est exacte.

Une proposition pour la distance Δ . Nous proposons de définir Δ par la formule suivante :

$$\Delta(S_{a \rightarrow b}, S_{c \rightarrow d}) = \text{Min} \begin{cases} \delta(a, b) + \delta(c, d) \\ \delta(a, c) + \delta(b, d) \end{cases}$$

Cette définition remplit les contraintes précédentes et assure que Δ possède la propriété de l'inégalité triangulaire : $\Delta(S_{a \rightarrow b}, S_{c \rightarrow d}) + \Delta(S_{c \rightarrow d}, S_{e \rightarrow f}) \geq \Delta(S_{a \rightarrow b}, S_{e \rightarrow f})$. Cependant, on n'a pas : $\Delta(S_{a \rightarrow b}, S_{c \rightarrow d}) = 0 \Leftrightarrow a : b :: c : d$

Fin de la construction de la relation “comme”. Une fois définie la distance Δ entre séquences d'opérations élémentaires, il est facile d'appliquer à nouveau l'algorithme de Wagner et Fisher avec Δ sur le couple de séquences $\mathcal{S}(u, v)$ et $\mathcal{S}(w, x)$. Le coût obtenu quantifie la relation “comme”.

Définition 5.5.1 Nous appelons dissemblance analogique approchée entre les phrases u , v , w et x , notée $\widehat{DA}(u, v, w, x)$, le coût d'édition obtenu avec la distance Δ entre $\mathcal{S}(u, v)$ et $\mathcal{S}(w, x)$. Ces deux derniers termes représentent les deux séquences optimales² de transformations élémentaires entre u et v , d'une part, et entre w et x , d'autre part, obtenues avec la distance δ .

Un exemple. Pour le tableau de distances :

δ	a	b	c	\sim
a	0	1.2	1.5	2
b	1.2	0	1.7	2
c	1.5	1.7	0	2
\sim	2	2	2	2

l'alignement optimal unique entre $u = cbacba$ et $v = babba$ (de valeur 3.7) est :

u	=	c	b	a	c	b	a
v	=	\sim	b	a	b	b	a

et celui, unique aussi, entre $w = cbacbc$ et $x = bcabbc$ (de valeur 5.1) est :

w	=	c	b	a	c	b	c
x	=	b	c	a	b	b	c

et finalement la dissemblance approchée est calculée comme valant 3.7, par l'alignement optimal suivant selon la distance Δ :

$\mathcal{S}(u, v)$	=	$S_{c \rightarrow \sim}$	$S_{b \rightarrow b}$	$S_{a \rightarrow a}$	$S_{c \rightarrow b}$	$S_{b \rightarrow b}$	$S_{a \rightarrow a}$
$\mathcal{S}(w, x)$	=	$S_{c \rightarrow b}$	$S_{b \rightarrow c}$	$S_{a \rightarrow a}$	$S_{c \rightarrow b}$	$S_{b \rightarrow b}$	$S_{c \rightarrow c}$

²Encore une fois, nous supposons pour le moment ces séquences optimales uniques.

Le cas où les séquences d'opérations élémentaires ne sont pas uniques. Nous avons supposé jusqu'ici que, dans le calcul de la relation analogique entre séquences $u : v :: w : x$, l'algorithme de Wagner et Fisher produisait une solution unique avec la distance δ pour calculer l'alignement optimal $\mathcal{S}(u, v)$ entre u et v , comme $\mathcal{S}(w, x)$ entre w et x .

Il est en principe facile de relâcher cette contrainte : notons $\aleph(u, v)$ (respectivement $\aleph(w, x)$) l'ensemble des séquences de transformations élémentaires de coût optimal entre u et v (respectivement entre w et x). Nous pouvons définir l'exactitude de la relation “comme”, c'est à dire la distance analogique approchée entre u et v d'une part, w et x d'autre part, comme le coût d'un alignement optimal entre deux séquences de transformations optimales, quand l'une parcourt $\aleph(u, v)$ et l'autre parcourt $\aleph(w, x)$. On effectue donc au pire $|\aleph(u, v)| \times |\aleph(w, x)|$ alignements entre séquences de transformations élémentaires. En pratique, il est possible de réduire les calculs, car les séquences de transformations optimales sont structurées en graphe sans cycle [SK83].

Quelques problèmes avec la dissemblance analogique approchée.

- on n'a pas en général : $\Delta(S_{a \rightarrow b}, S_{c \rightarrow d}) = 0 \Leftrightarrow a : b :: c : d$;
- pour être cohérent avec les axiomes de l'analogie, il serait souhaitable que l'on ait la même dissemblance analogique entre les huit quadruplets de séquences dont l'analogie se déduit des axiomes, comme on l'a vu au paragraphe 3.1. Ce n'est en général pas le cas de la dissemblance analogique approchée ;
- après que les séquences de transformations qui définissent les deux relations “est à” aient été construites, on peut se demander si cette construction en deux étapes est optimale : il pourrait exister une séquence de transformations $\mathcal{S}'(u, v)$, de coût supérieur à $\mathcal{S}(u, v)$, et une séquence $\mathcal{S}'(w, x)$, de coût supérieur à $\mathcal{S}(w, x)$, telles que la distance d'édition entre $\mathcal{S}'(u, v)$ et $\mathcal{S}'(w, x)$ soit inférieure à celle entre $\mathcal{S}(u, v)$ et $\mathcal{S}(w, x)$.

Reprenons l'exemple du paragraphe 5.5.2 : $\widehat{DA}(cbacba, babba, cbacbc, bcabbc) = 3.7$. Nous pouvons montrer que la construction de \widehat{DA} en deux étapes séparées n'est en effet pas optimale. Il existe deux couples d'alignements entre u et v d'une part, w et x d'autre part, qui fournissent un meilleur résultat. Le premier n'est pas optimal (il a une valeur de 5.4), le second est l'alignement optimal (valeur de 5.1).

$$\begin{array}{cccccccccccc}
 c & b & a & c & b & a & c & b & a & c & b & c \\
 | & | & | & | & | & | & | & | & | & | & | & | \\
 b & \sim & a & b & b & a & b & c & a & b & b & c
 \end{array}$$

Leur alignement optimal par Δ se fait comme suit pour une valeur de 2 :

$$\begin{array}{cccccc}
 S_{c \rightarrow b} & S_{b \rightarrow \sim} & S_{a \rightarrow a} & S_{c \rightarrow b} & S_{b \rightarrow b} & S_{a \rightarrow a} \\
 | & | & | & | & | & | \\
 S_{c \rightarrow b} & S_{b \rightarrow c} & S_{a \rightarrow a} & S_{c \rightarrow b} & S_{b \rightarrow b} & S_{c \rightarrow c}
 \end{array}$$

C'est en particulier pour remédier à ces difficultés que nous allons voir dans la section suivante comment définir et calculer directement ce que nous appelons la *dissemblance*

analogique entre quatre séquences sur Σ , une notion complètement cohérente avec les axiomes de l'analogie et répondant de plus à un critère d'optimalité.

5.6 Dissemblance analogique entre séquences

Nous avons présenté dans la section 5.5 l'algorithme de dissemblance analogique approchée permettant de calculer une solution de l'équation analogique dans les séquences quand il y a une analogie dans l'alphabet augmenté Σ' .

Nous présentons dans ce qui suit deux algorithmes : le premier, nommé SEQUANA4, calcule une dissemblance analogique entre quatre séquences de Σ^* . Le second, nommé SOLVANA, est une généralisation de la dissemblance analogique approchée : en partant d'une équation analogique sur les séquences et d'une analogie sur l'alphabet, SOLVANA produit le DAG (*Directed Acyclic Graph*) de toutes les séquences qui ont la plus petite dissemblance analogique quand elles sont associées aux trois premières séquences connues de l'équation analogique. En particulier, s'il y a des solutions à l'équation analogique, cet algorithme les produit toutes.

Ces deux algorithmes sont assez généraux, dans les sens où ils ne font pas de postulat particulier sur l'alphabet des séquences. Cet alphabet Σ est simplement augmenté à $\Sigma' = \Sigma \cup \{\smile\}$ pour produire un alignement comme décrit dans la section 3.6.3.2. La dissemblance analogique sur Σ' est telle que : $DA(\smile, \smile, a, a) = 0$, et $DA(\smile, a, b, c) > 0$ pour chaque $a, b, c \in \Sigma$, sans plus de contrainte. Nous verrons dans la section 5.6.3 que garder un tel niveau de généralité a des inconvénients. Pour le moment, définissons une dissemblance analogique entre séquences et donnons l'algorithme SEQUANA4 qui la calcule.

5.6.1 Définition

Soit Σ un ensemble dans lequel est défini une dissemblance analogique DA . Nous l'augmentons en Σ' en rajoutant un symbole spécial \smile , qui va être utilisé dans les alignements comme déjà décrit dans la section 3.6.3.2. Nous supposons maintenant qu'une dissemblance analogique est définie dans Σ' . Nous pouvons donc définir :

Définition 5.6.1 (Dissemblance analogique entre quatre séquences.) *la dissemblance analogique $DA(u, v, w, x)$ entre quatre séquences de Σ^* est le coût minimal d'un alignement des quatre séquences.*

Et nous avons les propriétés suivantes :

Propriété 5.6.1 *Cette définition assure que les propriétés suivantes restent vraies :*

Cohérence avec l'analogie. $\forall u, v, w, x \in (\Sigma^*)^4, DA(u, v, w, x) = 0 \Leftrightarrow u : v :: w : x$

Symétrie de la relation “ce que” et échange des moyens. $\forall u, v, w, x \in X,$

$$DA(u, v, w, x) = DA(w, x, u, v) = DA(u, w, v, x)$$

Non symétrie de la relation “est à”. $\exists (u, v, w, x) \in (\Sigma^*)^4,$

$$DA(u, v, w, x) \neq DA(v, u, w, x)$$

5.6.2 Calcul de la dissemblance analogique entre quatre séquences : algorithme SEQUANA4

Nous calculons $DA(u, v, w, x)$ avec un algorithme de programmation dynamique, appelé SEQUANA4, qui progresse de manière synchronisée dans les quatre séquences pour construire un alignement optimal.

L'entrée de l'algorithme est l'alphabet augmenté Σ' dans lequel il y a une dissemblance analogique $DA(a, b, c, d)$. La sortie de cet algorithme est la dissemblance entre quatre séquences de Σ^* , c'est à dire $DA(u, v, w, x)$.

La récurrence de l'algorithme est la suivante :

Initialisation

$$C_{w_0x_0}^{u_0v_0} \leftarrow 0;$$

for $i = 1, |u|$ **do** $C_{w_0x_0}^{u_i v_0} \leftarrow \sum_{k=1}^{k=i} DA(u_i, \smile, \smile, \smile)$ **done**;

for $j = 1, |v|$ **do** $C_{w_0x_0}^{u_0 v_j} \leftarrow \sum_{k=1}^{k=j} DA(\smile, v_j, \smile, \smile)$ **done**;

for $i = 1, |w|$ **do** $C_{w_kx_0}^{u_0 v_0} \leftarrow \sum_{i=1}^{i=k} DA(\smile, \smile, w_k, \smile)$ **done**;

for $j = 1, |x|$ **do** $C_{w_0x_l}^{u_0 v_0} \leftarrow \sum_{k=1}^{k=l} DA(\smile, \smile, \smile, x_l)$ **done**;

Récurrence

$$C_{w_kx_l}^{u_i v_j} = \text{Min} \left\{ \begin{array}{ll} C_{w_{k-1}x_{l-1}}^{u_{i-1}v_{j-1}} + DA(u_i, v_j, w_k, x_l) & [i \leftarrow i+1; j \leftarrow j+1; k \leftarrow k+1; l \leftarrow l+1] \\ C_{w_{k-1}x_l}^{u_{i-1}v_{j-1}} + DA(u_i, v_j, w_k, \smile) & [i \leftarrow i+1; j \leftarrow j+1; k \leftarrow k+1] \\ C_{w_kx_{l-1}}^{u_{i-1}v_{j-1}} + DA(u_i, v_j, \smile, x_l) & [i \leftarrow i+1; j \leftarrow j+1; l \leftarrow l+1] \\ C_{w_kx_l}^{u_{i-1}v_{j-1}} + DA(u_i, v_j, \smile, \smile) & [i \leftarrow i+1; j \leftarrow j+1] \\ C_{w_{k-1}x_{l-1}}^{u_i v_{j-1}} + DA(\smile, v_j, w_k, x_l) & [j \leftarrow j+1; k \leftarrow k+1; l \leftarrow l+1] \\ C_{w_kx_{l-1}}^{u_i v_{j-1}} + DA(\smile, v_j, \smile, x_l) & [j \leftarrow j+1; l \leftarrow l+1] \\ C_{w_{k-1}x_l}^{u_i v_{j-1}} + DA(\smile, v_j, w_k, \smile) & [i \leftarrow i+1; k \leftarrow k+1] \\ C_{w_kx_l}^{u_i v_{j-1}} + DA(\smile, v_j, \smile, \smile) & [j \leftarrow j+1] \\ C_{w_{k-1}x_{l-1}}^{u_{i-1}v_j} + DA(u_i, \smile, w_k, x_l) & [i \leftarrow i+1; j \leftarrow j+1; l \leftarrow l+1] \\ C_{w_kx_{l-1}}^{u_{i-1}v_j} + DA(u_i, \smile, \smile, x_l) & [i \leftarrow i+1; l \leftarrow l+1] \\ C_{w_{k-1}x_l}^{u_{i-1}v_j} + DA(u_i, \smile, w_k, \smile) & [i \leftarrow i+1; k \leftarrow k+1] \\ C_{w_kx_l}^{u_{i-1}v_j} + DA(u_i, \smile, \smile, \smile) & [i \leftarrow i+1] \\ C_{w_{k-1}x_{l-1}}^{u_i v_j} + DA(\smile, \smile, w_k, x_l) & [k \leftarrow k+1; l \leftarrow l+1] \\ C_{w_kx_{l-1}}^{u_i v_j} + DA(\smile, \smile, \smile, x_l) & [l \leftarrow l+1] \\ C_{w_{k-1}x_l}^{u_i v_j} + DA(\smile, \smile, w_k, \smile) & [k \leftarrow k+1] \end{array} \right.$$

End

avec $i = |u|$, $j = |v|$, $k = |w|$ et $l = |x|$.

Résultat

$C_{w_{|w|}x_{|x|}}^{u_{|u|}v_{|v|}}$ est $DA(u, v, w, x)$ dans Σ^* .

Complexité

Cet algorithme a une complexité temporelle en $\mathcal{O}(|u|.|v|.|w|.|x|)$.

Exactitude

L'exactitude de cet algorithme peut être démontrée par récurrence, puisqu'il utilise le principe de programmation dynamique. Il a seulement besoin de la dissemblance analogique sur Σ' afin de respecter les propriétés de la *cohérence avec l'analogie*, de la *symétrie de "ce que"* et *l'échange des moyens*. La propriété de l'*inégalité triangulaire* n'est pas nécessaire.

5.6.3 Remarques

5.6.3.1 Inégalité triangulaire

Dans les propriétés données dans la section 5.6.1, il en manque malheureusement une. Nous aurions aimé avoir l'*inégalité triangulaire* :

$$\forall (u, v, w, x, z, t) \in (\Sigma^*)^6, DA(u, v, w, x) \leq DA(u, v, z, t) + DA(z, t, w, x)$$

Malheureusement, elle n'est pas toujours vraie. Considérons par exemple l'alphabet $\{a, b, c, d, A, B, C, D, \alpha, \beta, \gamma, \delta\}$, défini par les attributs binaires suivants :

	f_1	f_2	f_3	f_4	f_5	f_6	f_7
a	1	0	0	0	1	0	0
b	0	1	0	0	1	0	0
c	0	0	1	0	1	0	0
d	0	0	0	1	1	0	0
A	1	0	0	0	0	1	0
B	0	1	0	0	0	1	0
C	0	0	1	0	0	1	0
D	0	0	0	1	0	1	0
α	1	0	0	0	0	0	1
β	0	1	0	0	0	0	1
γ	0	0	1	0	0	0	1
δ	0	0	0	1	0	0	1

La dissemblance analogique entre $u = a\alpha c$, $v = b\alpha C$, $w = A\alpha d$ et $x = B\alpha d$ est nulle, selon l'alignement optimal :

a	α	c
b	α	C
A	α	d
B	α	D

De façon similaire, pour $y = \alpha\alpha d$ et $t = B\alpha\delta$, nous calculons $DA(w, x, y, t) = 0$ par

l'alignement optimal :

\sim	\sim	A	α	d
B	α	D	\sim	\sim
\sim	\sim	α	α	d
B	α	δ	\sim	\sim

En calculant maintenant $DA(u, v, y, t)$, nous constatons qu'il n'existe pas d'alignement à DA nulle. L'alignement optimal est le suivant (il a une valeur de DA égale à 4) :

a	α	c
b	α	C
α	α	d
B	α	δ

5.6.3.2 Comment ajouter \sim à Σ ?

Nous n'avons toujours pas donné d'indication sur la manière pratique d'ajouter \sim . Tout ce qui est demandé est une dissemblance analogique dans $\Sigma' \cup \{\sim\}$, qui vérifie les propriétés de la *cohérence avec l'analogie*, la *symétrie de "ce que"* et l'*échange des moyens*. Une façon simple est de choisir \sim comme un nouvel élément de l'alphabet, par exemple l'origine de \mathbb{R}^n ou un élément dont tout les attributs sont nuls dans le cas d'un ensemble d'éléments définis par des attributs binaires.

5.7 Dissemblance Analogique vs. Dissemblance Analogique Approchée

Nous avons introduit la notion de Dissemblance Analogique Approchée (DAA) car la complexité de son calcul tant spatiale que temporelle est largement inférieure à celle de la Dissemblance Analogique exacte. En effet, la complexité de la Dissemblance Analogique Approchée est de

$$\mathcal{O}\left(\underbrace{(|u| \cdot |v| + |w| \cdot |x|)}_{\substack{u \text{ sur } v \quad w \text{ sur } x}} + \underbrace{((|u| + |v|) \cdot (|w| + |x|))}_{\text{taille des alignements de seq. d'édition}} \times \underbrace{|\aleph(u, v)| \times |\aleph(w, x)|}_{\text{nb. d'alignements opt. de seq. d'édition}}\right)$$

$$= \mathcal{O}(l^2), \text{ avec } l \text{ la longueur moyenne des séquences}$$

Alors que la complexité de la Dissemblance Analogique exacte est $\mathcal{O}(l^4)$.

Nous avons utilisé notamment la Dissemblance Analogique Approchée lors d'une application sur les séquences de protéine dont la longueur de séquence ne nous permettait pas d'utiliser la Dissemblance Analogique exacte.

On peut passer de la DAA à la DA . En effet, il suffit de calculer le coût minimal de la distance d'édition calculée par la distance Δ entre $\mathcal{S}'(u, v)$ et $\mathcal{S}'(w, x)$, quand ces deux termes parcourent : l'un tous les alignements entre u et v et l'autre tous les alignements entre w et x . Toutefois ce passage a un coût important en termes de complexité algorithmique.

5.8 Résolution généralisée des équations analogiques

5.8.1 Présentation

Jusqu'à maintenant, en particulier dans le chapitre 4 nous avons considéré qu'une équation analogique possède zéro, une ou plusieurs solutions exactes. Dans le cas où il n'y a pas de solution, la notion de dissemblance analogique va permettre d'être plus précis : elle autorise le concept de *solution approchée*.

Définition 5.8.1 (Meilleures solutions approchées d'une équation analogique.)

Soit X un ensemble dans lequel une dissemblance analogique est définie. Soit $a : b :: c : x$ une équation analogique dans X . L'ensemble des meilleures solutions approchées de cette équation est défini par : $\{ \underset{y \in X}{\text{ArgMin}} DA(a, b, c, y) \}$

Autrement dit, les meilleures solutions approchées sont les objets $y \in X$ qui sont les plus proches d'être en proportion analogique avec a , b et c . Elles produisent toutes la même valeur $DA(a, b, c, y)$, qui est la plus petite que l'on peut obtenir avec les trois premiers termes fixés à a , b et c . Évidemment, cette définition généralise celle de la solution de l'équation analogique donnée dans la section 3.3. Puisque nous avons défini la DA avec les bonnes propriétés sur certains alphabets et sur les séquences d'objets de ces alphabets, nous pouvons résoudre de manière généralisée les équations analogiques sur tous ces domaines.

Nous pouvons élargir encore ce concept en définissant les k -meilleures solutions de l'équation analogique $a : b :: c : x$. Nous supposons que les éléments de X sont ordonnées par dissimilarité analogique croissante avec a, b et c : l'ensemble des k meilleures solutions est constitué des k premiers éléments de cette liste. Dans le cas où l'élément de rang $k + 1$ a la même DA avec a , b et c que celui de rang k , cette définition demanderait à être affinée pour produire un ensemble unique. Nous traitons ce problème dans un cas pratique au chapitre 6 en augmentant k jusqu'à la plus petite valeur k' pour laquelle l'ensemble des meilleures solutions est défini de manière unique.

Après avoir rapidement traité le problème de la résolution généralisée d'équations analogiques dans le cas des ensembles finis et dans \mathbb{R}^n , nous détaillons un algorithme (SOLVANA) qui produit l'ensemble des meilleures solutions approchées à l'équation $a : b :: c : x$ quand les objets sont des séquences de lettres d'un alphabet dans lequel une DA est définie. Nous lui ajouterons aussi quelques modifications pour lui permettre de trouver l'ensemble des k -meilleures solutions.

5.8.2 Résolution généralisée dans les ensembles finis

Dans les ensembles finis d'objets définis par des attributs binaires, il y a huit équations analogiques, dont six possèdent une solution exacte (voir le paragraphe 4.2.2). La résolution généralisée a donc deux cas à traiter, l'équation $0 : 1 :: 1 : x$ et sa symétrique. Selon le principe de minimisation de la dissemblance analogique, la première équation aura pour solution $x = 1$, puisque $DA(0, 1, 1, 1) = 1$ et l'équation symétrique $1 : 0 :: 0 : x$ aura pour solution approchée $x = 0$, pour la même valeur de dissemblance analogique.

Remarquons qu'il n'y a pas d'ambiguïté : la solution $x = 0$ à l'équation $0 : 1 :: 1 : x$ produit une dissemblance analogique de valeur 2.

5.8.3 Résolution généralisée dans \mathbb{R}^n

La résolution d'une équation analogique, dont les objets appartiennent à un espace vectoriel, est immédiate. Il suffit de résoudre vectoriellement : $\vec{AB} = \vec{CD}$, avec A , B , C et D les objets de la proportion analogique. Notons que la résolution dans \mathbb{R}^n est toujours exacte.

5.8.4 Résolution généralisée dans les séquences

5.8.4.1 L'algorithme SOLVANA

Au paragraphe 4.4.3, nous avons remis à plus tard la résolution d'équations analogiques dans les séquences, selon notre définition de la proportion analogique entre séquences. L'introduction de la notion de dissemblance analogique va maintenant nous permettre de traiter ce point important. Nous présentons dans ce paragraphe un algorithme appelé SOLVANA, qui produit l'ensemble des meilleures solutions approchées à une équation analogique dans les séquences, ce qui inclut bien entendu les solutions exactes s'il y en a. Il utilise la programmation dynamique pour construire d'abord une matrice à trois dimensions. Quand cette phase de construction est terminée, il procède à un *backtracking* pour produire le DAG (*Directed Acyclic Graph*) de toutes les meilleures solutions.

Le principe est donc de réaliser un alignement des quatre séquences de longueur différente en insérant un certain nombre de lettres \sim , afin que toutes les séquences aient la même longueur. Par programmation dynamique, on assure que la somme sur cette longueur des dissemblances analogiques dans l'alphabet augmenté soit minimale.

