

**République Algérienne Démocratique et Populaire**  
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique  
**Université des Sciences et de la Technologie Houari Boumediene**  
Faculté d'Electronique et d'Informatique  
**Département Informatique**



# Mémoire de Master

Domaine Mathématiques et Informatique

## Spécialité

Systèmes Informatiques Intelligents

## Thème

**Classification et interprétation de scènes basées sur les objets**

### Présenté par :

- BENAMARA Soufian Przemyslaw
- DERARDJA Mohamed Elamine

### Proposé et encadré par :

- Mme N. BAHA
- Mr Lamine BENRAIS

### Devant le jury :

- Mme N. LAICHE (Président)
- Mr L. ABADA (Membre)

**Projet N° : 033/2020**

## **Remerciements et dédicaces**

Dans un premier lieu, je dédie toute ma gratitude à Allah, tout puissant, qui m'a donné la santé, la patience et la force pour tenir jusqu'au bout tout au long de mon parcours scolaire et universitaire, et surpasser tous mes obstacles en ne lâchant pas dans les moments difficiles.

Secondairement, bien que des simples remerciements ne sont pas en mesure de rembourser leur sacrifice, je m'adresse avec mes sincères remerciements à mon tout, mes parents, pour leur aide depuis mes premiers pas dans ce monde, leur soutien et encouragements dans mes bas, et leur fierté et présence dans mes hauts, je ne rendrai jamais votre sacrifice pour moi. Que dieu les protège et les gardera pour moi en pleine santé incha'Allah.

Je remercie mes deux promoteurs, madame Baha et monsieur Benrais, pour leur supervision et accompagnement tout au long de ce PFE. Un grand merci pour leur sérieux, même à distance, et leurs motivations afin de tirer le meilleur de moi-même dans ce travail. Merci à toutes leurs critiques constructives, remarques et conseils. J'apprécie leur esprit ouvert et acceptance des différentes idées et approches proposées par moi et mon binôme.

Je remercie mon binôme, Mohamed Elamine, pour tout son aide durant ces longs mois de travail, qu'il était toujours là pour discuter sur les idées proposées en donnant son point de vue et les éventuelles suggestions. C'était un plaisir de travailler avec toi.

Je remercie également les deux enseignants, membres de jury, madame Laiche et monsieur Abada d'avoir accepté d'évaluer notre modeste travail.

Je remercie mes proches, de ma famille et amis, qui m'ont soutenu et étaient là quand je leur avais besoin. Vous êtes ma force tant que je peux compter sur vous.

Je dirige mes remerciements, bien qu'ils n'aillent pas lire cela, à tous mes chats, animaux de compagnie, pour leur amour et affectation sans contrepartie. Vous êtes toujours là pour me calmer de déstresser.

Je dédie ce travail à mes chers parents, ma grande-mère maternelle décédée cette année (que dieu l'accueille dans son vaste paradis) qui a toujours cru en moi et voulait me voir parmi les meilleurs, et toute ma famille et amis.

**Soufian Przemyslaw**

« C'est en Dieu que je retrouve mon soutien, c'est en lui que je place ma confiance. Il est mon appui. A lui, louange et grâce. Par lui, assistance et protection. Louange à lui jusqu'à ce qu'il soit satisfait, louange à lui s'il est satisfait, et louange à lui après satisfaction »

Bien qu'un bon travail ne soit jamais établi sans un grand effort, je n'aurais jamais abouti à un tel résultat sans l'aide des personnes qui m'ont entouré. À travers ces petites phrases, je vais essayer de remercier ces personnes à leur juste valeur.

Mes sincères remerciements à Pr. Baha et M. Benrais d'avoir été avec nous tout au long de la réalisation de notre projet et pour tous leurs investissements personnels, tous leurs efforts, tous leurs conseils et plus particulièrement à la fin de cette expérience professionnelle.

Je remercie également le groupe de jury d'avoir accepté de soutenir notre travail de PFE Master.

Un grand merci à ma famille plus particulièrement à mes parents, mes grands-parents pour leur aide et leurs prières. Qu'Allah les récompense dans cette vie et dans l'au-delà.

Enfin, je n'oublie pas aussi à remercier tous mes amis pour toutes leurs conseils, leur aide, etc. et plus particulièrement mon binôme Soufian pour ses efforts particulièrement à la fin de cette expérience professionnelle.

**Mohamed Elamine**

# Sommaire

<b>Remerciements et dédicaces .....</b>	<b>i</b>
<b>Sommaire .....</b>	<b>iii</b>
<b>Liste des figures .....</b>	<b>v</b>
<b>Liste des tableaux .....</b>	<b>viii</b>
<b>Liste des algorithmes.....</b>	<b>ix</b>
<b>Introduction .....</b>	<b>10</b>
<b>Chapitre 1 – État de l'Art .....</b>	<b>12</b>
1.1    Introduction .....	12
1.2    État de l'art .....	12
1.2.1    État de l'art de la classification des scènes.....	12
1.2.1.1    Définition de la classification.....	12
1.2.1.2    Travaux associés.....	12
1.2.2    État de l'art de l'interprétation des scènes.....	23
1.2.2.1    Définition de l'interprétation.....	23
1.2.2.2    Travaux associés.....	23
1.3    Conclusion.....	31
<b>Chapitre 2 – Conception .....</b>	<b>32</b>
2.1    Introduction .....	32
2.2    Approches proposées pour la classification et l'interprétation des scènes .....	32
2.2.1    Classification des scènes basées sur les objets .....	32
2.2.1.1    Classifieur des scènes proposé .....	33
2.2.1.2    Enrichissement des scènes.....	40
2.2.1.3    Classification en catégories additionnelles.....	43
2.2.2    Interprétation des scènes basées sur les objets .....	44
2.2.2.1    Approche proposée pour l'interprétation des scènes .....	45
2.3    Conclusion.....	67
<b>Chapitre 3 – Implémentation et Résultats.....</b>	<b>68</b>
3.1    Introduction .....	68
3.2    Outils de test.....	68
3.3    Tests et résultats .....	68
3.3.1    Tests et résultats de la classification des scènes basées sur les objets.....	69
3.3.1.1    Datasets utilisés .....	69
3.3.1.2    Évaluation du classifieur des scènes proposé .....	70
3.3.1.3    Évaluation du classifieur de l'enrichissement des scènes.....	75
3.3.1.4    Évaluation des classifieurs des catégories additionnelles.....	78

3.3.2 Tests et résultats de l’interprétation des scènes basées sur les objets .....	80
3.3.2.1 Datasets utilisés .....	80
3.3.2.2 Évaluation de l’identificateur des relations .....	82
3.3.2.3 Évaluation des définitseurs des relations .....	82
3.3.2.4 Évaluation du générateur des descriptions .....	91
3.4 Application .....	93
3.4.1 Diagramme de cas d’utilisation .....	93
3.4.2 Outils de développement .....	94
3.4.2.1 Partie front-end.....	94
3.4.2.2 Partie back-end .....	94
3.4.3 Fonctionnalités de l’application .....	95
3.4.3.1 Données en entrée de l’application.....	95
3.4.3.2 Traitements du côté serveur de l’application.....	96
3.4.3.3 Données en sortie de l’application .....	96
3.5 Conclusion.....	97
<b>Conclusion.....</b>	<b>98</b>
1 Conclusion générale .....	98
2 Améliorations et perspectives .....	99
<b>Bibliographie.....</b>	<b>101</b>
<b>Annexe A .....</b>	<b>107</b>
A.1 Tests et résultats de la classification des scènes basées sur les objets .....	107
A.1.1 Évaluation du classifieur des scènes proposé .....	107
A.1.1.1 Tests sur le dataset MIT-Indoor.....	107
A.1.1.2 Tests sur le dataset Sun2012.....	113
A.1.2 Évaluation du classifieur de l’enrichissement des scènes.....	115
A.2 Tests et résultats de l’interprétation des scènes basées sur les objets .....	119
A.2.1 Évaluation de l’identificateur des relations .....	119
A.2.1.1 Test du LSTM.....	119
A.2.1.2 Test de l’identificateur des relations.....	119
<b>Annexe B .....</b>	<b>123</b>
B.1 Résultats détaillés des questionnaires sur la qualité des interprétations générées .....	123

# Liste des figures

<b>Figure 1.1</b> : Illustration de l'apprentissage du modèle bayésien hiérarchique proposé dans [38].....	13
<b>Figure 1.2</b> : Architecture globale proposée dans [42] .....	15
<b>Figure 1.3</b> : Différence entre une couche convolutionnelle linéaire et une couche <i>mlpconv</i> [47] .....	17
<b>Figure 1.4</b> : Architecture du <i>GCRBM</i> [52].....	18
<b>Figure 1.5</b> : Architecture d'un D-CNN [54].....	19
<b>Figure 1.6</b> : Exemple d'application de la technique RICAP [59].....	21
<b>Figure 1.7</b> : Exemple d'application de l'effacement aléatoire [59].....	22
<b>Figure 1.8</b> : Schéma global de l'approche de l'article [70] .....	25
<b>Figure 1.9</b> : Exemple d'application de l'approche de l'article [75] .....	26
<b>Figure 1.10</b> : Exemple d'application de l'approche de l'article [76] .....	27
<b>Figure 1.11</b> : Pipeline de l'approche de l'article [81].....	29
<b>Figure 1.12</b> : Différence entre le modèle d'un simple RNN et le modèle d'un m-RNN [82] .....	29
<b>Figure 1.13</b> : Architecture générale de l'article [84] .....	30
<b>Figure 2.1</b> : Exemple d'un fichier d'annotation manuelle d'une scène dans MIT-Indoor [12].....	33
<b>Figure 2.2</b> : Workflow du classifieur des scènes proposé .....	33
<b>Figure 2.3</b> : Exemple du calcul des importances des objets dans un dataset.....	36
<b>Figure 2.4</b> : Architecture détaillée d'un LSTM [15] .....	38
<b>Figure 2.5</b> : Exemple de classification d'une scène .....	39
<b>Figure 2.6</b> : Exemple de construction de l'ensemble d'apprentissage à partir d'une scène .....	42
<b>Figure 2.7</b> : Exemple de l'enrichissement d'une scène .....	42
<b>Figure 2.8</b> : Exemple de définition des catégories additionnelles dans un dataset.....	44
<b>Figure 2.9</b> : Workflow du processus de l'interprétation des scènes .....	45
<b>Figure 2.10</b> : Exemple de l'identification des relations d'une scène.....	48
<b>Figure 2.11</b> : Exemple des types des descriptions des relations unaires .....	49
<b>Figure 2.12</b> : Exemples de définition d'une relation unaire dans une scène .....	49
<b>Figure 2.13</b> : Exemple de définition des relations unaires en utilisant des classifieurs pré-entraînés..	50
<b>Figure 2.14</b> : Exemple des types des natures des relations binaires .....	50
<b>Figure 2.15</b> : Exemple sur la détermination de l'importance d'un objet dans une scène .....	52
<b>Figure 2.16</b> : Exemple sur la détermination de l'IOU entre deux objets dans une scène .....	52
<b>Figure 2.17</b> : Exemple sur la détermination des positions d'un objet par rapport à un autre dans une scène .....	53
<b>Figure 2.18</b> : Exemple sur la détermination de la taille d'un objet par rapport à un autre dans une scène .....	53
<b>Figure 2.19</b> : Exemple sur la détermination de la collision d'un objet par rapport à un autre dans une scène .....	54
<b>Figure 2.20</b> : Exemple sur la détermination de la distance qui sépare deux objets dans une scène .....	55
<b>Figure 2.21</b> : Structure d'un vecteur de caractéristiques pour une relation binaire.....	55
<b>Figure 2.22</b> : Exemple de définition d'une relation binaire.....	56
<b>Figure 2.23</b> : Exemple des différentes descriptions générées pour une scène .....	57
<b>Figure 2.24</b> : Exemple d'un graphe des relations d'une scène .....	58
<b>Figure 2.25</b> : Exemple de regroupement des sous-graphes isolés similaires.....	58
<b>Figure 2.26</b> : Exemple de la suppression des sous-graphes redondants .....	59
<b>Figure 2.27</b> : Exemple des phrases construites à partir des graphes de taille 1 .....	60
<b>Figure 2.28</b> : Exemple de phrases construites à partir des graphes de taille 2 .....	61
<b>Figure 2.29</b> : Exemple des phrases construites à partir des graphes de taille 3, avec exactement deux arcs entrants.....	62

<b>Figure 2.30</b> : Exemple des phrases construites à partir des graphes de taille 3, avec exactement deux arcs sortants .....	62
<b>Figure 2.31</b> : Exemple des phrases construites à partir des graphes de taille 3, avec exactement deux arcs, un entrant et un sortant.....	63
<b>Figure 2.32</b> : Exemple des phrases construites à partir des graphes de taille 3 avec trois arcs, ou de taille supérieure à 3 .....	63
<b>Figure 2.33</b> : Exemple du processus de l’interprétation d’une scène .....	66
<b>Figure 3.1</b> : Capture d’écran de la page principale de Google Colab.....	68
<b>Figure 3.2</b> : Exemple de fichier d’annotations obtenu en appliquant un détecteur d’objets sur une scène du MIT-Indoor complet.....	72
<b>Figure 3.3</b> : Exemple d’application du classifieur des scènes (entraîné sur MIT-Indoor complet).....	73
<b>Figure 3.4</b> : Exemple d’application du classifieur des scènes (entraîné sur Sun2012) .....	74
<b>Figure 3.5</b> : Exemple d’application des classifieurs en catégories additionnelles (entraînés sur MIT-Indoor partiel).....	79
<b>Figure 3.6</b> : Exemple des relations unaires et binaires d’une scène de VisualGenome .....	81
<b>Figure 3.7</b> : Exemple de l’annotation d’une scène de COCO dataset et ses interprétations associées.	81
<b>Figure 3.8</b> : Structure générale du dictionnaire global des attributs des objets .....	83
<b>Figure 3.9</b> : Exemple des ensembles des catégories définies pour deux objets.....	83
<b>Figure 3.10</b> : Précisions moyennes obtenues en fonction de la taille de l’image en entrée sur deux classifieurs des relations unaires .....	84
<b>Figure 3.11</b> : Exemple de l’ensemble d’apprentissage pour un classifieur d’une relation unaire .....	85
<b>Figure 3.12</b> : Taux des précisions obtenus pour certains classifieurs des relations unaires sans / avec la normalisation.....	85
<b>Figure 3.13</b> : Architecture des classifieurs des relations unaires.....	86
<b>Figure 3.14</b> : Liste des classifieurs des relations unaires entraînés .....	86
<b>Figure 3.15</b> : Liste des catégories du définitisseur des relations binaires .....	88
<b>Figure 3.16</b> : Précision obtenue (avec le définitisseur des relations binaires utilisant un CNN) en fonction de la taille de l’image .....	89
<b>Figure 3.17</b> : Précision obtenue (avec le définitisseur des relations binaires utilisant un NN) en fonction de la fonction d’activation et l’optimiseur.....	90
<b>Figure 3.18</b> : Précision obtenue (avec le définitisseur des relations binaires utilisant un NN) en fonction du nombre des couches et leurs neurones.....	90
<b>Figure 3.19</b> : Exemple d’un questionnaire sur la qualité des interprétations générées.....	92
<b>Figure 3.20</b> : Résumé des résultats obtenus à partir des questionnaires sur la qualité des interprétations générées.....	93
<b>Figure 3.21</b> : Diagramme de cas d’utilisation de l’application implémentée .....	94
<b>Figure 3.22</b> : Exemple des données en entrée de l’application implémentée.....	95
<b>Figure 3.23</b> : Exemple de l’affichage en sortie pour la tâche de la classification de l’application implémentée .....	96
<b>Figure 3.24</b> : Exemple de l’affichage en sortie pour la tâche de l’interprétation de l’application implémentée .....	97
<b>Figure A.1</b> : Précision du LSTM obtenue (sur MIT-Indoor partiel) en fonction de l’optimiseur et la fonction d’activation.....	107
<b>Figure A.2</b> : Précision obtenue (de classifieur de scènes, sur MIT-Indoor partiel) en fonction de $\beta$ .	108
<b>Figure A.3</b> : Précision obtenue (de classifieur de scènes, sur MIT-Indoor partiel) en fonction de N	109
<b>Figure A.4</b> : Précision du classifieur proposé (sur MIT-Indoor partiel) en fonction des meilleurs ensembles du $\beta$ et N .....	110
<b>Figure A.5</b> : Précision du LSTM obtenue (sur MIT-Indoor complet) en fonction de l’optimiseur et la fonction d’activation.....	110
<b>Figure A.6</b> : Précision obtenue (de classifieur de scènes, sur MIT-Indoor complet) en fonction de $\beta$ .....	111
<b>Figure A.7</b> : Précision obtenue (de classifieur de scènes, sur MIT-Indoor complet) en fonction de N .....	112

<b>Figure A.8</b> : Précision du classifieur proposé (sur MIT-Indoor complet) en fonction des meilleurs ensembles du $\beta$ et N .....	112
<b>Figure A.9</b> : Précision du LSTM obtenue (sur Sun2012) en fonction de l'optimiseur et la fonction d'activation.....	113
<b>Figure A.10</b> : Précision obtenue (de classifieur de scènes, sur Sun2012) en fonction de $\beta$ .....	114
<b>Figure A.11</b> : Précision obtenue (de classifieur de scènes, sur Sun2012) en fonction de N .....	114
<b>Figure A.12</b> : Précision du classifieur proposé (sur Sun2012) en fonction des meilleurs ensembles du $\beta$ et N .....	115
<b>Figure A.13</b> : Précision obtenue (de l'enrichissement des scènes, sur MIT-Indoor partiel) en fonction de T .....	115
<b>Figure A.14</b> : Précision obtenue (de l'enrichissement des scènes, sur MIT-Indoor partiel) en fonction de NFreq.....	116
<b>Figure A.15</b> : Précision du classifieur de l'enrichissement des scènes (sur MIT-Indoor partiel) en fonction des meilleurs ensembles du T et NFreq .....	117
<b>Figure A.16</b> : Précision obtenue (de l'enrichissement des scènes, sur MIT-Indoor complet) en fonction de T .....	117
<b>Figure A.17</b> : Précision obtenue (de l'enrichissement des scènes, sur MIT-Indoor complet) en fonction de NFreq.....	118
<b>Figure A.18</b> : Précision du classifieur de l'enrichissement des scènes (sur MIT-Indoor complet) en fonction des meilleurs ensembles du T et NFreq .....	118
<b>Figure A.19</b> : Précision du LSTM (de l'identificateur des relations) obtenue en fonction de l'optimiseur et la fonction d'activation.....	119
<b>Figure A.20</b> : Précision obtenue (de l'identificateur des relations) en fonction de $\beta$ .....	120
<b>Figure A.21</b> : Précision obtenue (de l'identificateur des relations) en fonction de N .....	120
<b>Figure A.22</b> : Précision de l'identificateur des relations en fonction des meilleurs ensembles du $\beta$ et N .....	121
<b>Figure B.1</b> : Résultats détaillés des questionnaires sur la qualité des interprétations générées.....	129

# Liste des tableaux

<b>Tableau 2.1 :</b> Valeurs de la caractéristique "Importance" et leurs significations.....	51
<b>Tableau 2.2 :</b> Valeurs de la caractéristique "IOU" et leurs significations.....	52
<b>Tableau 2.3 :</b> Valeurs de la caractéristique "Positions" et leurs significations .....	53
<b>Tableau 2.4 :</b> Valeurs de la caractéristique "Taille" et leurs significations.....	53
<b>Tableau 2.5 :</b> Valeurs de la caractéristique "Collision" et leurs significations .....	54
<b>Tableau 2.6 :</b> Valeurs de la caractéristique "Distance" et leurs significations .....	54
<b>Tableau 3.1 :</b> Meilleure combinaison des valeurs des paramètres du classifieur proposé (sur MIT-Indoor partiel) avec la précision associée .....	70
<b>Tableau 3.2 :</b> Comparaison du taux de précision obtenu (sur MIT-Indoor partiel) avec [14] .....	71
<b>Tableau 3.3 :</b> Meilleure combinaison des valeurs des paramètres du classifieur proposé (sur MIT-Indoor complet) avec la précision associée.....	72
<b>Tableau 3.4 :</b> Comparaison du taux de précision obtenu (sur MIT-Indoor complet) avec des travaux précédents.....	73
<b>Tableau 3.5 :</b> Meilleure combinaison des valeurs des paramètres du classifieur proposé (sur Sun2012) avec la précision associée.....	74
<b>Tableau 3.6 :</b> Meilleure combinaison des valeurs des paramètres du classifieur de l'enrichissement des scènes (sur MIT-Indoor partiel) avec la précision associée .....	76
<b>Tableau 3.7 :</b> Précision du meilleur modèle de la classification en catégories standard (sur MIT-Indoor partiel) avec/sans l'application de l'enrichissement des scènes .....	76
<b>Tableau 3.8 :</b> Meilleure combinaison des valeurs des paramètres du classifieur de l'enrichissement des scènes (sur MIT-Indoor complet) avec la précision associée .....	77
<b>Tableau 3.9 :</b> Précision du meilleur modèle de la classification en catégories standard (sur MIT-Indoor complet) avec/sans l'application de l'enrichissement des scènes .....	77
<b>Tableau 3.10 :</b> Précision des classifieurs des catégories additionnelles sur MIT-Indoor partiel .....	79
<b>Tableau 3.11 :</b> Précision des classifieurs des catégories additionnelles sur MIT-Indoor partiel .....	79
<b>Tableau 3.12 :</b> Meilleure combinaison des valeurs des paramètres pour les deux modèles des définitseurs des relations binaires (CNN et NN) .....	91

## Liste des algorithmes

<b>Algorithme 2.1 :</b> Cration des structures de correspondance "objet - frquence" pour un dataset .....	34
<b>Algorithme 2.2 :</b> Calcul des importances des objets d'un dataset.....	35
<b>Algorithme 2.3 :</b> Calcul des importances des objets dans une scne .....	37
<b>Algorithme 2.4 :</b> Apprentissage d'un LSTM pour l'enrichissement.....	41
<b>Algorithme 2.5 :</b> Gnration de l'information sur les catgories d'une scne .....	64

# Introduction

L'intelligence artificielle a connu un progrès impressionnant ces dernières années. Restreinte auparavant aux recherches scientifiques uniquement, de nos jours son application se propage de plus en plus dans les tâches pratiques. L'une de ces dernières est la vision par ordinateur, et plus particulièrement sa devise : l'image. L'image (ou scène) numérique n'est qu'un ensemble de points bruts (appelés : pixels) coloriés et juxtaposés l'un devant l'autre dans une grille sans aucune information significative pour l'ordinateur. L'intérêt de traiter une telle structure est -justement- d'extraire le sens contenu dans cette image, en imitant par conséquence le système visuel humain.

Le traitement d'images est une vaste discipline contenant une panoplie de sections hiérarchisées où chaque section s'intéresse aux détails plus fins que celle qui la précède, parmi les sections fondamentales est la classification. Cette dernière s'intéresse à l'affectation d'une scène à une classe bien définie (par exemple : aéroport, école, magasin, etc.) en se basant sur des caractéristiques propres à elle. Ces "features" peuvent être de bas niveau (contours, couleurs, textures, etc.) ou -en opposé- de haut niveau (objets nommés, contenus dans la scène). Pour ce deuxième type (notre cas d'étude), ses éléments (les objets) sont issus soit grâce à une identification (annotation) manuelle ou bien de la part d'un détecteur d'objets automatique. Bien que la procédure de classification des images semble basique et triviale, elle pose une difficulté non-négligeable qui est due au très grand nombre des objets potentiels, l'imprécision des noms des objets à cause des fautes humaines ou la basse performance des détecteurs automatiques ou le choix de la méthode de classification (plusieurs algorithmes sont proposés en l'IA, chacun d'eux est adapté à une certaine catégorie d'application).

Cependant, l'assignation correcte d'une image (ou scène) à un groupe particulier (classe) partageant plus ou moins les mêmes caractéristiques n'apporte qu'une seule connaissance (donnée supplémentaire) sur une image, ainsi cette dernière restera toujours décrite par un ensemble d'annotations (tags), ce qui est utile en lui-même et d'ailleurs utilisé dans diverses applications, la recherche d'informations par exemple. Cependant, l'état actuel ne permet pas de comprendre l'image étudiée de manière exacte qui éliminera le problème d'ambiguïté, prenons -par exemple- une image annotée avec les mots {"bleu", "ciel", "voiture"}, cela pourrait représenter "voiture bleue" ou "ciel bleu". D'où le recours à un niveau plus profond de l'analyse d'une image : l'interprétation. Cette dernière consiste à décrire textuellement une scène, en mettant en évidence les actions phares faites par les objets dans cette scène.

Notre travail s'appuie sur les deux concepts précédents (classification et interprétation des scènes). Pour chaque concept, nous présentons nos propres approches permettant d'aboutir au résultat attendu derrière ce concept.

Pour la classification, nous définissons trois variantes : Classification en catégories standard (étant donné une scène décrite par les objets qu'elle contient, il s'agit de déterminer la catégorie la plus plausible à laquelle cette scène appartient), enrichissement des scènes (révéler les objets manquants de la scène à classifier, afin d'augmenter la probabilité de la prédiction de la vraie catégorie de la scène) et la classification en catégories additionnelles (définition des ensembles des catégories à valeurs restreintes, afin d'extraire davantage informations sur la scène). Les résultats de la classification sont utilisés dans l'interprétation.

Pour l'interprétation, nous nous sommes basés sur un dataset qui contient, pour chaque scène, l'ensemble des relations entre ses objets. Afin d'interpréter une scène donnée, nous identifions ses relations (i.e. Nous déterminons quels objets sont en relation), puis nous les définissons (i.e. Nous attribuons un type bien défini à chaque relation identifiée), et enfin, nous générerons la description de la scène en exploitant un graphe des relations et en employant les résultats de la classification.

Ce mémoire est divisé en trois chapitres, qui se présentent comme suit :

- Chapitre 1 – État de l’art : Présentation de la définition formelle et les différents travaux liés aux deux concepts, la classification et l’interprétation.
- Chapitre 2 – Conception : Présentation détaillée du fonctionnement des différentes approches proposées.
- Chapitre 3 – Implémentation et résultats : Évaluation des approches proposées via des tests des paramètres et comparaison avec les travaux similaires, ainsi qu’une présentation de l’application développée

# Chapitre 1 – État de l'Art

## 1.1 Introduction

À travers de ce chapitre, nous détaillons les concepts initiés dans l'*Introduction* : La classification et l'interprétation des scènes. La présentation de chacun des deux concepts sera accompagnée d'une définition précise sur le sens et l'intérêt de s'intéresser à un tel concept dans notre travail. De plus, nous mettons un point sur les avancées dans la recherche scientifique pour chaque concept, en catégorisant les différents travaux (que nous résumons et expliquons), tout en citant les spécificités de chaque catégorie et ses éventuels avantages et inconvénients.

Nous passons, dans ce qui suit, à la partie phare de ce chapitre, à savoir : L'état de l'art.

## 1.2 État de l'art

Comme précisé dans l'introduction, dans cette partie, nous passons en revue de la classification et l'interprétation successivement. Pour chaque concept, nous présentons sa définition et quelques travaux associés.

### 1.2.1 État de l'art de la classification des scènes

La section courante est consacrée pour la classification des scènes : Sa définition et les travaux associés.

#### 1.2.1.1 Définition de la classification

La classification d'une scène est l'opération qui consiste à attribuer une classe à une image. Elle a beaucoup d'utilité, du fait qu'elle permet de donner une description des objets constituants l'image.

La classification de scène peut être trouvée sous différents noms, par exemple, dans certains articles elle est sous le nom "la reconnaissance de scène", dans certains autres, elle est sous le nom "la compréhension de scène", dans certains autres, elle est sous le nom "la catégorisation de scène".

Pour faire une classification, nous devons construire un modèle qui pour une ou plusieurs entrées, il retourne une réponse consistant en la classe attribuée à l'image visée. Ce modèle peut être basé sur l'un des deux critères : Le premier est l'ensemble des objets contenus dans l'image, ces objets sont obtenus en appliquant des filtres d'objet. L'autre est des caractéristiques propres extraites à partir de l'image.

L'étape de construction du modèle nécessite une phase d'apprentissage, où une base d'images est utilisée, cet apprentissage peut être supervisé ou non tout dépend de la technique utilisée.

Une fois le concept de la classification des scènes éclairci en présentant sa définition. Nous continuons son état de l'art par la présentation des travaux associés et leur catégorisation.

#### 1.2.1.2 Travaux associés

Comme mentionné dans la définition, les modèles de classification des scènes sont divisés en deux, suivant les propriétés des images exploitées : Modèles basés sur des caractéristiques propres et ceux basés sur les objets. Nous débutons, dans ce qui suit, par la présentation des travaux associés à la première catégorie des modèles (basés sur des caractéristiques propres).