Rappelons que nous définissons le coût d'un alignement comme la somme des dissemblances analogiques sur les colonnes, et qu'un *alignement optimal* possède par définition un coût minimal. La dissemblance analogique entre quatre séquences (DAS) est donc le coût d'un alignement optimal.

Wagner et Fisher [WF74] ont présenté un algorithme de programmation dynamique pour trouver un alignement optimal et la distance entre deux séquences, par la construction d'une matrice de $n_1 \times n_2$ (n_1 et n_2 sont respectivement la longueur de la première et de la deuxième séquence qu'on cherche à aligner). SOLVANA étend cette méthode

à un alignement de trois séquences, en construisant une matrice M à trois dimensions de taille $n_1 \times n_2 \times n_3$, les longueurs respectives de la première, de la deuxième et de la troisième séquence A , B et C de l'équation analogique “ A est à B ce que C est à ?”).

En notant a_i le i^{eme} objet de la séquence A (de même pour B et C) et x la ou les lettres produites à chaque étape de l'algorithme, la progression est la suivante :

$$M[i, j, k] \leftarrow \min_{1 \leq i, j, k \leq n_1, n_2, n_3} \begin{cases} M[i-1, j-1, k-1] + \min_{x \in \Sigma'} DA(a_{i-1}, b_{j-1}, c_{k-1}, x) \\ M[i, j-1, k-1] + \min_{x \in \Sigma'} DA(\wr, b_{j-1}, c_{k-1}, x) \\ M[i, j, k-1] + \min_{x \in \Sigma'} DA(\wr, \wr, c_{k-1}, x) \\ M[i, j-1, k] + \min_{x \in \Sigma'} DA(\wr, b_{j-1}, \wr, x) \\ M[i-1, j, k-1] + \min_{x \in \Sigma'} DA(a_{i-1}, \wr, c_{k-1}, x) \\ M[i-1, j-1, k] + \min_{x \in \Sigma'} DA(a_{i-1}, b_{j-1}, \wr, x) \\ M[i-1, j, k] + \min_{x \in \Sigma'} DA(a_{i-1}, \wr, \wr, x) \end{cases}$$

En réalité, nous sauvegardons à chaque étape dans la case $M[i, j, k]$ non seulement le coût, comme indiqué ci-dessus, mais aussi la ou les lettre(s) x trouvées par la résolution de l'équation analogique au long de la progression. Quand la construction de M est terminée, le *backtracking* produit toutes les séquences optimales générées avec la même dissemblance analogique optimale, structurées en DAG.

La complexité temporelle de cet algorithme est $O(m * n^3)$, avec $m = Card(\Sigma')$ et n la longueur moyenne des séquences.

5.8.4.2 Exemple

Soit $\Sigma = \{a, b, c, A, B, C\}$ un alphabet défini par 5 attributs binaires :

	f_1	f_2	f_3	f_4	f_5
a	1	0	0	1	0
b	0	1	0	1	0
c	0	0	1	1	0
A	1	0	0	0	1
B	0	1	0	0	1
C	0	0	1	0	1
\wr	0	0	0	0	0

Les trois premiers attributs indiquent la nature de la lettre (par exemple, f_1 est vrai pour a et A seulement) et les deux derniers attributs indiquent la casse de la lettre (f_4 est vrai pour les minuscules, f_5 pour les majuscules).

Soit $ab : Bc :: Bc : x$ une équation analogique dans Σ . On remarque qu'il n'existe pas de solution exacte, mais plusieurs meilleures solutions approchées y avec $DA(ab, Bc, Bc, y) = 4$, par exemple $y = BB$ ou $y = Cc$. La figure 5.2 montre le résultat de SOLVANA. Chaque chemin dans le graphe correspond à une solution optimale. Par exemple, le

chemin du bas décrit la solution $y = Cc$. La composition de son coût, qui vaut 4, est également lisible sur la figure 5.2.

$$\begin{array}{ccc}
 a & b & \sim \\
 | & | & | \\
 \sim & B & c \\
 | & | & | \\
 b & c & \sim \\
 | & | & | \\
 \sim & C & c
 \end{array}$$

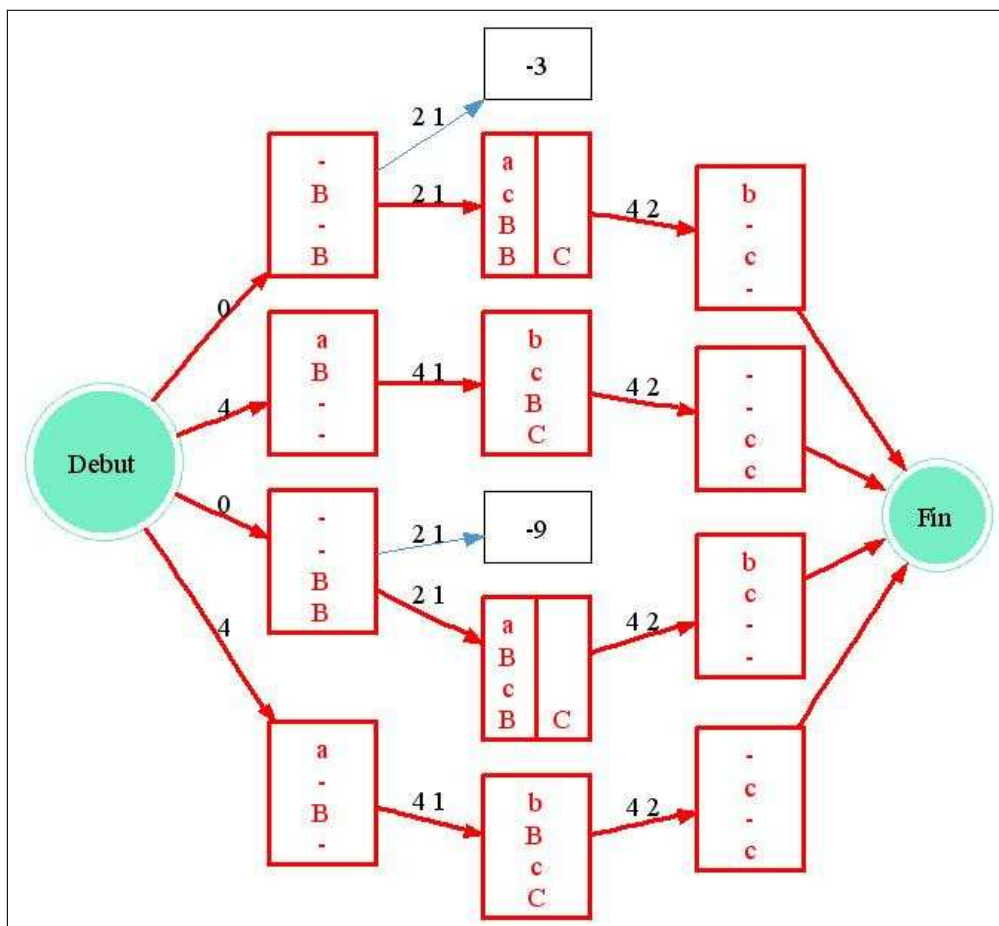


FIG. 5.2 – Résultat de SOLVANA : le DAG des meilleures solutions approchées de l'équation analogique sur les séquences $ab : Bc :: Bc : ?$.

5.8.4.3 Recherche des k meilleures solutions.

SOLVANA produit toutes les solutions de coût minimal à une équation analogique entre séquences. Il peut être intéressant de produire toutes les solutions de coût inférieur à une valeur donnée, ou les k meilleures solutions, pour k fixé. D'où l'utilisation de l'algorithme A^* .

Algorithme A^* L'algorithme A^* est un algorithme général de recherche [Nil80] qui peut être utilisé afin de trouver les k chemins les plus courts dans un graphe [Epp98]. Nous pouvons transposer la progression dans M en un problème de plus courts chemins, en remplaçant les éléments de M par des nœuds connectés à sept autres nœuds, comme c'est le cas dans la programmation dynamique vue dans la section 5.8.4.1. Pour être sûr d'obtenir des solutions exactes, nous avons affecté à l'heuristique de A^* la valeur zéro (et par conséquent réduit cet algorithme à une *breadth-first search*). L'étude d'heuristique admissible n'a pas été abordée dans ce travail.

La complexité de l'algorithme est $O(v + m * n^3 \lg_2(n) + k) = O(m * n^3 \lg_2(n))$, avec

- k le nombre de meilleurs solutions ;
- n la longueur moyenne d'une séquence ;
- $v = 7 * n^3$ le nombre de nœuds ;
- m le cardinal de Σ' .

En réalité, la complexité dans le pire des cas est $O(m * 7^n)$. D'un côté l'algorithme A^* est nettement plus lent en temps que celui par construction de matrice vu dans la section 5.8.4.1 même si on n'est jamais dans le pire des cas. D'un autre côté l'algorithme A^* est capable de produire l'ensemble des k meilleures solutions pour n'importe quelle valeur de k , au lieu de l'ensemble des meilleures solutions ayant la même dissemblance analogique.

5.9 Expérimentations

5.9.1 Expérimentations sur la dispersion des triplets dans \mathbb{R}^2

Ayant vu comment se faisait la résolution d'équations analogiques dans \mathbb{R}^n , on propose ici de prendre une base de données contenant m éléments dans \mathbb{R}^2 et de construire tous les triplets possibles à partir des m éléments, ce qui fera m^3 éléments. À partir des m^3 éléments, on résout par analogie pour trouver le quatrième élément pour ensuite voir la dispersion de ces éléments dans l'espace. On a choisi de classer les m éléments de la base de données en deux classes, représentés dans la figure 5.3 par des points rouges et des points bleus. Les éléments de la première (respectivement deuxième) classe produiront des éléments de la première (respectivement deuxième) classe. Lors d'un conflit de classe, on choisit de classer le nouvel élément dans une troisième classe représentée par des points verts (figure 5.3).

On peut voir sur la figure 5.3 que les points rajoutés par l'analogie respectent relativement bien la dispersion initiale des deux classes. On pourrait penser que dans

certaines applications on pourrait éventuellement agrandir ou élargir la base de données en utilisant cette méthode, comme on le verra ultérieurement dans le chapitre 7.

Dans l'exemple qu'on a pris, on n'a retenu que les éléments construits par analogie d'ordre 1. Par ordre 1, on entend que les trois premiers termes de l'analogie appartiennent à la base de données initiale. Mais on pourrait aussi résoudre une équation analogique dont le premier terme par exemple a lui même été construit (voir [Str05]), etc.

5.9.2 Expérimentations sur la classification d'éléments de \mathbb{R}^2

Soit \mathcal{S} une base d'exemples étiquetés. Ces exemples ne peuvent appartenir qu'à une des deux classes C_1 ou C_2 . Ces exemples appartiennent à un espace vectoriel à deux dimensions. Le but ici est de voir la surface de séparation d'un classificateur d'objets numérique par analogie sans aucun pré-traitement des données ni de pondération.

Afin d'y parvenir, pour chaque point de l'espace vectoriel, on calcule les 100 plus proches (au sens de la dissemblance analogique) triplets de l'ensemble d'apprentissage qui forment une analogie avec comme quatrième élément un point de l'espace vectoriel. Chacun de ces 100 triplets est choisi de façon à ce que leurs classes aient la forme suivante (C_a, C_a, C_b) , on dira donc que ce triplet classe le point de l'espace vectoriel dans C_b .

On a choisi de montrer la surface de séparation d'autres classificateurs classiques comme :

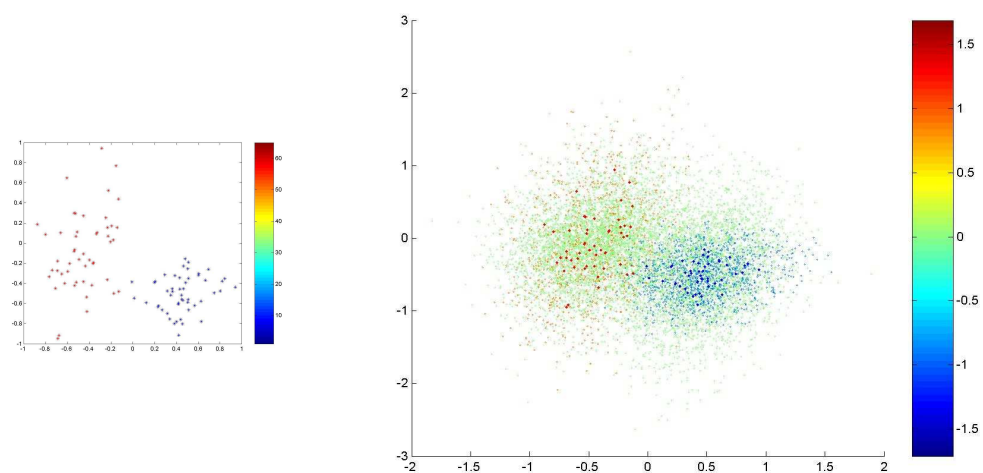
- *PPV* : on prend le 10 plus proches voisins ;
- *SVM* : noyau Gaussien, $\sigma = 0.05$;
- *Perceptron* : nombre itération = 500 ;
- *RBF* : nombre de couches cachées = 20 .

On remarque sur la figure 5.4, pour une séparation d'objets à deux classes, que tous les classificateurs, hormis le perceptron qui est un classificateur linéaire, ont donné à peu près la même surface de séparation.

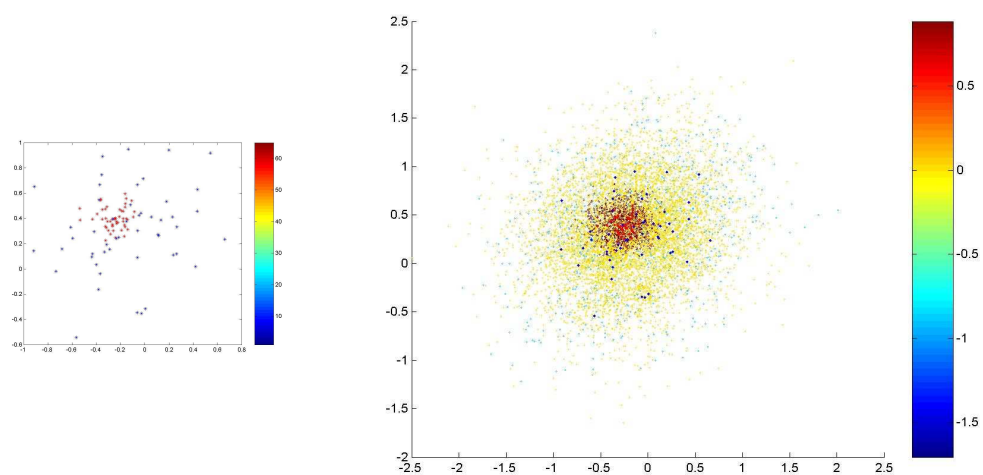
En revanche, on note sur la figure 5.5 que le classificateur par analogie sur les objets numériques se distingue des autres essentiellement en bas des images. On voit en haut à gauche de la figure 5.5 que la distribution des points d'apprentissage ressemble à celui d'un échiquier. Bien que tout l'échiquier ne soit pas totalement décrit par les points d'apprentissage, le classificateur par analogie garde cette régularité de l'alternance entre "carré rouge" et "carré bleu". Cela ne veut nullement dire que ce qu'a trouvé le classificateur par analogie est meilleur que les autres classificateurs, mais nous avons juste voulu donner un aspect visuel du comportement du classificateur par analogie d'une part et un exemple de la différence que pourrait apporter l'analogie d'autre part.

5.9.3 Expérimentations sur les dissemblances analogiques entre séquences

Le matériel expérimental que nous utilisons est un ensemble de données artificielles de petite taille. Nous construisons un ensemble de séquences dans lequel nous connais-

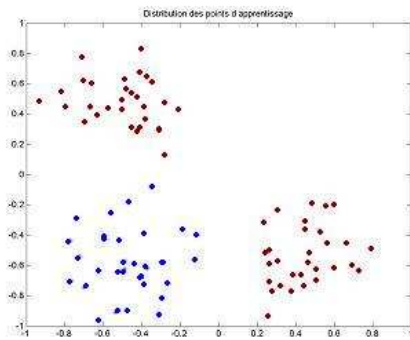


(a)

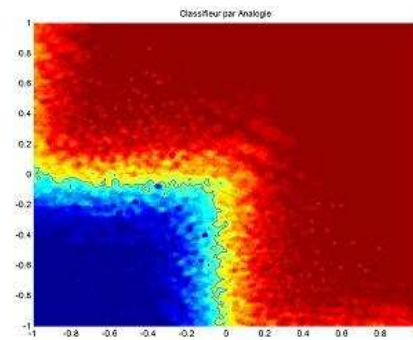


(b)

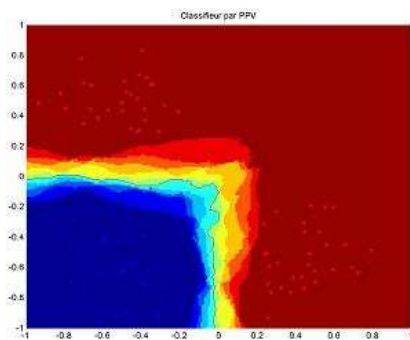
FIG. 5.3 – Répartition des triplets dans \mathbb{R}^2 selon deux dispersion initiale 5.3(a) et 5.3(b)



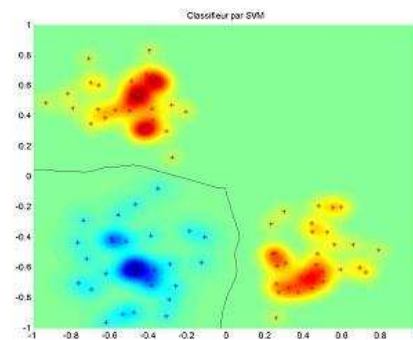
(a) Distribution des points.



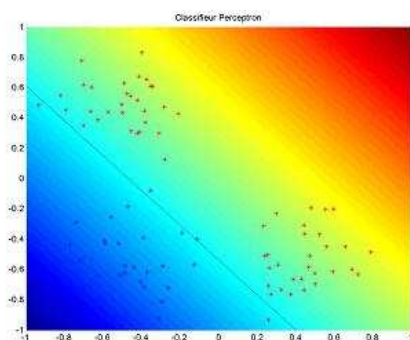
(b) Classificateur par analogie.



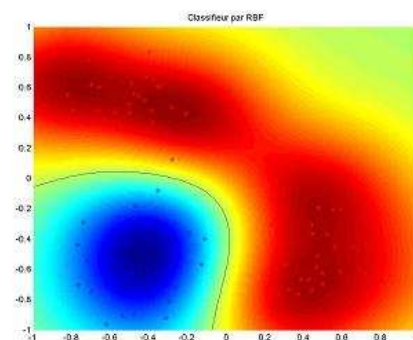
(c) PPV.



(d) SVM.

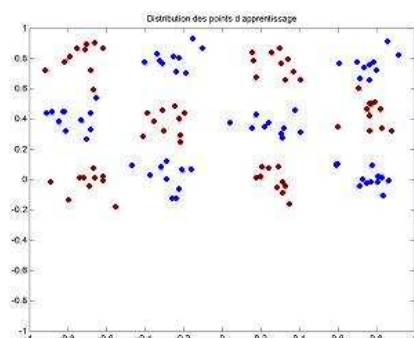


(e) Perceptron.

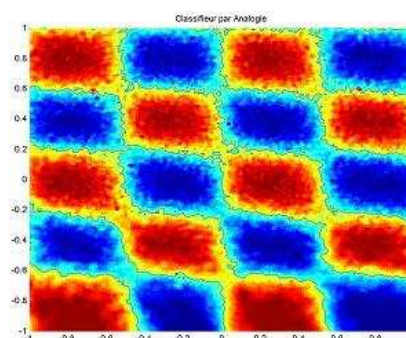


(f) RBF.

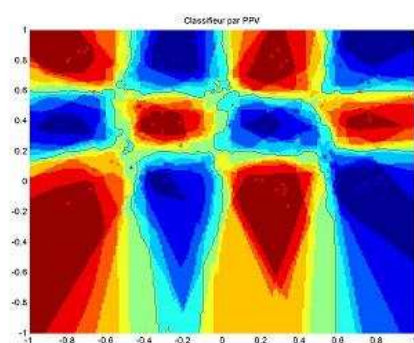
FIG. 5.4 – Surface de séparation (représentée par une ligne noire) pour une distribution des points d'apprentissage comme cela figure dans l'image 5.4(a)



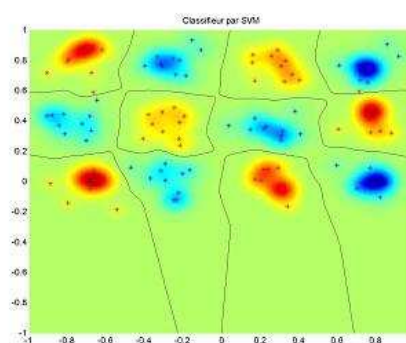
(a) Distribution des points.



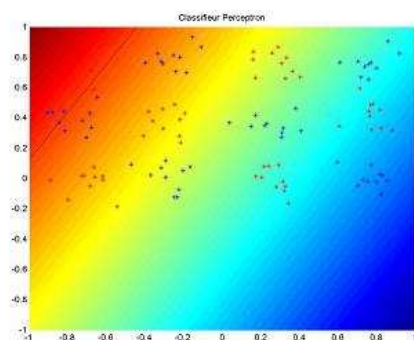
(b) Classificateur par analogie.



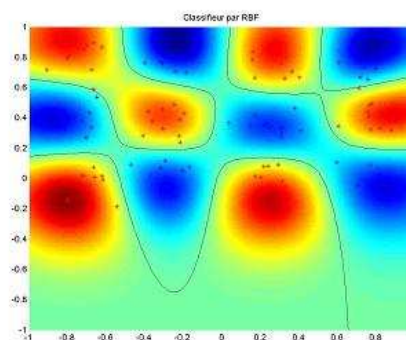
(c) PPV.



(d) SVM.



(e) Perceptron.



(f) RBF.

FIG. 5.5 – Surface de séparation (représentée par une ligne noire) pour une distribution des points d'apprentissage comme cela figure dans l'image 5.5(a)

sons les quadruplets en analogie. Nous prenons une de ces séquences et nous la bruitons. L'expérience réalisée consiste à constater si, oui ou non, le meilleur triplet, celui qui a la dissemblance la moindre avec la séquence bruitée, est encore le même. Il ne s'agit pas encore d'un véritable test de la capacité de l'apprentissage par analogie dans les séquences. Le but est pour le moment essentiellement de chercher à savoir comment la dissemblance analogique entre séquences, telle que nous l'avons définie par la distance d'édition, se comporte en présence de bruit.

5.9.3.1 Constitution de la base de données.

Les séquences. Une base de données est constituée de cent séquences de même longueur $2n$, qui sont quatre par quatre en relation d'analogie exacte (et de dissemblance analogique nulle). Pour assurer cette propriété, un ensemble de quatre séquences u , v , w et x de longueur $2n$ est composé en tirant au hasard quatre séquences X , Y , Z et T de longueur n , qui sont ensuite concaténées comme suit : $u = XZ$, $v = XT$, $w = YZ$, $x = YT$. Bien que les composants puissent être très différents, il y a une dissemblance analogique nulle entre les phrases composées. En répétant 25 fois cette opération, on dispose ainsi de 25 quadruplets en analogie exacte. On vérifie aussi qu'il n'existe pas d'autres quadruplets en analogie dans les 100 phrases. Ainsi, chaque séquence de la base de données n'est en analogie qu'avec le triplet composé avec elle.