## 1) Modèles basés sur des caractéristiques propres

L'idée de ces modèles consiste à extraire certaines caractéristiques propres (*low-level features*) de l'image permettant en premier lieu de définir de quelle scène elle s'agit et ensuite différencier les scènes dans le cas où certaines ont des caractéristiques communes (discrimination). De ce fait, plusieurs techniques ont été proposées distinguant la manière de voir ces caractéristiques et les algorithmes de classification utilisés. Nous pouvons classifier ces techniques en trois classes [37].

### a - Méthodes basées sur les caractéristiques artisanales (*Hancrafted Features*)

Ces méthodes nécessitent l'extraction des caractéristiques basées sur les compétences d'ingénierie humaine telles que la couleur, la forme, la résolution spectrale, la résolution spatiale, la taille, la texture, etc. Pour faire la classification, ces méthodes se basent sur ces caractéristiques individuellement ou combinées.

L'article [38] a proposé un modèle bayésien hiérarchique pour la reconnaissance des scènes naturelles, les chercheurs avaient considéré initialement une base de formes appelée "*form codebook*", pour la construction du modèle, ils extraient de chaque image contenue dans leur base des images, les formes "*codewords*" contenues dans "*form codebook*", puis ils donnaient ça comme entrée à l'algorithme bayésien hiérarchique qui à son tour retourne un modèle permettant la classification des scènes, dans ce modèle les caractéristiques sont vues comme des formes prédéfinies contenues dans l'image. La figure 1.1 illustre l'apprentissage du modèle proposé.

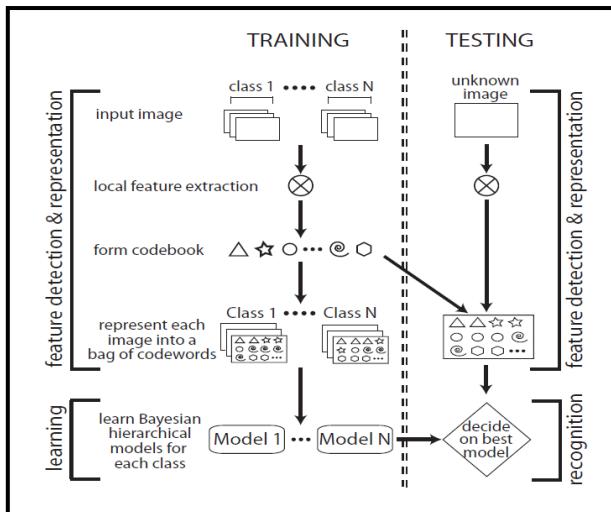


Figure 1.1 : Illustration de l'apprentissage du modèle bayésien hiérarchique proposé dans [38]

Les membres de l'équipe du [39] ont développé un framework appelé *modèle de classification en cascade* (CCM) où des classificateurs de pointe pour un ensemble de tâches connexes sont combinés pour améliorer les performances de certaines/toutes les tâches. Les tâches qu'ils avaient considérées sont : La catégorisation de la scène, la détection d'objets, la segmentation d'images multi-classes et la reconstruction 3D. Le framework crée plusieurs instantiations de chaque classificateur et les organise en niveaux tel que les modèles du premier niveau apprennent de manière isolée (ne nécessitant aucun résultat d'autres classificateurs), traitant les données pour produire les meilleures classifications, compte tenu uniquement des fonctionnalités (caractéristiques) d'instance brutes. Les niveaux inférieurs acceptent en entrée les fonctionnalités de l'instance de données (*features from the data instance*), ainsi que les fonctionnalités calculées à partir des classifications de sortie des modèles du niveau précédent.

Les chercheurs de l'article [40] ont proposé une méthode de reconnaissance des catégories des scènes basées sur une correspondance géométrique globale approximative. Cette technique fonctionne en partitionnant l'image en sous-régions de plus en plus fines et en calculant des histogrammes des caractéristiques locales trouvées à l'intérieur de chaque sous-région. La "*pyramide spatiale*" qui en résulte est une extension simple et efficace en termes de calcul d'une représentation d'images sans sac

de fonctionnalités, et elle montre des performances considérablement améliorées pour les tâches de catégorisation de scènes difficiles.

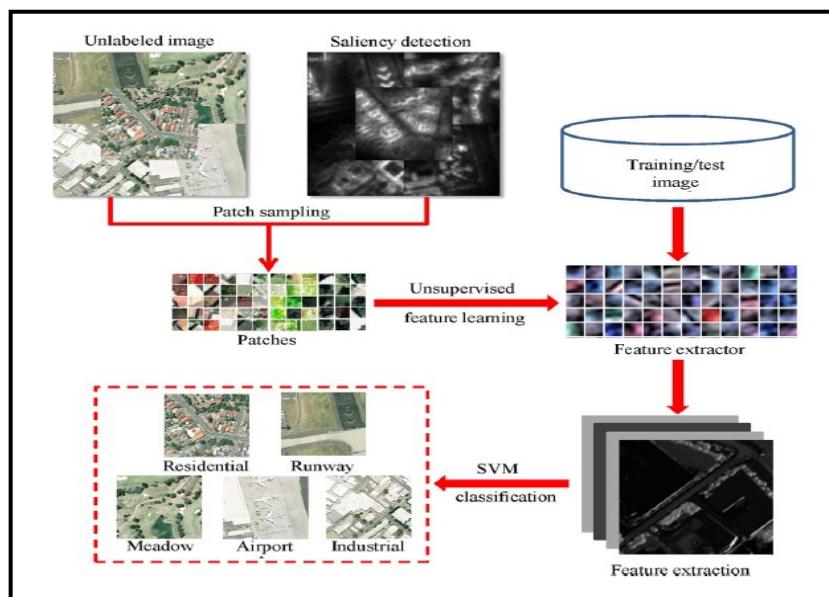
L'approche [41] propose un modèle de sujet latent discriminant (*discriminative latent topic model*) pour la reconnaissance de scène en modélisant la disposition spatiale globale (*global spacial layout*) spécifique à la catégorie des différents éléments de la scène et le renforcement de la cohérence visuelle dans des régions locales uniformes, ensuite en appliquant l'inférence Bayésienne pour la tâche de reconnaissance.

Le principal inconvénient de ces méthodes réside dans la difficulté d'obtenir les caractéristiques discriminantes. Pour cela, d'autres méthodes, basées sur l'apprentissage automatique, ont vu le jour.

### **b - Méthodes basées sur l'apprentissage non-supervisé des caractéristiques (Unsupervised Feature Learning)**

Les approches utilisant ces méthodes se basent sur des caractéristiques qui sont extraites automatiquement de l'image. Bien que ces méthodes soient capables de donner des résultats meilleurs par rapport à ceux donnés par la classe précédente (basées sur les caractéristiques artisanales), elles n'arrivent pas à donner les performances de l'état de l'art car elles n'étaient pas en mesure de fournir les meilleures caractéristiques discriminantes entre les classes en raison du manque d'informations sémantiques fournies par le label de catégorie.

Une variante de cette méthode a été utilisée dans [42], le but étant de classifier les images (aériennes et satellitaires) de haute résolution (abrégées VHR, "*Very High Resolution*") en utilisant un framework d'apprentissage des caractéristiques non-supervisé, et cela par le biais d'un algorithme de détection de saillance. Ce dernier procède par l'extraction d'un ensemble représentatif des "patches" à partir des régions *saillantes* (contenantes l'information la plus représentative) de l'image, en utilisant une approche originale, nommée "*saillance contextuelle*", qui indique la probabilité de la région saillante de l'image qui est "*distinctive*" en tenant compte de son environnement (entourage) local et global. Ensuite, ces patches de données non-étiquetées sont exploitées par une méthode d'apprentissage des caractéristiques (fonctionnalités) non-supervisée (en utilisant une variante des réseaux de neurones, nommée "*auto-encodeurs*") pour apprendre un ensemble d'extracteurs des caractéristiques qui est robuste et efficace. Finalement, ces extracteurs sont combinés avec un SVM linéaire (implémentation LIBSVM, réglage des paramètres : *fivefold cross-validation*), en utilisant une méthode nommée "*dropout*" avec l'augmentation des données pour réduire le surapprentissage. Les résultats obtenus sont satisfaisants, une exactitude de 82.72 % est notée pour *UC-Merced dataset* et 90.78 % pour le *Sydney dataset*. L'architecture globale proposée est illustrée dans la figure 1.2.



**Figure 1.2 : Architecture globale proposée dans [42]**

Une autre approche a fait l'objet d'étude dans [43], les auteurs du proposent l'utilisation d'une préformation gourmande non supervisée par couche couplée à un algorithme très efficace pour l'apprentissage non supervisé de fonctionnalités rares, ils illustrent le pouvoir expressif des représentations extraites dans plusieurs scénarios : classification des scènes aériennes, classification de l'utilisation des sols en très haute résolution (VHR) et classification de l'occupation du sol à partir d'images multispectrales et hyper-spectrales. Leur algorithme proposé surpassé l'analyse principale des composants principaux (PCA) et son homologue du noyau (kPCA), ainsi que les algorithmes de pointe de la classification aérienne, tout en étant extrêmement efficace sur le plan informatique pour apprendre les représentations des données.

Même en s'appuyant sur l'apprentissage non-supervisé, les caractéristiques extraites n'étaient pas suffisantes pour représenter correctement une scène. Cette insuffisance est dû, entre-autres, à la légèreté des architectures utilisées. Pour cela, une fois la puissance de calcul nécessaire disponible, les recherches se sont tournées vers les méthodes basées sur l'apprentissage profond.

### **c - Méthodes basées sur l'apprentissage profond (deep learning) des caractéristiques**

Ces méthodes se sont avérées efficaces pour extraire les informations cachées et les caractéristiques discriminantes des images. Bien qu'il existe de nombreuses architectures d'apprentissage profond, l'architecture CNN est celle la prédominante.

Les *Convolutional Neural Networks* (CNNs) représentent une architecture largement utilisée dans la reconnaissance des images et leur classification et la détection des objets. Leur particularité est qu'ils agissent sur des images quasi-brutes (avec un minimum de pré-traitement).

Un modèle CNN (ou ConvNet) typique reçoit en entrée une image (matrice de pixels) sur laquelle une série de filtres de convolution est appliquée, chacun d'eux vise à extraire (mettre en évidence) des caractéristiques de l'image concernée. Les matrices résultantes sont aplatis (converties en vecteurs) et introduites dans un réseau de neurones *feedforward* habituel.

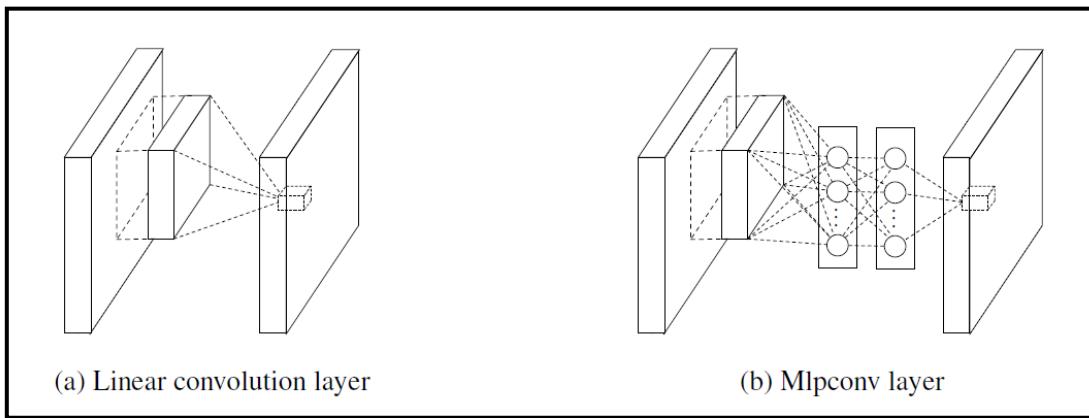
Différentes variantes d'architectures ont été conçues dans la recherche visant à minimiser le taux d'erreur.

L'approche présentée dans [44], nommée "*AlexNet*", vise à exploiter un dataset volumineux (millions des images annotées) nommé "*ImageNet*". Contrairement aux petits datasets (dizaine de milliers des images) dans lesquels la qualité de reconnaissance est assez bonne, leurs équivalents grands nécessitent une architecture plus adaptée pour gérer la complexité du problème. Pour cela, un modèle basé sur les réseaux neuronaux convolutifs (CNNs) a été implémenté, leur capacité est contrôlée en variant la largeur (nombre de neurones) et la longueur (nombre de couches), en détails, 60 millions de paramètres et 650 000 neurones répartis sur 5 couches convolutives. Ainsi, l'architecture proposée, comparée aux réseaux neuronaux standards, est facile à entraîner (peu de connexions et paramètres) avec quasiment la même performance. Les résultats obtenus (taux d'erreur 37,5 % contre 45,7 % [meilleur résultat déclaré dans une compétition avec le dataset ILSVRC-2010]) montrent de façon claire le potentiel du modèle proposé. Ainsi, les expérimentations suggèrent que ces résultats peuvent être améliorés en utilisant un GPU plus rapide (pour ce travail, l'apprentissage a été fait avec deux GTX 580 3Go, ce qui a pris entre 5 et 6 jours de calcul continu) et/ou un dataset plus large.

Le travail [45] cherchait à examiner la profondeur des CNNs et leur impact sur les résultats obtenus. Ainsi, les autres paramètres de l'architecture ont été fixés, tout en incrémentant progressivement le nombre des couches convolutionnelle du CNN de 8 (minimum, avec 113 millions de paramètres) à 16 (maximum, avec 144 millions de paramètres, abrégé *VGG-16*). Cela a été possible grâce à l'utilisation de très petits (3\*3) [la plus petite taille permettant de capturer la notion gauche/droite, haut/bas, centre] filtres convolutionnels dans toutes les couches. L'architecture utilisée, invariante dans tous les tests, recevait en entrée des images de taille fixe (RGB, 224\*224), avec le seul pré-traitement : soustraction de la valeur RGB moyenne, calculée sur l'ensemble d'apprentissage, pour chaque pixel. Les résultats obtenus ont permis de noter le taux d'erreur minimal dans ILSVRC-2014.

Dans [46], une mise au point sur la compréhension du fonctionnement interne des CNNs a été faite. En effet, malgré le succès éblouissant d'*AlexNet*, une ignorance sur le comportement interne de ces modèles complexes ne permettait pas de les améliorer d'une manière scientifique, le développement de meilleurs modèles se restreignait aux essais successifs. De ce fait, une nouvelle technique de visualisation du traitement appliquée dans les différentes couches intermédiaires a été proposée dans le but de comprendre le fonctionnement exact des CNNs (perçues comme des boîtes noires), et par conséquence, améliorer leur performance. De plus, cette méthode "diagnostique" permet de révéler les parties importantes de l'image pour la classification, et cela en occultant certaines parties de l'image en entrée. La visualisation des CNNs a été faite grâce à *Deconvnet*, qui consiste à réappliquer les actions faites sur une image (ou une partie seulement) au niveau d'une couche de façon inverse en mappant les caractéristiques aux pixels, récupérant ainsi l'image dans son état intermédiaire. Les résultats de cette analyse ont abouti en diminution du taux d'erreur par rapport au modèle *AlexNet* original.

Le travail présenté dans [47] a résulté par le développement d'une nouvelle structure nommée "*Network In Network*" (*NIN*). Cette dernière introduit des mini réseaux de neurones standards entre les différentes couches intermédiaires d'un CNN, le but étant de diminuer le niveau d'abstraction. En effet, les filtres de convolution dans CNN sont des *modèles linéaires généralisés (GLM)* qui présentent un bas niveau d'abstraction (la variation d'une caractéristique par rapport aux autres variantes du même concept). Par contre, la performance des GLMs dépend de la séparation linéaire des concepts eux-mêmes, dans le cas contraire (non linéairement séparables) elles sont inefficaces. Pour cela, dans *NIN*, les GLMs sont remplacés par des micro-réseaux qui permettent d'approximer toute sorte de fonction (avec les bons paramètres, nombre de couches/neurones adéquat), un tel micro-réseau entre deux couches est appelé "*mlpconv*". Les résultats des tests montrent une légère diminution du taux d'erreur sur le dataset *CIFAR-10* (~ 1% de gain par rapport à la meilleure configuration des CNNs) et sur *CIFAR-100* (~ 3% de gain). La figure 1.3 montre la différence entre une couche convolutionnelle linéaire standard (à gauche) et celle après l'introduction du *mlpconv* (à droite).



**Figure 1.3 : Différence entre une couche convolutionnelle linéaire et une couche *mlpconv* [47]**

L'idée derrière l'article [48] est de proposer un simple algorithme de classification des images basé sur la détection des objets en combinant des méthodes existantes. L'approche procède en deux phases, la première est de diviser l'image en entrée en régions sémantiques indépendantes, chacune d'elles désigne une zone jugée importante (porteuse d'information), nommée "*proposition*", qui affectera le résultat de la classification, le nombre approximatif de ces régions est 2000. Ensuite, pour chaque proposition, un vecteur de caractéristiques (*features*) est généré en utilisant les CNNs, cependant, une telle approche souffre d'un manque de données annotées, qui sont insuffisantes pour l'entraînement d'un CNN large (adapté à ce cas de l'utilisation), ainsi, un pré-entraînement non-supervisé est appliqué. Les caractéristiques générées sont injectées dans un *SVM* (*Support Vector Machine*) pour classifier la proposition. La méthode décrite est nommée : R-CNN (Regions with CNN features).

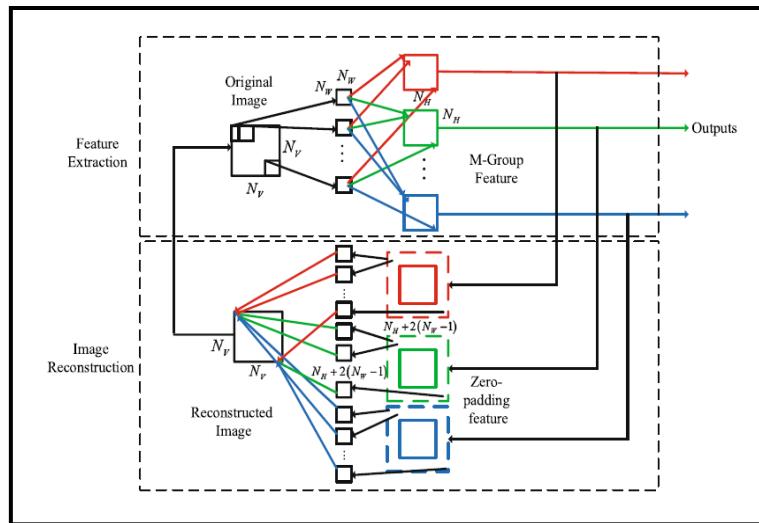
Le défi affronté dans [49] concerne l'amélioration de l'utilisation des ressources de calcul disponibles dans le CNN. L'équipe de Google visait à proposer une nouvelle architecture avec son algorithme associé qui diminuera le taux d'erreur sur le dataset *ImageNet* (par rapport aux résultats précédents) en

employant une machine avec des caractéristiques similaires. Le modèle conçu, nommé "*GoogLeNet*", utilisait 12 fois moins de paramètres que celui du [42].

Dans [50], une approche inverse pour le problème de classification des images a été développée, il s'agit d'une méthode d'apprentissage d'une architecture à partir d'un dataset. Pour surpasser la difficulté de conception d'une architecture adéquate à un dataset donné, l'idée est de rechercher la meilleure architecture pour un dataset réduit (de petite taille) et de l'adapter à ses équivalents de grande taille. Pour le faire, la couche convolutionnelle élue comme "meilleure" pour un petit dataset (par exemple, CIFAR-10) est appliquée (grâce à une méthode nommée NAS [*Neural Architecture Search*]) à un grand dataset (par exemple, ImageNet) en empilant ensemble plusieurs copies de cette couche (ou "cellule"), chacune avec ces propres paramètres (poids), cette transformation est assurée par la création d'un espace de recherche, nommé "*the NASNet search space*", ayant une complexité indépendante de la profondeur du réseau et la taille de l'image en entrée.

Dans [51], les chercheurs ont proposé l'utilisation d'une machine à vecteurs de support (SVM) en collaboration avec DeCAF (*DEep Convolutional Activation Feature*) pour classifier les scènes. En premier temps, ils extraient des caractéristiques de la scène, ensuite ils font l'apprentissage du modèle en utilisant LSVM (*Linear Support Vector Machine*).

Le travail du [52] propose un nouveau modèle génératif appelé "*Gaussian-Bernoulli based Convolutional Restricted Boltzmann Machines (GCRBM)*" en combinant les CNN avec les GRBM. De plus, en empilant le GCRBM proposé, un modèle profond correspondant est également proposé, qui s'appelle "*Gaussian-Bernoulli based Convolutional Deep Belief Network (GCDBN)*". Tirant parti des avantages de CNN et de GRBM, cette architecture profonde de GCDBN peut extraire des caractéristiques significatives à travers une image en taille réelle par des filtres de convolution génératifs, ce qui peut réduire un certain nombre de poids de connexion et peut apprendre plus efficacement les informations spatiales des correctifs d'image voisins. Ils prouvent que leurs méthodes proposées sont meilleures que RBM et ses modèles dérivés et ceux de SPM et CA-TM en termes de taux de précision. Cependant, le CaBow et le CNN (fusion) surpassent leur méthode GCDBN proposée, cela pourrait être dû au fait que le CaBow et le CNN ont été préformés sur PASCAL07 et que le GCDBN n'avait pas de procédures de pré-entraînement. De plus, en terme du temps, leur GCDBN a un coût inférieur à celui du CNN (fusion). L'architecture du *GCRBM* est présentée dans la figure 1.4.

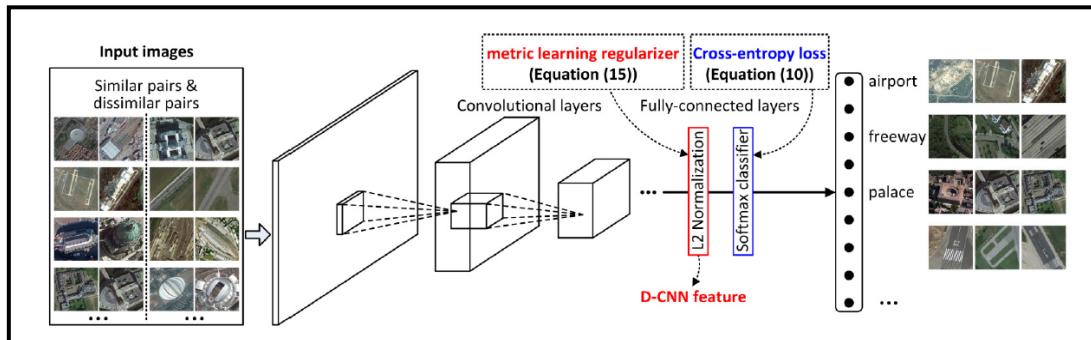


**Figure 1.4 : Architecture du *GCRBM* [52]**

En parallèle avec les images naturelles, celles de la télédétection ont eu aussi leur espace de manœuvre. Ces dernières, étant récupérées dans le but d'observer les différents aspects/phénomènes de notre planète, générées sous différents types et résolutions (spatiale, spectrale et temporelle) en utilisant différents dispositifs (satellites, radars, etc.). La classification de telles images a de nombreuses applications pratiques : détection des risques naturels, *LULC* (*Land Use and Land Cover*), détection des objets géospatiaux, récupération d'images géographiques, cartographie de la végétation, la surveillance

de l'environnement et l'urbanisme. Au début, les images de la télédétection étaient floues (petites, de faible résolution), la plupart des analyses se restaient à une étude statistique des pixels composants de l'image en question, ainsi, les pixels étaient considérés comme des unités isolées. Ensuite, avec l'émergence des dispositifs sophistiqués, l'apparition des datasets adéquats et l'avancée des techniques de la classification, notamment les CNNs, la recherche dans ce domaine a pris l'ampleur en présentant des résultats éblouissants issues de la modification/adaptation des algorithmes/architectures des modèles utilisés pour classifier les images naturelles [53].

Un travail traitant les images de la télédétection a été présenté dans [54], une modification des CNNs habituels a été faite pour tenir compte des deux critères "diversité intra-classe" (les images appartenant à deux classes différentes sont éloignées le plus possible les unes des autres) et "similarité inter-classe" (les images représentant la même scène sont proches). Plus précisément, l'adaptation, nommée D-CNN (*Discriminative CNN*), présentée dans la *figure 1.5*, prend en compte une nouvelle fonction objective (qui inclue les deux critères précédents) sans changement de l'architecture du CNN elle-même, ainsi, la "*cross entropy loss*" est toujours prise en compte. Les expérimentations ont permis de noter un nouveau record sur le dataset UC-Merced avec une précision de 98.93 % (pour la meilleure architecture du CNN), sur le dataset AID avec une précision de 96.89 % (pour un ratio d'entraînement égale à 50 %), et sur le dataset NWPU-RESISC45 avec une précision de 91.89 % (pour un ratio d'entraînement égale à 20 %).



**Figure 1.5 : Architecture d'un D-CNN [54]**

Bien que l'application du *deep learning* pour la classification des scènes a donné des résultats éblouissants, il s'est avéré que cette technique souffre d'un problème majeur. En effet, le *deep learning* nécessite des grands datasets, une contrainte qui n'est pas toujours satisfaite pour les datasets de la classification des scènes. Pour cela, l'approche de l'augmentation des données, que nous présentons dans ce qui suit, est née.

### *i / Augmentation des données*

L'augmentation des données est une approche développée pour rendre possible l'apprentissage des petits datasets, contrairement aux CNNs qui ont prouvé leur performance dans diverses tâches du domaine "vision par ordinateur", entre-autres, ces réseaux dominent l'exploitation des larges datasets tout en évitant le surapprentissage. Cependant, dans la pratique, la majeure disponibilité est pour les petits datasets (par exemple, le domaine médical : rareté des maladies, confidentialité des patients et l'exigence des experts médicaux pour d'étiquetage), d'où l'augmentation des données qui propose des mécanismes pour trouver des solutions aux problèmes où la quantité des données disponibles est limitée, et cela en ajoutant d'autres instances (images étiquetées) au dataset via des manipulations sur celles déjà existantes [55].

Diverses méthodes de l'augmentation des données ont vu le jour dans la recherche, en s'inspirant des différents aspects/caractéristiques de l'image.

Les transformations géométriques sont des actions triviales, facilement applicables aux images, elles définissent la base pour la compréhension des autres techniques, plus avancées, de l'augmentation des données. L'application de ces transformations est conditionnée par leur "*sécurité*" sur un problèmes donné, cela signifie la préservation de la même étiquette (point de vu sémantique) sur l'image post-

transformée, par exemple : les défis du type "chat" vs "chien" sont -généralement- sûrs, contrairement à la reconnaissance des chiffres (par exemple, "6" vs "9").

- Flipping : C'est l'application de l'effet miroir sur une image, il peut être appliquée par rapport à l'axe horizontal (la méthode la plus courante) ou l'axe vertical. Cette augmentation a prouvé d'être utile sur les datasets *CIFAR-10* ou *ImageNet*.

- Recadrage : C'est le changement de la taille de l'image, tel que celle résultante aura des dimensions (longueur et largeur) différentes que l'originale. Cette technique peut être faite manuellement (en fixant au préalable la valeur de réduction/agrandissement) ou aléatoire.

- Rotation : C'est la rotation d'une image à gauche ou à droite sur un axe entre  $1^\circ$  et  $359^\circ$ . La sécurité de cette augmentation est liée au degré de la rotation elle-même, ainsi, des faibles rotations ( $\pm 1^\circ$  à  $\pm 20^\circ$ ) peuvent être bénéfiques pour le dataset *MNIST* (chiffres en écriture manuscrite).

- Translation : C'est le décalage d'une image vers une (ou plusieurs) directions, le but étant de diversifier les positions potentielles des objets contenus dans l'image. Le nouvel espace vide restante après cette augmentation peut être rempli simplement avec une constance (entre 0 et 255) ou, plus intelligemment, avec du bruit aléatoire ou Gaussien. Cette technique se présente adéquate aux datasets dont les images sont majoritairement centrées (par exemple, la reconnaissance des visages).

L'ensemble des augmentations géométriques est souvent combiné pour tirer le meilleur de chacune d'elles. Une telle combinaison a fait la base d'une étude liée à la détection d'une tumeur, très agressive, de la peau, nommée "*Mélanome*", qui peut être guérie efficacement si détectée tôt. Plus précisément, des transformations linéaires (rotation, flipping, recadrage et translation) ont été appliquées pour enrichir le dataset avec des nouvelles images, ensuite, une imitation de la *consistance des couleurs* (l'invariance de la perception des couleurs, peu importe l'intensité de la lumière, dans le système visuel humain) a été faite par le *white-balancing* et la *gamma-correction*. Puis, les images résultantes ont été injectées dans un CNN conventionnel [56].

Une autre approche intéressante présentée dans [57] consiste à automatiser le processus du choix de la meilleure combinaison des augmentations pour un problème donné. Ce choix étant souvent facile aux experts du domaine pour les transformations individuelles, n'est pas aussi évident pour les transformations composées ce qui conduit à une perte de temps énorme. La stratégie adoptée est l'apprentissage par renforcement, permettant d'utiliser les connaissances liées au domaine pour prédire la façon de combiner les différentes augmentations et les paramétriser.