L'alphabet et la distance δ . Nous avons choisi un alphabet Σ de $2p$ lettres, défini à partir de $p + 1$ traits (voir [MD04]). Par exemple pour $p = 3$, l'alphabet est $\Sigma = \{a, b, c, A, B, C\}$ et il est défini à partir des 4 traits binaires suivants (en colonnes) :

a	1	0	0	0
b	0	1	0	0
c	0	0	1	0
A	1	0	0	1
B	0	1	0	1
C	0	0	1	1

Les trois premiers traits définissent le caractère et le dernier définit sa "casse" (majuscule ou minuscule). Nous en avons déduit trois distances, dont voici deux :

δ_1	a	b	c	A	B	C	\sim
a	0	2	2	1	3	3	4
b	2	0	2	3	1	3	4
c	2	2	0	3	3	1	4
A	1	3	3	0	2	2	4
B	3	1	3	2	0	2	4
C	3	3	1	2	2	0	4
\sim	4	4	4	4	4	4	

δ_3	a	b	c	A	B	C	\sim
a	0	1.5	1.5	1.2	1.7	1.7	2
b	1.5	0	1.5	1.7	1.2	1.7	2
c	1.5	1.5	0	1.7	1.7	1.2	2
A	1.2	1.7	1.7	0	1.5	1.5	2
B	1.7	1.2	1.7	1.5	0	1.5	2
C	1.7	1.7	1.2	1.5	1.5	0	2
\sim	2	2	2	2	2	2	

La première est la distance de Hamming entre les lettres vues comme des vecteurs binaires de traits, nous avons donc pour tout quadruplet en analogie $a : b :: d \delta_1$ vérifie : $\delta_1(a, b) = \delta_1(c, d)$ et $\delta_1(a, c) = \delta_1(b, d)$.

Le coût d'insertion et de suppression n'est pas défini par le système de traits. Lui donner une valeur forte comme dans δ_1 permet *a priori* d'éviter des dissemblances faibles dans des schémas du type $au : av :: wa : xa$. Nous constaterons expérimentalement cette propriété en comparant δ_1 avec une distance appelée δ_2 , qui ne diffère de δ_1 que par la valeur des insertions et des suppressions, fixée à 2 au lieu de 4.

Quant à δ_3 , elle n'est pas très différente de δ_1 , mais elle est construite pour fournir des ensembles $\aleph(u, v)$ et $\aleph(w, x)$ de taille aussi petite que possible. Ceci n'est pas indifférent compte tenu du protocole expliqué au paragraphe suivant.

5.9.3.2 Protocole expérimental.

Déroulement d'une expérience. Une expérience se déroule ainsi : une séquence est enlevée de la base de données et bruitée avec un certain taux (le calcul du bruitage sera expliqué au paragraphe suivant). Ensuite, on cherche dans les 99 phrases restantes le triplet qui a la plus petite dissemblance analogique (approchée ou non) avec la phrase bruitée. Si c'est le triplet original, le score de cette expérience augmente de 1, sinon il reste inchangé. On recommence pour chaque séquence. En fin d'expérience, on dispose d'un score entre 0 et 100 qui indique la robustesse au bruit de la dissemblance analogique. On sait, par construction, que pour un bruitage de taux nul, le score de l'expérience est de 100.

Le bruitage. Les séquences sont bruitées avec un certain taux de bruit τ , entre 0 et 100. La manière de bruite est définie par un *transducteur* représenté en figure 5.6. Le bruitage peut être *uniforme* ou varier en sens inverse de la distance δ d'une insertion, d'une suppression et d'une substitution. Chaque lettre est dans ce cas transformée selon la probabilité de transition dans le transducteur. Nous avons choisi deux façons de définir ces probabilités de transitions à partir d'une distance, donc au total trois types de bruit.

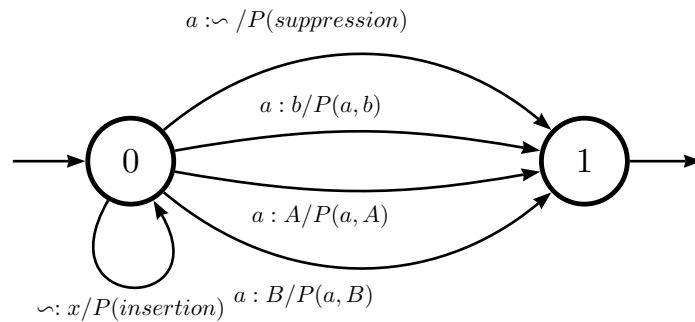


FIG. 5.6 – Automate de bruitage des séquences

Prenons un exemple avec un alphabet $\Sigma = \{a, b, A, B\}$. La première méthode de bruitage (dite en “ $1/\delta$ ”) fait dépendre la probabilité de remplacement d'une lettre a

vers une autre lettre x de l'inverse des distances entre lettres :

$$P(a, x) = \tau \cdot \frac{\frac{1}{\delta(a, x)}}{\sum_{y \in \Sigma - \{a\}} \frac{1}{\delta(a, y)} + \frac{1}{\delta_{insertion}} + \frac{1}{\delta_{suppression}}}$$

La seconde méthode (dite en “ $1 + max - \delta$ ”) définit une probabilité de transition comme dépendant de la différence des distances lettre à lettre avec le maximum de ces distances.

$$P(a, x) = \tau \cdot \frac{Sup}{Sup \cdot (|\Sigma| + 1) - \sum_{y \in \Sigma - \{a\}} \delta(a, y) - \delta_{insertion} - \delta_{suppression}}$$

où Sup est une quantité supérieure au maximum max des distances (en pratique, $Sup = 1 + max$).

Par exemple, en faisant varier le bruitage, la méthode $1/\delta$ donne, pour δ_3 :

Séquence originale	aAbdDD	adEdebbEaa	AdAcBec
$\tau = 0.1$	aAEdDD	adEdebbEaa	AdAcBec
$\tau = 0.2$	aAbdDD	adEdebbEaA	AdAcbbec
$\tau = 0.3$	AAbdDD	aadEdBbbCea	adacDBcc

5.9.3.3 Résultats et premières conclusions.

La figure 5.7(a) compare, pour la distance de traits donnée ci-dessus δ_1 ou δ_2 , pour un alphabet à 8 lettres et des séquences de longueur 10, les trois modes de bruitage pour la distance δ_1 et un bruitage uniforme. La figure 5.7(b) compare sur les mêmes données les distances δ_1 , δ_2 et δ_3 . Les figures 5.8(a) et 5.8(b) comparent les deux dissemblances analogiques et analysent l'influence de la longueur des séquences.

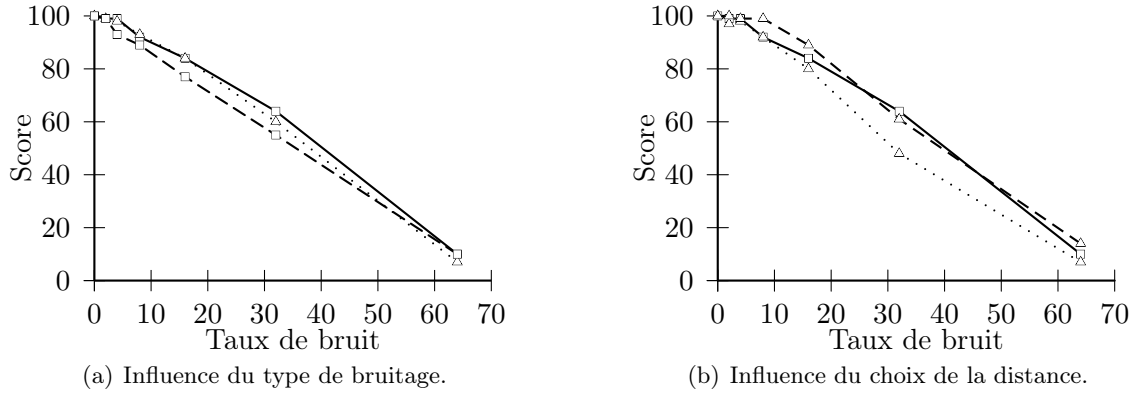


FIG. 5.7 – (a) : Comparaison de la méthode $1/\delta$ (en trait plein), de la méthode $1 + max - \delta$ (en pointillés) et d’une génération uniforme de bruit (en trait interrompu). (b) : Comparaison de δ_1 (en trait plein), de δ_2 (en pointillés) et de δ_3 (en trait interrompu). Le générateur de bruit est du type $1/\delta$ dans les trois cas. L’algorithme utilisé calcule la dissemblance analogique approchée.

En première analyse, on peut tirer les conclusions suivantes :

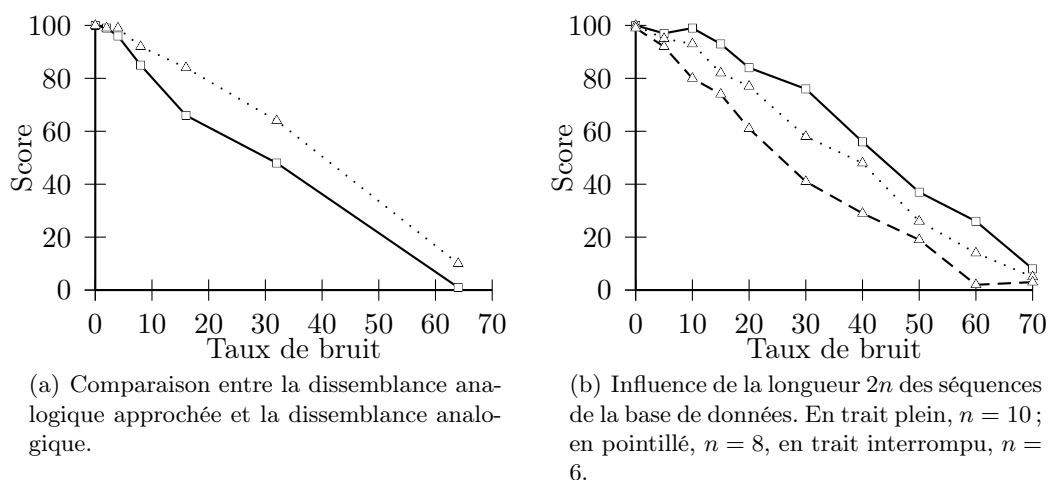


FIG. 5.8 – (a) : La distance analogique approchée est en pointillés et la dissemblance analogique en trait plein. La distance est δ_1 . Le générateur de bruit est du type $1/\delta$. (b) : Dans toutes les autres figures, la base de données est composée de phrases de longueur 8 sur un alphabet de taille 10. Ici, la taille des séquences varie, sur ce même alphabet. La base de données comporte 100 séquences, la distance est δ_3 , la méthode est la dissemblance analogique approchée et le bruit est en $1/\delta$.

- la dissemblance analogique offre une certaine résistance au bruit : le bon triplet analogique est retrouvé dans environ les deux tiers des cas, pour cent phrases de longueur 8, avec un taux de bruitage de 32% ;
- le type de bruit n'a pas d'importance ;
- un coût de suppression et d'insertion élevé est plutôt souhaitable. L'analyse fine de ce phénomène montre en effet qu'il permet d'éviter des dissemblances faibles dans des schémas du type $au : av :: wa : xa$.

Troisième partie

Applications

Chapitre 6

Apprentissage d'une règle de classification par analogie

Sommaire

6.1	Description du problème	86
6.1.1	Le codage des données nominales	87
6.1.2	Le problème à deux classes	87
6.1.3	Problème multi-classes et valeurs manquantes	88
6.1.4	Exemple	89
6.2	Pondération des attributs	90
6.2.1	Matrice de pondération analogique	90
6.2.2	Apprentissage des poids	91
6.2.3	Algorithme	92
6.3	Recherche Rapide	93
6.3.1	Solution naïve.	93
6.3.2	Transformation de la dissemblance analogique	96
6.3.3	Les kd-trees	97
6.3.4	Algorithme <i>Branch and Bound</i>	98
6.3.5	Recherche rapide du plus proche voisin : algorithme AESA	98
6.3.6	“FADANA” : Fast search of the least Dissimilar ANALogy	100
6.3.7	Résultats de FADANA et FADANA*	105
6.4	Expérimentations et Résultats	106
6.4.1	Protocole expérimental	106
6.4.2	Résultats	110
6.4.3	Variation du paramètre k	110
6.4.4	Discussion et Perspectives	110

Contenu du chapitre

Ce chapitre accorde une attention particulière à l'apport du raisonnement par proportion analogique à un problème classique d'apprentissage de règles de classification.

Il décrit la méthode d'apprentissage d'une règle de classification par analogie, dont le principe de base est semblable à celui de la méthode du plus proche voisin. La classification par analogie présentée dans ce chapitre ne traite que des objets binaires et nominaux. Nous proposons un algorithme rapide d'apprentissage et nous présentons des résultats qui montrent que cette méthode est robuste et efficace.

Motivation

Supposons qu'il existe une analogie définie sur un ensemble X , étendue à une dissemblance analogique DA qui vérifie les propriétés suivantes :

Cohérence avec l'analogie. $\forall (u, v, w, x) \in (\Sigma^*)^4 : DA(u, v, w, x) = 0 \Leftrightarrow u : v :: w : x$

Symétrie de "ce que". $\forall (u, v, w, x) \in (\Sigma^*)^4 : DA(u, v, w, x) = DA(w, x, u, v)$

Inégalité triangulaire. $\forall (u, v, w, x, z, t) \in (\Sigma^*)^6 : DA(u, v, w, x) \leq DA(u, v, z, t) + DA(z, t, w, x)$

Échange des moyens. $\forall (u, v, w, x) \in (\Sigma^*)^4 : DA(u, v, w, x) = DA(u, w, v, x)$

Non symétrie de "est à". $\exists (u, v, w, x) \in (\Sigma^*)^4 : DA(u, v, w, x) \neq DA(v, u, w, x)$

Nous nous intéressons à l'apprentissage d'une règle de classification par analogie, dont le principe de base est semblable à celui de la méthode du plus proche voisin. Il se décrit schématiquement comme suit. Soit un ensemble d'apprentissage \mathcal{S} composé d'éléments de X supervisés par des classes : $\mathcal{S} = \{(o_i, h(o_i)) \mid 1 \leq i \leq m\}$ où $h(o_i)$ la classe de l'objet o_i . Soit x un autre élément de X avec $x \notin \mathcal{S}$. Le problème est d'abord de trouver un triplet d'objets (u, v, w) dans \mathcal{S} tel que :

$$DA(u, v, w, x) = \underset{t_1, t_2, t_3 \in \mathcal{S}}{\text{Argmin}} DA(t_1, t_2, t_3, x)$$

Ensuite, si l'équation analogique $h(u) : h(v) :: h(w) : ?$ possède une solution h^* , nous affectons cette classe à x .

6.1 Description du problème

Le but de ce chapitre est de mesurer ce que peut apporter le raisonnement par proportion analogique à un problème classique d'apprentissage de règles de classification, comparé à des méthodes standard telles que les k -plus proches voisins, les réseaux de neurones, les arbres de décision, etc. Dans ce qui suit, nous ne nous sommes pas encore intéressés à classer des séquences, mais simplement à apprendre une règle de classification pour des objets symboliques, plus précisément des objets binaires et nominaux.

6.1.1 Le codage des données nominales

Rappelons que des données nominales prennent leurs valeurs dans un ensemble fini, la plupart du temps non ordonné. Classiquement, pour traiter les données nominales, deux approches sont possibles :

- la première (*one-per-value encoding*) consiste à éclater l’attribut nominal en attributs binaires. Par conséquent, un attribut nominal ayant n valeurs possibles est remplacé par n attributs binaires et sur ces n attributs, il y en a un et un seul qui prend la valeur 1 ;
- la deuxième approche consiste à garder l’attribut nominal comme étant un seul attribut. Ceci nécessite dans notre approche de définir une relation analogique et une dissemblance analogique sur le domaine de l’attribut.

La deuxième approche peut être employée quand on a une connaissance *a priori* sur la sémantique des classes : une relation d’ordre ou une mesure de distance, par exemple. Comme il n’y a pas d’information utilisable sur les classes dans les bases de données que nous avons utilisées [NHB98], nous avons choisi d’appliquer le codage *one-per-value* pour traiter les attributs nominaux. Ainsi, les données sur lesquelles nous avons travaillé peuvent être considérées comme décrites par des attributs binaires.

6.1.2 Le problème à deux classes

Soit $\mathcal{S} = \{(o_i, h(o_i)) \mid 1 \leq i \leq m\}$ l’ensemble d’apprentissage, avec $h(o_i)$ la classe de l’objet o_i . Les objets sont maintenant tous définis par des attributs binaires. Soit x un objet n’appartenant pas à \mathcal{S} . Il s’agit de trouver la classe de x , en utilisant l’ensemble d’apprentissage \mathcal{S} . Pour cela, nous définissons une règle d’apprentissage par analogie dépendant d’un paramètre k , que nous appelons la règle de classification par *les k triplets les moins dissemblants*.

Le principe est le suivant : parmi tous les triplets (a, b, c) de \mathcal{S}^3 , nous retenons le sous-ensemble de ceux qui ont la plus petite dissemblance analogique avec x . Pour certains d’entre eux, l’équation analogique sur les classes $h(a) : h(b) :: h(c) : g$ a une solution exacte. Nous ne gardons que ces triplets, puis nous choisissons la classe majoritairement représentée comme étant la classe de x . Voici plus en détails la procédure utilisée :

1. Calculer la dissemblance analogique entre x et tous les n triplets appartenant à \mathcal{S} qui donnent une solution exacte pour la classe de x ;
2. Ordonner ces n triplets par ordre croissant par rapport à leurs valeurs de DA , quand ils sont associés à x ;
3. Si le k^{eme} triplet a la valeur p , alors appeler k' le plus grand nombre tel que le triplet k'^{eme} a la même valeur p ;
4. Résoudre les k' équations analogiques sur les classes. Choisir la classe gagnante qui comptabilise le plus grand nombre de votes parmi les k' résultats.

Nous traitons sur un exemple le cas où il y a seulement deux classes ω_0 et ω_1 . Un exemple avec 3 classes sera présenté plus tard (exemple 6.1.4).

$h(a)$:	$h(b)$::	$h(c)$:	$h(x)$	resolution
ω_0	:	ω_0	::	ω_0	:	?	$h(x) = \omega_0$
ω_1	:	ω_0	::	ω_1	:	?	
ω_1	:	ω_1	::	ω_1	:	?	$h(x) = \omega_1$
ω_0	:	ω_1	::	ω_0	:	?	

TAB. 6.1 – Configurations acceptables des classes des triplets

Le point 1 veut dire qu'on ne garde que les triplets qui ont l'une des quatre¹ configurations suivantes pour leur classe donnée dans la table 6.1.

Nous ignorons par conséquent les deux configurations de classes de triplets qui ne permettent pas de classer x :

$h(a)$:	$h(b)$::	$h(c)$:	$h(x)$
ω_0	:	ω_1	::	ω_1	:	?
ω_1	:	ω_0	::	ω_0	:	?

Le point 2 est similaire à la méthode des k plus proches voisins. Cependant, puisque les données sont binaires et que les DA prennent des valeurs entières, on trouve généralement plusieurs triplets pour lesquels $DA = 0$, $DA = 1$, etc. C'est la raison pour laquelle le point 3 augmente k pour prendre en compte tous les triplets ayant la même valeur de DA . Enfin, le point 4 utilise la même technique de vote que la règle des k plus proches voisins [DHS01].

6.1.3 Problème multi-classes et valeurs manquantes

Pour le cas à plus de deux classes, on procède de la même manière que dans le problème à deux classes, en gardant seulement les triplets qui peuvent donner trivialement la classe du quatrième élément à partir des classes des trois premiers. Par exemple :

Exemples d'équations analogiques donnant une solution triviale						
ω_1	:	ω_3	::	ω_1	:	$? \Rightarrow h(x) = \omega_3$
ω_4	:	ω_4	::	ω_2	:	$? \Rightarrow h(x) = \omega_2$
ne donnant pas de solution						
ω_1	:	ω_3	::	ω_0	:	$? \Rightarrow h(x) = ?$
ω_0	:	ω_1	::	ω_3	:	$? \Rightarrow h(x) = ?$

Si l'ensemble d'apprentissage comprend des valeurs d'attributs manquantes, nous remplaçons chaque valeur d'attribut manquante par une valeur à distance égale de toutes les valeurs possibles de cet attribut nominal. Ainsi, en utilisant l'encodage *one-per-value*, un attribut nominal manquant prendra la valeur 0 dans tous les attributs binaires au lieu de prendre la valeur 1 dans une de ces valeurs. Donc pour un attribut "modèle de

¹En réalité, il y en a encore deux autres, mais ils sont équivalents à l'un des quatre.

o_1	o_2	o_3	$h(o_1)$	$h(o_2)$	$h(o_3)$	$h(x)$	DA	k
b	a	d	ω_0	ω_0	ω_1	ω_1	0	1
b	d	e	ω_0	ω_1	ω_2	\perp	1	
c	d	e	ω_1	ω_1	ω_2	ω_2	1	2
a	b	d	ω_0	ω_0	ω_1	ω_1	1	3
c	a	e	ω_1	ω_0	ω_2	\perp	2	
d	c	e	ω_1	ω_1	ω_2	ω_2	2	4
d	b	c	ω_1	ω_0	ω_1	ω_0	2	5
a	c	e	ω_0	ω_1	ω_2	\perp	2	
a	c	c	ω_0	ω_1	ω_1	\perp	3	
a	b	e	ω_0	ω_0	ω_2	ω_2	3	6
b	a	e	ω_0	ω_0	ω_2	ω_2	3	7
b	c	d	ω_0	ω_1	ω_1	\perp	3	
c	c	c	ω_1	ω_1	ω_1	ω_1	4	8
a	a	c	ω_0	ω_0	ω_1	ω_1	4	9
...

TAB. 6.2 – Exemple de classification par analogie. Les analogies qui ont \perp pour solution ne sont pas pris en compte. DA représente $DA(o_1, o_2, o_3, x)$.

voiture” par exemple, qui ne prend que les valeurs suivantes : Audi ou VW ou Mercedes, on remplace la valeur de l’attribut sur les objets de la manière suivante :

objets	modèle de voiture			
	attribut initial	attributs finaux		
o_1	Audi	1	0	0
o_2	VW	0	1	0
o_3	Mercedes	0	0	1
o_4	? = valeur manquante	0	0	0

6.1.4 Exemple

Soit $\mathcal{S} = \{(a, \omega_0), (b, \omega_0), (c, \omega_1), (d, \omega_1), (e, \omega_2)\}$ un ensemble de cinq objets étiquetés. Soit $x \notin \mathcal{S}$ l’objet à classer. Selon les axiomes de l’analogie, il y a seulement $(Card(\mathcal{S})^3 + Card(\mathcal{S})^2)/2 = 75$ équations analogiques non équivalentes parmi les $Card(\mathcal{S})^3 = 125$ équations qui peuvent être formées à partir de trois éléments de \mathcal{S} et x . Le tableau 6.2 donne un exemple de ce que peuvent être les 14 premières, après ordonnancement suivant leur dissemblance analogique. La table suivante donne la classification d’un objet x suivant la valeur de k :

k	1	2	3	4	5	6	7
k'	1	3	3	5	5	7	7
vote pour x	1	1	1	?	?	2	2

6.2 Pondération des attributs

Il est très classique en apprentissage de pondérer les attributs, en s'appuyant sur l'idée simple que s'ils n'ont pas la même importance pour la classification, il faut donner un plus grand poids aux attributs qui sont réellement discriminants. Cette constatation est en particulier importante dans des méthodes comme les k plus proches voisins, dans laquelle les valeurs des attributs servent à calculer une distance, dont la valeur dépend évidemment du facteur d'échelle attribué à chaque composante.