Les transformations géométriques énumérées précédemment s'avèrent très utiles en regard de leur simplicité. Cependant, elles souffrent d'une panoplie des problèmes : consommation additionnelle de mémoire, coût (temps de calcul) lié à l'application de ces transformations et le temps additionnel de l'entraînement. Ces désavantages ont poussé les chercheurs à penser à autres transformations plus performantes.

- Plage des couleurs : C'est l'ensemble des manipulations sur les canaux des couleurs compositrices de l'image. En effet, une image digitale est représentée par une matrice ordinaire (longueur \* largeur \* nombre de canaux), ainsi, des simples augmentations peuvent être appliquées en isolant un seul canal de couleur (pour chaque pixel, ne garder que la valeur d'un seul canal, les autres seront nulles). De plus, un ajustement de luminosité (incrémentation / diminution) peut être fait en parcourant l'image, en ajoutant (plus sombre) ou retranchant (plus clair) une valeur constante à chaque pixel. D'autres changements, plus poussés, bénéfiques peuvent être apportés en jouant avec l'histogramme. Ce type de transformation appartient à une famille, nommée "transformations photométriques".

Une comparaison a été faite dans [58] entre les différents espaces colorimétriques des images, utilisés pour le clustering (deux algorithmes ont été utilisés, un basé sur l'*entropie* et l'autre sur l'*ignorance*), le but étant d'identifier la meilleure représentation colorielle. Les résultats ont montré que la meilleure similarité (entre l'image idéale et les meilleures images segmentées) est celle de CMY (modèle coloriel

utilisé dans l'impression), ensuite, HSV, YUV (modèle imitant la vision humaine) et RGB successivement.

- Mixage des images : C'est la division d'une image en parties et leur jointure avec d'autres parties d'images pour former une nouvelle image. Malgré leur non-intuitivité (l'image produite est incompréhensible pour un observateur humain), une telle augmentation peut s'avérer bénéfique pour la classification des images (réduction du taux d'erreur).

Une telle technique, nommée *RICAP* (*Random Image Cropping And Patching*), a été proposée dans [59] qui consiste à diviser aléatoirement quatre images (d'une même dimension, prises à partir du ImageNet) et les regrouper pour construire une nouvelle image d'entraînement (avec une taille similaire aux quatre images). Ce traitement inclus aussi les labels (classes), ainsi, l'image résultante contiendra les quatre labels des parties d'images initiales composantes (avec des pourcentages proportionnels à la surface des parties prises pour chaque image). Les résultats obtenus ont marqué un nouveau record sur CIFAR-10 avec un taux d'erreur de 2.23 %, de bons résultats ont été aussi obtenus sur CIFAR-100 et ImageNet. La figure 1.6 montre un exemple d'application de la technique *RICAP*.

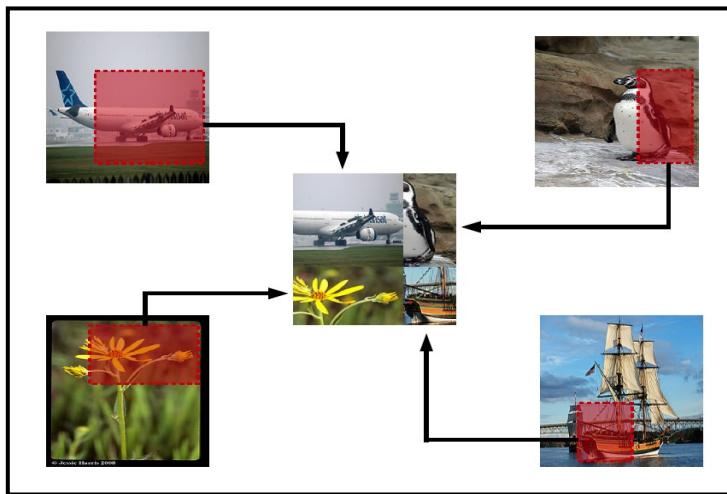


Figure 1.6 : Exemple d'application de la technique RICAP [59]

Une autre manière de mixage des images a été abordée dans [60], il s'agit de proposer une variante plus générale pour ce type d'augmentation. Plus précisément, un tas de mixages non-linéaires ont été appliqués aux images sources (deux images à la fois) pour produire des nouvelles images d'entraînement, entre-autres : concaténation horizontale/verticale (diviser les deux images horizontalement/verticalement en deux parties, et les interchanger), concaténation mixte (diviser les deux images selon une grille 2\*2 [4 parties], et les combiner [l'image résultante sera composée de deux parties de chaque image]), carré aléatoire (récupérer une surface [sous forme d'un carré] d'une image et le placer dans l'autre), lignes/colonnes aléatoires (récupérer des lignes/colonnes d'une image et les placer dans l'autre), pixels aléatoires (mixer les pixels des deux images arbitrairement). Pour les résultats, le taux d'erreur obtenu sur CIFAR-10 variait entre 6.2 % et 3.8 % (selon le type de mixage utilisé), sur CIFAR-100 entre 24.2 % et 19.7 % et sur CALTECH-256 entre 56.3 % et 59.7 %.

- Effacement aléatoire : Une nouvelle méthode d'augmentation, proposée dans [61], qui consiste à sélectionner aléatoirement une région -continue- rectangulaire dans une image et "effacer" ses pixels selon différentes manières : assignation des valeurs arbitraires dans l'intervalle [0, 255] aux pixels, assignation de la valeur moyenne d'un pixel dans ImageNet (125, 122 ou 114), assignation de la valeur 0 (noir) ou 255 (blanc) aux pixels. A travers ce processus, une variété des niveaux d'occlusion (masquer une ou plusieurs parties d'un objet dans une image) de l'image d'entraînement est générée, ainsi, le modèle résultant (une fois entraîné, en utilisant un CNN) sera robuste à l'effet d'occlusion et au bruit. Pour les différentes architectures des CNNs testées, le meilleur résultat noté (taux d'erreur minimal) pour CIFAR-10 est 3.08 %, pour CIFAR-100 est 17.73 % et pour Fashion-MNIST est 3.65 %. Un exemple d'application de l'effacement aléatoire sur quatre images en entrée est illustré dans la figure 1.7.



**Figure 1.7 : Exemple d'application de l'effacement aléatoire [59]**

Une fois les trois classes des méthodes des modèles basés sur des caractéristiques propres présentées, à savoir : Méthodes basées sur les caractéristiques artisanales, celles basées sur l'apprentissage non-supervisé et celles basées sur le deep learning. Nous passons dans ce qui suit à la deuxième catégorie des modèles : Modèles basés sur les objets.

## 2) Modèles basés sur les objets

L'idée de ces modèles est de se concentrer sur les objets contenus dans l'image et les considérer avec leurs propriétés (position, arrangement par rapport aux autres objets) comme entrée aux algorithmes d'apprentissage. Plusieurs techniques ont été proposées, chacune à sa manière pour l'extraction des objets dans l'image et pour l'algorithme appliqué dans la phase d'apprentissage.

Dans [62], les chercheurs ont proposé un modèle pour la recherche des scènes naturelles en se basant sur une étape de modélisation sémantique. Ils avaient extrait à partir de l'image un vecteur correspondant aux objets (concepts) avec leurs occurrences dans l'image en termes de pourcentage, et cela en utilisant les similarités et les dissimilarités entre les catégories de scènes naturelles, pour la phase d'apprentissage, ils ont utilisé un SVM (*Support Vector Machine*).

Dans l'article [63], une autre manière de classification a été proposé, cela en adaptant l'algorithme de pLSA (*probabilistic Latent Semantic Analysis*) pour l'extraction des objets et l'algorithme de KNN (*K-Nearest Neighbors*) pour la phase d'apprentissage.

Une autre approche a été proposée dans [64], il s'agit d'utiliser un ensemble de filtres d'objets pour extraire les objets de l'image (*Latent SVM object detectors*) pour les "*blobby objects*" (tables, véhicules, êtres-humains, etc.) et "*texture classifier*" pour les objets basés sur les textures (ciel, rue, sable, etc.) et un SVM pour la phase d'apprentissage.

Les chercheurs du [65] ont utilisé d'abord une segmentation sémantique pour obtenir des régions sémantiques, puis ils avaient obtenu une représentation histogramme objet de la scène en question en regroupant la sommation sur toutes les régions. Deuxièmement, ils ont construit des priors spatiaux et géométriques pour chaque objet et chaque paire d'objets co-occurrents à partir de scènes d'apprentissage, et ont intégré les informations spatiales et géométriques des objets dans la représentation de la scène. Enfin, ils avaient utilisé un SVM avec le noyau X2 pour la classification. Leurs résultats d'expérimentation sur deux datasets (*LabelMe Outdoor* et *MIT Indoor-67*) démontrent que la représentation proposée est efficace et compétitive.

La terminaison du listage des travaux liés aux modèles basés sur les objets, achève l'état de l'art de la classification. De même que pour cette dernière, et en suivant la succession élaborée au début de la partie de l'état de l'art, nous présentons l'état de l'art de l'interprétation des scènes.

## 1.2.2 État de l'art de l'interprétation des scènes

Dans la section courante, nous présentons deux parties liées à l'état de l'art de l'interprétation des scènes : Définition et les travaux associés.

### 1.2.2.1 Définition de l'interprétation

L'interprétation d'une scène est le passage d'une description structurelle à une description sémantique d'une scène. Autrement dit, à partir de l'analyse formelle et objective faite sur une image, un sens est attribué à cette dernière : La façon dont l'image fonctionne dans son contexte. Cette sémantique est assurée par un traitement automatique du langage naturel qui transforme l'ensemble des mots (annotations) reçus en une phrase concrète (séquence significative des mots) en ajoutant des relations linguistiques (ponctuations, verbes et propositions) à ces tags suivant une méthodologie particulière.

Le but de cette interprétation (ou description) est de trouver une relation entre les différents éléments identifiés dans une image. Ainsi, pour l'exemple précédent, la légende "voiture qui roule sous le ciel bleu" rendrait explicite la relation entre les mots.

Une fois le concept de l'interprétation des scènes défini. Nous passons à la présentation des travaux associés en les résumant et affectant à la catégorie adéquate.

### 1.2.2.2 Travaux associés

Suivant la description du [66], l'interprétation automatique des scènes visuelles est une tâche complexe dans l'intelligence artificielle, qui n'est pas encore résolue. Cette complexité est liée principalement à l'inexistence d'une méthode parfaite pour combler le vide entre les caractéristiques du bas niveau (*low-level features*) et les informations sémantiques du haut niveau. Ces dernières nécessitent de comprendre le visuel, en appliquant des techniques de transformation du contenu visuel en descriptions textuelles précises et concises, plus faciles à comprendre en les comparant aux simples annotations issues de la classification (sorties partielles et non-structurées), ainsi, la différence entre les deux niveaux (bas et haut) est le "*niveau d'abstraction*". Les défis majeurs de l'interprétation peuvent être résumés par :

- Reconnaissance des détails des contenus visuels et les interactions entre les objets qui les composent.
- L'apprentissage de la représentation intermédiaire entre les deux domaines : visuel et du langage naturel.
- Identification du degré des détails récupérés à partir du contenu visuel (l'importance des différents éléments par rapport à la thématique d'une scènes données) et la complexité du langage utilisée lui-même.
- L'existence des datasets appropriés (images avec leurs annotations sous forme de phrases, pour l'évaluation des méthodes et algorithmes développés).

À cause de cette difficulté, peu d'études ont abordé directement ce problème dans le passé. Les recherches se restreignaient uniquement au cas d'annotation (attribution des étiquettes) aux images. Cependant, malgré la quantité éblouissante des travaux autour l'étiquetage des images, la génération des modèles de langage naturel pour les systèmes de "compréhension" visuelle a été un paradigme de recherche populaire au cours de la dernière décennie. Ainsi, les techniques "visuel au texte" peuvent être classifiées en quatre différentes catégories.

Les méthodes d'interprétation d'images existantes calculent les scores de vraisemblance du journal pour évaluer leurs légendes générées. Les mesures *BLEU*, *METEOR*, *ROUGE*, *SPICE* et *CIDEr* sont utilisées comme métriques d'évaluation. Cependant, *BLEU*, *METEOR* et *ROUGE* ne sont pas bien corrélés avec les évaluations humaines de la qualité. *SPICE* et *CIDEr* ont une meilleure corrélation mais sont difficiles à optimiser. La qualité de la description d'image dépend de l'évaluation de deux aspects

principaux : l'adéquation et la fluidité. Une mesure d'évaluation doit se concentrer sur un ensemble diversifié de caractéristiques linguistiques pour atteindre ces aspects. Cependant, les mesures d'évaluation couramment utilisées ne prennent en compte que certaines caractéristiques spécifiques (par exemple, lexicales ou sémantiques) des langues [88].

Nous distinguons trois catégories des approches pour la génération des interprétations des scènes. Nous commençons, dans ce qui suit, par la première catégorie : Approches basées sur des règles de langage ou des modèles.

### **1) Génération des interprétations basées sur des règles de langage ou des modèles**

Cette catégorie est basée sur des modèles prédéfinis, qui précisent d'une manière exacte la façon dont les différents types de mots sont combinés. Ainsi, les concepts (objets, attributs, actions, ...) détectés dans l'image sont mises dans leur emplacement correspondant dans le modèle. Ce dernier peut être défini suivant différentes méthodes. Les techniques de cette catégorie sont spécifiques, liées à un domaine donné, pas générales, efficace uniquement si l'ensemble des éléments est restreint et les descriptions attendues sont simples.

Une partie des méthodes abordées utilisent des modèles fixes, désignés manuellement (par des experts du domaine, par exemple).

L'article [67] est basé sur la génération des descriptions pour les images prises dans un arrière-plan uniforme. Cela est fait par le biais d'une base de données manuelle indexant des signatures d'images (à partir de l'ombre, couleur et caractéristiques des textures) et deux mots-clés (nom et catégorie). En premier lieu, les images sont segmentées en objets. Ensuite, une signature est créée et comparée à celles dans la BD en récupérant la plus similaire. La dernière étape est la génération de la description en utilisant les mots-clés associés avec les objets similaires. Cette approche est peu pratique car elle opère sur des images relativement simples en supposant, de plus, l'inexistence du problème d'occlusion (et c'est justement, grâce à cette contrainte qu'une bonne segmentation est obtenue).

Le travail du [68] est la mise en correspondance des images et phrases dans un espace de signification construit de triplets <objet, action, scène>, ensuite, les résultats sont comparés. Celui de l'image est récupéré en résolvant un petit MRF (Multi-label Markov Random Field). Le potentiel d'un nœud est défini comme étant une combinaison linéaire des évaluations (degrés) de certains paramètres, le choix de la meilleure combinaison est fait par une méthode gourmande. De plus, le système développé est adaptatif pour les termes hors-vocabulaire (noms des marques par exemple). L'une des contributions marquantes de ce travail est la construction du PASCAL, un dataset des descriptions potentielles (phrases) pour une image donnée.

L'article [69] s'intéresse à la composition d'une description en employant les web-scale n-grams, des fonctions de préposition sont créées manuellement pour générer des triplets de forme <<adj1, obj1>, prep, <adj2, obj2>>. L'approche consiste, dans un premier lieu, de sélectionner les phrases candidates, potentiellement utiles, pour la génération d'une description de l'image concernée, en tenant compte des synonymes et le réordonnancement des mots pour améliorer la fluidité/lisibilité. Ensuite, ces phrases sont fusionnées en cherchant la combinaison optimale en utilisant la programmation dynamique, ainsi une nouvelle phrase plus complexe est générée et attribuée comme interprétation de l'image.

Dans l'article [70], l'approche proposée (illustrée dans la *figure 1.8*) consiste à créer une forte liaison entre le contenu d'une image et le processus de génération du texte. La technique commence par la division de l'image en parties concrètes (objets ou choses : arbre, route, ciel, etc.) en utilisant des détecteurs entraînés. Chaque élément détecté est procédé par classificateurs d'attributs et fonctions de préposition simples sont utilisées pour déterminer la relation spéciale entre les différents éléments. Un CRF (Conditional Random Field) où les nœuds correspondent aux entités de l'image (objets, attributs et prépositions) permettant de prédire la meilleure annotation pour l'image concernée est utilisé. La génération des phrases est faite via les n-grammes manipulant le labelling précédent.

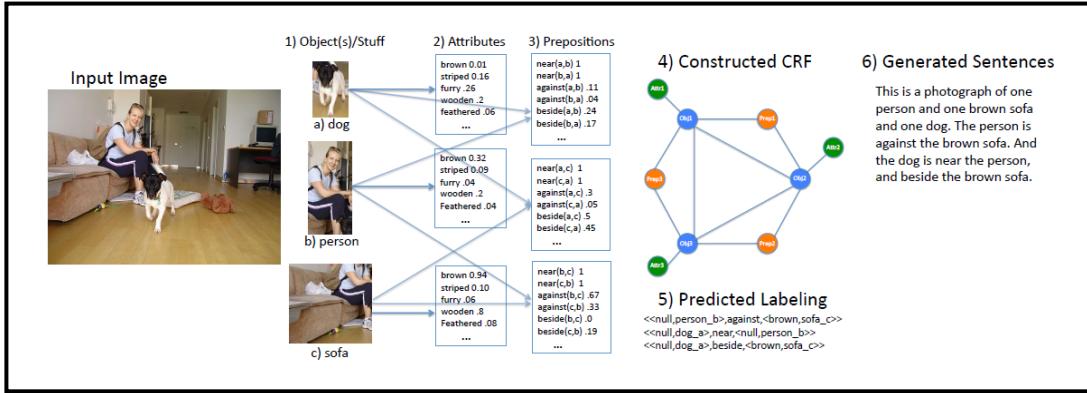


Figure 1.8 : Schéma global de l'approche de l'article [70]

D'autres méthodes proposent des modèles variables, ces derniers sont adaptatifs. Ainsi, le processus de la composition des mots diffère d'une image à autre.

L'article [71] propose l'utilisation des coordonnées géographiques issues du GPS contenues dans les métadonnées des images web (de plus d'autres informations pertinentes contenues dans celle-ci). Les documents pertinents sont ensuite récupérés et résumés (en utilisant différentes mesures : similarité, position, etc.) en faisant un matching avec les données précédentes. Cette approche est, elle aussi, peu pratique car elle se limite aux objets statiques (montagnes, ponts, monuments, etc.) et ne peut pas être appliquée sur les objets dynamiques de la vie quotidienne (gens, voitures, etc.).

Une autre approche a été présentée dans [72], il s'agit d'un framework de génération des descriptions textuelles pour les images (et vidéos également) en se basant sur la compréhension du contenu. Au début, un moteur d'analyse de l'image en entrée est lancé, une décomposition de l'image en motifs visuels est faite, suivie d'une construction d'un *graphe d'analyse (parse graph)*, ce dernier est sous forme d'une décomposition détaillée progressive des différents concepts de la scène (nœuds : un concept général donné [par exemple, personne avec équipements], descendants : les sous-concepts composants du concept père [par exemple, personne + sac-à-dos]). Ensuite, des relations sémantiques entre les différents concepts du graphe d'analyse créé sont établies, deux types de relations se présentent : entre objets (concepts différents) et entre parties (du même concept). Cela est fait en se servant d'un *AoG (And-or Graph)* comme étant une ontologie, une structure qui fournit les hypothèses de haut en bas, cet arbre est construit par la fusion de plusieurs graphes d'analyse en regroupant d'une manière cohérente les objets avec le même sens sémantique mais d'une apparence différente, les différentes relations sont extraites à partir du *WordNet* (un réseau sémantique lexical d'anglais). Par la suite, ces représentations sémantiques sont converties au format *RDF* et enrichies avec les connaissances générales associées (autres que l'image elle-même, par exemple : la géolocalisation), contenues dans le web sémantique. Finalement, ces représentations sémantiques enrichies sont converties en description au langage naturel lisible par l'humain, en utilisant un "*text planner*" (élection du contenu qui doit être exprimé, et décider comme l'organiser) et un "*text realizer*" (générer une structure grammaticale correcte basée sur l'output du "*text planner*").

Les chercheurs de l'article [73] supposent la disponibilité d'un document pertinent pour une image dans le domaine d'information (news), en récupérant un ensemble de mots-clés et expressions. Ce dernier est combiné avec l'annotation de l'image concernée suivant un modèle probabiliste, le résultat est la génération d'une légende décrivant brièvement l'image. Deux méthodes sont proposées : La génération extractive de légendes (calculer la similarité entre les phrases et les mots-clés de la description générée par le modèle d'annotation en utilisant certaines mesures) et la génération abstraite (basée sur les mots ou phrases).

La stratégie de génération des phrases qui décrivent les images, présentée dans [74] consiste à prédire les éléments noyaux de la structure de la phrase descriptive : les objets importants (noms), quelques descriptions des actions (verbes) associées à ces objets, la scène (entourage) d'où l'image est prise et les prépositions qui relient les objets à la scène. L'hypothèse posée est que les images naturelles reflètent fidèlement les scénarios quotidiens, par exemple : sachant que les bateaux sont généralement sur l'eau,

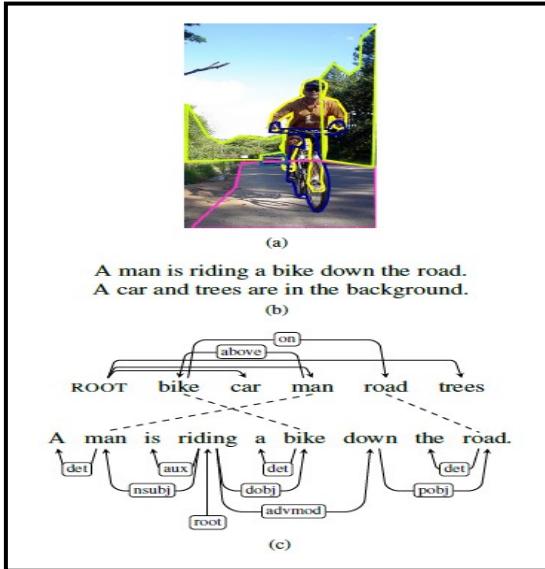
permet de limiter les scènes possibles de ce dernier, et donc exclure "rue" ou "autoroute", ce même raisonnement est applique aux verbes. Initialement, l'entrée représente des estimations des objets détectés en utilisant des détecteurs pré-entraînés (issus d'autres travaux), nommés : Pascal-VOC 2008, en se limitant aux deux objets les plus présents dans la scène. Ensuite, un modèle de langage est entraîné en utilisant le corpus English Gigaword (un ensemble d'environ dix millions documents [plus de quatre milliards mots]) comme un "support sémantique" pour obtenir les différentes probabilités conditionnelles de l'espace de langage (noyau de la phrase). Ces estimations sont utilisées comme des paramètres dans un HMM (Hidden Markov Model) pour la génération de la phrase (nœuds cachés : composantes de la phrase, émissions : détections de l'image). La méthode proposée est générique (aucune connaissance apriori du domaine considéré n'est utilisée). Les résultats obtenus sur UIUC Pascal Sentence dataset sont : une pertinence de 2.51 et une lisibilité de 4.10 (meilleurs résultats, en employant toutes les composantes de l'approche proposée).

L'article [75] s'adresse au problème de la génération automatique des descriptions des images à partir de leur annotation, en passant par diverses étapes. Premièrement, des ensembles sont extraits, chacun contenant différents types d'informations (objet, attribut, sujet, préposition, déterminant et verbe) de cardinalité variées (combinaisons de longueurs différentes). Ensuite, une annotation de l'image en entrée est faite avec les deux objets et les deux annotations les plus fréquents dans l'ensemble des descriptions disponibles pour l'image à partir du *PASCAL dataset*. Par la suite, une association des attributs récupérés avec les objets est faite en utilisant des ensembles spécifiques des types d'information, cette spécificité (qui diffère à chaque fois) est maintenue pour la détermination du *sujet de la description*, le *verbe*, le *déterminant* et la *préposition*. Finalement, la génération de l'interprétation est faite en mappant le quadruplet défini ((dét 1, attr 1, sujet), verbe, préposition, (dét 2, attr 2, objet)) dans une phrase grammaticalement correcte en utilisant *SimpleNLG* (un framework avec une couverture significative de la syntaxe/morphologie anglaise). La figure 1.9 présente un exemple d'application de l'approche de l'article [75].

INPUT	OUTPUT
 <i>&lt; dog, wall, white, brown &gt;</i>	Attr-Obj : < brown, dog > < white, wall > Subject : dog Verb : sit Preposition : next_to Description : A brown dog is sitting next to a white wall.

Figure 1.9 : Exemple d'application de l'approche de l'article [75]

Dans [76], les relations binaires entre les régions de l'image ont été prises en compte pour générer des interprétations qui reflètent le plus fidèlement la réalité. Pour cela, une annotation des régions de l'image a été faite en dessinant les polygones autour des contours des régions en utilisant l'outil *LabelMe*. Ensuite, un *CDG (Visual Dependency Grammar)* a été construit pour décrire les relations spatiales entre les paires des régions. Ce VDG est un *graphe acyclique dirigé (DAG)* composé des nœuds (qui représentent des régions, avec "l'acteur principal" comme racine), et les arcs sont les relations (huit types, par exemple : à côté de, opposé, en dessus, derrière, etc.), déterminés suivant trois propriétés géométriques : chevauchement des pixels, l'angles et la distance entre les régions. Enfin, les descriptions sont générées en utilisant différentes approches, toutes basées sur des modèles prédéfinis : Proximity, Corpus, Structure et Parallel. La figure 1.10 présente un exemple d'application de l'approche de l'article [76].



**Figure 1.10 : Exemple d'application de l'approche de l'article [76]**

Dans [77], une génération des descriptions améliorées dans un langage "*situé*" (supposer un contexte sans le lecteur, en proposant des étapes à suivre) a fait l'objet d'étude, le but étant d'augmenter la compréhension des instructions des recettes de cuisine en ajoutant des mots/phrases complémentaires dans l'instruction elle-même. Pour cela, différentes notions sont manipulées : *états* (ensemble d'objets [ingrédients et conteneurs], avec leurs informations [quantité, emplacement et état actuel]), *actions* (verbe avec des arguments, spécifiant comment transformer un état), *données textuelles* (phrases qui décrivent l'action primitive) et les *données visuelles* (les images statiques [ou trames des vidéos] pratiques avec leurs instructions textuelles associées aux actions, si disponibles). Ces notions sont modélisées par un *processus de Markov décisionnel orienté-objet (temps discret, partiellement observé)*, tel que les états/actions sont *cachés*, et les données textuelles/visuelles sont *observées*, le résultat étant une "vraie" couverture d'une recette (avec tous les détails).

L'achèvement du listing des travaux liés à première catégorie de la génération des interprétations des scènes (approches basées sur des règles de langage ou des modèles), nous conduit à la présentation de la deuxième catégorie : Approches basées sur la récupération des descriptions à partir des contenus visuellement similaires.

## 2) Récupération des descriptions à partir des contenus visuellement similaires

Une amélioration de l'approche précédente a été proposée dans [78] en construisant la description par la composition des phrases relatives à chaque objet de l'image en entrée (et non pas un seul texte pour toute l'image, comme dans [72]). Au début, pour chaque concept de l'image en entrée, un ensemble de sous-phrases (passages, contenus dans les descriptions) qui font référence à ce dernier (visuellement similaires [en comparant la couleur, texture ou la forme]) dans le dataset *SBU Captioned Photo* est récupéré. Ensuite, une construction de l'interprétation finale (un petit texte) est faite en générant un ensemble de phrases. Pour chaque phrase (qui décrit un objet à la fois), l'ILP (*Integer Linear Programming*) est exécuté : réordonnancement des passages relatifs à l'objet courant, et la sélection des passages favorables (selon de critères).

L'approche présentée dans [79] consiste à interpréter les images en se basant sur les descriptions disponibles sur le web pour les images similaires à celle d'entrée. Plus précisément, une nouvelle base d'images avec leur description textuelle est construite en interrogeant un réseau social spécialisé à cela (Flickr). Les données récupérées sont épurées en ne gardant que celles jugées "satisfaisantes" (en se basant sur la longueur de la description et d'autres critères liés aux types des mots employés), le résultat étant une collection d'environ un million d'images décrites, nommée "*SBU Captioned Photo dataset*". Ensuite, vient l'étape d'exploitation de ce dernier, pour cela, pour une image en entrée (à interpréter), un ensemble d'images (au nombre de 100) les plus similaires sont récupérées, la comparaison est faite

en sommant les sorties de deux descripteurs globaux d'images. Finalement, un réordonnancement de l'ensemble récupéré est fait en se basant sur la correspondance au contenu de l'image en entrée, cela est fait en générant un score pour chaque type de contenu (objets, choses, personnes, scènes et poids TFIDF [entre les descriptions de l'ensemble récupéré]), et en combinant ces scores pour obtenir un degré de similarité général (deux manières : *modèle de régression linéaire* ou *SVM*).