Même en ce qui concerne l'analogie, l'idée de sélectionner des attributs discriminants n'est pas nouvelle. Dans une application linguistique, [Tur05] propose pour trouver des reformulations de paires de mots comme par exemple (menuisier/bois, maçon/pierre), de compter la fréquence d'apparition des paires de mots dans un large corpus. Cela permet de contribuer à la décision de reformulation d'une paire de mots : il s'agit donc d'une forme de pondération.

Le problème de pondérer les attributs en classification par analogie est particulier, car un attribut peut intervenir de plusieurs façons dans la décision d'affecter une certaine classe à un objet. Reprenons l'exemple (6.1) précédent afin de mieux comprendre. Dans cet exemple, on remarque qu'il y a deux manières différentes permettant de conclure que la classe est ω_0 ou ω_1 (il y a encore une troisième configuration possible qui est équivalente à la deuxième par échange des moyens). Ainsi, la manière dont la classe a été obtenue n'est pas indifférente. C'est pourquoi, dans la suite, nous définissons un *ensemble* de poids pour chaque attribut, dont la taille dépend du nombre de classes. Tous ces ensembles de poids sont réunis dans une *matrice de pondération analogique*.

6.2.1 Matrice de pondération analogique

Définition 6.2.1 *La matrice de pondération analogique W est une matrice à trois dimensions. Dans la première dimension, on trouve les attributs, alors que dans la deuxième et la troisième dimension on trouve respectivement la classe du premier élément (classe dite de départ) et la classe du dernier élément (classe dite d'arrivée) de l'équation analogique.*

La matrice de pondération analogique est donc de dimension $(d \times C \times C)$, où d représente le nombre des attributs et C le nombre de classes. Pour l'attribut a_k de rang k , l'élément W_{kij} de la matrice indique le poids qu'on doit lui associer quand on est en présence d'une analogie dont la classe de départ est ω_i et la classe d'arrivée est ω_j .

Ainsi, pour chaque attribut a_k :

		Classe d'arrivée (décision)	
		classe ω_i	classe ω_j
Classe de départ	classe ω_i	W_{kii}	W_{kij}
	classe ω_j	W_{kji}	W_{kjj}

Puisqu'on ne prend en compte que les triplets d'apprentissage dont l'équation analogique sur les classes donne une solution, les configurations possibles ont l'une des formes suivantes :

Configurations possible	Classe de départ	Classe d'arrivée
$\omega_i : \omega_i :: \omega_j : \omega_j$	ω_i	ω_j
$\omega_i : \omega_j :: \omega_i : \omega_j$	ω_i	ω_j
$\omega_i : \omega_i :: \omega_i : \omega_i$	ω_i	ω_i

Cette remarque nous permet d'estimer la valeur W_{kij} à partir de l'ensemble d'apprentissage.

6.2.2 Apprentissage des poids

Le but de cette partie est de montrer comment remplir la matrice à trois dimensions des poids, à partir de l'ensemble d'apprentissage. Pour cela, on estime W_{kij} par le nombre de fois que l'attribut k peut être en analogie, avec la classe de départ ω_i et la classe d'arrivée ω_j .

Pour commencer, on compte le nombre de fois dans l'ensemble d'apprentissage qu'un attribut k prend la valeur 0 ou 1 sur un objet dont la classe est ω_i :

	... classe ω_i ...
$a_k = 0$... n_{0i} ...
$a_k = 1$... n_{1i} ...

Avec a_k représentant l'attribut k et n_{0i} (resp. n_{1i}) représentant le nombre d'objets dont la classe est ω_i et qui ont la valeur 0 (resp. 1) pour l'attribut k . Ainsi,

$$\sum_{p=0}^1 \sum_{i=1}^C n_{pi} = m$$

est égal au nombre d'objets dans l'ensemble d'apprentissage. Ensuite, on calcule W_{kij} en estimant la probabilité de trouver une analogie sur l'attribut k avec comme classe de départ ω_i et comme classe d'arrivée ω_j .

Le tableau suivant présente toutes les configurations possibles d'obtention d'une analogie sur l'attribut k . Ici, 0_i (resp. 1_i) représente la valeur 0 (resp. 1) de l'attribut k dont la classe de l'objet est ω_i .

1^{er}	$0_i : 0_i :: 0_j : 0_j$
$2^{ème}$	$0_i : 1_i :: 0_j : 1_j$
$3^{ème}$	$0_i : 0_i :: 1_j : 1_j$
$4^{ème}$	$1_i : 1_i :: 1_j : 1_j$
$5^{ème}$	$1_i : 0_i :: 1_j : 0_j$
$6^{ème}$	$1_i : 1_i :: 0_j : 0_j$

$\mathcal{P}_k(1^{er})$ estime la probabilité de trouver la première configuration.

$$\begin{aligned}\mathcal{P}_k(1^{er}) &= n_{0i}n_{0i}n_{0j}n_{0j}/m^4 \\ &\vdots\end{aligned}$$

À partir de

$$W_{kij} = \mathcal{P}_k(1^{er}) + \dots + \mathcal{P}_k(6^{ème})$$

On calcule

$$W_{kij} = ((n_{0i}^2 + n_{1i}^2)(n_{0j}^2 + n_{1j}^2) + 2 * n_{0i}n_{0j}n_{1i}n_{1j}) / (6 * m^4)$$

Une fois les poids déterminés, l'algorithme de décision (section 6.1.2) n'a besoin d'être modifié seulement au niveau du point 1, qui devient :

1. ayant x , trouver tous les n triplets appartenant à \mathcal{S} dont une résolution sur les classes est possible. Pour chaque triplet (a, b, c) parmi les n , ω_i est la classe de départ et ω_j est la classe d'arrivée. Calculer la dissemblance analogique entre x et le triplet en utilisant la matrice des poids :

$$DA(a, b, c, x) = \sum_{k=1}^d W_{kij} DA(a_k, b_k, c_k, x_k)$$

Sans modification du point 1, le classificateur est appelé *APC* en référence à *Analogical Proportion Classifier* ; si on modifie le point 1 par la pondération des attributs, le classificateur est appelé *WAPC* (*Weighted Analogical Proportion Classifier*).

6.2.3 Algorithme

On se propose dans ce paragraphe de présenter les algorithmes de pondération à travers un exemple jouet.

Soit $\mathcal{S} = \{(x_i, h(x_i)) \mid 1 \leq i \leq m\}$ l'ensemble d'apprentissage, avec $h(x_i)$ la classe de l'objet x_i . Chaque élément (x_i) est défini sur un ensemble d'attributs $(a_j)_{1 \leq j \leq d}$ où il vaut 1 s'il vérifie ou appartient à cet attribut et 0 sinon.

Par exemple, pour $d = 6$ et $h(x_i)$ valant “−1” ou “+1” :

x_i	aquatique	venimeux	prédateur	domestique	denté	plumes	$h(x_i)$
Ours	0	0	0	0	1	0	−1
Serpent	0	1	0	0	1	0	+1
crabe	1	0	0	0	0	0	+1
\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots

Pour un ensemble d'apprentissage contenant donc m éléments, l'algorithme 1 commence par retirer un élément x_l . Ensuite l'algorithme 2 choisit les n triplets (x_i, x_j, x_k) avec $i, j, k \neq l$ dont la dissemblance analogique avec l'élément x_l retiré est la moindre.

$$\begin{array}{l|l} (1) & -1 \quad : \quad -1 \quad :: \quad +1 \quad : \quad +1 \\ (2) & +1 \quad : \quad +1 \quad :: \quad +1 \quad : \quad +1 \end{array}$$

TAB. 6.3 – Analogies triviales sur les classes

$$\begin{array}{l} 0 \quad : \quad 1 \quad :: \quad 0 \quad : \quad 1 \\ 0 \quad : \quad 0 \quad :: \quad 1 \quad : \quad 1 \\ 0 \quad : \quad 0 \quad :: \quad 0 \quad : \quad 0 \\ \vdots \end{array}$$

TAB. 6.4 – Analogies élémentaires sur les attributs binaires

On ne prend que les triplets analogiques triviaux au sens des classes. Donc, par exemple, si la classe de l'élément x_l est “+ 1” alors les équations analogiques triviales possibles sont celles présentées dans le tableau 6.3 : Dans cet exemple, il n'y a que deux façons d'obtenir la classe “+ 1” et deux pour obtenir la classe “− 1”.

Pour chacune des n équations analogiques retenues, l'algorithme 3 regarde tout d'abord le type d'analogie triviale qui régit cette équation (voir tableau 6.3). Ensuite, l'algorithme 4 vérifie sur chaque attribut (a_i) qu'on a une analogie élémentaire (voir tableau 6.2.3).

Si l'analogie élémentaire est vérifié on donne plus de crédibilité à cet attribut pour ce type d'analogie en ajoutant une valeur positive à son poids (voir algorithme 1, la valeur positive est prise égale à 1).

À la fin du passage des n équations, on aura pour chaque attribut et pour chaque type d'analogie une valeur positive reflétant son poids, puisqu'une grande valeur indiquera que pour cet attribut et pour ce type d'analogie, quand les classes sont en analogie cet attribut l'est aussi. Et inversement, si pour un type d'analogie sur les classes, les analogies sur un attribut ne sont pas correctes, on peut conclure que cet attribut n'est pas discriminant pour ce type d'analogie, et le poids associé sera faible.

6.3 Recherche Rapide

6.3.1 Solution naïve.

Selon la technique d'apprentissage par analogie de la section 6.1.2, il faut trouver les k triplets analogiques les moins dissemblants dans \mathcal{S} . La méthode naïve revient à calculer la dissemblance analogique de chaque triplet possible de l'ensemble d'apprentissage avec l'objet x , dont la classe est inconnue et qui représentera le quatrième objet de l'analogie. Si l'on prend comme opération élémentaire le calcul d'une dissemblance analogique, cette méthode a une complexité *a priori* de m^3 , le nombre de tous les triplets qui peuvent être construits à partir de l'ensemble d'apprentissage \mathcal{S} . En se référant aux propriétés de la dissemblance analogique, ce nombre peut être divisé par 8, mais cela ne change en rien la complexité théorique de la recherche.

La situation est similaire à celle de la recherche du plus proche voisin en appren-

Algorithme 1 Algorithme de Pondération.

```

begin
 $\mathcal{E} \leftarrow \{x_i, y_i \mid i = 1, l\};$ 
 $\mathcal{A} \leftarrow \{a_i \mid i = 1, m\} = \text{attributs};$ 
pour  $i = 1 : \text{Card}(\mathcal{E})$  faire
     $\mathcal{E}' \leftarrow \mathcal{E} \setminus (x_i, y_i);$ 
     $\mathcal{S} \leftarrow \text{N\_Analogies}(\mathcal{E}', (x_i, y_i));$ 
    pour  $j = 1 : \text{Card}(\mathcal{S})$  faire
         $\text{type} \leftarrow \text{TypeAnalogie}(s_j \in \mathcal{S});$ 
        pour  $k = 1 : \text{Card}(\mathcal{A})$  faire
            si  $\text{AnalogieElémentaire}(s_j, k) = \text{vrai}$  alors
                 $\text{Poids}(k, \text{type}) \leftarrow \text{Poids}(k, \text{type}) + 1;$ 
            fin si
        fin pour
    fin pour
fin pour
end

```

Algorithme 2 N_Analogies : Renvoie les N quadruplets analogiques les moins disséminés.

```

N_Analogies( $\mathcal{E}', (x_i, y_i)$ )
begin
 $\mathcal{S} = (s_j)_{1 \leq j \leq N} = ((x_{1j}, y_{1j}), (x_{2j}, y_{2j}), (x_{3j}, y_{3j}), (x_{4j}, y_{4j}))_{1 \leq j \leq N}$ 
pour  $l, m, n = 1 : \text{Card}(\mathcal{E}')$  faire
    si  $DA(x_l, x_m, x_n, x_i) < DA(s_j), j = 1 : N$  alors
         $\mathcal{S} \leftarrow \mathcal{S} \setminus \{s_k\}, oDA(s_k) \geq DA(s_j), j = 1 : N;$ 
         $\mathcal{S} \leftarrow \mathcal{S} \cup \{((x_l, y_l), (x_m, y_m), (x_n, y_n), (x_i, y_i))\};$ 
    fin si
fin pour
return  $\mathcal{S}$ 
end

```

Algorithme 3 TypeAnalogie : Renvoie un nombre indiquant le type d'analogie qui relie le quadruplet (voir 6.3).

```

TypeAnalogie( $s_j \in \mathcal{S}$ )
begin
 $s_j = ((x_{1j}, y_{1j}), (x_{2j}, y_{2j}), (x_{3j}, y_{3j}), (x_{4j}, y_{4j}))$ 
si  $y_{1j} = y_{2j} = y_{3j} = y_{4j}$  alors
     $type \leftarrow 1$ ;
sinon si  $y_{1j} = y_{2j}$  ET  $y_{3j} = y_{4j}$  alors
     $type \leftarrow 2$ ;
sinon si  $y_{1j} = y_{3j}$  ET  $y_{2j} = y_{4j}$  alors
     $type \leftarrow 3$ ;
fin si
return  $type$ 
end

```

Algorithme 4 AnalogieElémentaire : retourne vrai si sur l'attribut a_k du quadruplet s_j on a une analogie élémentaire.

```

AnalogieElémentaire( $s_j, k$ )
begin
 $s_j = ((x_{1j}, y_{1j}), (x_{2j}, y_{2j}), (x_{3j}, y_{3j}), (x_{4j}, y_{4j}))$ 
 $x_a = (x_a^k)_{1 \leq k \leq Card(\mathcal{A})}$ 
si  $DA(x_{1j}^k, x_{2j}^k, x_{3j}^k, x_{4j}^k) = 0$  alors
     $elementaire \leftarrow vrai$ 
sinon
     $elementaire \leftarrow faux$ 
fin si
return  $elementaire$ 
end

```

tissage artificiel, pour lequel l'algorithme naïf fait m calculs de distance. Plusieurs méthodes ont été proposées afin de réduire cette complexité. Nous allons dans les sections suivantes nous focaliser sur l'extension de l'algorithme AESA et sur ses variantes aux proportions analogiques, puisque ces algorithmes reposent sur les propriétés des distances, que nous avons étendues à la dissemblance analogique.

Premièrement, nous rappelons le principe de AESA et ses améliorations (LAESA, TLAESA). Deuxièmement, nous proposons l'algorithme FADANA. Auparavant, nous expliquons comment transformer une dissemblance analogique en distance, en changeant de représentation.

6.3.2 Transformation de la dissemblance analogique

On peut voir la dissemblance analogique d'un autre point de vue : au lieu de comparer quatre objets en même temps, on peut diviser les quatre objets en deux groupes de deux et se ramener à un calcul de distance entre deux couples de points. Par exemple, dans le cas où les objets A , B , C et D sont des éléments de \mathbb{R}^d (voir la section 3.5.2) ou d'un autre espace où la propriété de la transitivité est vérifiée².

$$DA(A, B, C, D) = \sum_{l=1}^d \left\| \underbrace{(A_l - B_l)}_{P_l} - \underbrace{(C_l - D_l)}_{Q_l} \right\|$$

d'où

$$DA(A, B, C, D) = \sum_{l=1}^d \|P_l - Q_l\|$$

L'intérêt de transformer ainsi la formulation de la dissemblance analogique est évidemment de pouvoir utiliser des variantes des algorithmes bien rodés de recherche rapide des plus proches voisins.

Ainsi, partant d'un ensemble d'apprentissage $\mathcal{S} = \{(s_i, h(s_i)) \mid 1 \leq i \leq m\}$, on construit l'ensemble $\mathcal{S}^* = \{s_{ij}^* \mid 1 \leq i \leq m, 1 \leq j \leq m\}$, où $s_{ij}^* = s_i - s_j$. On associe une étiquette à s_{ij}^* qui est le couple des classes $(h(s_i), h(s_j))$.

Pour finir de transformer le problème, au lieu d'avoir un seul objet inconnu x , on aura l'ensemble des m objets $\mathcal{X} = \{s_i - x \mid 1 \leq i \leq m\}$. Ensuite, en utilisant les algorithmes de recherche rapide du plus proche voisin on calcule le plus proche voisin de chacun des éléments de cet ensemble à ceux de \mathcal{S}^* , pour enfin ne retenir que le couple qui a la distance la plus faible. De la sorte, le problème initial, qui recherchait le triplet d'objets le moins dissemblant, est devenu la recherche des deux objets les plus proches dans deux ensembles, figure 6.3.2.

En supposant que la recherche rapide du plus proche voisin d'un objet x dans un ensemble de taille N soit de complexité moyenne $\mathcal{O}(f(N))$, avec $f(N) < N$, la reformulation précédente mène maintenant à une complexité moyenne en $\mathcal{O}(m \times f(m^2))$ pour la recherche du triplet le moins dissemblant.

Nous allons examiner dans les paragraphes suivants différentes méthodes de recherche rapide des plus proches voisins, en particulier celles de la famille AESA.

²Ce qui n'est pas toujours le cas, notamment pour les séquences, voir le paragraphe 5.6.3.1.

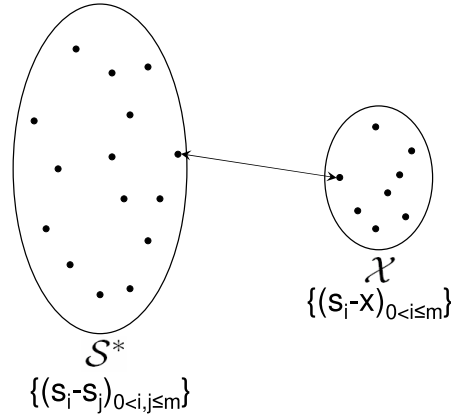


FIG. 6.1 – Recherche des deux points les plus proches de deux ensembles

6.3.3 Les *kd-trees*

Un *kd-tree* [FBF77] (*k – dimensional tree*) est une structure de données servant à l'enregistrement, au rangement et à la recherche de m données dans un espace de dimension k . Il s'agit d'un arbre binaire où chaque nœud contient un sous-ensemble d'exemples, en particulier la racine contient tous les points. Seuls les nœuds terminaux renferment des données. Chaque nœud non terminal résume ses fils par :

- une valeur qui sépare les exemples du fils gauche de ceux du fils droit ;
- la dimension sur laquelle se fait la séparation. Cette dimension est choisie de façon à avoir la plus grande dispersion des points dans ce nœud.

Chaque nœud terminal contient un ensemble d'exemples inférieur à une valeur arbitraire choisie ultérieurement.

Donc le *kd-tree* procède au rangement des points en faisant de la séparation de ces derniers par des plans qui sont toujours perpendiculaires à l'une des dimensions. La recherche du plus proche voisin se fait ensuite en parcourant l'arbre de haut en bas en élaguant suivant deux procédés.

1. Éliminer toutes les branches dont les rayons sur chaque dimension ne croisent pas les frontières de la meilleure solution obtenue à cet instant.
2. Ignorer toutes les branches qui à une certaine profondeur dans l'arbre sont d'ores et déjà à une distance des frontières supérieure à la distance minimale trouvée à ce stade.

Complexité

- La complexité temporelle pour la création du *kd-tree* est égale à $O(m^2 \log m)$.
- La complexité temporelle lors de la recherche dans le *kd-tree* est égale à $O(m \log m)$

6.3.4 Algorithme *Branch and Bound*

Fukunaga et Narendra [FN75] ont proposé une méthode fondée sur la technique générale du *branch and bound*³. Leur algorithme de recherche rapide des k plus proches voisins est basé sur la division de l'ensemble d'apprentissage initial en plusieurs sous-ensembles suivant une structure d'arbre ; ensuite pour la deuxième phase de recherche, ils utilisent une méthode d'élagage *branch and bound*.

Durant la première phase, ils décomposent récursivement l'ensemble d'apprentissage en l sous-ensembles (l est un entier choisi *a priori* de façon arbitraire). La décomposition de ces sous-ensembles se fait de façon à favoriser un meilleur élagage dans la deuxième phase et peut donc ne pas avoir d'interprétation simple et directe au sens des données (une des méthodes utilisées est le clustering par k -means, avec $k = l$). Chaque sous-ensemble est résumé par sa moyenne ou son centre dans l'ensemble d'apprentissage et des informations utiles à l'élagage. Sur chaque nœud on trouve :

- M_p un représentant (point moyen) ;
- les distances r_{min} et r_{max} qui séparent ce représentant du plus proche et du plus lointain point du sous-ensemble ;
- l'ensemble des indices des points S_p du sous-ensemble et leur nombre N_p .

La deuxième phase consiste à parcourir l'arbre avec un élagage *branch and bound* pour éviter le calcul de distance à des points dont la distance au nouveau point ne peut être que supérieure à la distance minimale trouvée à ce stade. L'élimination de ces points suit quatre règles dont deux ont été introduites par Fukunaga et Narendra (point 1 et 3 des inégalités ci après) et les deux autres par Kamgar-Parsi et Kanal [KPK85] (point 2 et 4 des inégalités ci après). On note :

- B la distance au plus proche voisin actuel ;
- $d(.,.)$ l'opérateur de distance ;
- x le point dont on cherche le plus proche voisin ;
- (x_i) les points appartenant à S_p .

Inégalités d'élagage :

1. $B + r_{max} < d(x, M_p) \Rightarrow$ on élimine l'ensemble des points S_p ;
2. $B + d(x_i, M_p) < d(x, M_p) \Rightarrow$ on élimine le point x_i ;
3. $B + d(x, M_p) < r_{min} \Rightarrow$ on élimine l'ensemble des points S_p ;
4. $B + d(x, M_p) < d(x_i, M_p) \Rightarrow$ on élimine le point x_i .

On remarque que les inégalités 1 et 3, contrairement aux inégalités 2 et 4, ont un élagage plus important mais une condition d'élagage plus forte.

6.3.5 Recherche rapide du plus proche voisin : algorithme AESA

6.3.5.1 Principe

AESA ([MOV94]) est une technique de recherche efficace par élagage, selon une certaine distance d , du plus proche voisin d'un objet $y \in X$ parmi un ensemble \mathcal{S}

³L'algorithme précédent utilisant les kd -trees pourrait aussi être vu comme une application de cette technique.

d'objets dans X . AESA utilise cette propriété pour réduire la complexité de calcul, *a priori* égale à $m = \text{Card}(\mathcal{S})$ calculs de distances par la méthode naïve, de la façon suivante :

1. **Initialisation.** Initialiser U par \mathcal{S} ;
2. **Sélection.** Choisir $p_0 \in U$;
3. **Calcul de distance.** Calcul de la distance $d(p_0, y)$ entre p_0 et y ;
4. **Actualisation du plus proche voisin.** Si p_0 est le plus proche voisin rencontré jusqu'à maintenant, rafraîchir le résultat ;
5. **Élimination.** Utiliser p_0 afin d'éliminer des éléments de U . ;
6. **Itération.** Répéter les étapes 2 jusqu'à 5 tant que U n'est pas vide.

L'efficacité de cet algorithme dépend des phases 2 et 5. Cette dernière utilise les propriétés de distance de la manière suivante.

6.3.5.2 Élimination

Soit pp le plus proche voisin de y trouvé jusqu'à maintenant dans \mathcal{S} , avec $d(y, pp) = \delta$.

Parmi les éléments de U , tous les suivants n'ont aucune chance d'être plus proche de y que l'est pp et peuvent donc être éliminé de U :

- ceux qui sont localisés à l'intérieur de l'hypersphère centrée en p_0 et de rayon $d(y, p_0) - \delta$, comme p_1 dans la Figure 6.2 ;
- ceux qui sont localisés à l'extérieur de l'hypersphère centrée en p_0 et de rayon $d(y, p_0) + \delta$, comme p_2 dans la Figure 6.2.

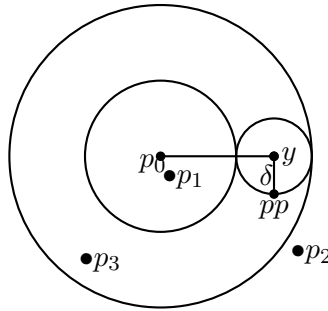


FIG. 6.2 – La géométrie de “éliminer” dans l'algorithme AESA.

6.3.5.3 Sélection

Maintenant, le problème se résume à choisir un élément p_0 aussi efficace que possible, i.e. celui qui est très proche de y et donc qui nous permettra de faire un élagage très efficace et de garder que peu de candidats pour la suite. Comme on ne peut pas le connaître à ce stade, on va choisir un élément de U avec une heuristique. On choisit le point le plus proche de l'intersection entre toutes les hypersphères centrées en un

élément $u \in Q$ (de rayon $d(x, u)$). Q est le sous-ensemble de \mathcal{S} composé de tous les éléments pour lesquels les valeurs $d(x, u)$ ont déjà été calculées.

$$p_0 = \underset{q \in U}{\operatorname{Argmin}} \quad \underset{u \in Q}{\operatorname{Max}} |d(q, u) - d(x, u)|$$

6.3.5.4 Réduction du pré-calcul : *LAESA*

Principe. La procédure de *sélection* utilise des termes tels que $d(p_0, p_1)$, $d(p_0, p_2)$, \dots , qui sont des distances entre éléments de \mathcal{S} . Si nous voulons que la phase d'*élimination* soit efficace, ces valeurs doivent être calculées et stockées dans la phase de pré-calcul. Ceci nous donne une complexité temporelle et spatiale en $\mathcal{O}(m^2)$.

Une version améliorée de *AESA* réduit cette complexité à $\mathcal{O}(m)$, mais en contrepartie l'efficacité de la phase d'*élimination* est réduite. Cette version est appelée *LAESA* ([MOV94]). Elle consiste à choisir un sous-ensemble \mathcal{T} de n éléments de \mathcal{S} , appelés *prototypes bases*, et à pré-calculer seulement les $n \times m$ distances entre les éléments de \mathcal{T} et ceux de \mathcal{S} . Ensuite, dans la version la plus simple, l'algorithme est décrit comme l'algorithme 5, dans cette section, avec :

- $G(p)$ est la limite inférieure de $d(p, y)$: c'est la valeur minimale que peut prendre $d(p, y)$ d'après les données qu'on a vu jusqu'à maintenant ;
- p^* est le plus proche voisin de y , à une distance $d^* = d(p^*, y)$.

Choisir \mathcal{T} . Les auteurs de l'algorithme *LAESA* recommandent de composer \mathcal{T} avec des prototypes bases les plus séparés par une distance d , et suggèrent une procédure pour les choisir parmi \mathcal{S} . Ils ont montré expérimentalement qu'il y a une taille optimale de \mathcal{T} , qui dépend de la dimension de l'espace euclidien dans lequel ils travaillent, et non de la taille de \mathcal{S} , une fois que la dimension est fixée. Ils montrent que le nombre de distance calculé avec *LAESA* est autour de 1,5 fois celui de *AESA*, alors que le temps et l'espace alloué en pré-calcul sont linéaires avec m . Pour donner un ordre de grandeur, dans \mathbb{R}^8 , avec $m = 1024$ vecteurs aléatoirement choisis dans une hypersphère de rayon 1 pour composer \mathcal{S} , la taille optimale de \mathcal{T} est autour de 25 et le nombre de distance euclidienne calculé est en moyenne égal à 50 avec *LAESA*.

6.3.6 “FADANA” : FAsT search of the least DIssimilar ANalogy

Cette section décrit un algorithme rapide pour trouver, à partir d'un ensemble d'objets \mathcal{S} de cardinal m et un objet y , les trois objets (z^*, t^*, x^*) dans \mathcal{S} tel que la dissemblance analogique $DA(z^*, t^*, x^*, y)$ est minimale. Cet algorithme est basé sur la technique de *AESA*, qui peut être étendue aux proportions analogiques grâce aux propriétés 6.3.2. Ainsi, puisqu'une dissemblance analogique $DA(z, t, x, y)$ peut être vue, sous certaines conditions, comme une distance entre les deux couples (z, t) et (x, y) , nous allons principalement travailler avec des couples d'objets. Nous utilisons dans la suite de façon équivalente le terme “distance entre deux couples (u, v) et (w, x) ” ou “dissemblance entre quatre éléments u, v, w et x ” pour décrire $DA(u, v, w, x)$.

Algorithme 5 Algorithme LAESA.

```

begin
   $U \leftarrow \mathcal{S}; V \leftarrow \mathcal{T};$ 
   $p^* \leftarrow \text{anything}; d^* \leftarrow +\infty$ 
  pour chaque élément  $p$  de  $U$  faire
     $G(p) \leftarrow +\infty$ 
  fin pour
  tant que  $U \neq \emptyset$  faire
    si  $V \neq \emptyset$  alors
      choisir l'élément  $s$  dans  $V$  avec le plus petit  $G(p)$ 
    sinon
      choisir l'élément  $s$  dans  $U$ 
    fin si
    calculer  $d(s, y)$ 
    si  $d(s, y) < d^*$  alors
       $d(s, y) \leftarrow d^*; p^* \leftarrow s$ 
    fin si
    si  $V \neq \emptyset$  alors
      pour chaque élément dans  $V$  faire
         $G(p) \leftarrow \text{Min} \begin{cases} G(p) \\ d(p, s) - d(y, s) \end{cases}$ 
      fin pour
    fin si
    pour chaque élément de  $U$  faire
      si  $G(p) \geq d^*$  alors
         $U \leftarrow U - \{p\}$ 
      fin si
      si  $(p \in V)$  and  $(G(p) \geq d^*)$  alors
         $V \leftarrow V - \{p\}$ 
      fin si
    fin pour
  fin tant que
end

```

6.3.6.1 Calcul préliminaire

Dans cette partie, qui est faite hors ligne et une seule fois, nous devons calculer la dissemblance analogique entre chaque 4-uplet de l'ensemble d'apprentissage. Cette étape a une complexité spatiale et temporelle de $\mathcal{O}(m^4)$, avec m la taille de \mathcal{S} . Nous allons revenir sur ce point dans la section 6.3.6.7 pour réduire cette complexité en passant à FADANA*.

6.3.6.2 Principe de l'algorithme

L'opération de base est de composer un couple d'objets en ajoutant à y un $x_i \in \mathcal{S}$ avec $i = 1, m$. Le but est alors de trouver un couple d'objets dans \mathcal{S}^2 ayant la plus petite distance de (x_i, y) , ensuite de changer x_i en x_{i+1} . Boucler m fois sur une sélection et une élimination "AESAs-like" nous assure de trouver le triplet dans \mathcal{S}^3 qui a la plus petite dissemblance analogique avec y .

6.3.6.3 Notations

Les conventions de notation dans la description de l'algorithme sont les suivantes :

- \mathcal{C} l'ensemble des couples (u, v) dont la distance à (x_i, y) a déjà été calculée ;
- $\delta = \underset{(z,t) \in \mathcal{U}}{\text{ArgMin}} (DA(z, t, x_i, y))$;
- $\delta_i = \underset{(z,t) \in \mathcal{U}, 1 \leq j \leq i}{\text{ArgMin}} (DA(z, t, x_i, y))$;
- $\text{Dist} = \{DA(z, t, x_i, y), (z, t) \in \mathcal{C}\}$;
- $\text{Dist}(j)$ le $j^{\text{ème}}$ élément de Dist ;
- $\text{Quad}_{\mathcal{U}} = \{(z, t, x_i, y), (z, t) \in \mathcal{C}\}$;
- $\text{Quad}_{\mathcal{U}}(j)$ le $j^{\text{ème}}$ élément de $\text{Quad}_{\mathcal{U}}$.

L'algorithme est construit en trois phases qui sont décrites dans la suite.

6.3.6.4 Initialisation

Chaque fois que x_i change (quand i croit de 1), l'ensemble \mathcal{U} est rempli à nouveau avec tout les couples d'objets possibles $\in \mathcal{S}^2$. L'ensemble \mathcal{C} et Dist qui contiennent respectivement les couples et les distances à (x_i, y) qui ont été mesurées durant une boucle, sont réinitialisés à l'ensemble vide. Le minimum local Min , contenant la dissemblance analogique minimale dans une boucle, est réinitialisé à l'infini. $k = \text{Card}(\mathcal{C})$, représentant le nombre de couples dont la distance a été calculée à (x_i, y) dans la boucle courante, est réinitialisé à zéro.

6.3.6.5 Sélection

Le but de cette fonction est d'extraire à partir de l'ensemble \mathcal{U} le couple (zz, tt) qu'on pense être le plus proche de (x_i, y) , en utilisant le critère suivant :

$$(zz, tt) = \underset{(u,v) \in \mathcal{U}}{\text{Argmin}} \quad \underset{(z,t) \in \mathcal{C}}{\text{Max}} \quad | DA(u, v, z, t) - DA(z, t, x_i, y) |$$

Algorithm 6 Algorithmme FADANA : initialisation.

```

begin
 $\mathcal{U} \leftarrow \{(x_i, x_j), i = 1, m \text{ and } j = 1, m\};$ 
 $\mathcal{C} \leftarrow \emptyset;$ 
 $Min \leftarrow +\infty;$ 
 $Dist \leftarrow \emptyset;$ 
 $k \leftarrow 0;$ 
end

```

Algorithm 7 Algorithmme FADANA : sélection du couple le plus prometteur.

```

sélection( $\mathcal{U}, \mathcal{C}, (x_i, y), Dist$ )
begin
 $s \leftarrow 0$ 
pour  $i = 1, Card(\mathcal{U})$  faire
  si  $s \leq \sum_{j \in \mathcal{C}} |DA(z_j, t_j, u_i, v_i) - Dist(j)|$  alors
     $s \leftarrow \sum_{j \in \mathcal{C}} |DA(z_j, t_j, u_i, v_i) - Dist(j)|;$ 
     $Argmin \leftarrow i;$ 
  fin si
fin pour
Return ( $u_{Argmin}, v_{Argmin}$ );
end

```

6.3.6.6 Élimination

Cette phase consiste en l'élimination de tous les couples $(u, v) \in \mathcal{U}$ dont la dissemblance analogique avec (x_i, y) ne peut pas être plus petite que celle trouvée jusqu'à maintenant. L'utilisation de ces deux critères, grâce à la propriété de l'inégalité triangulaire, permet de trouver ces couples (u, v) tel que :

$$DA(u, v, z, t) \leq DA(z, t, y, x_i) - \delta \Rightarrow DA(u, v, x_i, y) \geq \delta$$

et

$$DA(u, v, z, t) \geq DA(z, t, y, x_i) + \delta \Rightarrow DA(u, v, x_i, y) \geq \delta$$

avec $\delta = DA(z^*, t^*, x^*, y)$ représentant la dissemblance analogique minimale trouvée à ce stade. Notons que δ est actualisé durant tout l'algorithme et n'est jamais réinitialisé quand i croît.

6.3.6.7 Sélection des prototypes bases dans FADANA

Jusqu'ici *FADANA* dérive directement de *AESA* et a le désavantage d'exiger un pré-calcul en temps et en espace en $\mathcal{O}(m^4)$, ce qui en pratique n'est plus utilisable à partir de $m > 100$. Afin d'y remédier, nous avons aussi développé un algorithme *LAESA*-like, dans lequel les pré-calculs se font en $N.m^2$, avec N un certain nombre de *couples* d'objets. Le principe est similaire à celui de *LAESA*. Les N couples de *prototypes bases*

Algorithme 8 Algorithme FADANA : élimination des couples inutiles.

```

eliminate( $\mathcal{U}, \mathcal{C}, (x_i, y), \delta, k$ )
( $z_k, t_k$ ) is the  $k^{th}$  element of  $Quad_{\mathcal{U}}$ 
begin
  pour  $i = 1, Card(\mathcal{U})$  faire
    si  $DA(z_k, t_k, u_i, v_i) \leq Dist(k) + \delta$  alors
       $\mathcal{U} \leftarrow \mathcal{U} - \{(u_i, v_i)\}$ ;
       $\mathcal{C} \leftarrow \mathcal{C} \cup \{(u_i, v_i)\}$ ;
    sinonsi  $DA(z_k, t_k, u_i, v_i) \geq Dist(k) - \delta$  alors
       $\mathcal{U} \leftarrow \mathcal{U} - \{(u_i, v_i)\}$ ;
       $\mathcal{C} \leftarrow \mathcal{C} \cup \{(u_i, v_i)\}$ ;
    fin si
  fin pour
end

```

Algorithme 9 Algorithme FADANA : procédure principale.

```

begin
 $\mathcal{S} \leftarrow \{x_i, i = 1, m\}$ ;
 $DA^* \leftarrow +\infty$ ;
pour  $i = Card(\mathcal{S})$  faire
  Initialiser;
  tant que  $\mathcal{U} \neq \emptyset$  faire
    ( $z, t$ )  $\leftarrow$  sélection( $\mathcal{U}, \mathcal{C}, (x_i, y), Dist$ );
     $Dist(k) \leftarrow DA(z, t, x_i, y)$ ;
     $k = k + 1$ ;
     $\mathcal{U} \leftarrow \mathcal{U} - \{(z, t)\}$ ;
     $\mathcal{C} \leftarrow \mathcal{C} \cup \{(z, t)\}$ ;
    si  $Dist(k) \geq Min$  alors
      éliminer( $\mathcal{U}, \mathcal{C}, (x_i, y), \delta, k$ )
    sinon
       $Min \leftarrow Dist(k)$ ;
      si  $Dist(k) < DA^*$  alors
         $DA^* \leftarrow Dist(k)$ ;
         $z^* \leftarrow z, t^* \leftarrow t, x^* \leftarrow x_i$ ;
      fin si
    pour  $k = 1, Card(\mathcal{C})$  faire
      éliminer( $\mathcal{U}, \mathcal{C}, (x_i, y), \delta, k$ )
    fin pour
  fin si
fin tant que
fin pour
le meilleur triplet dans  $\mathcal{S}^3$  est  $(z^*, t^*, x^*)$ ;
La plus petite dissemblance analogique est  $DA^* = DA(z^*, t^*, x^*, y)$ ;
end

```

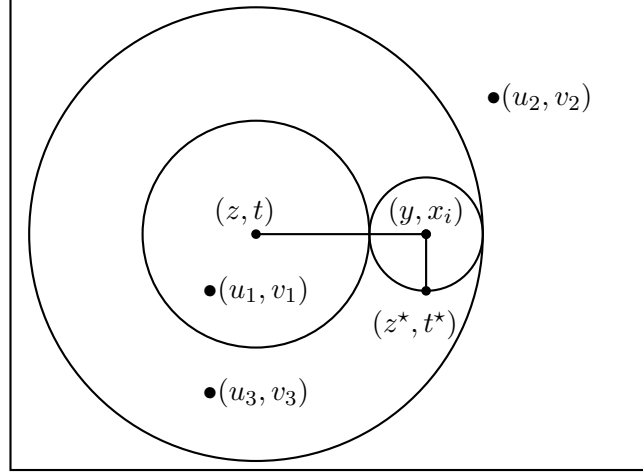


FIG. 6.3 – Processus d'élimination dans FADANA.

sont sélectionnés parmi m^2 possibilités par un processus bien défini : le premier est choisi de façon aléatoire, le deuxième est le couple le plus loin du premier, le troisième est le couple le plus loin des deux premiers Le plus loin est exprimé en termes de dissemblance analogique entre quatre éléments :

$$\delta((x, y), (z, t)) = DA(z, t, x, y)$$

6.3.7 Résultats de FADANA et FADANA*

Afin de prouver expérimentalement l'apport de la méthode FADANA, nous avons appliqué cette dernière sur quelques bases de données en relevant à chaque fois le nombre de DA calculées en ligne pour différents nombres de prototypes bases (*bp*) (voir le tableau 6.5). Ce nombre de prototypes bases est lié au nombre d'exemples dans l'ensemble d'apprentissage, c'est pourquoi, dans le tableau 6.5, les prototypes bases sont exprimés en pourcentage. Bien évidemment, si l'ensemble d'apprentissage contient m éléments, le nombre de triplets qu'on peut construire est m^3 , ce qui explique le fait que le pourcentage de prototypes bases par rapport à l'ensemble d'apprentissage dépasse 100%. On notera que lorsque le nombre de prototypes bases est égal à zéro, cela revient à appliquer la méthode naïve. Elle sera notre valeur de référence pour montrer l'éventuel apport de FADANA. On a divisé chaque base de données en un ensemble d'apprentissage et un ensemble de test ; les valeurs des nombres de calculs de la DA en ligne sont donc une moyenne sur l'ensemble de test. Dans le tableau 6.5, on présente :

- temps en pré-calcul : c'est le temps (en secondes) mis par la méthode FADANA dans la partie hors ligne et qui n'est donc faite qu'une seule fois ;
- temps en ligne : c'est le temps (en secondes) mis par FADANA en ligne pour trouver le triplet le moins dissemblant ;
- nb DA calculée en ligne : c'est le nombre de dissemblances analogiques calculées par la méthode naïve et donc en ne retenant aucun prototype base ;

- % DA calculée : c'est le pourcentage de dissemblances analogiques calculées par rapport à celle calculée par la méthode naïve.

Le temps calculé en secondes est pris sur une machine ayant les caractéristiques suivantes :

- système d'exploitation : Windows 2000 SP4 ;
- processeur : Intel Pentium M 2.00 GHz, 1 Go RAM ;
- compilateur Microsoft Visual C++ .NET release.

La figure 6.4 montre la courbe suivie par la méthode FADANA en termes de taux de calculs de DA en ligne (a) et en termes de temps de calcul en ligne (b). Les deux courbes sont en échelle logarithmique en abscisse et en ordonnée. On note sur ces bases de données, qui suivent la même allure, que le nombre de prototypes bases optimal est entre 10 et 20% de l'ensemble d'apprentissage.

FADANA vs kd-tree vs Branch et Bound En comparant la rapidité des trois algorithmes de recherche rapide FADANA 6.3.6, *kd-tree* 6.3.3 et Branch and Bound 6.3.4, nous avons remarqué que selon les données sur lesquelles on travaille l'ordre de performance des algorithmes changeait. Ainsi, pour des données binaires et nominales le *kd-tree* est le meilleur alors que dans les séquences c'est plutôt FADANA qui donne les meilleurs résultats. La raison est que FADANA calcule le moins de dissemblances analogiques possibles au dépend du temps pris pour le faire. Donc si le calcul de la dissemblance analogique est rapide à faire comme dans les données binaires ou nominales alors FADANA est plus lent que le *kd-tree* bien qu'il calcule moins de *DA*. Il passe en effet plus de temps à éliminer un candidat (point) qu'à le calculer. En revanche, quand le temps de calcul de la *DA* est très important, c'est le cas des séquences, il vaut mieux passer du temps à éliminer les candidats car ce temps est négligeable devant celui du calcul de la *DA*. Donc pour savoir quel est l'algorithme à utiliser, il faut comparer le temps de calcul d'une *DA* avec celui de l'élagage d'un candidat.

6.4 Expérimentations et Résultats

6.4.1 Protocole expérimental

Nous avons appliqué le classifieur pondéré par proportion analogique (*WAPC*) sur huit bases de données classiques, à attributs binaires ou nominaux, issues de l'*UCI Repository* [NHB98].

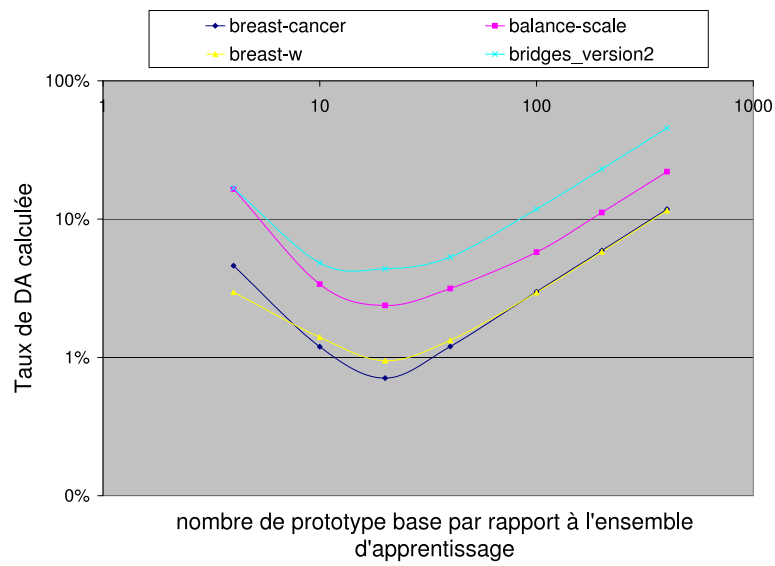
- **MONK 1,2 et 3** (*MO.1*, *MO.2* et *MO.3*). Le domaine de ces données est le même et concerne des moines, décrits par six attributs nominaux ($a_1 \dots a_6$). Les trois problèmes sont régis par des équations logiques :

1. *MO.1* : $(a_1 = a_2) \text{ ou } (a_5 = 1)$;
2. *MO.2* : exactement deux des six attributs ont la valeur 1 ;
3. *MO.3* : $((a_4 = 1) \text{ et } (a_5 = 3)) \text{ ou } ((a_5 \neq 4) \text{ et } (a_2 \neq 3))$.

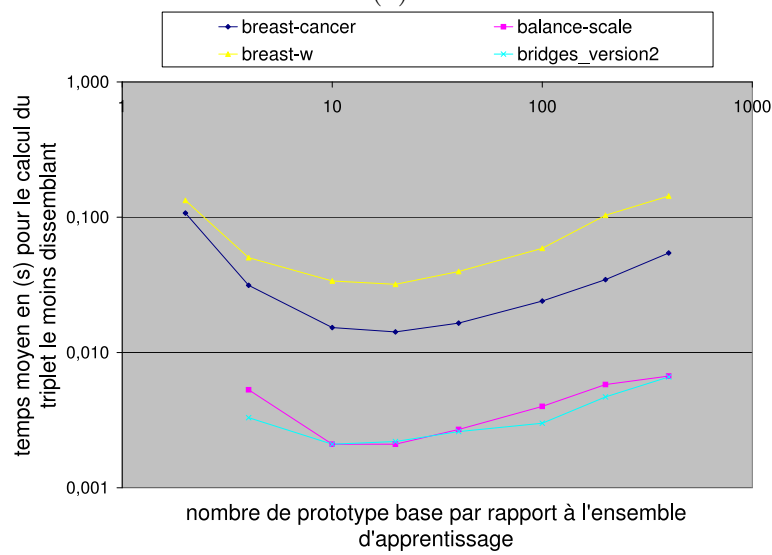
MONK3 a subi un bruitage de 5% sur les classes.

base de données % de bp par rapport à l'ens. d'app.		Breast Cancer	Balance Scale	Breast-W	Bridges
ensemble de test		228	593	629	73
ensemble d'apprentissage		58	32	70	34
0%	temps en pré-calcul (s)	0	0	0	0
	temps en ligne (s)	0.122	0.011	0.187	0.010
	nb DA calculée en ligne	110409	17515	165945	8463
2%	temps en pré-calcul (s)	0.01	0.00	0.01	0.00
	temps en ligne (s)	0.107	0.026	0.133	0.018
	% DA calculée	19.7	100	10.3	100
4%	temps en pré-calcul (s)	0.01	0.00	0.01	0.01
	temps en ligne (s)	0.031	0.005	0.050	0.003
	% DA calculée	4.6	16.5	3.0	16.8
10%	temps en pré-calcul (s)	0.01	0.00	0.02	0.01
	temps en ligne (s)	0.015	0.002	0.034	0.002
	% DA calculée	1.2	3.4	1.4	4.8
20%	temps en pré-calcul (s)	0.03	0.00	0.03	0.01
	temps en ligne (s)	0.014	0.002	0.032	0.002
	% DA calculée	0.7	2.4	0.9	4.4
40%	temps en pré-calcul (s)	0.05	0.01	0.08	0.01
	temps en ligne (s)	0.017	0.003	0.040	0.006
	% DA calculée	1.2	3.2	1.3	5.3
100%	temps en pré-calcul (s)	0.12	0.02	0.23	0.02
	temps en ligne (s)	0.024	0.004	0.059	0.003
	% DA calculée	2.3	5.8	3.0	11.8
200%	temps en pré-calcul (s)	0.32	0.03	0.76	0.03
	temps en ligne (s)	0.035	0.006	0.103	0.005
	% DA calculée	5.9	11.2	5.8	23.0
400%	temps en pré-calcul (s)	1.02	0.06	2.31	0.06
	temps en ligne (s)	0.054	0.007	0.144	0.007
	% DA calculée	11.8	22.1	11.5	45.6

TAB. 6.5 – Performances des capacités d'élagage de FADANA en fonction du nombre de prototypes bases.