Le travail du [80] présente le mappage des phrases et les images dans un espace d'intégration commun pour pouvoir déduire la description à partir d'une image en entrée. Pour cela, une étape primaire est nécessaire, elle consiste à calculer des représentations vectorielles des mots/images séparément. Pour les phrases, les représentations sont calculées en utilisant un DT-RNN (*Dependency-Tree Recursive Neural Network*) qui exploite directement l'arbre de dépendance (fonctions séparées/relationnelles des mots) des phrases, la particularité de cette technique est qu'elle permet de produire des représentations qui soient robustes aux changements de la structure syntaxique et l'ordre des mots dans la phrase. Les représentations des images sont capturées à partir d'un réseau de neurones profond non-convolutionnel (un peu modifié, pour accélérer l'apprentissage, entraîné avec des données annotées et autres non) en récupérant les caractéristiques trouvées dans la dernière couche (avant le résultat final [sortie] de la classification), qui sera la représentation vectorielle de l'image. Finalement, l'entraînement de la fonction objective pour les représentations jointes image-phrase est fait en employant les images (sous forme de vecteurs) et leurs représentations des phrases descriptives (au nombre de 5, récupérées d'un dataset).

Une fois la présentation de la deuxième catégorie de la génération des interprétations (approches basées sur la récupération des descriptions à partir des contenus visuellement similaires) achevée. Nous passons, dans ce qui suit, à la troisième catégorie : Approches basées sur la génération de nouvelles descriptions basées sur l'apprentissage profond.

### **3) Génération de nouvelles descriptions basées sur l'apprentissage profond**

Le modèle de langage construit dans [81] est lié à l'image entrée elle-même, la génération de la description de cette dernière passe par trois étapes. Au début, un apprentissage faiblement-supervisé est utilisé pour la création des détecteurs pour un ensemble de mots couramment trouvés dans les interprétations des images. Cependant, une difficulté liée au accès d'un tel système aux "*supervisory signals*" (par exemple : le "bounding box" pour les mots "bondé" ou "dans" est flou) est noté. Pour lui faire face, un raisonnement par "sous-régions" (au lieu de l'image complète) est adopté, plus précisément, une caractéristique (nom, verbe ou adjectif) est attribuée à chaque région en utilisant un CNN, ensuite, en utilisant un *MIL (Multiple Instance Learning)*, chaque caractéristique est mappée avec les mots les plus probables, contenus dans la description. La seconde étape consiste à générer des descriptions à partir des mots précédents (appelées : sac de mots), ce qui est vu comme un problème d'optimisation, la tâche principale est de prendre un ensemble de mots détectés avec leurs scores, et de trouver les phrases les plus probables qui couvrent chaque mot exactement une fois. Cela est fait en entraînant un *ME LM (Maximum Entropy Langage Model)* à partir des descriptions des images récupérées d'un dataset, ainsi, il capture les connaissances sur le "monde" en employant des statistiques linguistiques. Finalement, des degrés sont attribués aux descriptions générées, pour en choisir la meilleure, en pondérant les caractéristiques des phrases avec un *MERT (Minimum Error Rate Training)*, de plus, une nouvelle caractéristique est introduite, basée sur *DMSM (Deep Multimodal Similarity Model)*. Le pipeline général de l'approche de l'article [81] est présenté dans la *figure 1.11*.

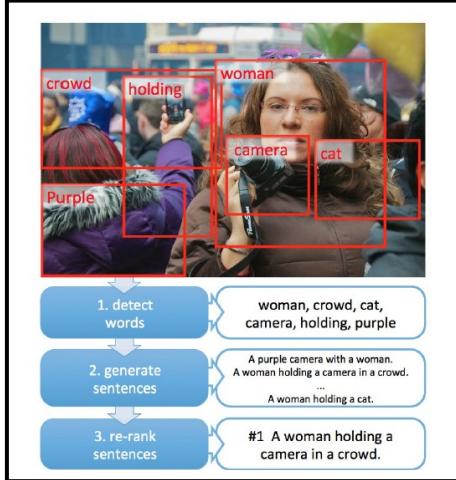


Figure 1.11 : Pipeline de l'approche de l'article [81]

La structure proposée dans [82] pour la génération des interprétations aux images, nommée *m-RNN* (*Multimodal RNN*), permet de modéliser directement la distribution de probabilités pour générer un mot sachant un mot précédent dans l'image, cela est fait en combinant deux modèles de sous-réseaux. La structure contient cinq couches : deux couches d'intégration (qui encodent le sens syntaxique et sémantique des mots), une couche récurrente (permet de concaténer progressivement les mots obtenus pour former la phrase finale), une couche multimodale (connecte la partie du modèle de langage, et la partie de vision, elle reçoit trois entrées : couche 2 d'intégration des mots, la couche récurrente et la représentation de l'image [en utilisant la 7ème couche d'activation du CNN AlexNet]) et la couche Softmax (génération la distribution de probabilités pour le prochain mot). Le m-RNN est utilisé dans trois tâches différentes, la plus importante (l'interprétation des images) procède en entrant un signe de début (ou un nombre arbitraire des mots de référence), en calculant à chaque fois la probabilité du mot suivant pour récupérer le mot suivant, jusqu'à l'obtention du signe d'arrêt. La figure 1.12 montre la différence entre le modèle d'un simple RNN (à gauche) et le modèle d'un m-RNN (à droite).

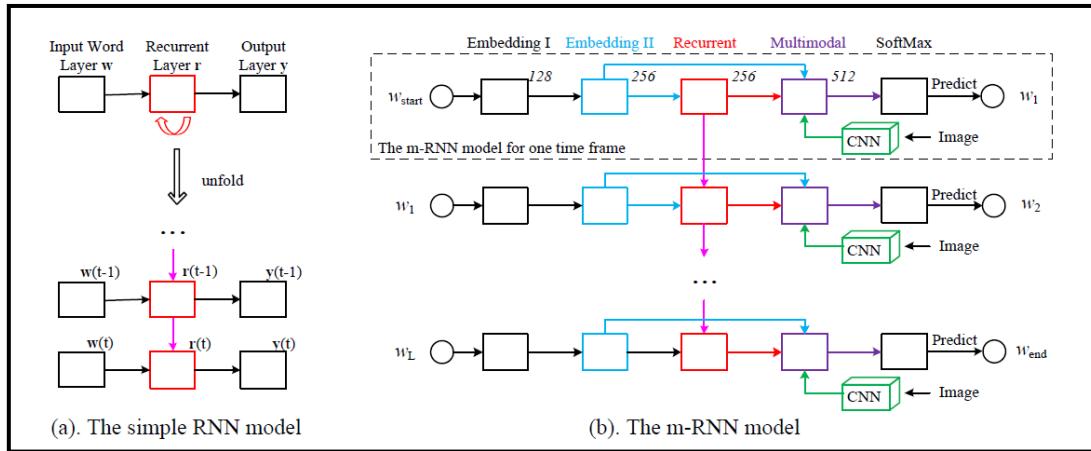
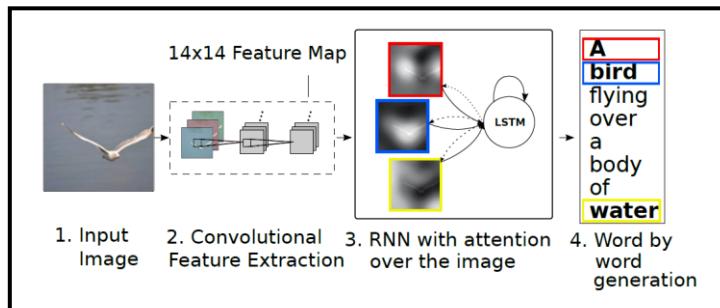


Figure 1.12 : Différence entre le modèle d'un simple RNN et le modèle d'un m-RNN [82]

La technique développée dans [83] permet de générer des descriptions denses (détails pour chaque élément) pour les images, cela est fait en passant par deux étapes. Premièrement, en exploitant le fait que les phrases écrites par les humains font références à des zones particulières, mais inconnues dans l'image, un modèle de réseaux de neurones profonds est développé pour la mise en correspondance des segments des phrases d'interprétation et les régions des images à lesquelles elles réfèrent. Pour cela, des représentations pour les images (en utilisant un *Recursive-CNN*) et les phrases (en utilisant un *Bidirectional-RNN*) sont générées séparément, puis mappées ensemble (en formulant un score d'appariage image-phrase). Ensuite, une architecture M-RNN (modification de celle du [17], en posant des conditions sur le contenu de l'image) est introduite, qui prend une image en entrée et génère sa description (interprétation) textuelle.

Dans [84], une nouvelle architecture basée sur l'attention pour générer les descriptions d'images est proposée (illustrée dans la *figure 1.13*). Le réseau de neurones convolutionnel (CNN) est utilisé comme un encodeur pour extraire des caractéristiques d'une image en entrée, et un *Long Short-Term Memory (LSTM)* network (qui produit une légende en générant un mot à chaque pas de temps conditionné sur un vecteur de contexte, l'état caché précédent et les mots générés précédemment) est utilisé comme décodeur. Entre les deux, deux générateurs de légendes basés sur l'attention dans un cadre commun : 1) un mécanisme d'attention déterministe "doux" entraînable par des méthodes de rétropropagation standard et 2) un mécanisme d'attention stochastique "dur" entraînable en maximisant une limite inférieure variationnelle approximative ou de manière équivalente par REINFORCE sont utilisés aidant le décodeur à se pencher sur les parties importantes de l'image. Afin de montrer l'efficacité du modèle proposé, les datasets *Flickr8k*, *Flickr30k* et *Microsoft COCO* sont utilisés, les résultats expérimentaux montrent que le modèle proposé atteint un score BLEU1 de 67 sur le jeu de données *Flickr8k*, 66,9 sur le jeu de données *Flickr30k* et 71,8 sur l'ensemble de données *MS COCO*.



**Figure 1.13 : Architecture générale de l'article [84]**

Dans [85], pour construire le modèle *NIC (Neural Image Caption)*, les chercheurs utilisent d'abord un CNN comme "encodeur" d'image, en le pré-entraînant pour une tâche de classification d'image et en utilisant la dernière couche cachée comme entrée pour le décodeur RNN qui génère des phrases. Des expériences sur plusieurs ensembles de données montrent la précision du modèle et la maîtrise du langage qu'il apprend uniquement à partir des descriptions d'images. Par exemple, sur l'ensemble de données *Pascal*, leur approche atteint un score BLEU-1 de 59. Sur l'ensemble de données *Flickr30k*, ils atteignent un score BLEU-1 de 66 et un score SBU de 28. Enfin, sur le nouvel ensemble de données *MS COCO*, ils ont atteint un score BLEU-4 de 27,7.

Pour montrer que de nouveaux objets peuvent être correctement intégrés dans les légendes générées, les chercheurs de l'article [86] ont proposé un modèle "*Image Captioning using Novel Word Injection*", qui utilise un générateur de légendes préformé [85] et travaille sur la sortie du générateur pour injecter des objets qui ne sont pas renvoyés dans l'ensemble de données dans la légende à l'aide de deux modules : 1) un modèle de détection d'objet préformé [87] qui est chargé de générer avec précision une liste d'objets présents dans l'image et 2) un "*Semantic Word Embedding*". L'espace d'intégration est utilisé pour déterminer la proximité de deux objets, contextuellement et sémantiquement, dans cet espace. Les objets présents dans la légende et ceux extraits de l'image sont ensuite comparés et remplacés suivant leurs probabilités s'ils sont suffisamment proches les uns des autres, la préférence étant donnée aux objets extraits de l'image. Leur évaluation du modèle se fait sur les métriques standardisées à savoir, BLEU, CIDEr et ROUGE-L. Les résultats surpassent quantitativement et qualitativement le modèle sous-jacent [85].

En finissant la présentation des travaux de la catégorie courante ("Génération de nouvelles descriptions basées sur l'apprentissage profond"), nous achevons l'état de l'art de l'interprétation des scènes. Pour conclure ce chapitre, nous récapitulons, dans la section suivante, les différents points abordés.

### **1.3 Conclusion**

Dans ce chapitre, nous avons éclairci les deux concepts traités par notre travail, à savoir : La classification et l'interprétation des scènes. En accompagnant chaque concept par sa définition (permettant de comprendre, et donc cerner le sens d'un concept donné) et l'ensemble des travaux (permettant de connaître du près les avancés scientifiques d'un concept donné), résumés et catégorisés, proposant diverses approches liées à ce concept.

Dans le chapitre suivant, nous parlons sur nos propres approches proposées pour les deux concepts précédents. Et cela en présentant la conception de chaque approche.

# Chapitre 2 – Conception

## 2.1 Introduction

Ce chapitre présente la conception de nos différentes approches mises en place, liées aux deux thématiques principales, discutées dans le "*Chapitre 1 – État de l'art*", à savoir, la classification et l'interprétation des scènes. La présentation de chaque approche englobera une introduction générale sur son utilité, son type (parmi ceux existants, définis dans l'état de l'art) et le format des données manipulées. De plus, pour chaque approche, nous passons en revue détaillée des différentes étapes avec les détails algorithmiques et exemples sous forme de figures (illustrations)

Nous passons, dans la section suivante, au cœur de ce chapitre, à savoir, la conception des approches proposées pour la classification et l'interprétation des scènes basées sur les objets.

## 2.2 Approches proposées pour la classification et l'interprétation des scènes

Les approches proposées sont présentées dans cette section suivant leur thématique, en commençant pas la classification, puis l'interprétation. Un tel ordonnancement est choisi du fait que les résultats obtenus par la classification seront utilisés dans la partie interprétation. Chaque thématique est accompagnée avec des phases intermédiaires nécessaires à l'aboutissement au résultat final : Catégorie de la scène pour la classification et la description textuelle pour l'interprétation.

Nous commençons, dans ce qui suit, par la présentation des approches proposées pour la classification des scènes basées sur les objets.

### 2.2.1 Classification des scènes basées sur les objets

La classification des scènes est une tâche importante dans le domaine du traitement des images, elle vise à affecter à une scène une catégorie bien définie en se basant sur ces caractéristiques de bas niveau (couleur, forme, etc.) ou celles de haut niveau (objets, actions, etc.). Dans notre travail, nous nous focalisons sur cette dernière modélisation de la scène (haut niveau) pour effectuer la classification. Notre choix est justifié par le fait que les caractéristiques de bas niveau ne sont pas en mesure de fournir les meilleures propriétés discriminantes entre les classes, contrairement aux caractéristiques de haut niveau, en raison de manque d'informations sémantiques.

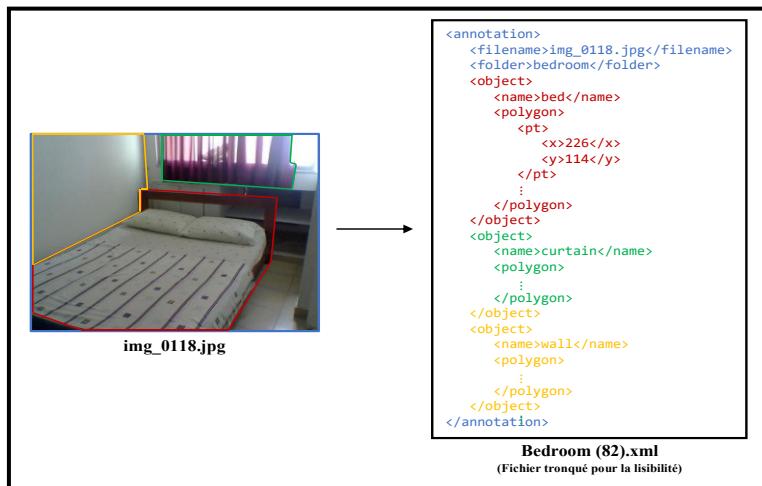
La classification des scènes basées sur les objets commence, bien évidemment, par la détection des objets (appelés également "annotations"), les objets détectés sont utilisés dans le processus de la classification afin de déduire la classe (catégorie).

Cependant, puisque les détecteurs d'objets sont en développement continu (Fast R-CNN (2015) [1], SSD (2016) [2], Faster R-CNN (2016) [3], 2017 [4], 2018 [5, 6], 2019 [7] et YOLOv4 (2020) [8]), ils ont des précisions variables et peuvent retourner des résultats erronés. Par conséquence, les détecteurs d'objets ne forment pas un support rigoureux pour évaluer l'efficacité des classifieurs des scènes (i.e. la précision d'un détecteur d'objets influe sur celle du classifieur). Ainsi, pour faire abstraction des problèmes pouvant être émis par le détecteur d'objets et focaliser sur l'évaluation du problème de classification des scènes, nous supposons l'existence au préalable des objets des scènes. Cela nécessite une préparation des datasets dont leurs scènes sont annotées (objets identifiés) manuellement.

L'annotation manuelle est faite soit par des travaux personnels (MIT-Indoor [12]), ou bien, par un système de collaboration (LabelMe [11]). Quel que soit le type d'annotation, elle doit assurer la

cohérence de l'ensemble des données entrées en employant diverses méthodes telles que la correction orthographique ou la vérification par des tiers.

L'annotation d'une scène est mise dans un fichier structuré (XML [11, 12, 13], JSON [9, 10], etc.). La *figure 2.1* présente un exemple d'une annotation manuelle d'une scène dans laquelle le contour de chaque objet est délimité par un polygone irrégulier. L'objet identifié est mis dans le fichier d'annotation XML (associé à la scène dont le nom est sous le tag `<filename>` et la catégorie sous le tag `<folder>`) sous le tag `<object>` qui contient, entre-autres, le nom de l'objet (tag `<name>`) et l'ensemble des points (coordonnés `<x>` et `<y>`) définissant le polygone.



**Figure 2.1 : Exemple d'un fichier d'annotation manuelle d'une scène dans MIT-Indoor [12]**

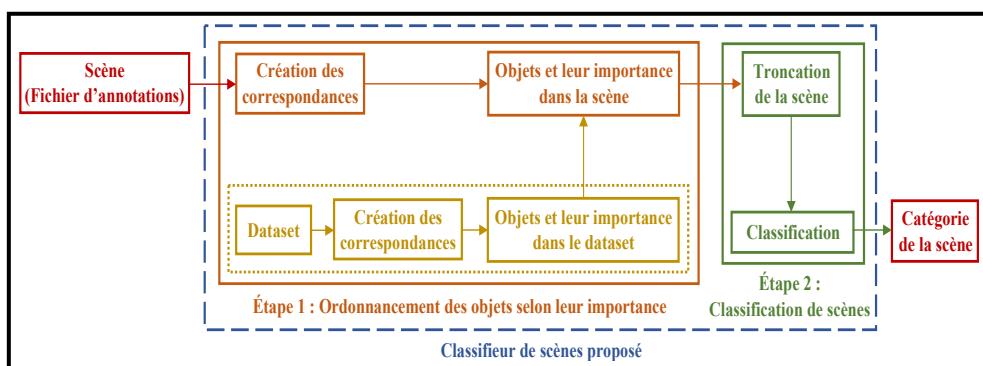
Après cette brève introduction aux systèmes de classification des scènes avec les caractéristiques de haut niveau, les arguments derrière le choix d'un tel type et les datasets. Nous présenterons notre première contribution qui consiste en un classifieur des scènes basées sur les objets.

### 2.2.1.1 *Classifieur des scènes proposé*

Le fonctionnement de notre classifieur (pour la classification en catégories standard des scènes basées sur les objets) se résume par les deux étapes successives suivantes :

- Ordonnancement des objets selon leur importance : Il s'agit de définir un ordre entre les objets de la scène à l'aide d'une mesure qui prenne en compte l'occurrence des objets dans la scène et l'importance de l'objet dans la partie d'apprentissage du dataset.
- Classification des scènes : En exploitant les importances des objets, une troncation de la scène est faite de façon à ne considérer qu'un nombre limité d'objets. Ensuite, la scène est injectée dans un LSTM qui se chargera de déterminer la catégorie de la scène en entrée.

La *figure 2.2* présente des étapes les étapes générales expliquées précédemment.



**Figure 2.2 : Workflow du classifieur des scènes proposé**

Après avoir présenté de manière générale le flux et les étapes importantes que renferme notre classifieur. Nous présenterons, dans ce qui suit, les détails de chaque étape.

### **1) Ordonnancement des objets selon leur importance**

Nous allons détailler, au cours de cette première étape, la façon avec laquelle les importances des objets du dataset et les scènes à classifier sont obtenues, notamment la conversion des scènes annotées en structures de correspondance et les équations définies.

#### **a - Création des correspondances "objet - fréquence"**

Cette étape primaire consiste à parcourir la partie d'apprentissage du dataset (sous forme d'annotations, en fichiers textuels structurés), catégorie par catégorie, scène par scène et d'en extraire leurs objets avec leurs occurrences (fréquences d'apparition), le résultat étant une structure de correspondance (dictionnaire, table de hachage, etc.) "objet – fréquence" isolée (individuellement, pour chaque scène). L'*algorithme 2.1* présente la manière avec laquelle la création des correspondances est faite.

#### **Algorithme 2.1 : Création des structures de correspondance "objet - fréquence" pour un dataset**

```

Algorithme création_correspondance;
Entrée: D: Dataset contenant des catégories avec des fichiers
d'annotations des scènes;
Sortie: Dataset sous forme de structure de correspondance;
Début;
ds_corresp ← initialiser_dictionnaire();
Pour chaque c ∈ D.catégories faire
    ds_corresp[c] ← []; /* Initialiser par une liste vide */
    Pour chaque s ∈ c.scènes faire
        dict_obj_freq ← extraire_objets(s);
        ds_corresp[c].ajouter(dict_obj_freq);
    Fait;
    Fait;
    retourner(ds_corresp);
Fin;
/* Extraction des objets et leur fréquence à partir d'un fichier
d'annotation */
Algorithme extraire_objets;
Entrée: F: Fichier d'annotation d'une scène;
Sortie: Structure de correspondance de la scène;
Début;
dict_obj_freq ← initialiser_dictionnaire();
/* Récupérer le texte contenu dans le fichier */
F_texte ← lire_fichier(F);
liste_objets ← récupérer_objets(F_texte);
Pour chaque objet ∈ liste_objets faire
    objet_propre ← supprimer_blancks(minuscule(objet));
    Si (objet dans dict_obj_freq) alors
        dict_obj_freq[objet]++;
    Sinon
        dict_obj_freq[objet] = 1;
    Fsi;
Fait;
retourner(dict_obj_freq); Fin;

```

Après avoir mis en place les structures de données adéquates à l'exploitation des fréquences d'apparitions de chaque objet du dataset, nous pouvons avancer sereinement vers l'introduction des différentes mesures d'importance que nous proposons dans notre travail afin d'évaluer la saillance de chaque objet dans une scène quelconque.

### **b - Calcul des importances des objets du dataset**

Cette étape consiste à calculer les importances des objets du dataset en utilisant sa partie d'apprentissage afin d'exploiter ces importances, par la suite, lors de l'identification des objets saillants dans la scène. Ainsi, le fait de pouvoir instaurer une hiérarchie entre les objets permet de définir un ordre, et donc, rendre possible leur comparaison.

Le calcul s'appuie sur des notions propres au problème traité, soit :

- "**D**" : le dataset.
- "**DA**" : la partie d'apprentissage du *D*.
- "**C**" : L'ensemble  $\{C_1, C_2, \dots, C_n\}$  des catégories contenues dans *D*.
- "**S**" : L'ensemble  $\{S_1, S_2, \dots, S_n\}$  des scènes contenues dans une catégorie donnée ( $C_i$ ).
- "**O**" : L'ensemble  $\{O_1, O_2, \dots, O_n\}$  des objets contenus dans une scène donnée ( $S_i$ ).

Afin de nous aider dans la mise en place des calculs de l'importance des objets, Nous définissons quelques mesures, inspirées de l'article [14] :

- **FO** ( $C_i, O_j$ ) : Une fonction qui retourne toutes les occurrences de l'objet  $O_j$  dans les différentes scènes qui appartiennent à la classe  $C_i$ .
- **NO** ( $C_i, O_j$ ) : Une fonction qui retourne le nombre de scènes de la catégorie  $C_i$  pour lesquelles l'objet  $O_j$  leur fait partie.
- **FO<sub>all</sub>** ( $O_i$ ) : Une fonction qui retourne toutes les occurrences de  $O_i$  dans DA.
- **NO<sub>all</sub>** ( $O_i$ ) : Une fonction qui retourne le nombre de scènes du DA contenant  $O_i$ .
- **Max<sub>FO<sub>oi</sub></sub>** : L'occurrence maximale de l'objet  $O_i$  dans une catégorie donnée.
- **Max<sub>NO<sub>oi</sub></sub>** : Le nombre d'apparitions maximal de l'objet  $O_i$  dans une catégorie donnée.

Après avoir défini l'ensemble des mesures nécessaires au calcul des importances des objets du dataset, nous présentons l'*équation 2.1* utilisée pour ce calcul :

$$I[O_i] = Max_{FO_{oi}} * \log_{10}(Max_{NO_{oi}} + 1) - (\log_{10}(FO_{all}[O_i]) + \log_{10}(NO_{all}[O_i] + 1)) \quad (2.1) \quad [14]$$

L'*algorithme 2.2* montre le processus explicite du calcul des importances des objets d'un dataset, en démarrant par la structure de correspondance (objets et leurs fréquences) et finissant avec une structure contenant les objets et leurs importances.

#### **Algorithme 2.2 : Calcul des importances des objets d'un dataset**

```

Algorithme calculer_importance_dataset;
Entrée: D: Dataset sous forme de structure de correspondance;
Sortie: Objets du dataset avec leur degré de l'importance;
Début;
    importance_objets <- initialiser_dictionnaire();
    D_apprentissage <- récupérer_partie_apprentissage(D);
    liste_objets <- récupérer_touts_objets(D_apprentissage);

```

```

Pour chaque objet ∈ liste_objets faire
    importance_objets[objet] ← calculer_I(objet, D_apprentissage);
Fait;
/* Pour une exploitation ultérieure */
exporter_vers_fichier(importance_objets);
Fin;
/* Récupération de la liste de tous les objets (uniques) dans un dataset*/
Algorithme récupérer_touts_objets;
Enrée: D: Dataset sous forme de structure de correspondance;
Sortie: Liste des objets (uniques) dans D;
Début;
liste_objets ← [];
Pour chaque c ∈ D.catégories faire
    Pour chaque s ∈ c.scènes faire
        Pour chaque obj ∈ s.objets faire
            Si non (obj dans liste_objets) alors
                liste_objets.ajouter(obj);
            Fsi;
        Fait;
    Fait;
Fait;
retourner(liste_objets);
Fin;

```

La figure 2.3 présente un exemple dans lequel les deux étapes précédentes (la création de structures de correspondance et le calcul de l'importance des objets d'un dataset) sont mises en pratique. Au début, les fichiers d'annotations des scènes de la partie d'apprentissage du dataset (dans cet exemple, quatre) sont récupérées pour les transformer en structures "objet - fréquence" (en utilisant l'*algorithme 2.1*). Ensuite, ces dernières sont regroupées dans un seul tableau nommé "structure de correspondance de toute la partie d'apprentissage du dataset", contenant pour chaque scène, sa catégorie (Intérieur [Indoor] ou extérieur [Outdoor]) et la fréquence d'apparition de chaque objet du dataset dans la scène (une valeur nulle signifie que l'objet n'existe pas dans la scène). À la fin, le tableau précédent est exploité (en utilisant l'*équation 2.1*) dans l'*algorithme 2.2* pour calculer les importances des objets du dataset, le dernier tableau en dessous (dans la *figure 2.3*) contient les résultats obtenus.

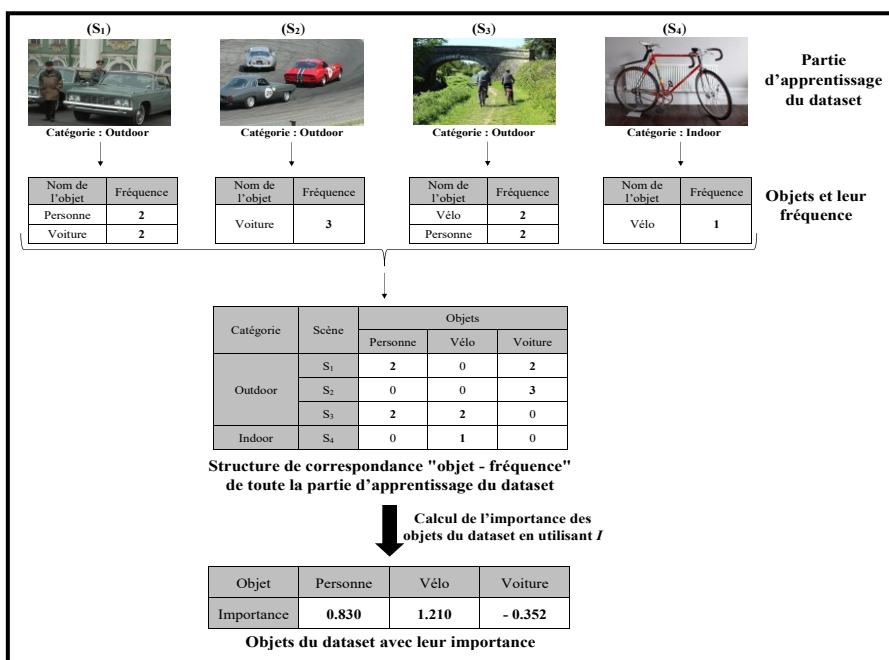


Figure 2.3 : Exemple du calcul des importances des objets dans un dataset

Une fois les importances des objets du dataset calculées, nous procéderons à les exploiter pour le calcul des importances des objets de chaque scène en entrée du processus de la classification.

### c - Calcul des importances des objets dans une scène

Nous avons, dans l'étape précédente, introduit la notion de l'importance d'un objet dans un dataset. Nous allons nous intéresser maintenant par les importances des objets dans une scène. Pour cela, la formule, nommée "*F-mesure*", est utilisée (*équation 2.2*).

$$\text{F-mesure}[O_i] = (1 + \beta^2) * \frac{I[O_i] * Occ_{oi}}{\beta^2 * I[O_i] + Occ_{oi}} \quad (2.2)$$

Où les différentes fonctions de l'*équation 2.2* sont :

- $\beta$  : Un nombre réel variable (paramètre).
- $Occ_{oi}$  : L'occurrence (fréquence) de l'objet dans la scène.

*L'algorithme 2.3* présente les détails du calcul des importances des objets dans une scène, en ayant en entrée une scène sous forme d'une liste des objets (sans redondance), le chemin vers le fichier contenant les importances des objets du dataset calculées précédemment (avec *l'algorithme 2.2*) et la valeur du paramètre  $\beta$ . Une fois les importances calculées (avec *l'équation 2.2*), la sortie de l'algorithme sera une structure de correspondance de forme "Objet de la scène - Importance".

#### Algorithme 2.3 : Calcul des importances des objets dans une scène

```

Algorithme calculer_importance_scène;
Entrée: io_chemin: Chemin vers le fichier contenant les importances des objets dans le dataset;
          S: Scène sous forme d'une liste d'objets (sans redondance);
          beta: Paramètre de la fonction "F-mesure";
Sortie: Objets de la scène avec leur degré de l'importance;
Début;
  importance_objets_dataset ← importer_depuis_fichier(io_chemin);
  importance_objets_scène ← initialiser_dictionnaire();
  Pour chaque obj ∈ objets_scène faire
    objet_w_tilde ← importance_objets_dataset[obj]
    importance_objets_scène[obj] ← calculer_f_mesure(obj, beta,
    objet_w_tilde);
  Fin;
  retourner(importance_objets_scène);
  Fin;
```

L'exemple de calcul des importances des objets dans une scène est présenté dans la *Figure 2.5*.

En calculant les importances des objets dans une scène, le but de la première étape de la classification des scènes (qui consiste à la capacité d'ordonner les objets d'une scène) est atteint. Ainsi, nous pouvons avancer droit vers la classification elle-même dans laquelle la scène sera tronquée (i.e. qu'un certain nombre des objets de la scène seront gardés) et entrée dans un classifieur qui se chargera de déterminer la catégorie la plus probable de la scène.

### 2) Classification des scènes

Nous présenterons, au cours de cette étape, les détails de la classification d'une scène, en commençant par sa liste des objets avec leurs importances, en passant par la troncation et finissant par l'assignement de la catégorie la plus probable à cette scène.

### a - Troncation d'une scène

Avant de la soumettre au classifieur, la scène subit un léger prétraitement nommé "une troncation". Ceci consiste à ne garder que les objets les plus importants dans la scène, en se basant sur la liste ordonnée de ses objets selon leurs importances. Le nombre des objets à garder est défini selon un paramètre, nommé "N", qui est un entier positif, déterminé par les expérimentations.

L'application de la troncation a comme but de ne garder que les objets de la scène jugés saillants, ainsi, le fait d'ignorer les objets les moins importants permettra guider le classifieur (en enlevant l'ambiguïté potentielle) pour la prédiction de la catégorie correcte de la scène.

### b - Classification de la scène tronquée

Pour traiter le processus de la classification, nous avons utilisé les LSTMs (*Long Short Term Memory*). Ces derniers, comme expliqué dans [15], sont un type particulier des RNNs (*Recurrent Neural Network*) ayant la capacité d'apprendre les dépendances à long terme.

La *figure 2.4* représente l'architecture interne détaillée d'un LSTM. Chaque ligne transporte un vecteur entier, de la sortie d'un nœud aux entrées des autres. Les cercles roses sont des opérations ponctuelles (telles que l'addition et la multiplication de vecteurs), tandis que les boîtes jaunes sont des couches de réseau neuronal apprises. La fusion des lignes indique la concaténation, tandis qu'une séparation de ligne (sous forme d'une fourche) indique que son contenu est copié, et que les copies vont à différents emplacements.

Les LSTMs ont la capacité d'ajouter ou supprimer les informations de la couche correspondante à l'état courant. Pour cela, des portes (*gates*) sont utilisées, elles permettent de contrôler les informations circulantes en les laissant (ou pas) passer. Une porte contient deux composants :

- Fonction d'activation sigmoïde : Elle génère des nombres compris entre zéro et un, décrivant la quantité (taux) de chaque composant qui doit être laissée passer.
- Opération de multiplication ponctuelle.

Un LSTM standard est composé de trois portes :

- Porte de l'oubli (forget gate) : Décide quelles informations doivent être oubliées (ou conservées). Elle examine la sortie précédente  $h_{t-1}$  et l'entrée actuelle  $x_t$ .
- Porte d'entrée (Input gate) : Décide quelles valeurs seront mises à jour.
- Porte de sortie (Output gate) : Décide quelle serait la sortie de la couche courante ( $h_t$ ), qui est aussi l'entrée du prochain état.

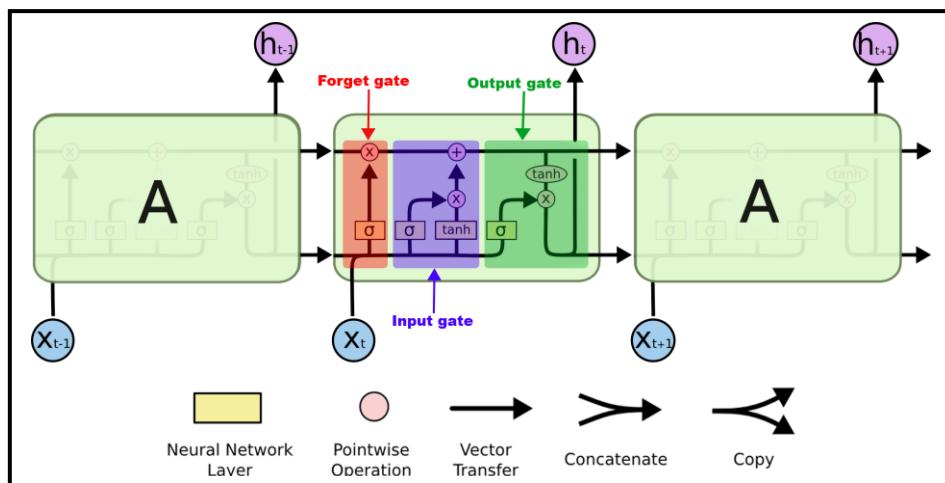


Figure 2.4 : Architecture détaillée d'un LSTM [15]

Le choix d'un tel classifieur est fait suite aux essais des différents classificateurs, notamment les arbres de décision (*Decision Trees*), SVMs (*Support Vector Machines*) et les réseaux de neurones. Nous avons opté pour les LSTMs suite aux arguments suivants :

- L'architecture des LSTMs a été conçue de telle sorte pour recevoir une structure ordonnée en entrée, et ce comportement correspond parfaitement à la première étape de notre méthode qui consiste à établir un ordre entre les objets d'une scène.
- Les LSTMs ont un temps d'exécution très réduit, ce qui revient essentiellement à leur simple et légère architecture (en les comparant, par exemple, avec les NNs [*Neural Network*]).

Après avoir présenté les deux étapes majeures de notre classifieur des scènes, nous énumérons les étapes successives du processus de la classification d'une scène entrée :

- Extraire les différents objets contenus dans la scène avec leurs fréquences (en utilisant son fichier d'annotation [*algorithme 2.1*] ou bien un détecteur d'objets).
- Trier les objets en ordre descendant suivant leur importance (le plus important est le premier) en utilisant F-mesure (*algorithme 2.2*).
- Effectuer la troncation de la scène en récupérant les N objets les plus importants.
- Entrer les objets précédents (sous forme d'un vecteur) dans un LSTM (qui se chargera par l'encodage des objets suivant le dictionnaire défini pour cela).
- Récupérer la catégorie de la scène après le décodage de la sortie du classifieur. Cette dernière est représentée par un vecteur *one-hot encoded* (vecteur nul, avec un seul "1" qui correspond à la position de la catégorie prédite).

La figure 2.5 montre un exemple pratique du processus présenté ci-dessus, en commençant par une scène et finissant par sa catégorie (dans ce cas, un "salon").

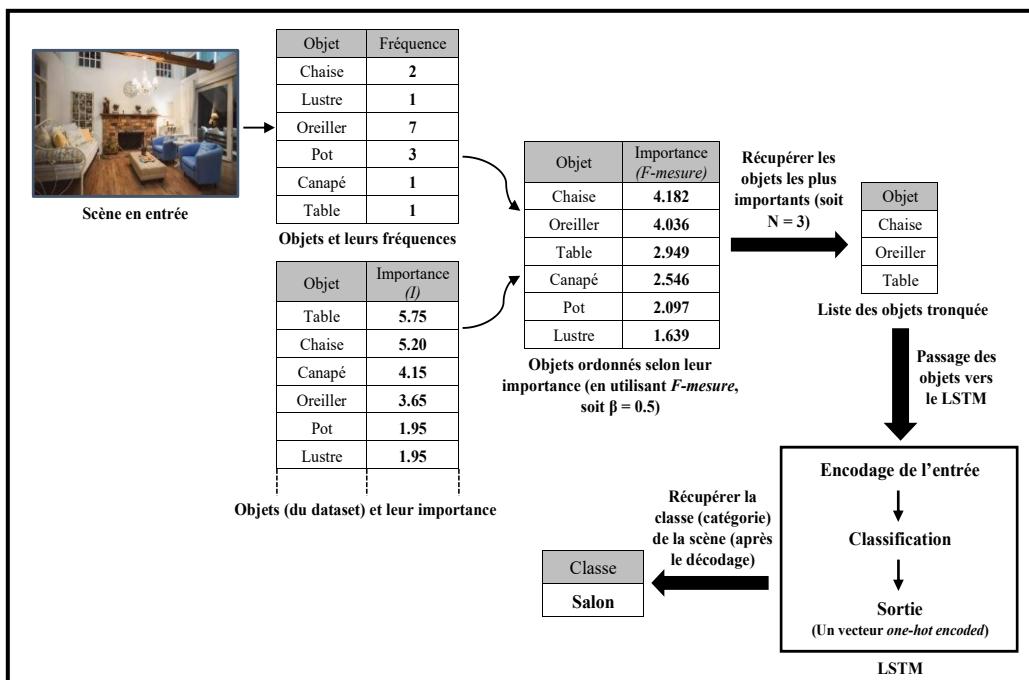


Figure 2.5 : Exemple de classification d'une scène

Ayant introduit notre classifieur des scènes proposé, et dans le but d'amélioration de sa précision, nous présentons, dès à présent, une approche complémentaire à notre classifieur que nous nommons l'enrichissement des scènes.

### 2.2.1.2 Enrichissement des scènes

L'enrichissement des scènes est une idée ayant comme fondement une simple observation : Une scène est susceptible de contenir quelques objets qui sont occlus et donc non-reconnus (lors de l'annotation), ou bien absents du cadre lui-même, mais existants -très probablement- dans la scène. Ce manque d'objets peut conduire à une ambiguïté lors du processus de classification, même au niveau humain, par exemple : Soit une image contenant quatre objets : laptop, livre, réfrigérateur et un four. Quelle serait sa catégorie : cuisine ou bureau ?

L'observation précédente nous a conduit à supposer que le rapprochement de la scène à classifier à une autre existante dans le dataset par le biais d'ajout intelligent des objets, peut être efficace à combler le manque des objets (dans la scène à classifier), afin d'améliorer le taux de précision dans classification des scènes.

Une telle hypothèse est défendue par le fait qu'il est très probable d'exister, parmi les scènes d'apprentissage, au moins une scène pour laquelle une grande portion d'objets de la scène à enrichir sont parmi ces objets, et l'ajout des objets manquants, paraissant importants, dans cette scène à la scène à enrichir aura un impact positif lors de la classification.

Ainsi, l'ajout intelligent des objets non-disponibles dans la scène à classifier, récupérés à partir de la scène la plus proche sémantiquement de celle à classifier, reliés avec ceux contenus initialement dans la scène à classifier, aura comme effet de lever l'ambiguïté sur la catégorie réelle de la scène.

L'approche proposée pour l'enrichissement suit la même logique que celle pour la classification où chaque objet du dataset est accompagné de son importance, ainsi que l'utilisation d'un LSTM comme classifieur.

*L'algorithme 2.4* présente les détails d'apprentissage d'un LSTM pour l'enrichissement. Cet algorithme reçoit en entrée quatre données :

- Les importances des objets du dataset, calculés en utilisant l'*algorithme 2.2* (dans le classifieur de scènes proposé).
- La partie d'apprentissage du dataset, où chaque élément est une catégorie qui contient un ensemble de scènes sous forme de structures de correspondance "objet - fréquence" (crées en utilisant l'*algorithme 2.1*).
- Liste des nombres des objets importants (calculés et ordonnés, en utilisant l'*algorithme 2.3*) à considérer pour les scènes de chaque catégorie.
- Taux des fréquences des objets importants à enlever pour chaque catégorie.
- Valeur du  $\beta$  utilisée (dans l'*équation 2.2*) dans la phase de réadaptation.
- Nombre des objets importants à garder dans la scène lors de la troncation dans la phase de réadaptation.

Au début, l'algorithme parcourt toutes les scènes de la partie d'apprentissage du dataset. Pour chaque scène, un taux de fréquence (lié à la catégorie de la scène courante) est appliqué à chaque objet parmi ces objets les plus importants (paramètre, spécifique pour chaque catégorie), ainsi, l'objet enlevé est ajouté dans une liste nommée "objets importants enlevés pour la scène courante" autant de fois que la valeur par laquelle la fréquence de cet objet dans la scène courante doit-être diminuée (en appliquant le taux de fréquence).

Ensuite, toutes les combinaisons des objets contenus dans la liste construite sont générées, chaque combinaison contient un certain nombre d'objets avec leurs occurrences. Pour chaque combinaison, une nouvelle scène est créée, similaire à la scène courante, mais avec les fréquences des objets importants (dans la scène courante) diminués par la valeur de l'occurrence des objets correspondants dans la

combinaison courante. À chaque fois, la nouvelle scène créée est insérée comme une nouvelle entrée dans l'ensemble de *features* (liste X), ayant comme *target* (liste Y) la combinaison courante.

Par la suite, l'ensemble d'apprentissage construit (*features* et leurs *targets*) est réadapté. Cette réadaptation est faite en réexécutant, pour chaque scène (dans la partie "*features*"), les deux étapes : Calcul des importances des objets (en utilisant l'équation 2.2, avec les importances des objets de la partie d'apprentissage du dataset primaire [à partir lequel, l'ensemble de l'enrichissement a été construit]) et la troncation.

À la fin, l'ensemble d'apprentissage réadapté est entré dans un LSTM. Ce dernier, une fois entraîné, est renvoyé comme sortie de l'*algorithme 2.4*.

#### **Algorithme 2.4 : Apprentissage d'un LSTM pour l'enrichissement**

```

Algorithme apprentissage_classifieur_enrichissement;
Entrée: I: Ensemble des objets du dataset avec leurs importances;
          C: Ensemble des catégories avec leurs scènes sous forme d'une
              structure de correspondance;
          N: Ensemble des nombres des objets à considérer pour chaque
              catégorie;
          T: Ensemble des taux des fréquences des objets importants à
              enlever pour chaque catégorie;
          beta_réadapt: Valeur du  $\beta$  pour la réadaptation;
          N_réadapt: Nombre des objets à considérer dans la réadaptation;
Sortie: Le classifieur entraîné;
Début;
ensemble_X_apprentissage  $\leftarrow$  [];      ensemble_y_apprentissage  $\leftarrow$  [];
Pour chaque c  $\in$  C faire
    n  $\leftarrow$  N[c];      t  $\leftarrow$  T[c];      scènes  $\leftarrow$  c.scènes;
    Pour chaque s  $\in$  scènes faire
        indices  $\leftarrow$  extraire_objets_importants(s, I, n);
        vecteur_objets  $\leftarrow$  [];
        Pour chaque i  $\in$  indices faire
            freq_enlev  $\leftarrow$  arrondir(s[i].fréquence * t);
            Pour num  $\leftarrow$  0 jusqu'à freq_enlev faire
                vecteur_objets.ajouter(i);
            Fait;
        Fait;
        combinaisons  $\leftarrow$  générer_toutes_combinaisons(vecteur_objets);
        Pour chaque comb  $\in$  combinaisons faire
            /* Copier le contenu vers une autre variable */
            s_copie = copie(s);
            Pour chaque indice  $\in$  comb faire
                s_copie[indice]--;
            Fait;
            ensemble_X_apprentissage.ajouter(s_copie);
            ensemble_y_apprentissage.ajouter(créer_nouvelle_scène(comb,
I.longeur));
        Fait;
    Fait;
/* Réadaptation du dataset d'apprentissage pour l'enrichissement */
nouveau_X_apprentissage  $\leftarrow$  [];
Pour chaque s  $\in$  ensemble_X_apprentissage faire
    importances_s  $\leftarrow$  calculer_importance_scène(I, s, beta_réadapt);

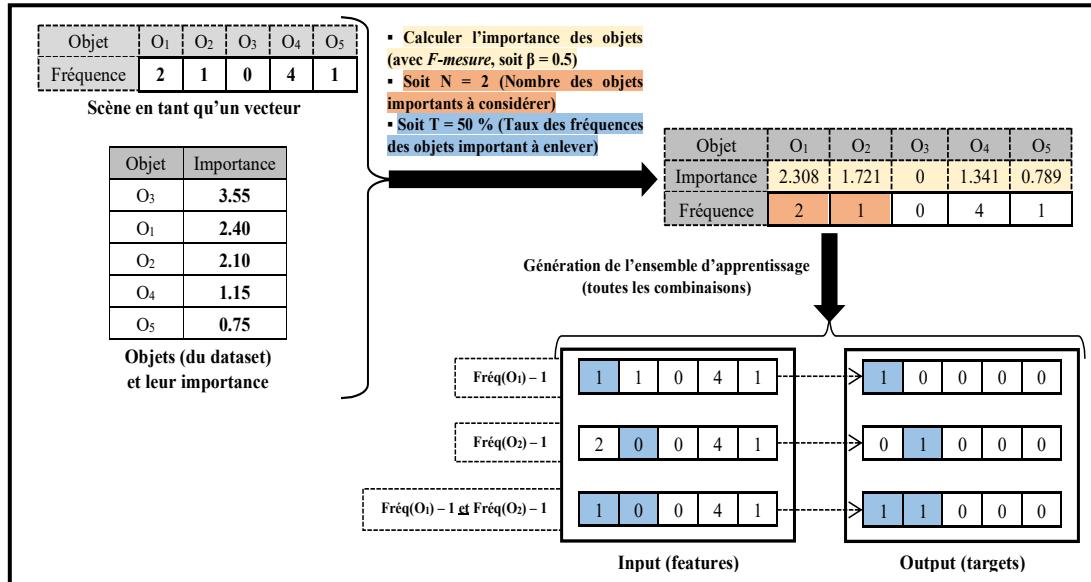
```

```

s_réadapt ← troncation_scène(importances_s, N_réadapt);
nouveau_X_apprentissage.ajouter(s_réadapt);
Fait;
classifieur ← entraîner_LSTM(nouveau_X_apprentissage,
ensemble_y_apprentissage);
retourner(classifieur);
Fin;

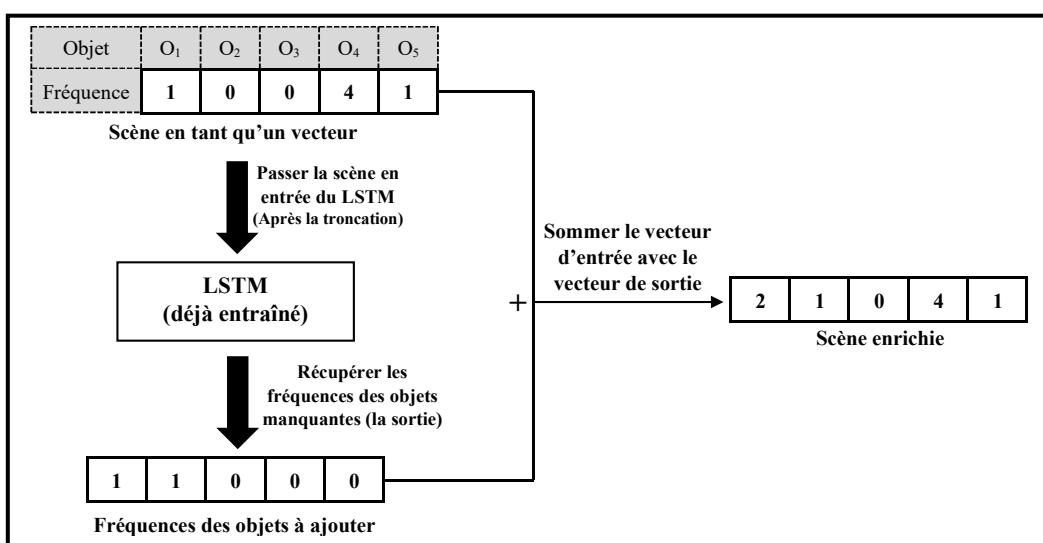
```

Afin de rapprocher l'idée de l'*algorithme 2.4*, nous illustrons une partie de son fonctionnement (sans la réadaptation) à travers la *figure 2.6* qui présente un exemple sur la construction de l'ensemble d'apprentissage du classifieur de l'enrichissement à partir d'une scène.



**Figure 2.6 : Exemple de construction de l'ensemble d'apprentissage à partir d'une scène**

Une fois l'ensemble d'apprentissage construit et le LSTM entraîné, son exploitation pour enrichir une scène en entrée consiste à -tout simplement- passer cette scène (sous forme vectorielle) comme entrée du LSTM et récupérer la sortie (output). Ensuite, il suffit de sommer le vecteur de la scène avec cette sortie obtenue (voir la *figure 2.7* qui présente un exemple de l'enrichissement d'une scène).



**Figure 2.7 : Exemple de l'enrichissement d'une scène**

L'enrichissement des scènes a pour but d'augmenter la probabilité d'une prédiction juste des catégories des scènes. Cependant, même si la catégorie d'une scène est prédite correctement, qu'une seule information sera ajoutée dans l'interprétation des scènes. Ainsi, afin d'avoir plus d'informations

sur la scène, une classification de cette dernière en d'autres catégories sera intéressante. Dans la section suivante, nous présenterons en détails la manière par laquelle une telle classification est faite.

### 2.2.1.3 Classification en catégories additionnelles

La classification en catégories standard des scènes présentée précédemment permet d'acquérir une information sur la catégorie de la scène. Aussi importante qu'elle paraisse, elle ne décrit pas toutes les spécificités de l'image classifiée. Ainsi, dans le but de récupérer davantage d'informations sur une scène, une autre classification en catégories additionnelles est faite. Cette dernière, comme son appellation l'indique, consiste à définir des classes supplémentaires (autres que celles déjà existantes) à valeurs réduites (binaire, ternaire, etc.).

Le choix d'une telle stratégie est argumenté par la simplicité (relative au nombre primaire des classes dans un dataset) dans l'ajout de telles classes vu leur petit ensemble de valeurs possibles, tout en ajoutant le gain d'informations reçu sur la scène. Ces informations seront ultérieurement utilisées dans l'interprétation des scènes.

La conception du classifieur des scènes en catégories additionnelles est identique à celle de la classification en catégories standard, notamment l'ordonnancement des objets selon leur importance (avec ses sous-étapes) et la classification elle-même (en utilisant un LSTM).

Le choix des catégories additionnelles dépend des classes initiales dans un dataset. Par exemple, si la totalité des classes concernent "l'intérieur" (comme dans le cas de MIT-Indoor [12]), cela ne sert à rien de définir une classe binaire "intérieur/extérieur". D'autre part, si une répétition d'un certain type de classes est remarquée (par exemple, moyen/lieu de transport), il serait intéressant d'introduire une classe binaire : Une scène représente le type remarqué ou non.

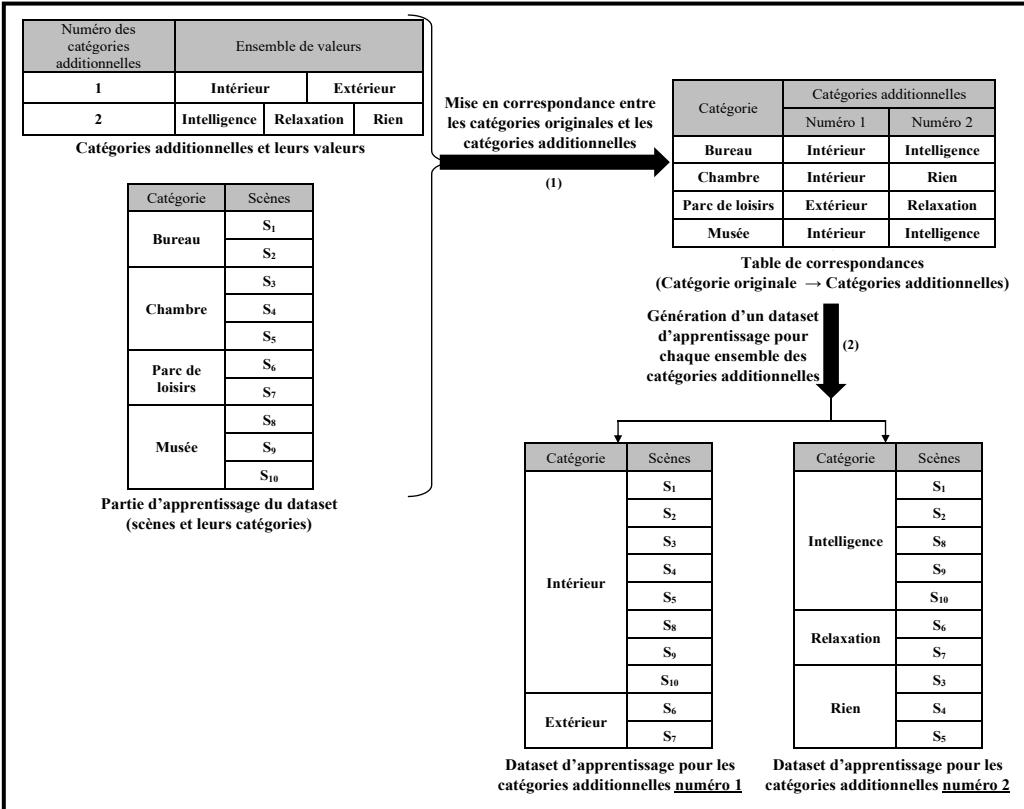
La *figure 2.8* représente un exemple de définition des catégories additionnelles dans un dataset. Au début, une liste (numérotée) de catégories additionnelles et l'ensemble de leurs valeurs possibles est définie :

- Catégories additionnelles 1 : De type "binaire", indique s'il s'agit d'une scène de l'intérieur d'une construction (e.g. bâtiment) ou de l'extérieur.
- Catégories additionnelles 2 : De type "ternaire", indique s'il s'agit d'une scène qui représente un endroit de relaxation (détente) ou demandant de l'intelligence (concentration et le sérieux) ou rien (ambiguïté de déterminer s'il s'agit d'une scène de relaxation ou de l'intelligence).

Ensuite, une mise en correspondance entre les catégories originales (contenues initialement dans le dataset) et les catégories additionnelles (définies précédemment) est faite. Cette correspondance consiste à attribuer à chaque catégorie originale (dans cet exemple : Bureau, chambre, parc de loisirs et musée), une valeur de chaque ensemble de catégories additionnelles définis.

Par la suite, un nouveau dataset pour chaque catégorie additionnelle est généré, en réorganisant la distribution des scènes suivant les correspondances définies précédemment.

À la fin, chaque dataset généré est injecté dans le processus de la classification en catégories standard (ordonnancement des objets selon leur importance et la classification des scènes), ainsi, il y aura autant de classifieurs que le nombre des catégories additionnelles définies.



**Figure 2.8 : Exemple de définition des catégories additionnelles dans un dataset**

La partie de la classification des scènes basées sur les objets est achevée. Au cours de cette dernière, nous avons présenté en détails, trois concepts :

- La classification en catégories standard des scènes, avec les LSTMs, en se basant sur les importances de leurs objets.
- L'enrichissement des scènes, une amélioration de la classification en catégories standard, visant à lever l'ambiguïté sur la catégorie correcte de la scène.
- La classification en catégories additionnelles, qui a pour but d'ajouter davantage d'informations sur une scène.