(a)



(b)

FIG. 6.4 – Apport de FADANA en temps de calcul en fonction du nombre de prototypes bases.

- **SPECT** (*SP.*). Ces données décrivent un diagnostic à partir d'images de tomographie cardiaque (SPECT : Single Proton Emission Computed Tomography). Les données se composent de 22 attributs binaires et de la classe d'appartenance des données (du patient) qui a l'une des deux valeurs : *normale* (classe 0) ou *anormale* (classe 1).
- **Balance-Scale** (*B.S.*). Base de données à trois classes faite pour modéliser les résultats expérimentaux psychologiques. La vraie règle de classification est :
 1. $(a_1 * a_2) > (a_3 * a_4) \implies \text{classe} = 1$;
 2. $(a_1 * a_2) = (a_3 * a_4) \implies \text{classe} = 2$;
 3. $(a_1 * a_2) < (a_3 * a_4) \implies \text{classe} = 3$.
- **Hayes Roth** (*H.R.*). Base de données multi-classes.
- **Breast-W** (*Br.*). C'est une base de données qui sert à classer les cancers de sein des patientes du Wisconsin en affection bénigne ou maligne. Cette base de données contient des exemples avec des valeurs manquantes.
- **Mushroom** (*Mu.*). Base de données contenant des valeurs manquantes et servant à classer les champignons en *comestible* ou *toxique*. Ceux dont la classe est inconnue sont classés comme *toxique*.
- **kr-vs-kp** (*k.k.*). Ces données décrivent si dans l'état actuel du jeu d'échecs les blancs peuvent gagner.

Afin de mesurer l'efficacité de *WAPC*, nous avons appliqué aux mêmes bases de données d'autres classifieurs communément utilisés dans la littérature. Dans le but de montrer l'apport de la nouvelle méthode de pondération (Section 6.2), nous avons aussi utilisé le classifieur par proportion analogique sans pondération.

Voici les paramètres utilisés pour les méthodes de comparaison présentes dans le tableau 6.6 :

- **Table de Décision** : Le nombre de tables de décisions non améliorées à prendre en compte avant d'abandonner la recherche vaut 5 ;
- **Part** : Construit un arbre de décision C4.5 partiel à chaque itération et transforme la "meilleure" feuille en une règle. L'encodage utilise le *One-per-value*. Le facteur de confiance est de 0.25, le nombre minimum d'exemples par règle est 2, la quantité de données utilisées pour la réduction d'erreur de l'élagage est 3 ;
- **Perceptron multi-couches** : Apprentissage par rétro propagation. Codage en *One-per-value* ; le nombre d'époques est égal à 500 ; une couche cachée avec $(\text{nombre de classes} + \text{nombre d'attributs})/2$ nœuds ;
- **LMT** (Logistic Model Trees : arbres de modèles logistiques) : ce sont des arbres de classification avec des fonctions de régression logistique aux feuilles. Codage *One-per-value*. Le nombre minimum d'exemples pour lequel un nœud peut se doubler est 15, le nombre d'itérations pour LogitBoost est 10 ;
- **IB1** : Classificateur par le plus proche voisin utilisant la distance Euclidienne normalisée.
- **IBk** (**k=10**) : classifieur par les *k* plus proches voisins avec pondération des distances (poids en $1/\text{distance}$) ;

- **IB1 ($k=5$)** [Gui04] : classifieur par les k plus proches voisins avec pondération des attributs, pertinence du poids calculé avec le rapport de gain.

Nous avons utilisé WEKA [WF05] et Timbl [Gui04] dans la partie expérimentale pour obtenir les taux de classification des classifieurs ci-dessus. Certains de ces classifieurs sont bien appropriés pour les données binaires telles que ID3, PART, Decision Table. D'autres le sont moins comme IB1 ou Multilayer Perceptron qui sont en revanche plus adaptés aux données numériques.

6.4.2 Résultats

Les résultats de classification sont donnés dans le Tableau 6.6. Nous avons dans un premier temps choisi arbitrairement $k = 100$ pour *WAPC* et *APC*. Nous avons tiré les conclusions suivantes pour des études comparatives préliminaires. Premièrement, au vu du bon taux de classification de *WAPC* sur les données *Br.* et *Mu.*, on peut dire que *WAPC* traite correctement les données manquantes. Deuxièmement, les résultats obtenus sur les données *B.S* et *H.R* confirment que *WAPC* manipule bien les problèmes à multi-classes. Troisièmement, comme *MO.3* a subi un bruitage sur les classes, on peut dire *WAPC* résiste bien aux données bruitées. On remarquera que les résultats de *WAPC* et *APC* sont exactement les mêmes sur les données de *MO.* et *B.S*. Cela s'explique par le fait que les dissemblances analogiques les plus petites sont à chaque fois égales à zéro, donc on trouve toujours des analogies exactes et par conséquent la pondération ne change rien aux résultats. Autrement, sur le reste des bases de données, la pondération améliore bien les taux de classification. Néanmoins, dans la dernière base de donnée, on note que le classifieur par PA a des mauvais taux de classification comparés aux autres classifieurs, ce qui prouve, comme attendu, que *WAPC* ne marche pas bien sur toutes les bases de données.

6.4.3 Variation du paramètre k .

La figure 6.4.3 montre l'évolution des résultats du taux de reconnaissance sur certaines bases de données en faisant varier k qui représente le nombre de triplets les moins dissemblants retenus dans la phase de décision. Ce paramètre n'est pas très sensible dans le cas des données à attributs binaires et nominales et dans le cas d'ensemble d'apprentissage réduit comme c'est le cas ici. Il est possible cependant de le régler par l'utilisation d'un ensemble de validation.

6.4.4 Discussion et Perspectives

Mise à part son originalité, le classifieur par analogie avec pondération semble, d'après des expériences préliminaires, appartenir aux meilleurs classifieurs sur les données binaires et nominales, du moins pour des ensembles d'apprentissage de petite taille. La raison en est que le classifieur *WAPC* profite des régularités sur les valeurs alternées dans les classes, contrairement aux arbres de décision ou aux méthodes métriques qui corrélaient directement les attributs avec les classes. Ce qui signifie par exemple, que la dissemblance analogique entre quatre objets appartenant à la même classe est diminuée

Methodes	<i>MO.1</i>	<i>MO.2</i>	<i>MO.3</i>	<i>SP.</i>	<i>B.S</i>	<i>Br.</i>	<i>H.R</i>	<i>Mu.</i>	<i>k.k.</i>
nb. d'att. nominaux	7	7	7	22	4	9	4	22	34
nb. d'attributs binaires	15	15	15	22	4	9	4	22	38
nb. d'exemples d'app.	124	169	122	80	187	35	66	81	32
nb. d'exemples de test	432	432	432	172	438	664	66	8043	3164
nb. de classes	2	2	2	2	3	2	4	2	2
WAPC ($k = 100$)	98%	100%	96%	79%	86%	96%	82%	98%	61%
APC ($k = 100$)	98%	100%	96%	58%	86%	91%	74%	97%	61%
Decision Table	100%	64%	97%	65%	67%	86%	42%	99%	72%
Id3	78%	65%	94%	71%	54%	<i>mv</i>	71%	<i>mv</i>	71%
PART	93%	78%	98%	81%	76%	88%	82%	94%	61%
Multi layer Perceptron	100%	100%	94%	73%	89%	96%	77%	96%	76%
LMT	94%	76%	97%	77%	89%	88%	83%	94%	81%
IB1	79%	74%	83%	80%	62%	96%	56%	98%	71%
IBk ($k = 10$)	81%	7%	93%	57%	82%	86%	61%	91%	
IB1 ($k = 5$)	73%	59%	97%	65%	78%	95%	80%	97%	

TAB. 6.6 – Tableau de comparaison entre *WAPC* et d'autres méthodes standard de classification sur huit bases de données.

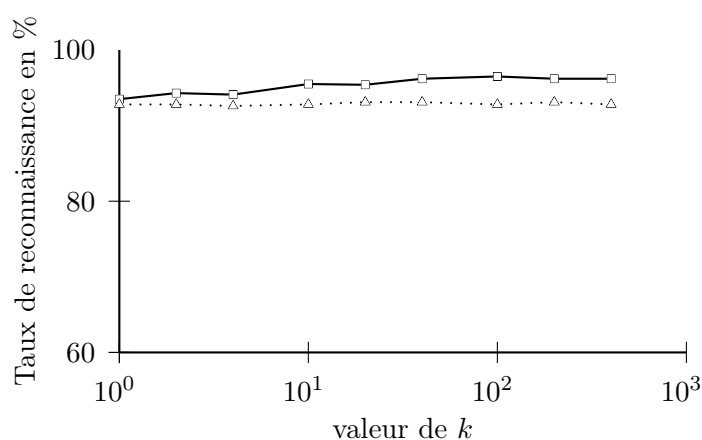


FIG. 6.5 – Variation du taux de reconnaissance en fonction de k . En trait plein le taux de reconnaissance sur la base de données “breast-w”, en pointillés le taux de reconnaissance sur la base de données “vote”

si on tient compte d'un nouvel attribut avec les valeurs $(0, 1, 0, 1)$ sur les quatre objets mais qu'elle est augmentée avec les valeurs $(0, 1, 1, 0)$.

Nous croyons qu'il est encore possible de faire progresser cette technique. En particulier, en étudiant une meilleure manière de pondérer des attributs qui soit plus précise, s'accordant plus finement au type d'analogie dans laquelle ils sont impliqués. Une éventuelle amélioration peut être apportée en appliquant *WAPC* sur des séquences, l'alphabet étant soit symbolique, soit numérique. Entre autres, beaucoup de travail peut

être effectué sur les aspects calculatoires, car même avec la méthode FADANA* (section 6.3.6), l'étape de décision prend toujours trop de temps pour un calcul réaliste sur de grands ensembles de données. C'est pourquoi nous avons choisi pour les expériences, des jeux d'essais relativement restreints en nombre d'exemples.

Un autre point intéressant serait de connaître les limites de ce type de classification. Nous savons définir des relations analogiques et des dissemblances analogiques sur des objets numériques, et nous avons étendu ces notions aux séquences d'objets symboliques ou numériques [DM05]. Mais il reste un certain nombre de questions ouvertes, dont les suivantes :

Dans quels cas l'analogie entre les classes est-elle liée à celle des objets qui constituent les classes dans l'ensemble d'apprentissage ? Y a-t-il des données pour lesquelles une classification analogique est "normale" et des données pour lesquelles c'est un non-sens ? Pour aller plus loin, pouvons-nous définir une relation par analogie entre les étiquettes des classes en examinant l'ensemble d'apprentissage ?

Chapitre 7

Expériences en génération de séquences.

Sommaire

7.1 Description du signal d'écriture manuscrite en ligne	114
7.1.1 Processus d'acquisition de données	114
7.1.2 Points d'ancrage	115
7.1.3 Vue d'ensemble	116
7.2 Génération basée sur l'analogie	116
7.2.1 Dissemblance analogique entre objets	117
7.2.2 Dissemblance analogique entre séquences	117
7.2.3 Matrice de pondération	118
7.2.4 Exemple	118
7.3 Génération à base de connaissances	118
7.3.1 Génération par distorsions d'images : échelle et pente	118
7.3.2 Génération par distorsions en ligne : vitesse et pente	119
7.4 Expérimentations	120
7.4.1 Protocole expérimental	120
7.4.2 Exemples de caractères synthétiques d'écriture manuscrite . .	123
7.4.3 Résultats	123
7.4.4 Validité et significativité des tests	123
7.5 Conclusion	126

Contenu du chapitre

Ce chapitre traite d'expériences en génération de séquences pour améliorer un système de reconnaissance d'écriture manuscrite.

Introduction

Dans de nombreux systèmes de reconnaissance de formes, l'acquisition de données étiquetées constitue un processus lent, coûteux, pas toujours agréable pour l'utilisateur. Par exemple, lors d'un achat d'un *smartphone* équipé d'un système de reconnaissance d'écriture manuscrite, il ne serait pas très incitatif de demander au client de saisir des dizaines d'exemplaires de chaque lettre de l'alphabet pour donner au système de reconnaissance un ensemble d'apprentissage assez conséquent pour un bon taux de classification. Le même problème se présente lorsque l'utilisateur introduit de nouveaux modèles à reconnaître, comme les gestes d'édition de texte sur un écran. Cependant, pour être efficace, n'importe quel système de classification statistique doit s'adapter au style d'écriture de la nouvelle personne (ou à une nouvelle forme) avec le plus d'exemples possible, ou au moins un nombre suffisant d'exemples bien choisis, qui couvrent le style d'écriture de la nouvelle personne.

Pour surmonter ce paradoxe et donc rendre possible l'apprentissage d'un classifieur avec très peu d'exemples, une idée est de générer des nouveaux exemples en ajoutant aléatoirement du bruit aux éléments du petit ensemble d'apprentissage. Le bruit peut être rajouté dans l'espace des attributs où l'apprentissage et la classification sont faits ou dans la représentation brute des données.

Nous proposons ici d'appliquer ce principe à l'écriture manuscrite en générant des nouvelles séquences par analogie : à partir de trois exemples de séquences, nous pouvons générer autant de séquences synthétiques que possible, en garantissant qu'elles aient une faible *dissemblance analogique* avec les trois premières séquences originales.

7.1 Description du signal d'écriture manuscrite en ligne

Pour faciliter la compréhension des sections suivantes, nous présentons dans cette section le système d'acquisition de signaux d'écriture manuscrite en ligne. Ce système et le logiciel de reconnaissance associé sont développés au sein de l'équipe IMADOC de l'IRISA ; cette partie de notre travail a ainsi été faite en collaboration avec elle. Cette équipe travaillait déjà sur la génération de séquences.

7.1.1 Processus d'acquisition de données

Les caractères en ligne utilisés dans ce chapitre ont été saisis sur un PDA (Personal Digital Assistant) en utilisant un stylet. Les caractères sont isolés : le problème de segmentation des caractères d'un mot n'est pas traité dans ce qui suit. Ils sont saisis sans aucun contexte, dans un ordre aléatoire, et en ligne. Ce protocole est opposé à celui de l'acquisition hors ligne, dans lequel les caractères sont plutôt des images numérisées à l'aide d'un scanner ou d'une caméra.

Lors de l'acquisition en ligne, un caractère est considéré comme une fonction temporelle $p(t)$ décrivant la position du stylet. Chaque point $p(t)$ est défini par ses coordonnées géométriques x et y mais aussi par la pression du stylet sur la surface d'entrée. Dans le cas où la pression est nulle, cela veut dire que le stylet ne touche pas la surface, et

dans ce cas les positions x et y ne sont pas définies. Par conséquent, un caractère est composé de un ou plusieurs segments, chaque segment commence par une pose de stylet et finit par un levé de stylet.

Nos deux stratégies de génération n'ont pas la même description des données en entrée. D'un côté, la génération par *connaissances a priori* utilise directement les segments sans aucune transformation au préalable, car ils transportent des informations comme la vitesse qui peuvent être perdues lors d'un lissage ou d'un ré-échantillonnage. D'un autre côté, la génération par *proportion analogique* (PA) demande une représentation légèrement plus élaborée du signal. Ainsi, avant la génération par PA , le signal est ré-échantillonné et ensuite transformé en une séquence en code de Freeman. Le ré-échantillonnage a pour but de produire un nouveau signal dans lequel les distances géométriques entre deux points consécutifs sont égales (les points entre le lever de stylet et le poser de stylet sont interpolés). On passe donc d'un échantillonnage temporel, lors de la saisie en ligne, à un échantillonnage spatial. Ensuite, l'encodage de Freeman prend le relais pour transformer les séquences de vecteurs en une séquence de symboles représentant la direction discrète.

L'alphabet du code de Freeman est constitué de 4, 8 ou 16 ... directions régulièrement distribuées. Nous avons ajouté la direction 0 qui correspond à un point isolé dans le signal original comme dans les lettres i et j par exemple. Dans ce chapitre nous utilisons le code de Freeman sur 16 directions $\Sigma = \{0, 1, \dots, 16\}$. La génération par proportion analogique produit des caractères en code de Freeman, qui sont ensuite décodés en signal en ligne, en retraçant les directions des séquences avec la même distance point à point.

7.1.2 Points d'ancrage

Pour les besoins de la génération par proportion analogique, nous avons étendu le code de Freeman par un ensemble de 8 lettres capitales $\{C, D, E, H, K, L, M, N\}$, chacune d'elle correspondant à point spécifique dans l'écriture manuscrite des caractères. Ces points sont appelés *points d'ancrage* et, comme on le verra dans la section 7.2.3, ils conduisent à une génération par proportion analogique plus homogène. Ces points d'ancrage proviennent de propriétés stables de l'écriture manuscrite, comme défini dans [AL97] : stylet-levé/posé, extrema-en- y , points angulaires et extrema-en- y dans une boucle. La table 7.1 donne une description et des exemples des différents points d'ancrage. Un point dans un caractère peut à la fois être un point angulaire, un extremum-en- y , un extremum-en- y dans une boucle et un levé de stylet. C'est pourquoi il y a au sein de l'ensemble des points d'ancrage une hiérarchie (une relation d'ordre) qui indique, en cas de possibilités multiples, quelle étiquette mettre à ce point. Dans l'ordre, en commençant par le moins important, on a : extrema-en- y , point angulaire, extrema-en- y dans une boucle et enfin stylet-levé/posé.

Symbole	Description	Exemples
C	extrema-en- y supérieur dans une boucle	
D	extrema-en- y inférieur dans une boucle	
E	point angulaire supérieur	
H	point angulaire inférieur	
K	extrema-en- y supérieur	
L	extrema-en- y inférieur	
M	stylet-levé	
N	stylet-posé	

TAB. 7.1 – Description des points d’ancrage.

7.1.3 Vue d’ensemble

La figure 7.1 montre une vue d’ensemble du processus global de génération de nouveaux exemples en utilisant des distorsions (connaissances *a priori*) et les proportions analogiques sur les séquences du code de Freeman. Suite à la phase de génération, les caractères, qui sont jusqu’à maintenant représentés par des séquences, sont transformés en vecteurs à 21 dimensions. Ces vecteurs sont les entrées du système de reconnaissance de l’écriture manuscrite. La transformation de séquences en vecteurs est basée sur des propriétés stables de l’écriture manuscrite [AL97, AB02]. Les processus de classification et de test dans la section 7.4 sont faits dans cet espace de représentation à 21 dimensions.

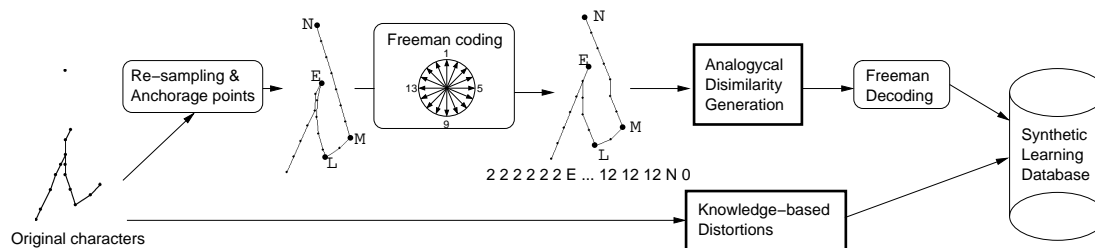


FIG. 7.1 – Vue d’ensemble du processus de génération de données synthétiques pour l’apprentissage en utilisant les distorsions (connaissances *a priori*) et les proportions analogiques sur les chaînes de code de Freeman.

7.2 Génération basée sur l’analogie

Dans cette section, nous montrons comment la notion de proportion analogique peut être utilisée pour la génération de nouveaux exemples.

Nous savons que *résoudre une équation analogique* consiste à trouver l’élément x inconnu dans une équation analogique. Par exemple, dans l’équation “*fins is to fish as*

wings is to x, x serait *bird* par résolution dans le domaine sémantique. En revenant à l'écriture manuscrite, supposons maintenant que le caractère “a” est encodé comme une séquence et que nous avons trois exemplaires a_1 , a_2 et a_3 de ce caractère. Nous pouvons générer d'autres caractères “a” en utilisant la résolution exacte ou approchée des équations analogiques sur les séquences : “ $a_1 : a_2 :: a_3 : x$ ” ou “ $a_3 : a_2 :: a_1 : x$ ”. Ainsi, si nous avons n exemples, le nombre de nouveaux “a” synthétiques que nous pouvons générer est $n^2 \times (n - 1)/2^1$.

Pour aller plus loin, l'introduction de la notion de dissemblance analogique autorise ne pas se contenter de prendre le ou les éléments générés avec la plus petite DA mais encore ceux avec les k plus petites DA (on produira alors $k \times n^2 \times (n - 1)/2$ exemples synthétiques). On peut aussi imaginer de générer toutes les séquences à DA inférieure à un seuil et d'en choisir aléatoirement un nombre donné.

7.2.1 Dissemblance analogique entre objets

L'alphabet que nous utilisons ici $\Sigma = \{1, 2, \dots, 16, 0, C, \dots, N\}$ est donc constitué d'un code de Freeman augmenté. Chaque lettre ou objet de l'alphabet peut être vue comme un élément d'un espace vectoriel \mathbb{R}^m , où $m = 18$. Chaque lettre du code de Freeman prend donc une dimension et les deux dimensions qui restent permettent de rendre l'alphabet plus cyclique et prennent donc les valeurs x et y du code de Freeman sur un cercle de rayon 1. Ceci permet d'avoir un ordre de préférence sur les analogies, puisque :

$$\underbrace{DA(a : a :: b : b)}_{\text{trivial}} < \underbrace{DA(a : b :: e : f)}_{\text{cyclique}} < \underbrace{DA(a : b : e : g)}_{\text{quelconque}}$$

La dissemblance analogique associée à cet espace vectoriel dans cette application est la suivante : $DA(u, v, w, x) = \|(u - v) - (w - x)\|$.

La détermination du meilleur quatrième objet x par résolution de l'équation analogique $u : v :: w : x$ est donnée par la formule suivante :

$$x = \underset{x_i \in \Sigma}{ArgMin} \ DA(u, v, w, x_i)$$

Cela nous permet d'obtenir l'ordre suivant : nous privilégions les analogies de type $a : a :: b : b$ qui ont donc une $DA = 0$, puis suivent les analogies de type cyclique comme $1 : 2 :: 5 : 6$ qui ont une DA de valeur α non nulle, puis les autres analogies du type $1 : 2 :: 5 : 8$ qui ont une DA de valeur $\beta > \alpha$.

7.2.2 Dissemblance analogique entre séquences

Compte tenu de la représentation des caractères manuscrits, la génération de nouveaux caractères se fera sur des séquences. Pour cela nous utilisons deux algorithmes.