Ces concepts seront impliqués dans la partie suivante de notre travail, comme source d'informations, qui nous permet d'interpréter les scènes en se basant, justement, sur leurs objets.

### 2.2.2 Interprétation des scènes basées sur les objets

La classification des scènes permet d'acquérir une information générale sur une scène (sa catégorie). Cependant, elle néglige les interactions existantes entre les objets d'une scène.

Contrairement à la classification, l'interprétation qui consiste à décrire une scène en révélant son aspect sémantique par l'identification et la définition précise des relations phares entre les objets d'une scène, prend en considération les interactions entre les objets. Ainsi, une fois interprétée, la scène aura une description ciblée, non-monotone (la description change d'une scène à l'autre, en changeant le positionnement relatif des objets) et informative (apporte des détails particuliers sur les objets de la scène).

L'interprétation des scènes peut être réalisée via diverses approches, comme mentionné dans l'état de l'art, en utilisant des modèles préétablis (templates), en extrayant des descriptions à partir des contenus visuellement similaires ou en utilisant les réseaux de neurones profonds. Notre travail présente une approche hybride, entre l'utilisation des réseaux des neurones (pour l'extraction d'un ensemble de données à partir d'une scène) et l'emploi des templates (pour la procédure de génération des

interprétations des scènes, en utilisant les données précédentes). Une telle approche permet de tirer le meilleur de l'intelligence artificielle et les méthodes basées sur des règles.

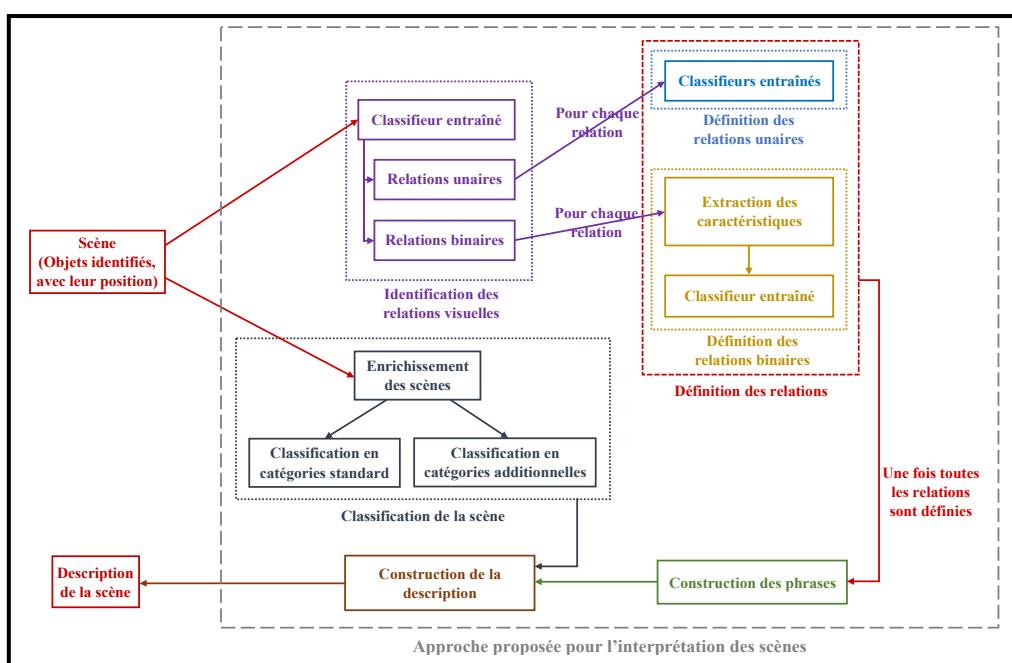
Après cette brève introduction à l'interprétation des scènes et les arguments derrière le choix d'une approche hybride (entre les réseaux de neurones et les templates). Nous présenterons dans la prochaine section, les détails de notre approche proposée pour l'interprétation des scènes.

### 2.2.2.1 Approche proposée pour l'interprétation des scènes

Une fois les objets contenus dans une scène sont détectés avec l'information sur leurs positions dans la scène sous forme d'un rectangle conteneur (*bounding-box*), nous procéderons à l'interprétation de cette scène en suivant les étapes successives ci-dessous.

- Identification des relations dans la scène : Cette étape a comme but d'identifier toutes les relations existantes entre les objets que ce soit des relations unaires ou binaires. Pour le faire, un modèle génératif sera conçu permettant de prédire les relations existantes entre les objets (en entrée de ce modèle).
- Définition des relations dans la scène : Il s'agit de déterminer, pour chaque relation identifiée, sa description.
  - Pour les relations unaires (l'objet lui-même) : un détecteur, qui dépend du type de l'objet impliqué dans la relation, sera utilisé. Il se chargera de déterminer l'attribut le plus probable pour cet objet de la scène.
  - Pour les relations binaires (entre deux objets) : une extraction des caractéristiques relatives entre les deux objets sera faite. Ensuite, ces caractéristiques seront injectées dans un classifieur pour définir la préposition le plus probable pour la relation binaire.
- Construction des phrases : En exploitant un graphe, qui englobe les objets d'une scène et l'ensemble de ses relations (unaires et binaires).
- Construction de la description : Consiste à construire une description complète (sous forme d'une ou quelques phrases reliées entre-elles) et cohérente à partir des phrases (séparées) définies précédemment ainsi que les informations extraites à partir de la phase de classification : Catégorie de la scène et les catégories additionnelles.

Le workflow de notre système d'interprétation des scènes est présenté dans la *figure 2.9*.



**Figure 2.9 : Workflow du processus de l'interprétation des scènes**

Nous présenterons, dans la section suivante, la première étape du processus de l'interprétation des scènes qui consiste à identifier les objets en relation dans une scène.

### **1) Identification des relations entre les objets d'une scène**

Il est primordial de connaître les relations unaires et binaires entre les objets d'une scène donnée afin qu'on puisse déterminer une description prenant en compte les interactions entre les objets existants dans la scène. Ces interactions peuvent être de cardinalité supérieure (relations ternaires, quaternaires, etc.), toutefois, pour des raisons de simplification nous nous sommes restreints à ces deux types de relations : Unaires et binaires.

Pour cela, nous avons conçu un modèle permettant à partir des objets de la scène de prédire ce qui existe comme relations entre ses objets.

Avant de présenter le modèle que nous avons adopté, nous allons, dans un premier temps, présenter ce qu'il va recevoir en entrée, et ensuite ce qu'il va générer en sortie.

#### **a - Entrée du modèle de l'identification des relations**

Comme mentionné précédemment, l'entrée du modèle consiste en une liste des d'objets de la scène triée selon leurs importances et tronquée pour ne pas tenir compte des objets non-significatifs et par conséquence, non représentatifs de la scène.

L'importance des objets dans la scène est calculée en utilisant une formule, appelée F-mesure (*équation 2.3*), similaire à l'*équation 2.2* (de la classification), prenant en compte deux paramètres :

- La fréquence de l'objet dans la scène ( $OC_{oi}$ ).
- L'importance de l'objet dans le dataset sur lequel le modèle est entraîné (*équation 2.4*).

$$F\text{-mesure}(O_i) = (1 + \beta^2) \frac{I(O_i) * OC_{O_i}}{\beta^2 * I(O_i) + OC_{O_i}} \quad (2.3)$$

L'importance de l'objet dans le dataset est calculée de la manière suivante, soit :

- $D$  : le dataset choisi pour entraîner le modèle sur.
- $N$  : le nombre de scènes du  $D$ .
- $OCoi$  : le nombre de scènes du  $D$  auxquelles l'objet  $O_i$  leur fait partie.
- $AllF$  : la fréquence totale des objets apparaissant dans les scènes du  $D$ .
- $Foi$  : la fréquence totale de l'objet  $O_i$  dans les scènes du  $D$ .

L'importance de l'objet  $O_i$  dans  $D$  est calculé comme le montre l'*équation 2.4* :

$$I(O_i) = \frac{OC_{O_i}/N}{F_{O_i}/AllF} \quad (2.4)$$

Après avoir présenté l'input du modèle de l'identification des relations, nous passons, dans ce qui suit, à la présentation de son output.

#### **b - Sortie du modèle**

Vu que pour chaque scène, il peut y avoir une variété de relations dépendant de ses objets, et que le fait d'avoir un vecteur *One-Hot Encoded* reflétant toutes les relations pouvant être générées aura une très grande taille dépendant de toutes les combinaisons pouvant être générées à partir des objets du

dataset, nous avons décidé d'utiliser un modèle génératif et par conséquence ne pas avoir nécessairement une sortie figée.

Ce que nous avons adopté comme représentation de la sortie est de représenter chaque relation (unaire ou binaire) par ses objets et en les séparant par un symbole (par exemple, #), l'exemple suivant donne plus de clarification :

Soit les relations suivantes comme sortie d'un modèle :

$$(O_1, O_2), (O_4, O_3), (O_5)$$

La représentation de cette sortie sera sous la forme suivante :

$$O_1 \ O_2 \ # \ O_4 \ O_3 \ # \ O_5$$

À noter que pour certains cas où il y aura beaucoup des relations résultantes. Ainsi, il est nécessaire de choisir celles les plus importantes pour en parler dans la description de la scène. De ce fait, il paraît indispensable dans un premier temps d'établir un ordre entre les relations résultantes et ensuite de ne garder qu'un seul nombre de relations, jugé suffisant, pour décrire la scène en question.

L'ordonnancement des relations se fait de la manière suivante, soient :

- $I(O_i)$  l'importance de l'objet  $O_i$  dans le dataset (voir la partie "*a - Entrée du modèle*").
- $(O_1, O_2), (O_3), \dots, (O_i, O_j), \dots, (O_n, O_m)$  les relations existantes entre les objets de la scène en entrée.
- $RI((O_i, O_j))$ , l'importance de la relation  $(O_i, O_j)$ .

Les importances des relations sont calculées comme suit :

- Pour une relation unaire :  $RI((O_i)) = 2 * I(O_i)$
- Pour une relation binaire :  $RI((O_i, O_j)) = I(O_i) + I(O_j)$

Une fois la présentation de l'entrée et la sortie du modèle de l'identification des relations achevée, nous nous intéressons, dans ce qui suit, au modèle lui-même.

### c - Le modèle

Comme mentionné dans la partie "*b - Sortie du modèle*", le modèle adopté est génératif et reçoit comme entrée une liste ordonnée d'objets, ce qui correspond exactement à ce que les LSTMs prennent en entrée. Pour cela, nous avons utilisé un LSTM pour la tâche d'identification des relations.

Ce modèle est entraîné sur un dataset ayant bien évidemment pour chaque scène la liste de ses objets et les relations existant entre eux (qu'elles soient unaires ou binaires).

La *figure 2.10* présente un exemple du processus de l'identification des relations entre les objets d'une scène.

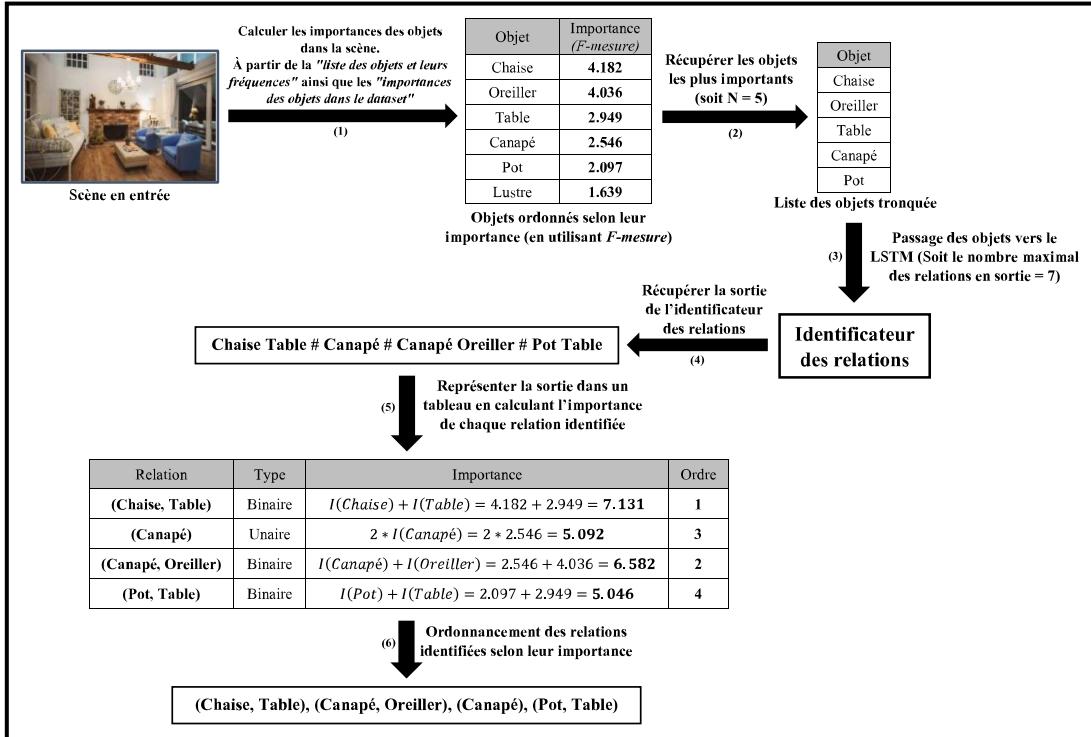


Figure 2.10 : Exemple de l’identification des relations d’une scène

Le fait qu’il y a certains datasets célèbres et très répandus pour l’interprétation des scènes mais n’ayant pas cette structure (i.e. ils ne donnent pas les relations existantes entre les objets) nous a poussé à rechercher des datasets ayant cette structure (e.g. Visual Genome [16]). Mais, le problème rencontré avec le changement du dataset est la différence entre les objets. Pour faire face à ce problème, nous avons fait une adaptation des objets du dataset choisi avec les objets du(des) dataset(s) de l’interprétation.

Les relations identifiées, résultantes du modèle de l’identification des relations, restent vides de tout sens sémantique. Pour cela, dans la section suivante, nous procédons à la présentation des approches proposées pour la définition (attribution d’un sens) des relations identifiées d’une scène.

## 2) Définition des relations dans la scène

L’identification des relations entre les objets présents dans une scène est le premier pas vers l’interprétation de cette dernière. Cependant, malgré l’importance de l’étape précédente, les relations obtenues sont abstraites et restent vides de tout sens sémantique.

Pour remédier à cela, une définition des relations est nécessaire. Une telle définition vise à déterminer la description d’une relation, représentée par la description pour un objet (dans le cas d’une relation unaire) et la liaison entre le premier et le deuxième objet (dans le cas d’une relation binaire).

L’approche adoptée pour la définition des relations dépend du type de cette relation, unaire ou binaire. De ce qui suit, la définition du premier type des relations (unaires) sera présentée.

### a - Définition des relations unaires

Les relations unaires s’intéressent aux objets pris individuellement. Leur identification permet de spécifier les objets les plus importants et remarquables dans une scène. Quant à leur définition, elle a comme but de leur associer l’attribut (description) le plus adéquat (si possible) permettant ainsi de dévoiler une particularité de l’objet concerné.

Les relations unaires ne contiennent qu’un seul objet. Ces relations ont des types différents des descriptions. La figure 2.11 présente un exemple des divers types des descriptions des relations unaires.

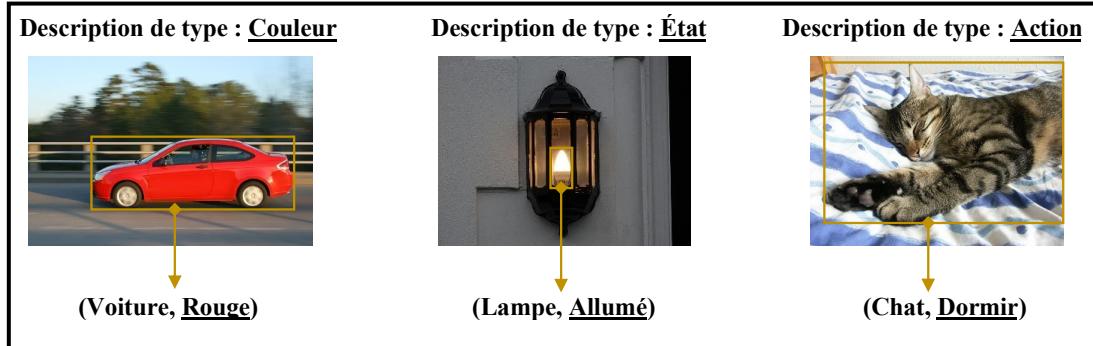


Figure 2.11 : Exemple des types des descriptions des relations unaires

La détermination de la description d'une relation unaire peut être ambiguë, surtout dans le cas où qu'une scène est considérée (contrairement aux flux vidéo).

La figure 2.12 montre deux exemples de définition d'une relation unaire dans une scène :

- La relation à gauche, avec description de type "couleur", peut être facilement définie.
- La relation à droite pose un problème d'ambiguité pour la définir. Est-ce que la personne est en train de : Marcher, parler, ou bien un mélange entre les deux actions.

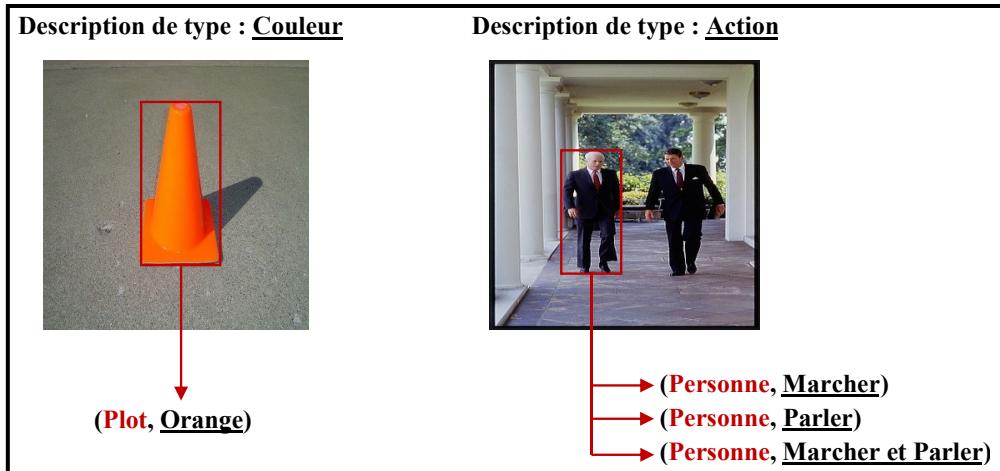


Figure 2.12 : Exemples de définition d'une relation unaire dans une scène

Les classifieurs des relations unaires sont définis de telle sorte à prédire des catégories (et donc, des descriptions) de même nature, tel que : la taille (petit, grand), matière (bois, métal, plastique, cuir, etc.), état (assis, debout, marcher, courir, etc.), etc.

Les définitiseurs des relations unaires sont entraînés séparément pour chaque objet. Une telle stratégie est argumentée par le fait que même si les catégories du classifieur sont similaires pour deux objets (e.g. "petit" / "grand"), la prédiction d'une catégorie particulière diffère d'un objet à autre (e.g. "petit chat" est différent visuellement de "petit laptop"). L'argument précédent est maintenu, bien-évidemment, pour les catégories différentes des objets (e.g. "un chat assis" ne peut pas être regroupé avec "un laptop allumé").

Ces classifieurs sont des CNNs (Convolutional Neural Networks). L'utilisation d'une telle architecture est due au fait que les relations unaires concernent les objets eux-mêmes, ainsi, la définition d'une telle relation nécessite une analyse directe (en utilisant les pixels, pas les bounding-boxes par exemple) de la partie de la scène qui représente cet objet.

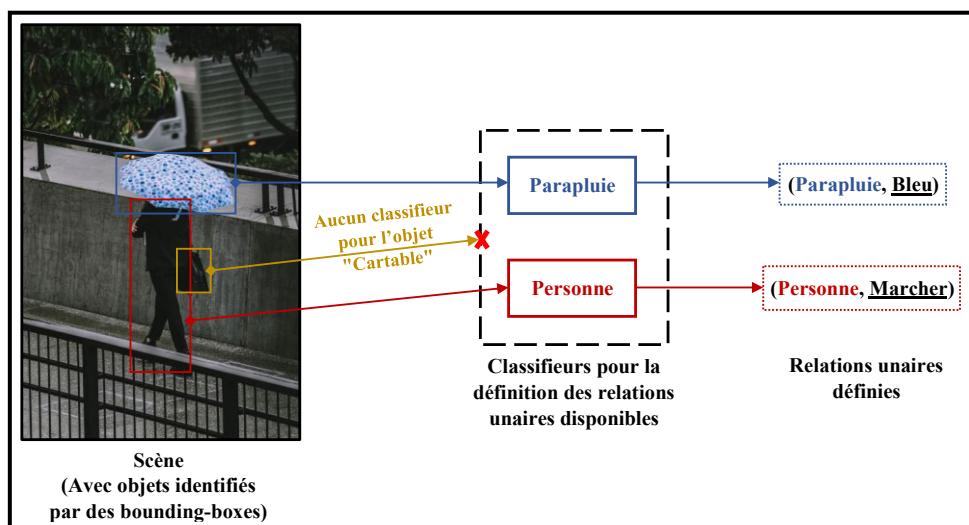
Cependant, afin d'obtenir une exactitude acceptable des définitiseurs (classifieurs de définition) des relations unaires, un ensemble de contraintes doit-être défini. Ces contraintes sont :

- Choisir des catégories du définitisseur qui appartiennent au même type (couleur, état, action, etc.).

- Construire un dataset d'apprentissage équilibré, dans lequel, les catégories contiennent un nombre d'instances (d'apprentissage et de test) à peu près similaire.
- Effectuer les tests nécessaires pour déterminer les meilleures valeurs des paramètres du définitisseur.
- Fixer un seuil minimal pour la précision du classifieur. Si cette dernière est inférieure au seuil fixé, le classifieur n'est pas pris en considération.

Pour le nombre des classificateurs à utiliser, il n'est pas limité, le plus, le mieux. Ainsi, le cas parfait correspond à la situation où le nombre des classificateurs est égal au nombre des objets.

La figure 2.13 présente un exemple pratique de définition des relations unaires à l'aide des classificateurs pré-entraînés où chaque partie d'une scène, délimitée par un bounding-box, représentant un objet particulier, est injectée dans son classifieur correspondant. La sortie d'un tel classifieur est une structure, qui représente une relation définie, ayant le nom de l'objet et sa description. Certains objets n'ont pas de classificateurs correspondants, ils seront, tout simplement, ignorés (i.e. aucune relation définie pour cet objet ne sera générée).

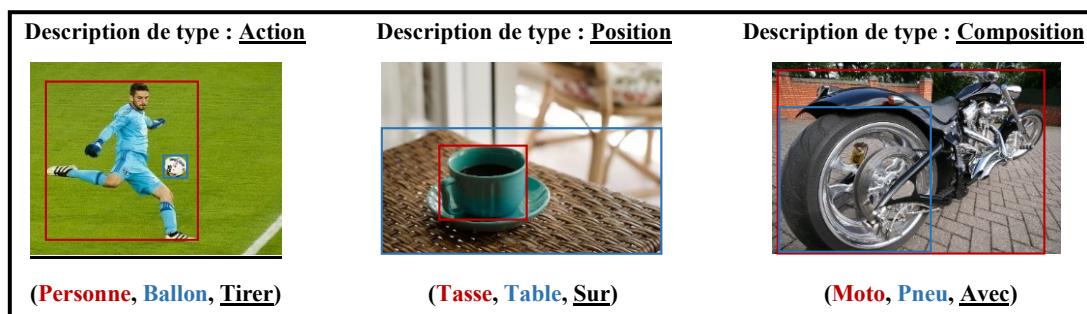


**Figure 2.13 : Exemple de définition des relations unaires en utilisant des classificateurs pré-entraînés**

Une fois les relations unaires définies, nous nous intéressons, par la suite, par la définition des relations binaires.

#### b - Définition des relations binaires

Les relations binaires contiennent deux objets impliqués. À la différence des relations unaires, celles binaires ont des types de descriptions plus variées. Un exemple de ces dernières est présenté dans la figure 2.14, dans laquelle, le premier objet ( $o_1$ ) est en "rouge" et le deuxième objet ( $o_2$ ) est en "bleu".



**Figure 2.14 : Exemple des types des natures des relations binaires**

La méthode proposée pour la définition des relations binaires consiste à extraire un ensemble de caractéristiques (des deux objets de la relation binaire) à partir des bounding-boxes qui représentent

l'emplacement des objets dans la scène. Ces caractéristiques, une fois encodées, sont injectées dans un réseau de neurones feedforward afin de prédire la description de la relation courante le plus plausible.

Nous introduirons, ci-dessous, la première étape de la définition des relations binaires qui consiste à extraire les caractéristiques des objets.

### *i / Extraction des caractéristiques des objets*

L'idée principale est d'identifier, pour chaque objet d'une relation binaire, un ensemble de caractéristiques unaires (propres à lui) et binaires (entre deux objets). Cela est fait en analysant les bounding-boxes de chaque objet.

Le choix d'une telle approche est argumenté par les deux faits suivants :

- Les caractéristiques extraites forment, en quelque sorte, une discréétisation des propriétés (individuelles ou relatives) des bonding-boxes des objets. Ainsi, le classifieur à entraîner (dans ce cas, un réseau de neurones) aura une architecture bien plus légère (moins complexe) que celle en utilisant les bounding-boxes de façon brute (par exemple, avec les CNNs [*Convolutional Neural Network*]).
- Les caractéristiques extraites sont générales (i.e. elles ne sont pas liées à un "nom" d'un objet particulier), ce qui permet de définir des relations non-vues auparavant, en imitant un aspect important du raisonnement humain : La généralisation. Par exemple : Si le classifieur est entraîné sur la relation (personne, cheval, *sur*), et une nouvelle relation est identifiée (personne – chameau), il pourra parfaitement généraliser en définissant cette relation par la proposition "sur" (en supposant que la relation [personne – cheval] a les mêmes caractéristiques que celles de la relation [personne – chameau]).

Afin de palier au problème de l'imprécision des bounding-boxes, des seuils (ou intervalles) ont été définis pour déterminer les différentes caractéristiques. L'ensemble des caractéristiques définies est divisé en caractéristiques unaires et binaires, il se présente comme suit :

#### *Caractéristiques unaires*

Les caractéristiques unaires concernent un bounding-box, pris individuellement, uniquement.

- **Importance** : Désigne à quel point l'objet est remarquable dans la scène, le degré de cette dernière est défini selon le ratio entre la surface de l'objet et la surface (aire) entière de la scène. Les valeurs possibles de l'*importance* sont décrites dans le *tableau 2.1*.

**Tableau 2.1 : Valeurs de la caractéristique "Importance" et leurs significations**

Valeur de "Importance"	Signification
0	L'objet courant a <b>la plus petite</b> importance dans la scène
1	L'objet courant a une importance <b>meilleure que celle du '0'</b>
2	L'objet courant a une importance <b>meilleure que celle du '1'</b>
3	L'objet courant a <b>la plus grande</b> importance dans la scène

La *figure 2.15* présente un exemple sur la détermination de l'*importance* d'un objet (une feuille) dans une scène.

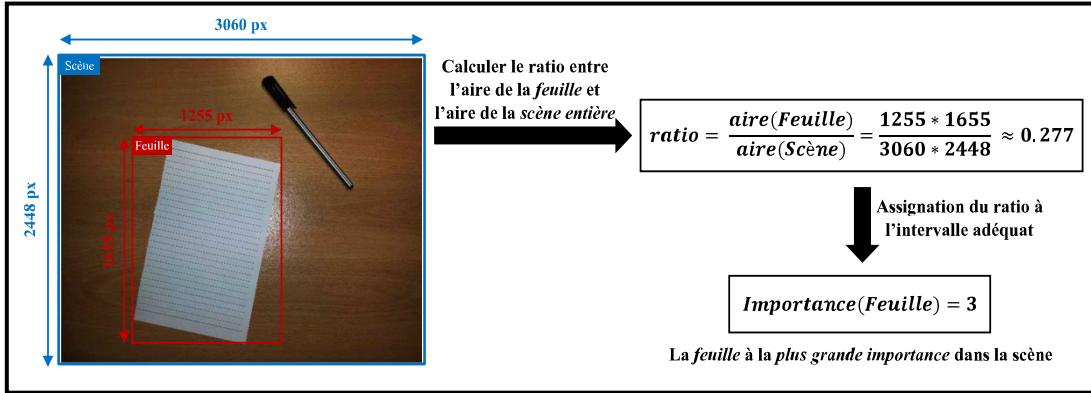


Figure 2.15 : Exemple sur la détermination de l'importance d'un objet dans une scène

### Caractéristiques binaires

Les caractéristiques binaires sont relatives, définies pour chaque couple d'objets, un objet par rapport à un autre.

- IOU (Intersection Over Union) : Il s'agit de calculer le ratio entre l'intersection (l'aire de la zone résultante par l'intersection de deux bounding-boxes) et l'union (la somme des aires des deux bounding-boxes, sans l'aire de l'intersection). Les valeurs possibles de l'*IOU* sont décrites dans le tableau 2.2.

Tableau 2.2 : Valeurs de la caractéristique "IOU" et leurs significations

Valeur de "IOU"	Signification
<b>0</b>	L'intersection entre les deux objets est <b>très petite</b>
<b>1</b>	L'intersection prend <b>une petite surface</b> de l'objet courant
<b>2</b>	L'intersection prend <b>une surface moyenne</b> de l'objet courant
<b>3</b>	L'intersection prend <b>une grande surface</b> de l'objet courant
<b>4</b>	L'objet courant <b>contient (inclus)</b> l'autre objet
<b>-1</b>	L'intersection entre les deux objets est <b>vide</b>

La figure 2.16 présente un exemple sur la détermination de l'*IOU* entre deux objets (une feuille et un stylo) dans une scène.

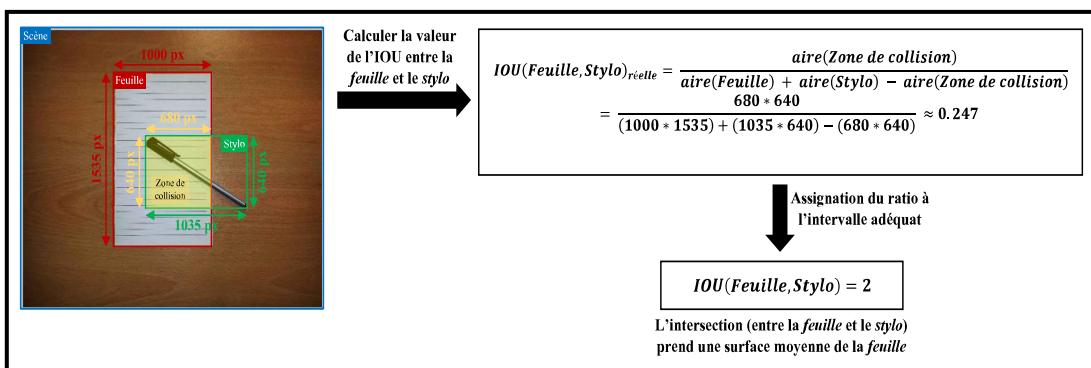


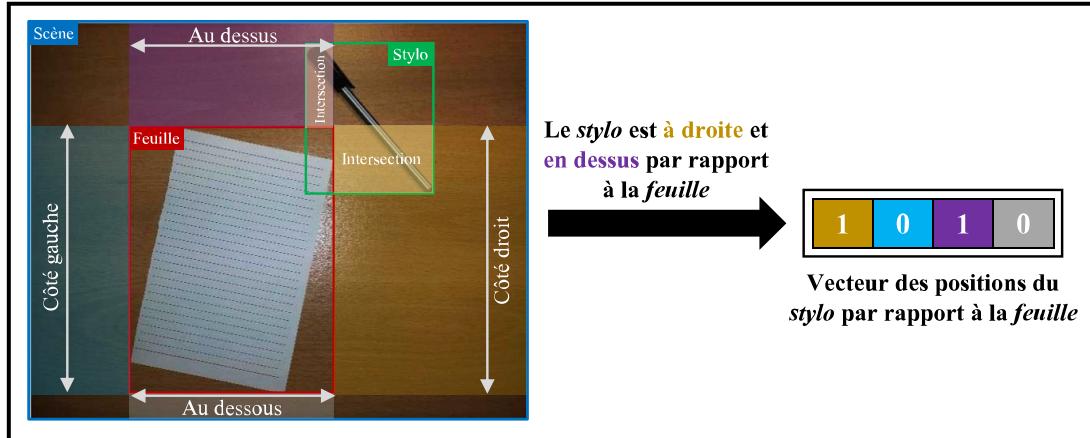
Figure 2.16 : Exemple sur la détermination de l'*IOU* entre deux objets dans une scène

- Positions : C'est l'emplacement d'un objet par rapport à l'objet courant, selon les quatre directions. L'information est mise dans un vecteur de quatre colonnes qui représentent, respectivement, "côté gauche", "côté droit", "en dessus (haut)" et "en dessous (bas)". Les valeurs possibles de *positions* sont décrites dans le tableau 2.3.

**Tableau 2.3 : Valeurs de la caractéristique "Positions" et leurs significations**

Valeur "Positions"	Signification
<b>0</b>	L'autre objet est <b>à droite</b> de l'objet courant
<b>1</b>	L'autre objet est <b>à gauche</b> de l'objet courant
<b>2</b>	L'autre objet est <b>au-dessus</b> de l'objet courant
<b>3</b>	L'autre objet est <b>au-dessous</b> de l'objet courant

La figure 2.17 présente un exemple sur la détermination des positions d'un objet (un stylo) par rapport à un autre (une feuille) dans une scène.



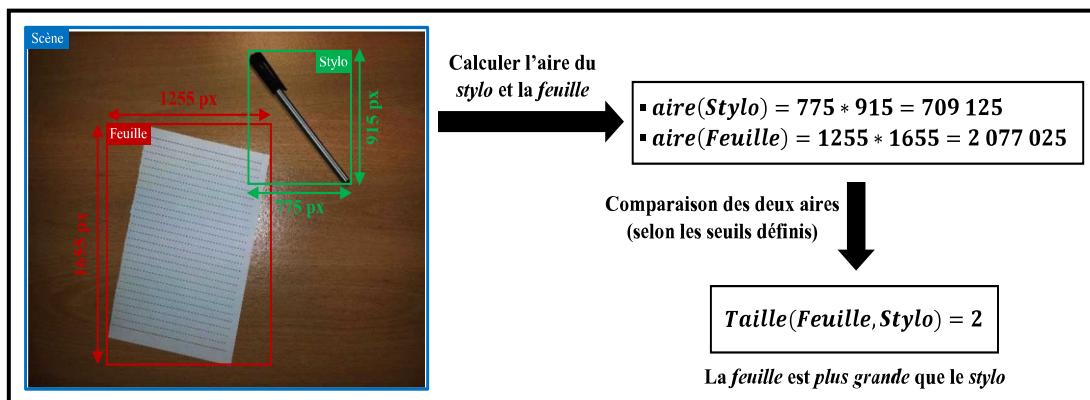
**Figure 2.17 : Exemple sur la détermination des positions d'un objet par rapport à un autre dans une scène**

- **Taille (Size)** : C'est la comparaison de la surface de l'objet courant par rapport à un autre objet. Les valeurs possibles de *taille* sont décrites dans le tableau 2.4.

**Tableau 2.4 : Valeurs de la caractéristique "Taille" et leurs significations**

Valeur de "Taille"	Signification
<b>1</b>	L'objet courant a <b>la même taille (ou presque)</b> que l'autre objet
<b>2</b>	L'objet courant est <b>plus grand</b> que l'autre objet
<b>0</b>	L'objet courant est <b>plus petit</b> que l'autre objet

La figure 2.18 présente un exemple sur la détermination de la taille d'un objet (une feuille) par rapport à un autre (un stylo) dans une scène.



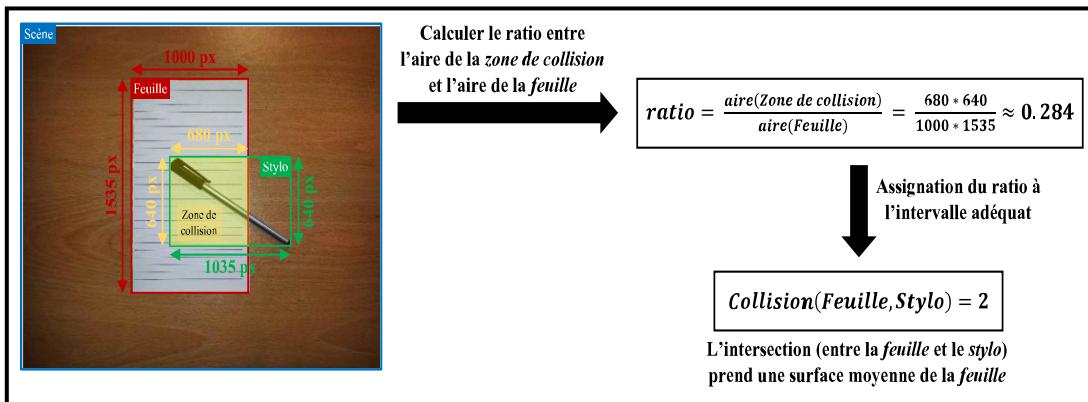
**Figure 2.18 : Exemple sur la détermination de la taille d'un objet par rapport à un autre dans une scène**

- Collision: C'est le degré de chevauchement d'un objet par rapport à l'objet courant. La valeur est déterminée selon le ratio entre la surface d'intersection (des deux objets) et la surface de l'objet courant, i.e. combien de surface prend cette intersection de l'objet courant. Les valeurs possibles de *Collision* sont décrites dans le *tableau 2.5*.

**Tableau 2.5 : Valeurs de la caractéristique "Collision" et leurs significations**

Valeur de "Collision"	Signification
<b>0</b>	Les deux objets <b>se touchent (ou presque)</b> uniquement
<b>1</b>	L'intersection prend <b>une petite surface</b> de l'objet courant
<b>2</b>	L'intersection prend <b>une surface moyenne</b> de l'objet courant
<b>3</b>	L'intersection prend <b>une grande surface</b> de l'objet courant
<b>4</b>	L'objet courant <b>contient (inclus)</b> l'autre objet
<b>-1</b>	Pas de collision (les deux objets sont distants)

La *figure 2.19* présente un exemple sur la détermination de la collision d'un objet (un stylo) par rapport à un autre (une feuille) dans une scène.



**Figure 2.19 : Exemple sur la détermination de la collision d'un objet par rapport à un autre dans une scène**

- Distance: C'est l'éloignement des deux objets. La projection de la distance, à partir de l'espace des nombres réels vers un espace discret (valeurs avec un sens sémantique), est faite selon le ratio entre la distance réelle (représentée en chiffres) entre les deux objets et la longueur de référence (la distance maximale dans la scène, en appliquant le théorème de Pythagore). Les valeurs possibles de *Distance* sont décrites dans le *tableau 2.6*.

**Tableau 2.6 : Valeurs de la caractéristique "Distance" et leurs significations**

Valeur de "Distance"	Signification
<b>0</b>	L'objet courant <b>touche (ou presque)</b> l'autre objet
<b>1</b>	L'objet courant est <b>près</b> de l'autre objet
<b>2</b>	L'objet courant est <b>à proximité (pas très loin)</b> de l'autre objet
<b>3</b>	L'objet courant est <b>loin</b> de l'autre objet
<b>-1</b>	Aucune distance séparante (les deux objets sont en collision)

La *figure 2.20* présente un exemple sur la détermination de la distance qui sépare deux objets (une feuille et un stylo) dans une scène.

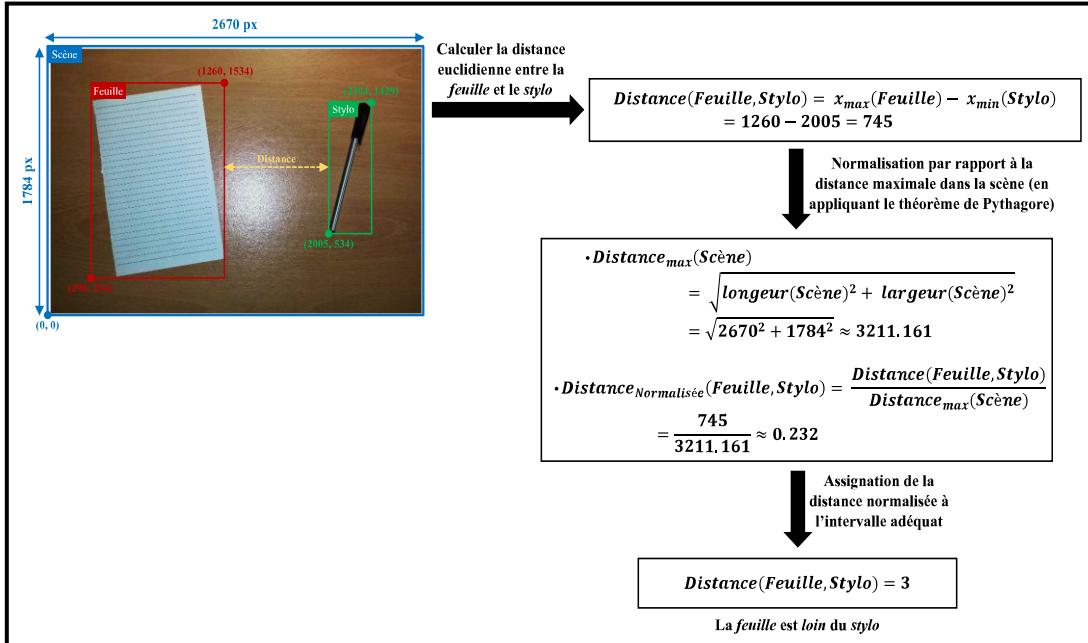


Figure 2.20 : Exemple sur la détermination de la distance qui sépare deux objets dans une scène

### ii / Prédiction de la description à partir d'un vecteur de caractéristiques

Une fois les caractéristiques extraites à partir d'une relation binaire, elles sont mises dans un vecteur structuré où chaque (ensemble) de colonne(s) désigne une caractéristique particulière. La figure 2.21 présente la structure d'un tel vecteur.

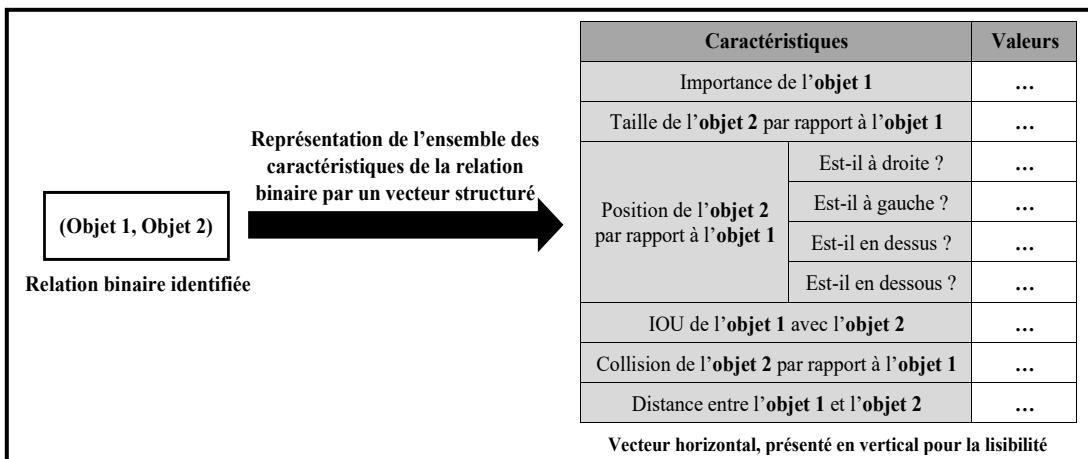
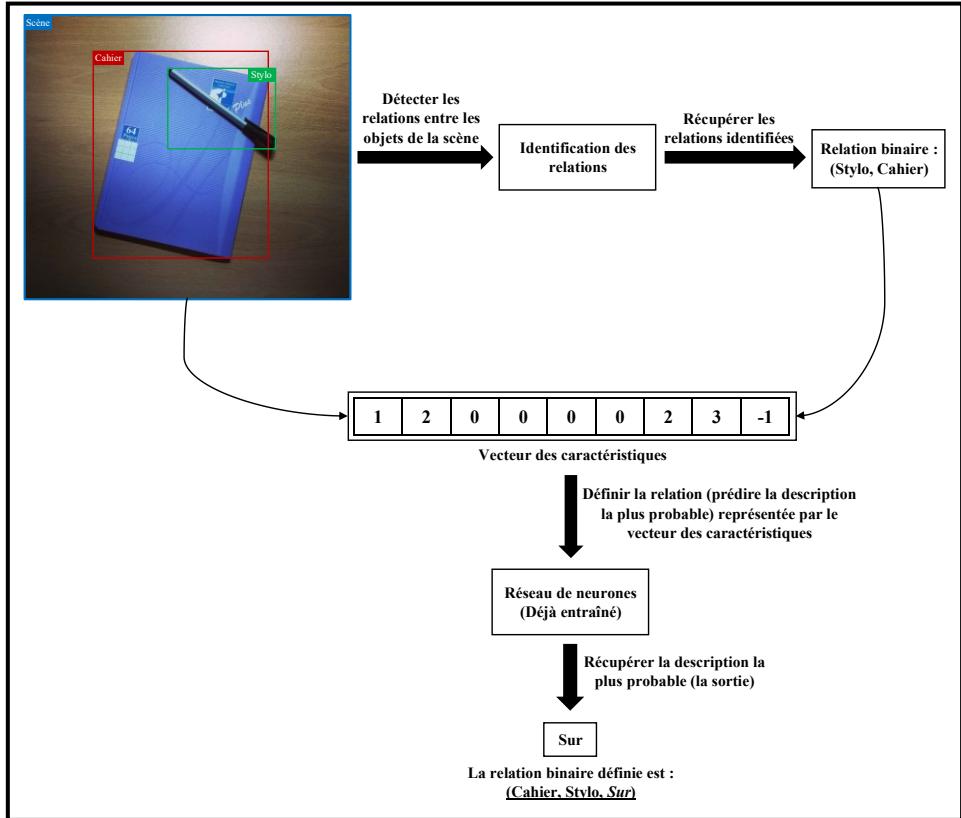


Figure 2.21 : Structure d'un vecteur de caractéristiques pour une relation binaire

Le vecteur des caractéristiques est injecté dans un réseau de neurones pré-entraîné dans le but de prédire la description le plus probable pour une relation binaire donnée. La figure 2.22 montre un exemple de définition d'une relation binaire, en démarrant par une scène (avec objets identifiés), passant par l'identification des relations (dans ce cas, une seule relation binaire) et la construction du vecteur des caractéristiques et finissant par la prédiction de la description le plus plausible pour la relation binaire.



**Figure 2.22 : Exemple de définition d'une relation binaire**

L'achèvement de la présentation détaillée des deux approches proposées pour la définition des relations binaires, nous mène à la phase finale : La génération des interprétations (descriptions) des scènes. Cette dernière mettra en œuvre les résultats des phases précédentes (à savoir : La classification avec ses variantes, l'identification des relations et leur définition) pour créer une description textuelle pour une scène donnée.

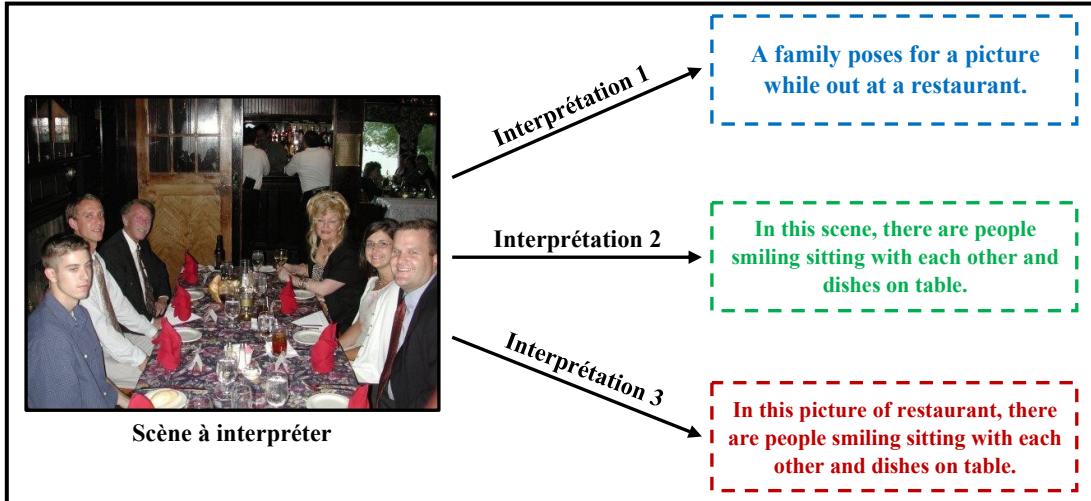
### 3) Génération des interprétations des scènes

L'interprétation d'une scène est une description textuelle, en langage naturel, des différents faits concernant la scène. Nous distinguons trois types d'interprétations des scènes :

- Descriptions présentant l'idée générale qui existe dans la scène : Elles s'intéressent à l'extraction du sens derrière une scène ou l'action (visible ou cachée) dans la scène.
- Descriptions présentant les interactions entre les objets d'une scène : Elles concernent les relations brutes qui existent entre les objets qui composent une scène.
- Descriptions hybrides : Elles présentent un mélange entre les deux types précédents, avec un éventuel biaisement, plus ou moins important, pour l'un des deux types.

La figure 2.23 présente un exemple de différentes descriptions générées pour scène :

- La première description (couleur bleue) est une description présentant l'idée générale qui existe dans la scène, l'accent est mis sur l'action déduite en analysant la scène (non-visible du premier coup).
- La deuxième description (couleur verte) est une description présentant les interactions entre les objets d'une scène, il est clair que la description générée ne contient que les relations phares de la scène.
- La troisième description (couleur rouge) est une description hybride, nous remarquons que parmi les informations existantes, à part les interactions entre les objets, nous trouvons l'ensemble des catégories auxquelles cette scène fait partie.



**Figure 2.23 : Exemple des différentes descriptions générées pour une scène**

Notre approche proposée pour l'interprétation des scènes génère des descriptions hybrides. Le choix d'un tel type de descriptions est justifié par la variété des informations présentées dans la description. Cette variété permettra de révéler différents aspects (ou points de vus) relatifs à la scène, et par conséquence, apporter une meilleure compréhension à la scène.

Notons que les descriptions générées sont en Anglais. Cela est dû à la disponibilité des datasets qui ont été créés avec cette langue. Cependant, rien n'empêche d'utiliser une autre langue (l'approche proposée est générique), par exemple, en traduisant les datasets déjà existants vers la langue désirée ou en créant de nouveaux datasets.

Afin de générer une interprétation d'une scène, l'approche proposée procède en trois étapes successives : Raffinement du graphe des relations d'une scène, construction des phrases et la génération de la description.

Nous débutons, dans ce qui suit, par la première étape : Raffinement du graphe des relations d'une scène.

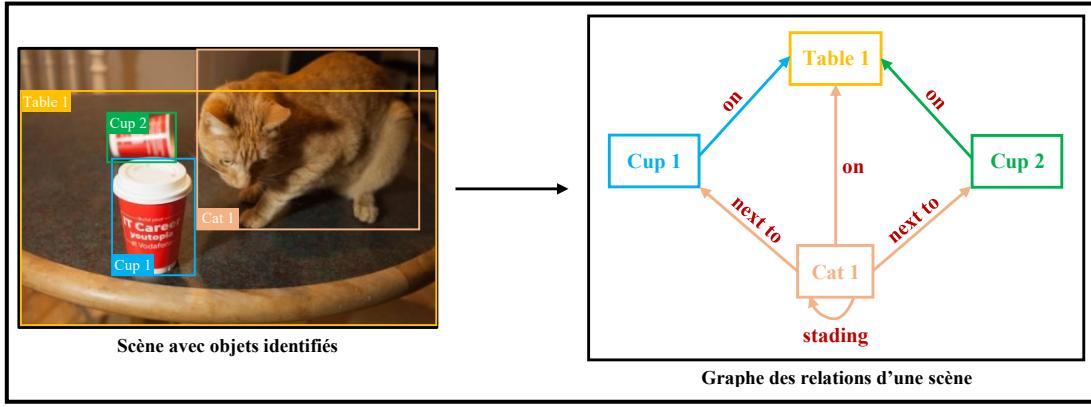
#### *a - Raffinement du graphe des relations d'une scène*

Pour faire fonctionner l'approche proposée de l'interprétation des scènes, nous faisons appel à la structure de graphe. Ce dernier permet de représenter, de manière claire et concise, l'ensemble des relations unaires et binaires, identifiées et définies, entre les objets d'une scène.

Le graphe des relations d'une scène est défini par un ensemble de nœuds (sommets) et d'arcs (arrêtes) où :

- Les nœuds représentent les différents objets d'une scène. Un identifiant (valeur entière incrémentale) est associé à chaque nœud, ce qui permettra de représenter toutes les instances des objets dont le nom est similaire.
- Les arcs sont étiquetés, ils représentent les relations entre les nœuds (objets) dont la description est l'étiquette de l'arc. En particulier, une relation unaire est représentée par une boucle d'un nœud vers lui-même, et une relation binaire est représentée par un arc, d'un nœud vers un autre.

La figure 2.24 présente un exemple d'un graphe des relations d'une scène.



**Figure 2.24 : Exemple d'un graphe des relations d'une scène**

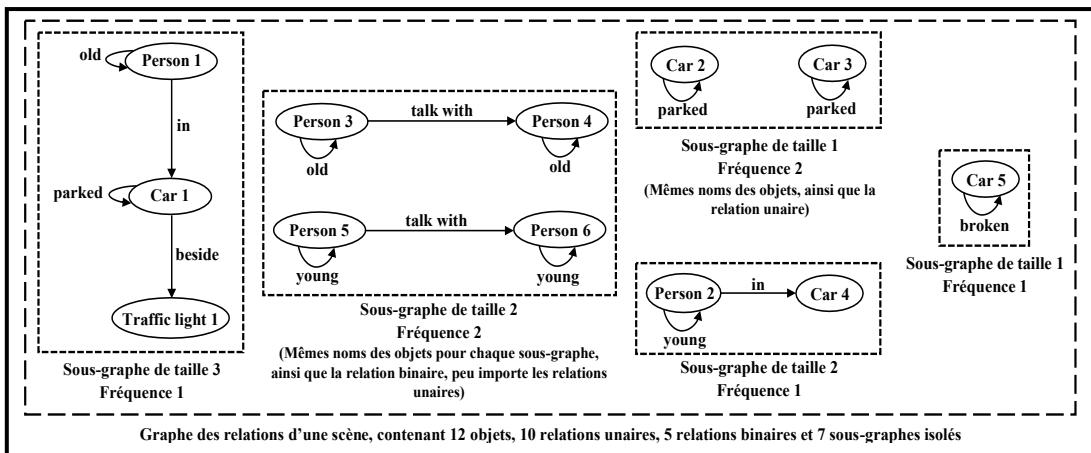
Une fois construit, le graphe des relations obtenu est raffiné, dans le but de rendre possible son exploitation par le générateur des descriptions des scènes proposé. Pour cela, nous procédons par deux étapes.

#### i / Regroupement des sous-graphes isolés similaires

Cette étape consiste à établir une structure dont chaque entrée est un sous-graphe (du graphe des relations général) et sa fréquence d'apparition. Deux sous-graphes sont regroupés s'ils sont :

- Isolés : Les nœuds du sous-graphe sont en relation uniquement avec les autres nœuds du même sous-graphe.
- Similaires : Les nœuds et les relations binaires d'un sous-graphe sont exactement les mêmes que pour l'autre sous-graphe. Pour les relations unaires, elles ne sont pas prises en compte (elles doivent être similaires pour le même nœud des deux sous-graphes) que pour les sous-graphes de taille 1 (contenant un seul nœud).

La figure 2.25 montre un exemple d'établissement d'une structure des fréquences lors de regroupement des sous-graphes isolés similaires à partir d'un graphe des relations d'une scène.



**Figure 2.25 : Exemple de regroupement des sous-graphes isolés similaires**

L'intérêt de ce traitement est la génération des phrases (qui constituent l'interprétation) plus concises (par exemple, en ajoutant les marques du pluriel pour les noms des objets des sous-graphes regroupés).

Après la première sous-étape du regroupement des sous-graphes isolés similaires, vient la deuxième sous-étape permettant de supprimer les sous-graphes redondants.

## ii / Suppression des sous-graphes redondants

Une fois la structure des fréquences des sous-graphe isolés similaires établie, nous procédons à la suppression des sous-graphes redondants. Il s'agit d'enlever les entrées de la structure des fréquences précédente (sous-graphes) existantes déjà dans les entrées (sous-graphes) de taille (nombre de noeuds) supérieure. Autrement dit, un sous-graphe isolé de taille "T" est supprimé si et seulement s'il est identifiable dans les sous-graphes isolés ayant une taille supérieure ou égale à "T+1". La figure 2.26 présente un exemple du processus de la suppression des sous-graphes redondants à partir d'une structure des fréquences.