¹Normalement le nombre de combinaisons de trois éléments est de n^3 , mais comme “ $a_1 : a_2 :: a_3 : x$ ” est équivalent à “ $a_1 : a_3 :: a_2 : x$ ” par échange des moyens le résultat serait le même, ce qui explique le rapport de 2. Et comme “ $a_1 : a_1 :: a_2 : x$ ” donnerait comme résultat $x = a_2$ qui n'est pas un nouveau élément, on remplace n par $n - 1$.

Le premier est l'algorithme SOLVANA décrit au paragraphe 5.8.4.1, fondé sur la programmation dynamique, qui génère les séquences ayant la dissemblance analogique la plus petite. Le deuxième est un algorithme *A toile* modifié qui génère les k meilleures séquences, mais qui est évidemment plus lent que le premier.

7.2.3 Matrice de pondération

L'utilisation des points d'ancrage (voir section 7.1.2) dans les séquences comme séparateurs de segments permet un alignement plus pertinent des trois premières séquences de l'équation analogique, qui par conséquent vont générer un caractère manuscrit homogène avec le type d'écriture de l'utilisateur. Afin de forcer les segments de même nature à s'aligner ensemble, nous pénalisons lourdement les analogies non exactes sur les points d'ancrage par rapport aux analogies non exactes sur le code de Freeman. Et comme les points d'ancrage suivent une certaine hiérarchie, la dissemblance analogique entre les points d'ancrage est différente, suivant le même ordre. Ce qui nous donne une matrice de pondération avec les propriétés suivantes :

- une DA intra-code de Freeman faible et à moitié cyclique 7.2.1 ;
- une DA intra-points d'ancrage importante et ordonnée ;
- une DA entre les points d'ancrage et les codes de Freeman très importante ;
- les distances entre \smile et les codes de Freeman faibles et égales ;
- les distances entre \smile et les points d'ancrage importantes et égales.

7.2.4 Exemple

Dans cet exemple de la Figure 7.2, nous avons choisi le caractère “h”. Nous avons donc utilisé trois exemples de caractères manuscrits h_1 , h_2 et h_3 . Nous générons par la suite en utilisant l'algorithme SOLVANA la séquence x_0 , représentant ainsi le nouveau caractère synthétique “h” ayant la plus petite dissemblance analogique avec h_1 , h_2 et h_3 . Nous voyons ici que la séquence générée n'est pas en dissemblance analogique nulle, car par exemple $DA(\smile, 2, 3, 3) \neq 0$ ou encore $DA(6, \smile, 8, 8) \neq 0$

7.3 Génération à base de connaissances

Dans cette section nous présentons deux types classiques de distorsions d'image et deux autres distorsions en ligne basées sur la spécificité du scripteur. Chaque distorsion dépend d'une ou de plusieurs paramètres aléatoires. Afin de générer un caractère synthétique à partir d'un original en utilisant ces distorsions, nous choisissons une valeur aléatoire pour chaque paramètre que nous appliquons. Les paramètres sont choisis dans un intervalle fixé de façon empirique par des connaissances expertes sur l'écriture manuscrite, afin d'éviter des distorsions trop importantes.

7.3.1 Génération par distorsions d'images : échelle et pente

Pour générer de nouveaux exemples, on a besoin de choisir les distorsions qu'on doit appliquer en tenant compte du processus d'acquisition des données. Les vraies images,

$h_1 =$	9	~	9	~	9	9	9	9	9	~	H	1	2	~	4	K	6	9	9	9
$h_2 =$	1	K	~	8	9	9	9	9	9	10	H	~	2	2	4	K	~	8	8	9
$h_3 =$	~	~	9	8	9	9	9	9	9	10	H	2	2	3	3	K	8	9	9	~
$x =$	1	K	~	8	9	9	9	9	9	10	H	2	2	3	3	K	8	8	8	~

(a)

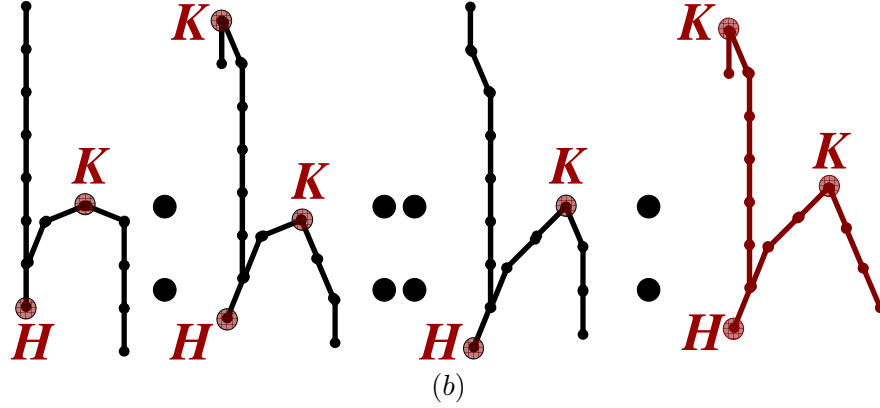


FIG. 7.2 – (a) Génération d’une nouvelle séquence par résolution approchée d’équation analogique. (b) Représentation graphique associée.

capturées par des appareils photo ou par des caméras, peuvent être perturbées par la perspective, la rotation, le bruit... tandis que l’écriture manuscrite scannée ne peut être perturbée que par la variation de l’épaisseur de l’encre. Dans cette application, on ne traitera que des déformations par inclinaison ou des déformations de taille vu la nature de nos données qui sont acquises isolées et en ligne sur un PDA. Deux paramètres correspondent aux transformations d’échelle, α_x et α_y qui sont le rapport des tailles sur les deux coordonnées. La pente permet de générer une écriture plus ou moins inclinée. Elle dépend du paramètre α_s qui quantifie la tangente de la pente. Une valeur positive de la pente incline l’écriture vers la droite et vice versa. La Figure 7.4 montre des exemples de distorsions de caractères par pente et échelle.

7.3.2 Génération par distorsions en ligne : vitesse et pente

Certains travaux dans la littérature se sont intéressés à la génération d’écriture manuscrite afin d’accroître l’ensemble d’apprentissage en ligne. Dans [VKB05], les auteurs utilisent les travaux sur la génération d’écriture manuscrite [PG98] pour l’augmentation de la taille de l’ensemble d’apprentissage afin d’apprendre un système de reconnaissance de phrases pour un scripteur particulier. Cette approche est proche de la nôtre mais n’est pas utilisable dans notre contexte car nous avons besoin d’un modèle dépendant du scripteur. C’est pourquoi nous utilisons deux distorsions en ligne d’écriture manuscrite : la *variation de la vitesse* et la *modification de la courbure*.

7.3.2.1 Variation de la vitesse

Le but de la distorsion par *variation de la vitesse* est de modifier la taille relative des parties verticales et horizontales du caractère, comme le montre la figure 7.4, selon un paramètre α_v . En vérité, la partie droite de l'écriture peut varier sans que le style d'écriture du scripteur change. C'est pourquoi on modifie la vitesse

$$\vec{V}(t) = (x(t+1) - x(t), y(t+1) - y(t))$$

suivant sa direction. Si cette direction est proche de l'un des axes alors elle croît ou décroît le rapport α_v . La nouvelle écriture manuscrite synthétique est définie par $p'(t)$:

$$p'(t) = p'(t-1) + \beta * \vec{V}(t-1), \text{ avec } \beta = \begin{cases} 1 & \text{si } \arg(\vec{V}(t-1)) \left[\frac{\pi}{2} \right] \in \left[\frac{\pi}{8}, \frac{3\pi}{8} \right], \\ \alpha_v & \text{sinon.} \end{cases}$$

7.3.2.2 Modification de la courbure

La distorsion par *modification de la courbure* modifie la courbure de l'écriture comme le montre la Figure 7.4. Elle permet de fermer ou d'ouvrir la boucle du caractère. La modification de la courbure utilise un paramètre α_c . La courbure au point $p(t-1)$ est définie par l'angle $\hat{\theta}(t-1) = \widehat{(p(t-2), p(t-1), p(t))}$ dans $] -\pi, \pi]$. Afin de garder la structure du caractère, nous ne modifions pas les lignes droites. L'équation 7.2 donne la position du point $p'(t)$ suivant les deux points précédents et la courbure originale $\hat{\theta}(t-1)$ modifiée par α_c . La modification angulaire maximale est pour $\hat{\theta}(t-1) = \frac{\pi}{2}$.

$$\hat{\theta}(t-1) = \widehat{(p(t-2), p(t-1), p(t))} \in] -\pi, \pi], \quad (7.1)$$

$$(p'(t-2), \widehat{p'(t-1)}, p'(t)) = \hat{\theta}(t-1) - \alpha_c * 4 * \frac{|\hat{\theta}(t-1)|}{\pi} * \left(1 - \frac{|\hat{\theta}(t-1)|}{\pi}\right) \quad (7.2)$$

7.4 Expérimentations

Dans cette section, nous présentons en premier lieu le protocole expérimental suivi, puis quelques exemples de génération de caractères. Nous montrons ensuite que les résultats obtenus par les nouvelles méthodes de génération ont permis d'améliorer le taux de reconnaissance de trois classifieurs classiques, entraînés sur peu de données.

7.4.1 Protocole expérimental

Le projet IMADOC dispose d'une base de données d'écriture manuscrite de 12 scripteurs différents. Chacun a saisi sur un *PDA* 40 exemplaires de chaque lettre de l'alphabet romain (qui contient 26 lettres). Pour chaque scripteur, on a divisé les 40 exemplaires en quatre ensembles de 10, dont un servira d'ensemble d'apprentissage et un autre d'ensemble de test. On itérera l'opération quatre fois (sur les 12 possibles) en prenant

soin que les ensembles d'apprentissage et de test soient toujours différents, afin que les erreurs ne soient pas dépendantes.

Comme le nombre de données obtenu est insuffisant pour faire un apprentissage correct, l'idée est de générer des données synthétiques d'apprentissage de la façon suivante : à partir de $10 * 26$ écritures manuscrites pour chaque scripteur dans l'ensemble d'apprentissage, on ne garde que $k * 26$ écritures, avec $2 \leq k \leq 10$, en tirant aléatoirement à trois reprises k exemplaires parmi les 10 (chaque tirage est nommé aussi validation). La prochaine étape revient à enrichir par des instances synthétiques les $k * 26$ instances pour qu'elles atteignent le nombre de $300 * 26$ instances. Le nombre $300 * 26$ est choisi arbitrairement, mais nous le jugeons suffisamment grand pour apprendre correctement les classifieurs. On utilise trois algorithmes de génération d'instances synthétiques qui sont les suivants :

1. génération par distorsions d'image (paragraphe 7.3.1) ;
2. génération par distorsions d'image et par distorsions en ligne (paragraphe 7.3.2) : les proportions de génération de chaque méthode de génération sont égales ;
3. génération par distorsions d'image et par distorsions en ligne et par résolution approchée d'équation analogique (paragraphe 7.2) : les proportions de génération de chaque méthode de génération sont égales.

L'étape suivante consiste à récupérer les instances d'apprentissage ($300 * 26$) pour apprendre un des trois classifieurs suivant :

1. **KNN** : k plus proches voisins (on a pris $k = 1$) ;
2. **SVM** : Machine à Vecteurs Supports (noyau Gaussien, C bound = 100) ;
3. **RBFN** : Réseau de Fonctions à Base Radiale (1 neurone par classe, ce qui fait un total de 26 neurones).

On calcule ensuite les taux de reconnaissance sur l'ensemble de test retenu antérieurement ($10 * 26$ éléments).

Ainsi, le taux de reconnaissance est dépendant du scripteur, de l'ensemble d'apprentissage et de test, du nombre de lettres retenues, de la méthode de génération et du classifieur utilisé (voir la Figure 7.3).

Enfin, on calcule la moyenne des taux de reconnaissance sur les douze scripteurs et sur les quatre ensembles d'apprentissage et de test possibles et sur les trois ensembles de validation. Afin de mesurer l'apport de la génération d'instance par les trois méthodes citées auparavant, l'idée est d'apprendre respectivement sur deux références $10 * 26$ et $30 * 26$ sans génération. On pourrait croire que l'amélioration du taux de reconnaissance est due au nombre important de données d'apprentissage. En réalité, ce taux est amélioré par la méthode de génération. Pour montrer cela, on générera des nouvelles instances, même pour les deux références, pour atteindre le même nombre de données, soit $300 * 26$. La génération est faite à l'aide d'un bruit normal sur chaque dimension de 0.5%. Ce bruit est jugé assez important pour que les données générées soit différentes et pas très grand pour que ces dernières appartiennent à la même classe.

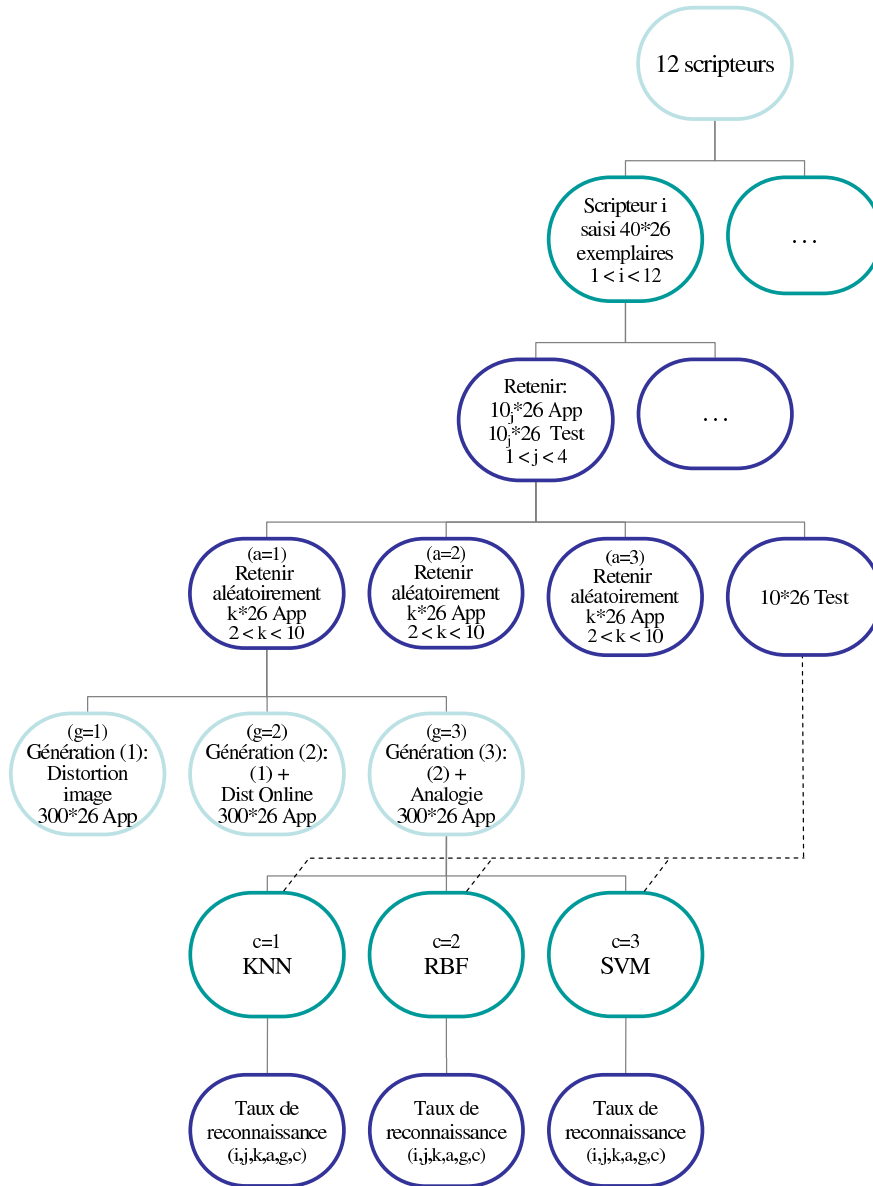


FIG. 7.3 – Organigramme du protocole expérimental

- $1 \leq i \leq 12$: indice du scripteur
- $1 \leq j \leq 4$: indice de la division en 4 ensembles
- $1 \leq k \leq 10$: nombre de lettres original retenu pour la génération
- $1 \leq a \leq 3$: indice du tirage aléatoire des k lettres
- $1 \leq g \leq 3$: indice de la méthode de génération
- $1 \leq c \leq 3$: indice du classifieur

7.4.2 Exemples de caractères synthétiques d'écriture manuscrite

La Figure 7.4 montre des exemples synthétiques d'écriture manuscrite générés par distorsion d'image, par distorsion en-ligne et par analogie. Nous notons que chaque




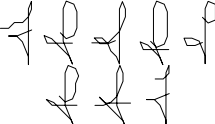




	Distorsions d'image		Distorsions en-ligne		Analogie
Original					
Synthétique	Échelle	Pente	Vitesse	Courbure	
					

FIG. 7.4 – Exemples de caractères synthétiques générés par les trois approches.

stratégie de génération permet une distorsion du caractère original non accomplie par aucune autre stratégie. La distorsion d'image permet de changer et la taille et l'inclinaison du caractère. La distorsion en-ligne permet de déplacer des parties du caractère et de changer la courbure des segments en gardant la même forme. La génération par analogie permet la modification du caractère suivant la transformation détectée entre les trois caractères originaux.

7.4.3 Résultats

Les courbes présentes sur la figure 7.5 montrent la moyenne des taux de reconnaissance de caractères isolés en mono-scripteur sur les 12 scripteurs et la moyenne des écart-types associée suivant le nombre k des caractères originaux. Les taux de reconnaissance des références avec 10 et 30 caractères sont aussi représentés. Ces résultats sont obtenus pour les trois classifieurs SVM, KNN et RBFN.

La figure 7.5 compare le taux de reconnaissance réalisé par les trois stratégies de génération pour les trois classifieurs. Premièrement, nous dirons que le comportement global est le même pour les trois classifieurs. Cela veut dire que les conclusions suivantes sont indépendantes du type de classifieur. Deuxièmement, les trois stratégies de générations sont complémentaires car en utilisant la méthode *distorsion d'image* et *distorsion en-ligne* on obtient de meilleurs résultat qu'avec *distorsion d'image* seulement, et en utilisant *distorsion d'image* et *distorsion en-ligne* et *génération par analogie*, on a de meilleurs résultats qu'avec *distorsion d'image* et *distorsion en-ligne*.

7.4.4 Validité et significativité des tests

Cette qualité relative de chaque méthode de génération doit être validée statistiquement pour pouvoir affirmer que la différence entre les méthodes n'est pas due au hasard. On utilisera dans ce qui suit deux méthodes de validation. L'une est paramétrique, le *t-test* [GC89], et l'autre non-paramétrique, le SIGN TEST [Hul93]. Plutôt que de comparer les performances des différents algorithmes de génération, ces deux méthodes comparent

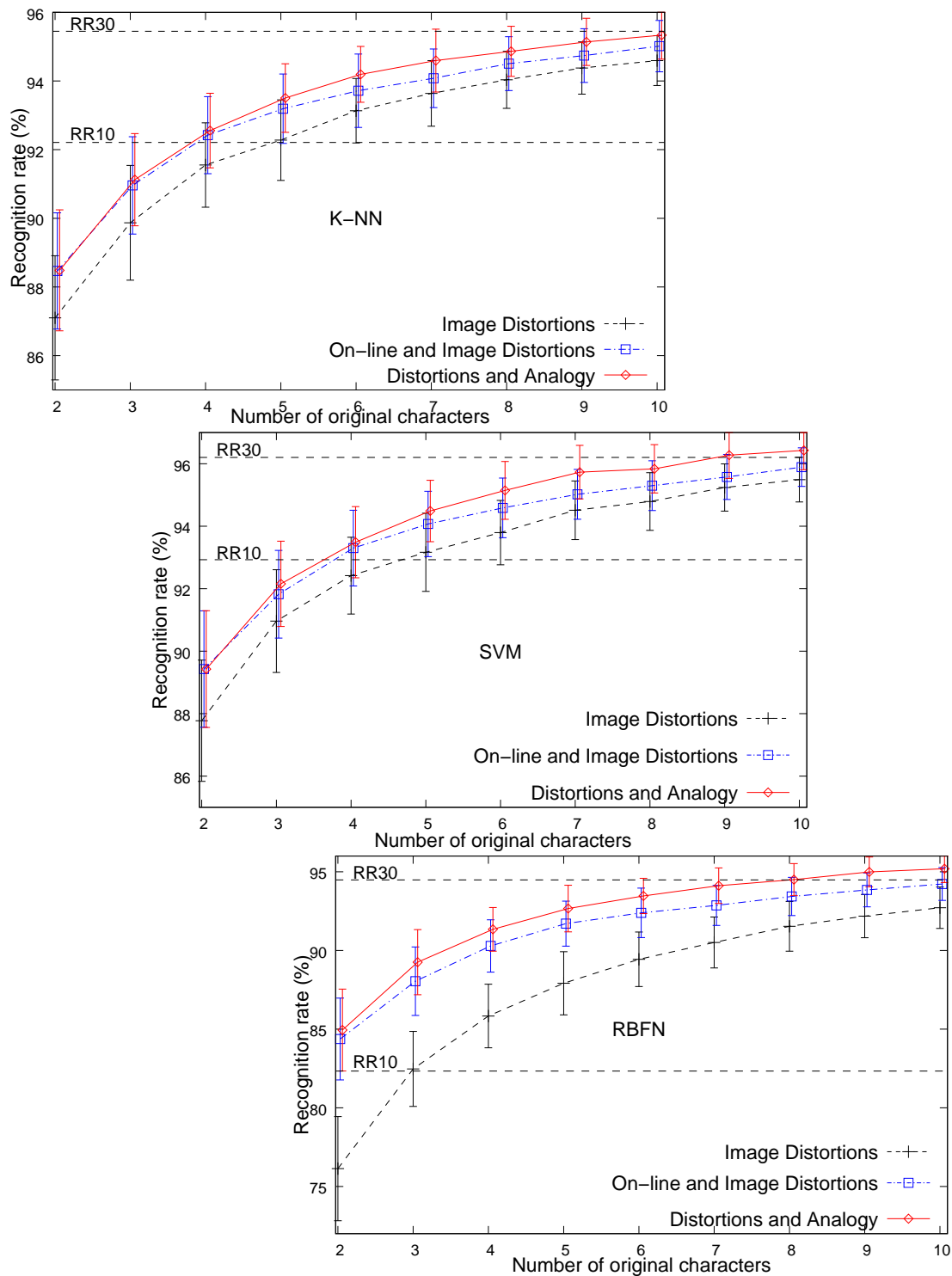


FIG. 7.5 – Taux de reconnaissance et écart-type de caractères isolés en mono-scripteur suivant le nombre de caractères originaux.

les différences entre les reconnaissances lettre par lettre. On comparera donc les trois algorithmes de génération deux par deux en observant la reconnaissance des caractères séparément.

Chaque observation prend l'une des trois valeurs suivantes :

- “0/3” si le caractère n’est reconnu par le classifieur sur aucune des trois validations (rappelons que chaque validation permet au classifieur d’apprendre sur des ensembles de caractères différents, tout en testant sur un ensemble unique de caractères) ;
- “1/3”, “2/3” ou “3/3” si le caractère est reconnu respectivement une, deux ou trois fois sur trois par le classifieur.

7.4.4.1 t-test

Ce test compare la grandeur de la différence entre les méthodes de génération par rapport à la variation entre les différences. Si la différence moyenne est grande au regard de l’écart-type, les deux méthodes sont significativement différentes.

Le t-test [GC89] n’est valable que si :

- Les erreurs suivent une loi de distribution normale ;
- Il y a indépendance des observations : l’erreur sur une observation n’influe aucunement sur les autres observations (condition vérifiée dans notre protocole expérimental, voir 7.4.1).

Supposons l’hypothèse nulle “H0” qui signifie que les deux méthodes de génération ne sont pas statistiquement différentes. On a :

$$t = \frac{\overline{D}}{s(D)/\sqrt{n}}$$

- n : nombre d’observations ;
- $\overline{D} = \frac{1}{n} \sum_{i=1}^n D_i$: moyenne des différences sur les observations
- $s(D) = \sqrt{\frac{1}{n-1} \sum_{i=1}^n (D_i - \overline{D})^2}$: écart-type des différences sur les observations.

D’après [GC89], cette statistique t obéit à une loi de Student à $(n - 1)$ degrés de liberté. La probabilité associée à l’hypothèse “H0” est :

$$P = 2Pr(Z \geq t)$$

Si on prend comme indice de confiance α pour rejeter l’hypothèse “H0”, il faut que $P < \alpha$.

Dans notre cas : $n = 12480$ et nous fixons $\alpha = 0.05$. Le calcul donne $t = 7,33544$ pour la comparaison des méthodes de génération 2 et 3 en utilisant le classifieur *SVM*, d’où $P = 2.22695E^{-13}$, valeur largement inférieure à $\alpha = 0.05$. La méthode du t -test nous permet de conclure que la différence entre les deux méthodes est statistiquement significative.

7.4.4.2 Sign test

Le *sign test* [Hul93] est une méthode de comparaison non-paramétrique et a donc l'avantage de ne pas émettre l'hypothèse sur la normalité de la distribution des observations ou des erreurs.

Le *sign test* remplace chaque différence dans les observations par un signe positif ou négatif. Puis la somme des occurrences des signes est comparée à la valeur attendue par l'hypothèse "H0". Ainsi, si une méthode est meilleure plus fréquemment que la moyenne attendue, alors elle est statistiquement supérieure à l'autre.

$$T = \frac{2 * \sum_{i=1}^n I[D_i > 0] - n}{\sqrt{n}}$$

- n : nombre d'observations ;
- $I[D_i > 0] = \begin{cases} 1 & \text{si } D_i > 0 \\ 0 & \text{sinon} \end{cases}$.

Cette statistique T obéit à une loi normale. La probabilité associée à l'hypothèse "H0" est :

$$P = 2Pr(Z \geq T)$$

Pour rejeter l'hypothèse "H0", il faut que $P < \alpha$.

Dans notre cas : $n = 12480$, $\alpha = 0.05$ ce qui nous donne $T = 107.446$ pour la comparaison des méthodes de génération 2 et 3 en utilisant le classifieur *SVM*, d'où $P < 1E^{-30}$, qui est largement inférieur à 0.05. Donc la différence entre les deux méthodes est statistiquement significative.

7.4.4.3 Résultats

Suite aux résultats obtenus dans la figure 7.5, on montre dans ce qui suit que l'amélioration constatée est significative et non l'œuvre du hasard. On a donc calculé la probabilité, suivant le t test et le *sign test* sur chaque nombre de caractère original, que la différence entre deux méthodes soit due au hasard. On a comparé en premier la différence entre la méthode de génération par distorsions d'image face à la méthode de génération par distorsions d'image et en ligne, et le tableau 7.2 (a) montre bien que la différence est effectivement significative sur tous les points. On a comparé ensuite la méthode de génération par distorsions d'image et en ligne face à la méthode de génération par distorsions d'image et en ligne et proportion analogique (voir tableau 7.2 (b)), et on trouve aussi que la différence est significative.

7.5 Conclusion

Dans cette partie nous avons montré comment générer des exemples synthétiques pour un apprentissage rapide d'un classifieur de caractère manuscrit pour un nouveau scripteur. Nous avons montré que la méthode de génération de séquence par analogie apportait de nouveaux caractères par rapport aux autres méthodes de déformation

nb caractères	3	4	5	8	9	10
	SVM					
<i>t</i> test	$7,6E^{-09}$	$2,8E^{-07}$	$1,1E^{-11}$	$5,4E^{-08}$	$2,0E^{-05}$	$6,1E^{-07}$
<i>sign</i> test	$< 1E^{-30}$	$< 1E^{-30}$	$< 1E^{-30}$	$< 1E^{-30}$	$< 1E^{-30}$	$< 1E^{-30}$
	RBFN					
<i>t</i> test	$3,7E^{-12}$	$2,3E^{-12}$	$1,3E^{-12}$	$9,2E^{-17}$	$2,7E^{-21}$	$4,1E^{-29}$
<i>sign</i> test	$< 1E^{-30}$	$< 1E^{-30}$	$< 1E^{-30}$	$< 1E^{-30}$	$< 1E^{-30}$	$< 1E^{-30}$

(a)

nb caractères	3	4	5	8	9	10
	SVM					
<i>t</i> test	$1E^{-02}$	$8E^{-02}$	$3E^{-03}$	$3E^{-04}$	$1E^{-08}$	$2E^{-06}$
<i>sign</i> test	$< 1E^{-30}$	$< 1E^{-30}$	$< 1E^{-30}$	$< 1E^{-30}$	$< 1E^{-30}$	$< 1E^{-30}$
	RBFN					
<i>t</i> test	$4E^{-12}$	$2E^{-12}$	$1E^{-12}$	$9E^{-17}$	$3E^{-21}$	$2E^{-13}$
<i>sign</i> test	$< 1E^{-30}$	$< 1E^{-30}$	$< 1E^{-30}$	$< 1E^{-30}$	$< 1E^{-30}$	$< 1E^{-30}$

(b)

TAB. 7.2 – Probabilité que la différence entre les moyennes de la génération par :
 –(a) (distorsions d'image) *vs* (distorsions d'image et en ligne)
 –(b) (distorsions d'image et en ligne) *vs* (distorsions d'image et en ligne et analogie)
 sur les taux de classification par le SVM et le RBFN soit due au hasard.

utilisées dans le domaine de l'écriture manuscrite. Ces méthodes combinées ont mené à des résultats meilleurs dans le cas de reconnaissance d'un monoscripteur.

Conclusion

“And I cherish more than anything else the Analogies, my most trustworthy masters. They know all the secrets of Nature, and they ought to be least neglected”
Johannes Kepler [[Kep73](#)] [[Hun04](#)]

Dans ce mémoire de thèse, nous avons étudié une notion formelle de l’analogie entre quatre objets dans le même univers, ou proportion analogique. Nous avons donné des définitions de la proportion analogique et des algorithmes pour résoudre les équations analogiques sur différents objets. Nous avons en particulier traité le cas des séquences, avec une définition originale de l’analogie basée sur des alignements optimaux sur quatre séquences. Nous avons aussi décrit une nouvelle notion, la dissemblance analogique, qui quantifie la distance qui sépare quatre objets d’être en analogie. Nous avons montré l’importance et l’utilité de cette notion dans l’apprentissage supervisé paresseux et dans la génération de nouvelles séquences. Nous avons montré que la classification par proportion analogique pouvait être meilleure que d’autres classificateurs standard. Nous avons montré que la génération de séquences par analogie crée de nouveaux objets, différents de ceux créés par les autres méthodes de génération.

Cependant, beaucoup reste à faire, et tout spécialement les questions suivantes sont encore ouvertes en classification :

- Quels types de données sont les mieux appropriées pour l’apprentissage par proportion analogique ?
- La classification par analogie que nous avons développée ne concerne que les objets à attributs binaires et nominaux et fonctionne en considérant seulement les analogies triviales sur les classes. Il y a plusieurs directions de généralisation :
- évaluer la méthode de pondération par rapport à des pondération plus grossières et plus fines,
- prendre en compte si possible les relations d’analogie entre classes,
- intégrer les données numériques,
- travailler sur des séquences longues et complexes, comme les bioséquences
- trouver une méthode plus rapide que FADANA pour la classification pour pouvoir utiliser des bases de données ayant un nombre d’exemples plus important,
- traiter l’apprentissage dans les séquences en le comparant par exemple aux méthodes d’inférence grammaticale ou aux modèles de Markov cachés.

En ce qui concerne la génération de séquences par analogie, nous avons utilisé des équations analogiques du type “ $a_1 : a_2 :: a_3 : a_x$ ”, mais on pourrait générer des nouveaux caractères “ a ” par des équations de type “ $b_1 : b_2 :: a_1 : a_x$ ”, ou même pour aller encore plus loin, générer des caractères d’un scripteur à partir de ceux d’un second scripteur. Nous pensons aussi à appliquer la génération par analogie à d’autres domaines, dont la transformation de la voix pour un système de synthèse de la parole.

Bibliographie

- [AA94] E. Plaza A. Aamodt. Case-based reasoning : Foundational issues, methodological variations, and system approaches. *AI Communications*, 7(1) :39–59, 1994.
- [AB02] Éric Anquetil and H. Bouchereau. Integration of an on-line handwriting recognition system in a smart phone device. In *Sixteenth ICPR International Conference on Pattern Recognition (ICPR'2002)*, volume 3, pages 192–195, 2002.
- [AL97] E. Anquetil and G. Lorette. Perceptual model of handwriting drawing application to the handwriting segmentation problem. In *Proc. of the 4th Int. Conf. on Document Analysis and Recognition*, pages 112–117, 1997.
- [AP98] The University of Chicago ARTFL Project. Dictionnaire de l'académie française, 5th edition, 1798.
- [AP32] The University of Chicago ARTFL Project. Dictionnaire de l'académie française, 8th edition, 1932.
- [ATI] Université Henri Poincaré ATILF. Trésor de la langue française informatisé.
- [CM02] A. Cornuéjols and L. Miclet. *Apprentissage artificiel : concepts et algorithmes*. Eyrolles, Paris, 2002.
- [Dae96] W. Daelemans. Abstraction considered harmful : lazy learning of language processing. In H. J. Van den Herik and A. Weijters, editors, *Proceedings of the sixth Belgian-Dutch Conference on Machine Learning*, pages 3–12, Maastricht, The Netherlands, 1996.
- [DHS01] R. Duda, P. Hart, and D. Stork. *Pattern classification*. Wiley, 2001.
- [DM04] Arnaud Delhay and Laurent Miclet. Analogical equations in sequences : Definition and resolution. In *Proceedings of the 7th International Colloquium on Grammatical Inference*, pages 127–138, 2004.
- [DM05] A. Delhay and L. Miclet. Analogie entre séquences : Définitions, calcul et utilisation en apprentissage supervisé. *Revue d'Intelligence Artificielle.*, 19 :683–712, 2005.
- [dS16] Ferdinand de Saussure. *Cours de Linguistique Générale*. Payot, 2nde édition, 1967 edition, Paris, 1916.
- [Epp98] David Eppstein. Finding the k shortest paths. *SIAM J. Computing*, 28(2) :652–673, 1998.

- [FBF77] J.H. Friedman, J.L. Bentley, and R.A. Finkel. An algorithm for finding best matches in logarithmic expected time. 3 :209–226, September 1977.
- [FN75] K. Fukunaga and P.M. Narendra. A branch and bound algorithm for computing k -nearest neighbors. 24 :750–753, 1975.
- [GC89] L. Gillick and S. J. Cox. Some statistical issues in the comparison of speech recognition algorithms. In IEEE, editor, *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 532–535, Glasgow, Scotland, 1989.
- [GHK01] D. Gentner, K. J. Holyoak, and B. Kokinov. *The analogical mind : Perspectives from cognitive science*. MIT Press, Cambridge, MA, 2001.
- [Gui04] Version Reference Guide. Timbl : Tilburg memory-based learner, 2004.
- [HJO⁺01] Aaron Hertzmann, Charles E. Jacobs, Nuria Oliver, Brian Curless, and David H. Salesin. Image analogies. In Eugene Fiume, editor, *SIGGRAPH 2001, Computer Graphics Proceedings*, pages 327–340. ACM Press / ACM SIGGRAPH, 2001.
- [Hof84] Douglas Hofstadter. The copycat project : An experiment in nondeterminism and creative analogies. Technical Report AIM-755, 1984.
- [Hul93] David A. Hull. Using statistical testing in the evaluation of retrieval experiments. In *Research and Development in Information Retrieval*, pages 329–338, 1993.
- [Hun04] Dan Hunter. Teaching and using analogy in law. In *Journal of the Association of Legal Writing Directors*, volume 2, 2004.
- [IH97a] E. Itkonen and J. Haukioja. *A rehabilitation of analogy in syntax (and elsewhere)*, pages 131–177. 1997.
- [IH97b] Esa Itkonen and Jussi Haukioja. A rehabilitation of analogy in syntax (and elsewhere). *András Kertész*, pages 131–177, 1997.
- [Jam90] William James. *The Principles of Psychology*. 1890.
- [JGCM83] Ryszard S. Michalski Jaime G. Carbonell and Tom M. Mitchell. *Machine Learning : An Artificial Intelligence Approach*, volume 1. Tioga Publishing Company, 1983.
- [Jus99] W. Just. Computational complexity of multiple sequence alignment with sp-score, 1999.
- [Kep73] Johannes Kepler. *Optics*. Quoted in Polya, 1973.
- [KF03] Boicho Kokinov and Robert M. French. Computational models of analogy-making. In Nature Publishing Group, editor, *Encyclopedia of Cognitive Science*, volume 1, pages 113–118, 2003.
- [KPK85] B. Kamgar-Parsi and L.N. Kanal. An improved branch and bound algorithm for computing k -nearest neighbors. 3 :7–12, 1985.
- [LA96] Y. Lepage and S. Ando. Saussurian analogy : a theoretical account and its application. In *Proceedings of COLING-96*, pages 717–722, København, August 1996.

- [Lav03] René Joseph Lavie. *Le locuteur analogique ou la grammaire remise à sa place*. Thèse de doctorat de l'Université de Paris X Nanterre, 2003.
- [Lea96] David Leake. CBR in context : The present and future. *Case-Based Reasoning : Experiences, Lessons, and Future Directions*, pages 3–30, 1996.
- [Lep98] Y. Lepage. Solving analogies on words : an algorithm. In *Proceedings of COLING-ACL'98*, volume 1, pages 728–735, Montréal, Canada, 1998.
- [Lep01] Y. Lepage. Défense et illustration de l'analogie. In *Actes de la 8^{ème} conférence sur le Traitement Automatique des Langues Naturelles (TALN)*, pages 373–377, Tours, France, 2001.
- [Lep03] Y. Lepage. *De l'analogie rendant compte de la commutation en linguistique*. Grenoble, 2003. Habilitation à diriger les recherches.
- [MD04] L. Miclet and A. Delhay. Relation d'analogie et distance sur un alphabet défini par des traits. Technical Report 5244, INRIA, July 2004. in French.
- [MH] James B. Marshall and Douglas R. Hofstadter. Making sense of analogies in metacat.
- [Mit93] M. Mitchell. *Analogy-Making as Perception*. MIT Press, Cambridge, MA, 1993.
- [Mit97] T. Mitchell. *Machine Learning*. McGraw-Hill, 1997.
- [MOV94] L. Micó, J. Oncina, and E. Vidal. A new version of the nearest-neighbour approximating and eliminating search algorithm aesa with linear preprocessing-time and memory requirements. *Pattern Recognition Letters*, 15 :9–17, 1994.
- [NHBM98] D.J. Newman, S. Hettich, C.L. Blake, and C.J. Merz. UCI repository of machine learning databases, 1998.
- [Nil80] N. Nilsson. *Principles of Artificial Intelligence*. Tioga Publishing Company., 1980.
- [Par98] François Parmentier. *Spécification d'une architecture émergente fondée sur le raisonnement par analogie : Application aux références bibliographiques*. PhD thesis, Université Henri Poincaré - Nancy 1, 1998.
- [PG98] Réjean Plamondon and Wacef Guerfali. The generation of handwriting with delta-lognormal synergies. *Biological Cybernetics*, 78 :119–132, 1998.
- [PY99] V. Pirrelli and F. Yvon. Analogy in the lexicon : a probe into analogy-based machine learning of language. In *Proceedings of the 6th International Symposium on Human Communication*, Santiago de Cuba, Cuba, 1999.
- [qdllf06] Office québécois de la langue française. Grand dictionnaire terminologique, 2006.
- [Rob05] Le Robert. *Dictionnaire Culturel en langue Française*. 2005.
- [SB02] Florence Durieux Sandrine Berger. *Questions de méthodes en Sciences de Gestion*. Editions EMS, 2002.

- [Sca03] Celso Carnos Scaletsky. *Rôle des références dans la conception initiale en architecture*. PhD thesis, Institut National Polytechnique de Lorraine, 2003.
- [SK83] D. Sankoff and J. Kruskal, editors. *Time Warps, String Edits and Macromolecules : the Theory and Practice of Sequence Comparison*. Addidon-Wesley, 1983.
- [Str05] Nicolas Stroppa. *Définitions et caractérisations de modèles à base d'analogies pour l'apprentissage automatique des langues naturelles*. PhD thesis, École Nationale Supérieure des Télécommunications, Paris, Novembre 2005.
- [SY05] N. Stroppa and F. Yvon. Analogical learning and formal proportions : Definitions and methodological issues. Technical Report ENST-2005-D004, École Nationale Supérieure des Télécommunications, June 2005.
- [TLFU05] Emmett Tomai, Andrew Lovett, Kenneth D. Forbus, and Jeffrey Usher. A structure mapping model for solving geometric analogy problems. In *Proceedings of the Annual Conference of the Cognitive Science Society*, volume 27, 2005.
- [Tur05] Peter D. Turney. Measuring semantic similarity by latent relational analysis. *Proceedings Nineteenth International Joint Conference on Artificial Intelligence (IJCAI-05)*, 05 :1136, 2005.
- [VKB05] Tamás Varga, Daniel Kilchhofer, and Horst Bunke. Template-based synthetic handwriting generation for the training of recognition systems. In *Proc. of 12th Conf. of the International Graphonomics Society*, pages 206–211, 2005.
- [WF74] R.A. Wagner and M.J. Fisher. The string-to-string correction problem. *JACM*, 1974.
- [WF05] Ian H. Witten and Eibe Frank. *Data Mining : Practical machine learning tools and techniques, 2nd Edition*. Morgan Kaufmann Publishers, 2005.
- [Wik05] Wikipédia. Analogie — wikipédia, l'encyclopédie libre, 2005. [En ligne ; Page disponible le 21-février-2006].
- [YSMD04] F. Yvon, N. Stroppa, L. Miclet, and A. Delhay. Solving analogical equations on words. Technical Report D005, Ecole Nationale Supérieure des Télécommunications, Paris, France, 2004.
- [Yvo] F. Yvon. *Des apprentis pour le traitement automatique des langues*. l'Université Pierre et Marie Curie - Paris VI. Habilitation à Diriger des Recherches.
- [Yvo97] F. Yvon. Paradigmatic cascades : a linguistically sound model of pronunciation by analogy. In *Proceedings of the 35th annual meeting of the Association for Computational Linguistics (ACL)*, Madrid, Spain, 1997.
- [Yvo99] F. Yvon. Pronouncing unknown words using multi-dimensional analogies. In *Proceeding of the European conference on Speech Application and Technology (Eurospeech)*, volume 1, pages 199–202, Budapest, Hungary, 1999.

- [Yvo03] F. Yvon. Finite-state transducers solving analogies on words. Technical Report D008, Ecole Nationale Supérieure des Télécommunications, Paris, France, 2003.

Table des figures

1.1	Le Parthénon à Athènes	21
1.2	L’Institut du Monde Arabe à Paris de Jean Nouvel	22
1.3	Traitement d’images par analogie	24
2.1	Réutilisation transformationnelle	31
2.2	Réutilisation dérivationnelle	31
2.3	Modèle analogique pour la conjugaison	32
2.4	Raisonnement à Partir de Cas	33
2.5	Analogie	33
3.1	parallélogramme de l’analogie	39
5.1	Illustration de la DA	59
5.2	SOLVANA : DAG des solutions approchées sur les séquences	72
5.3	Répartition des triplets	75
5.4	Surface de séparation : cas 1	76
5.5	Surface de séparation : cas 2	77
5.6	Automate de bruitage des séquences	79
5.7	Influence du bruitage et de la distance	80
5.8	DA vs. DAA	81
6.1	Distance entre ensemble	97
6.2	La géométrie de “éliminer” dans l’algorithme AESA.	99
6.3	Processus d’élimination dans FADANA.	105
6.4	Apport de FADANA en temps de calcul	108
6.5	Variation du taux de reconnaissance en fonction de k	111
7.1	Processus de génération de données synthétiques	116
7.2	Génération d’une nouvelle séquence	119
7.3	Organigramme du protocole expérimental	122
7.4	Les trois approches de génération	123
7.5	Taux de reconnaissance en mono-scripteur	124

Liste des tableaux

3.1	Table de l'opérateur \oplus	40
6.1	Configurations acceptables des classes des triplets	88
6.2	Exemple de classification par analogie	89
6.3	Analogies triviales sur les classes	93
6.4	Analogies élémentaires sur les attributs binaires	93
6.5	capacités d'élagage de FADANA	107
6.6	taux de reconnaissance de WAPC	111
7.1	Description des points d'ancrage.	116
7.2	Probabilité que la différence soit due au hasard	127

Liste des Algorithmes

1	Algorithme de Pondération.	94
2	N_Analogies	94
3	TypeAnalogie	95
4	AnalogieElémentaire	95
5	Algorithme LAESA.	101
6	Algorithme FADANA : initialisation.	103
7	Algorithme FADANA : sélection du couple le plus prometteur.	103
8	Algorithme FADANA : élimination des couples inutiles.	104
9	Algorithme FADANA : procédure principale.	104

Résumé

Apprentissage par proportion analogique

Les travaux présentés dans cette thèse s'inscrivent dans le cadre du raisonnement par analogie. Nous nous intéressons à la *proportion analogique* (*A est à B ce que C est à D*) et nous décrivons son utilisation et surtout son apport en apprentissage artificiel. Nous abordons tout d'abord le cas des proportions analogiques exactes. Ensuite, nous nous attachons plus particulièrement à définir une nouvelle notion, la *dissemblance analogique* qui mesure si quatre objets sont éloignés d'être en proportion analogique, et à l'appliquer en particulier à des séquences. Après avoir défini la proportion analogique, la dissemblance analogique et la résolution approchée d'équations analogiques, nous décrivons deux algorithmes qui rendent opérationnels ces notions d'apprentissage et de résolution pour des objets numériques ou symboliques et pour des séquences de ces objets. Nous montrons ensuite leur efficacité au travers de deux cas pratiques : le premier est l'apprentissage d'une règle de classification par proportion analogique pour des objets décrits par des attributs binaires et nominaux ; le second montre comment la génération de nouveaux exemples (par résolution approchée d'équations analogiques) peut aider un système de reconnaissance de caractères manuscrits à s'adapter très rapidement à un nouveau scripteur.

Mots-clés : Analogie, Proportion analogique, Dissemblance analogique, Apprentissage artificiel, Classification, Génération de séquences.

Abstract

Learning by Analogical Proportion.

The work presented in this thesis lies within the scope of reasoning by analogy. We are interested in the *analogical proportion* (*A is to B as C is to D*) and we describe its use and especially its contribution in machine learning. Firstly, we are interested in defining exact analogical proportions. Then, we tackle the problem of defining a new concept, the *analogical dissimilarity* which is a measure of how close four objects are from being in analogical proportion, including the case where the objects are sequences. After having defined the analogical proportion, the analogical dissimilarity and the approximate resolution of analogical equations, we describe two algorithms that make these concepts operational for numerical or symbolic objects and sequences of these objects. We show their use through two practical cases : the first is a problem of learning a classification rule on benchmarks of binary and nominal data ; the second shows how the generation of new sequences by solving analogical equations enables a handwritten character recognition system to rapidly be adapted to a new writer.

keywords : Analogy, Analogical proportion, Analogical Dissimilarity, Lazy learning, Classification, Sequence generation.