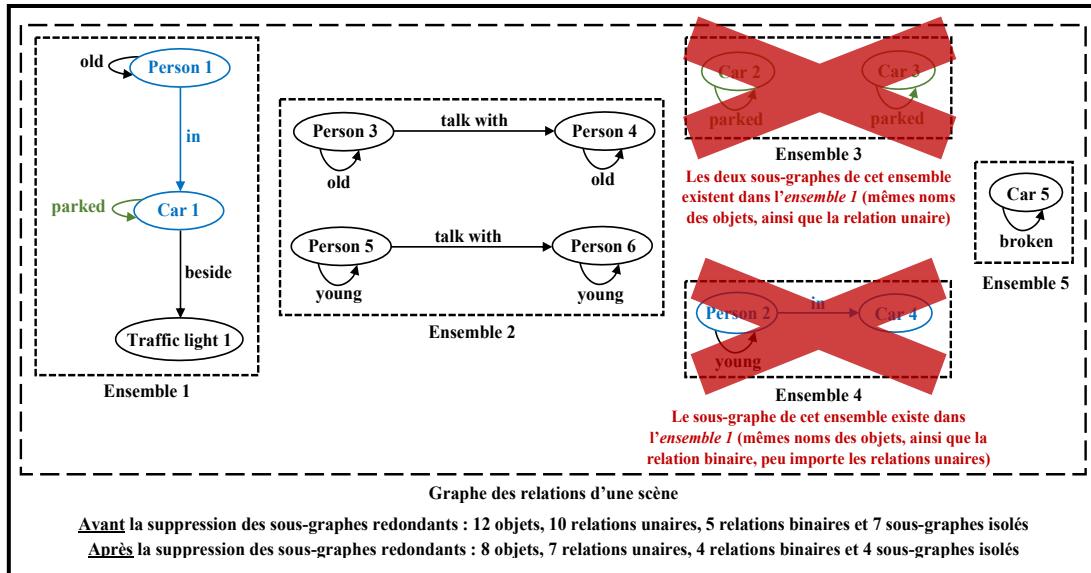


Figure 2.26 : Exemple de la suppression des sous-graphes redondants

Le but de cette sous-étape est d'enlever la redondance potentielle dans les interprétations des scènes générées. Ainsi, chaque phrase constituant l'interprétation apportera une nouvelle information.

Le graphe des relations raffiné est la structure des fréquences résultante après l'application des deux sous-étapes précédentes.

Notons qu'à partir de maintenant, nous utiliserons les deux termes "graphe" et "sous-graphe" mutuellement pour désigner une entrée de la structure des fréquences (graphe des relations raffiné).

Une fois raffiné, le graphe des relations d'une scène est prêt à être exploité pour la construction des phrases (qui constituent, une fois regroupées, l'interprétation de la scène).

### b - Construction des phrases

La construction des phrases est la génération des sous-parties qui constituent l'interprétation d'une scène, où chaque sous-partie apporte davantage informations sur la scène interprétée.

La construction des phrases est faite en parcourant l'ensemble des graphes contenus dans le graphe des relations raffiné, ainsi, la forme de la phrase dépend de la taille du graphe. Pour chaque taille, une panoplie des facteurs est prise en compte, notamment : La similarité des noms des objets, les descriptions des relations (unaires et binaires) et l'orientation des arcs des relations binaires.

Les différents traitements effectués (sur les graphes, pour générer les phrases) sont basés sur des règles. Ces derniers prennent en considération l'ensemble des facteurs précédents afin de générer une phrase cohérente suivant une certaine syntaxe.

La syntaxe est sous forme d'une suite de balises, symboles et fonctions, elle présente la forme des phrases à construire. Les différents éléments d'une syntaxe sont :

- <det> : Un déterminant Anglais (a / an), qui dépend du terme qui le suit.

- <desc\_rel\_unr\_i> : Description de la relation unaire pour l'objet 'i'.
- <desc\_rel\_bin\_i\_j> : Description de la relation binaire entre les deux objets 'i' et 'j' successivement.
- <nom\_obj\_i> : Nom de l'objet i.
- PLURIEL(<nom\_obj\_i>) : Transformer le nom de l'objet 'i', du singulier au pluriel.
- Accolades {...} : L'élément entre les accolades est facultatif.
- Terme entre apostrophes ('...') : Le terme est mis tel qu'il est.

Les règles mises en place ont été définies suivant une analyse manuelle des formes des descriptions faites par les humains. Le but étant de générer des phrases dont l'aspect robotique (résultat statique, redondant) est minimal d'un côté, et de l'autre, obtenir de phrases *human-friendly* (imiter les spécificités des descriptions humaines)

Parmi les résultats de cette analyse manuelle, nous citons est le bornage des cardinalités des objets. En effet, les phrases construites ne considèrent que trois cardinalités : 1 (one), 2 (two) et "more than two". Cette dernière valeur est attribuée aux objets (dans un graphe donné) dont la cardinalité est supérieure ou égale à 3. La définition d'une telle valeur générique se défend par le comportement des humains, qui ont tendance à ne spécifier explicitement que les cardinalités des objets dont le nombre est petit (facilement ou rapidement comptables).

La procédure de construction des phrases traite indépendamment trois tailles des graphes : Un, deux et "trois ou plus". Dans ce qui suit, nous commençons par le premier cas : Graphes de taille 1.

### *i / Traitement des graphes de taille 1*

Les graphes contenant un seul nœud, avec une description (relation unaire) ou non, sont facilement traitables. En effet, la construction de la phrase pour ce cas est intuitive, la syntaxe suivant la fréquence est :

- Fréquence du graphe égale à 1 : <det> {<desc\_rel\_unr\_1>} <nom\_obj\_1>.
- Fréquence du graphe égale à 2 : 'two' {<desc\_rel\_unr\_1>} PLURIEL(<nom\_obj\_1>).
- Fréquence du graphe supérieure ou égale à 3 : 'more than two' {<desc\_rel\_unr\_1>} PLURIEL(<nom\_obj\_1>).

La figure 2.27 présente un exemple du traitement des graphes de taille 1.

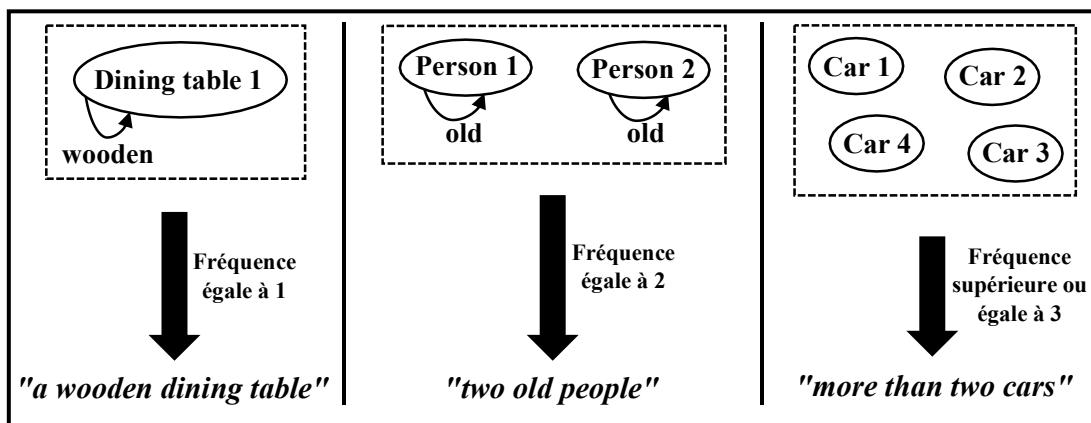


Figure 2.27 : Exemple des phrases construites à partir des graphes de taille 1

Une fois la présentation détaillée du traitement des graphes de taille 1, et les syntaxes associées achevée, nous passons au deuxième cas : Graphes de taille 2.

## *ii / Traitement des graphes de taille 2*

Les graphes avec deux nœuds contiennent un arc de relation binaire, et au plus, deux arcs de relations unaires (dans le cas où chacun des deux nœuds a son propre attribut). Ces graphes sont traités, eux-aussi, selon leur fréquence, nous distinguons deux situations :

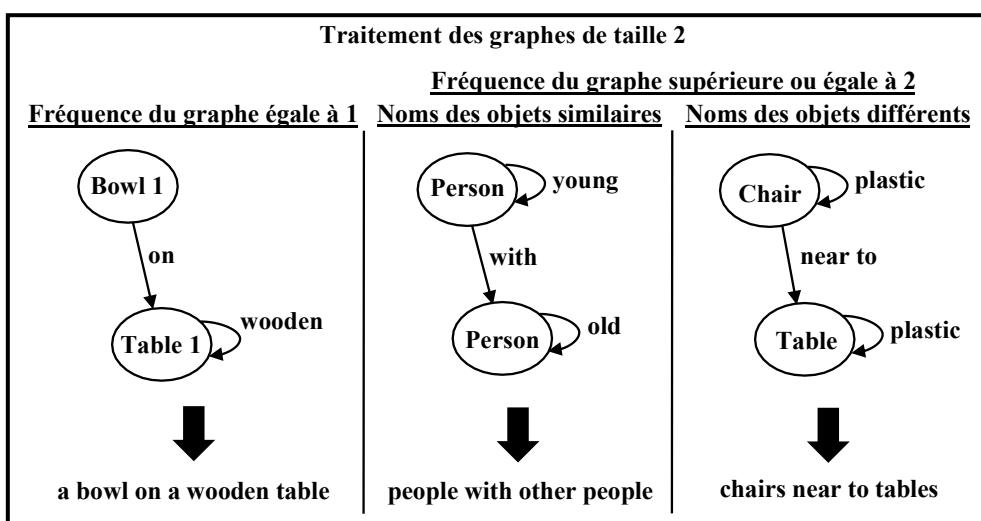
- Fréquence du graphe égale à 1 : La phrase générée est triviale, elle contient deux objets (précédés par un déterminant et éventuellement leur attribut) et la relation binaire reliant ces objets. La syntaxe est : <det> {<desc\_rel\_unr\_1>} <nom\_obj\_1> <desc\_rel\_bin\_1\_2> <det> {<desc\_rel\_unr\_2>} <nom\_obj\_2>.

- Fréquence du graphe supérieure ou égale à 2 : Deux cas se présentent, selon la similarité des noms des objets des deux nœuds :

- Noms des objets similaires : Transformer des noms des objets au pluriel, et ajouter "other" après la description de la relation binaire, et ne pas tenir compte des attributs des objets (relations unaires). La syntaxe est : PLURIEL(<nom\_obj\_1>) <desc\_rel\_bin\_1\_2> 'other' PLURIEL(<nom\_obj\_2>).

- Noms des objets différents : Même syntaxe que pour le cas des noms des objets similaires, mais sans le terme "other". La syntaxe est : PLURIEL(<nom\_obj\_1>) <desc\_rel\_bin\_1\_2> PLURIEL(<nom\_obj\_2>).

La figure 2.28 présente un exemple des phrases construites en traitant les graphes de taille 2 (en tenant compte des différents cas présentés précédemment).



**Figure 2.28 : Exemple de phrases construites à partir des graphes de taille 2**

Après la présentation des différents cas de traitement des graphes de taille 2. Nous passons au dernier cas : Graphes de taille supérieure ou égale à 3.

## *iii / Traitement des graphes de taille supérieure ou égale à 3*

Le traitement des graphes avec trois nœuds ou plus est plus complexe que pour les deux cas précédents (taille 1 et 2). Cette complexité est liée au nombre énorme des cas à considérer. Pour cela, nous nous restreignons aux deux cas seulement :

- Graphes de taille 3, avec exactement deux arcs (trois nœuds connectés, mais ne formant pas de clique<sup>1</sup>) : En fonction du nœud ayant un degré<sup>2</sup> de 2 (référencié par : obj\_deg2), nous définissons trois sous cas, en fonction de l'orientation des arcs (entrants ou sortants) :

<sup>1</sup> Dans la théorie des graphes, une clique est un ensemble de sommets deux-à-deux connectés.

<sup>2</sup> Dans la théorie des graphes, le degré d'un sommet d'un graphe est le nombre d'arcs reliant ce sommet.

▪ Nœud du degré 2 avec deux arcs entrants : Trois situations possibles :

- Mêmes relations, mêmes noms : Les descriptions des deux arcs entrants sont similaires, ainsi que les noms des deux objets (nœuds) à partir lesquelles démarrent ces deux arcs sont similaires. La syntaxe : PLURIEL(<nom\_obj\_1>) <desc\_rel\_bin\_1\_obj\_deg2> <det> <obj\_deg2>.

- Mêmes relations, noms différents : Les descriptions sont similaires, mais les noms des deux objets sont différents. La syntaxe : <det> <nom\_obj\_1> ' and ' <det> <nom\_obj\_2> <desc\_rel\_bin\_1\_obj\_deg2> <det> <obj\_deg2>.

- Relations différentes : Les descriptions des deux arcs entrants sont différents (peu importe les noms des deux objets). La syntaxe : <det> <nom\_obj\_1> <desc\_rel\_bin\_1\_obj\_deg2> <det> <obj\_deg2> ' and ' <det> <nom\_obj\_2> <desc\_rel\_bin\_2\_obj\_deg2> ' the same ' <obj\_deg2>.

La figure 2.29 présente un exemple de traitement des graphes de taille 3, avec exactement deux arcs entrants.

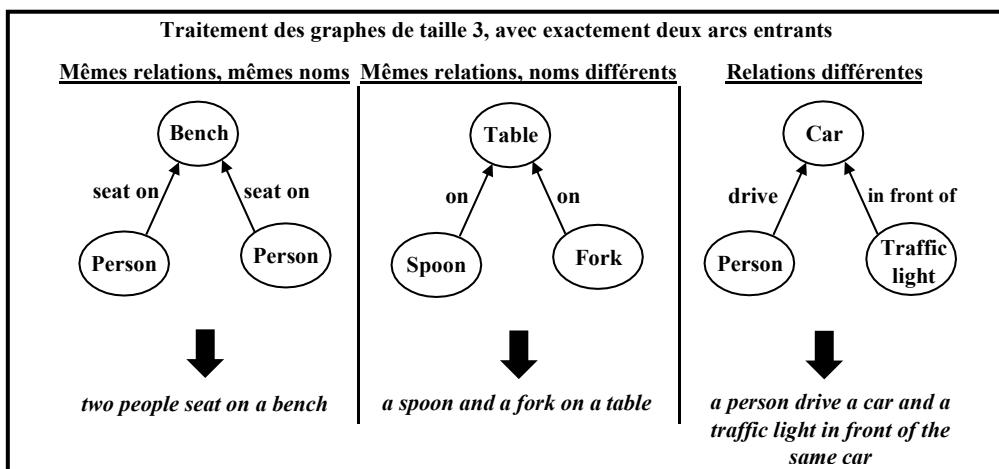


Figure 2.29 : Exemple des phrases construites à partir des graphes de taille 3, avec exactement deux arcs entrants

▪ Nœud du degré 2 avec deux arcs sortants : Deux situations possibles :

- Mêmes noms : <det> <obj\_deg2> ' that ' <desc\_rel\_bin\_obj\_deg2\_1> <det> <nom\_obj\_1> ' , ' <desc\_rel\_bin\_obj\_deg2\_2> ' another ' <nom\_obj\_2>.

- Noms différents : <det> <obj\_deg2> ' that ' <desc\_rel\_bin\_obj\_deg2\_1> <det> <nom\_obj\_1> ' , ' <desc\_rel\_bin\_obj\_deg2\_2> <det> <nom\_obj\_2>.

La figure 2.30 présente un exemple de traitement des graphes de taille 3, avec exactement deux arcs sortants.

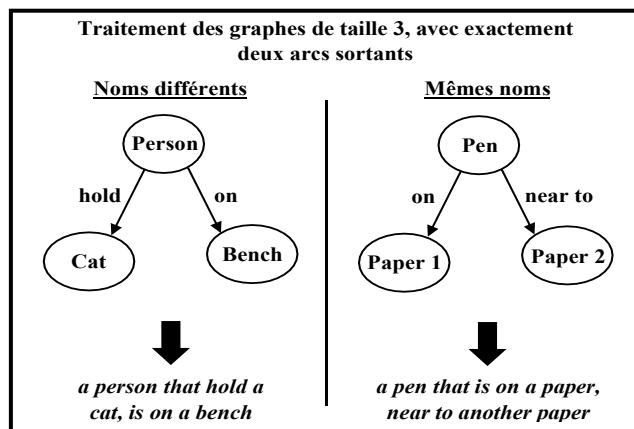


Figure 2.30 : Exemple des phrases construites à partir des graphes de taille 3, avec exactement deux arcs sortants

▪ Nœud du degré 2 avec un arc entrant, et un sortant : Deux situations possibles :

◦ Mêmes noms : <det> <nom\_obj\_1> <desc\_rel\_bin\_1\_obj\_deg2> <det> <obj\_deg2> ' that ' <desc\_rel\_bin\_obj\_deg2\_2> ' another ' <nom\_obj\_2>.

◦ Noms différents : <det> <nom\_obj\_1> <desc\_rel\_bin\_1\_obj\_deg2> <det> <obj\_deg2> ' that ' <desc\_rel\_bin\_obj\_deg2\_2> <det> <nom\_obj\_2>.

La figure 2.31 présente un exemple de traitement des graphes de taille 3, avec exactement deux arcs, un entrant et un sortant.

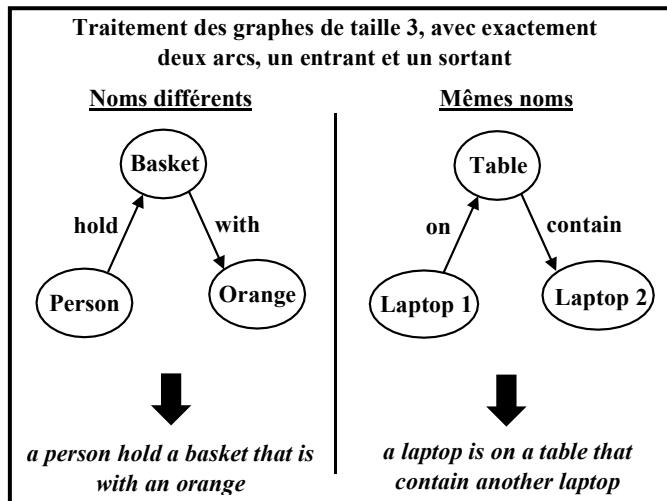


Figure 2.31 : Exemple des phrases construites à partir des graphes de taille 3, avec exactement deux arcs, un entrant et un sortant

• Autres graphes (taille 3 avec trois arcs [une clique], ou bien une taille supérieure à 3) : À cause de la complexité de tels graphes, aucun traitement particulier ne leur a été assigné. Nous procédons à leur division en graphes de taille 2. Pour chaque graphe, nous appliquons les traitements déjà définis pour les graphes de taille 2.

La figure 2.32 présente un exemple de traitement des autres graphes (de taille 3 avec trois arcs ou d'une taille supérieure à 3).

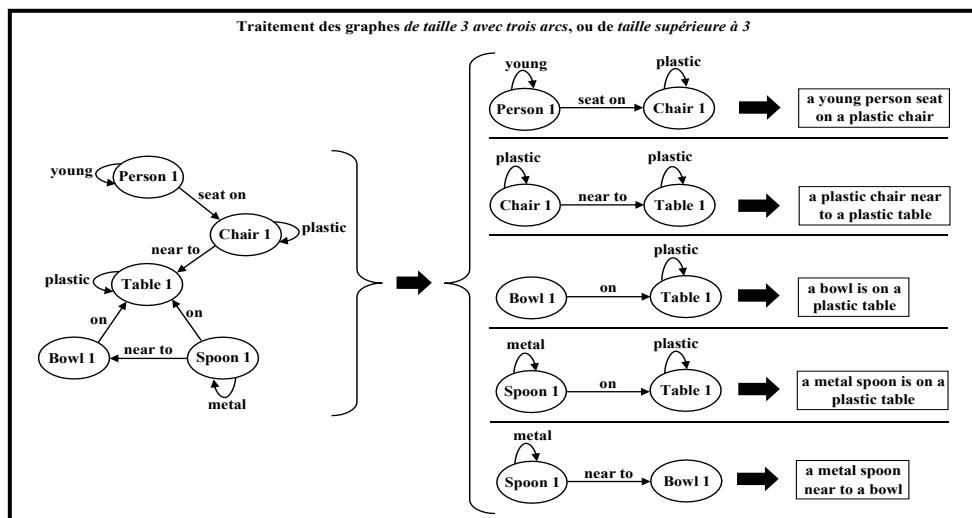


Figure 2.32 : Exemple des phrases construites à partir des graphes de taille 3 avec trois arcs, ou de taille supérieure à 3

La présentation des détails du traitement des graphes de taille supérieure ou égale à 3 clôture la partie de construction des phrases. À partir de cette dernière, une liste de phrases descriptives séparées est obtenue. Dans la section suivante, nous présentons la façon avec laquelle ces phrases sont combinées et reliées afin de générer une interprétation complète et cohérente.

### c - Génération de la description

Dans notre cas, la génération des interprétations est la procédure de création des descriptions textuelles en se basant sur un ensemble de données. Ces dernières consistent en phrases partielles décrivant les objets et les relations entre eux, ainsi que les différentes catégories auxquelles la scène à interpréter appartient.

En effet, le résultat de la partie précédente (l'ensemble des phrases construites) reste insuffisant, même regroupé, pour décrire complètement une scène. Cette incomplaisance reste levée tant que la catégorie de la scène, et donc le contexte général, n'est pas précisé.

Ainsi, dans le but de compléter les descriptions des scènes, nous faisons appel aux différentes variantes de la classification, présentés précédemment, à savoir : La classification en catégories standard, l'enrichissement des scènes et la classification en catégories additionnelles.

Cependant, les deux classifieurs précédents prédisent les catégories à une certaine probabilité (taux de précision). Ainsi, pour gérer cette probabilité, un degré de certitude, sous forme de mots (par exemple : Probablement, peut-être, etc.) doit être attribué à la catégorie prédite suivant sa probabilité. De plus, un ordonnancement doit être établi entre les sorties (catégories prédites) des deux variantes de la classification.

Pour cela, nous présentons l'*algorithme 2.5*, qui reçoit en entrée la liste des catégories prédites (avec leurs probabilités) à partir des deux variantes de la classification des scènes (standard et catégories additionnelles) et renvoi le début de l'interprétation de la scène, qui contient l'information sur les catégories de cette scène.

#### Algorithme 2.5 : Génération de l'information sur les catégories d'une scène

```
Algorithme ajout_prefixe_description;
Entrée: L: Liste des catégories (standards et additionnelles) avec leurs probabilités;
Sortie: Partie de la phrase qui sera mise au début de la description;
Début;
/* Récupérer la catégorie standard la plus probable */
cat_standard ← L['categories_standards'][0];
Si cat_standard['probabilité'] >= 0.6 alors
    retourner('In this ' + cat_standard['nom'] + ' there ');
Fsi;

/* Récupérer les catégories additionnelles ayant une probabilité ≥ 0.6 */
cat_addit ← Récupérer_toutes_les_catégories_additionnelles(L, 0.6);
Si taille(cat_addit) = 1 alors
    retourner('In this ' + cat_addit[0]['nom'] + ' place there ');
Fsi;

Si taille(cat_addit) > 1 alors
    noms_cat_addit ← joindre_noms_catégories(cat_addit);
    retourner('In this ' + noms_cat_addit + ' there ');
Fsi;

/* Récupérer les catégories standards ayant une probabilité ≥ 0.4 */
cats_standards ← Récupérer_toutes_les_catégories_standards(L, 0.4);
/* Récupérer les catégories additionnelles ayant une probabilité ≥ 0.4 */
cats_addit ← Récupérer_toutes_les_catégories_additionnelles(L, 0.4);

Si taille(cats_standards) >= 1 alors
    noms_cats_standards ← joindre_noms_catégories(cats_standards);
```

```

    retourner('In this scene that represents probably ' + noms_cats_standards
+ ' there ');
Fsi;

Si taille(cats_addit) = 1 alors
    retourner('In this scene that represents probably ' +
cats_addit[0]['nom'] + ' place there ');
Fsi;

Si taille(cats_addit) > 1 alors
    noms_cats_addit ← joindre_noms_catégories(cats_addit);
    retourner('In this scene that represents probably ' + noms_cats_addit +
there ');
Fsi;

/* Récupérer les catégories standards ayant une probabilité ≥ 0.2 */
cats_standards ← Récupérer_toutes_les_catégories_standards(L, 0.2);
/* Récupérer les catégories additionnelles ayant une probabilité ≥ 0.2 */
cats_addit ← Récupérer_toutes_les_catégories_additionnelles(L, 0.2);

Si taille(cats_standards) >= 1 alors
    noms_cats_standards ← joindre_noms_catégories(cats_standards);
    retourner('In this scene that represents perhaps ' + noms_cats_standards
+ ' there ');
Fsi;

Si taille(cats_addit) = 1 alors
    retourner('In this scene that represents perhaps ' + cats_addit[0]['nom']
+ ' place there ');
Fsi;

Si taille(cats_addit) > 1 alors
    noms_cats_addit ← joindre_noms_catégories(cats_addit);
    retourner('In this scene that represents perhaps ' + noms_cats_addit +
there ');
Fsi;

retourner('In this scene there ');
Fin;

```

L’interprétation finale se constitue du début renvoyé par l’*algorithme 2.5*, et l’ensemble des phrases (obtenues dans partie “Construction des phrases”) séparées par des virgules.

La *figure 2.33* présente un exemple du processus complet de l’interprétation d’une scène, en commençant par la liste de ces objets, et finissant par l’obtention d’une description textuelle de la scène.

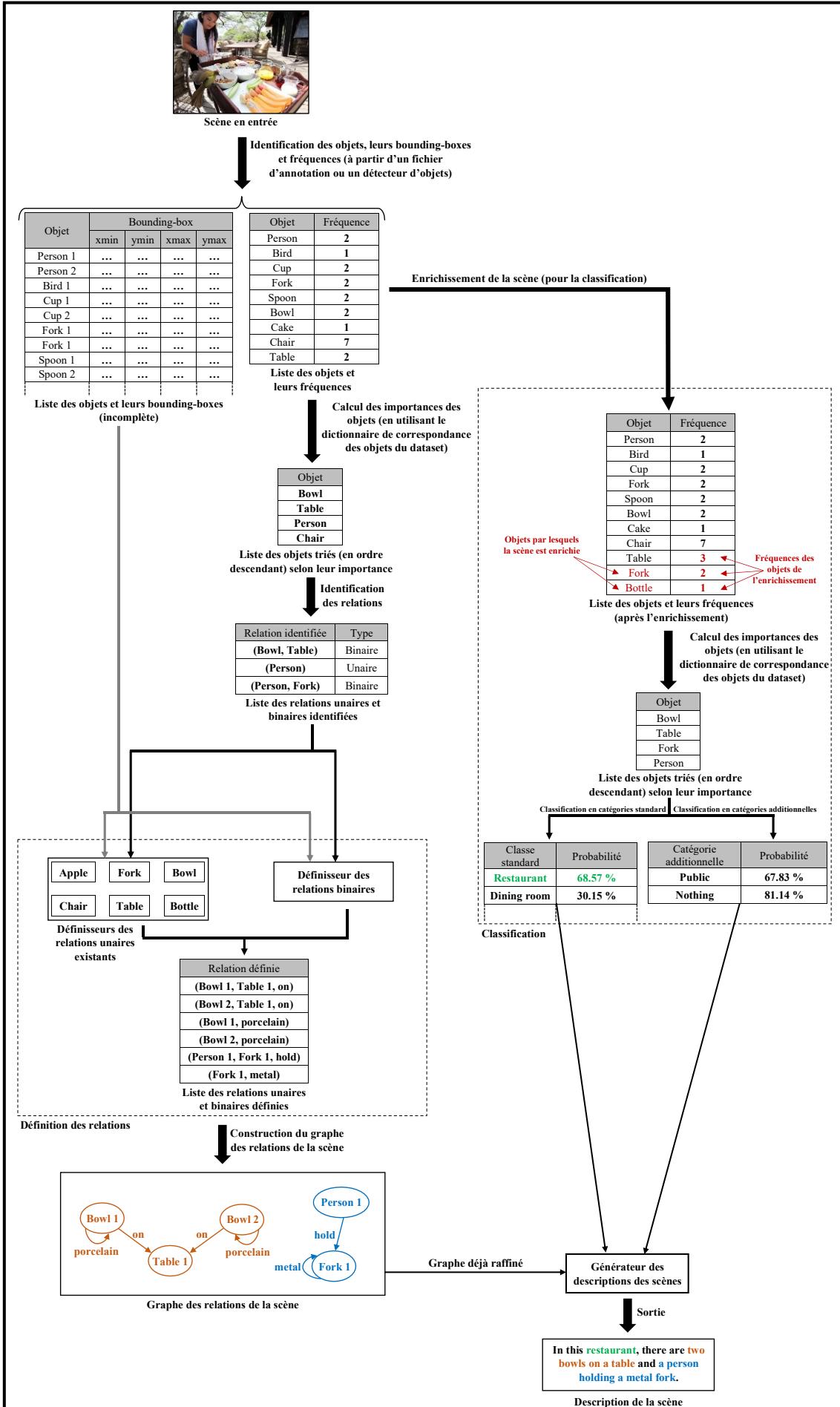


Figure 2.33 : Exemple du processus de l'interprétation d'une scène

La présentation de l'approche proposée pour l'interprétation des scènes, et l'arrivé au but final (avoir une scène interprétée), concluent la conception des approches développées dans notre travail. Dans la section suivante, nous récapitulons les points les importants sur lesquels nous avons parlé dans ce chapitre.

### **2.3 Conclusion**

Au cours de ce chapitre, nous avons mis en évidence les approches développées, et proposées comme solution aux deux thématiques de notre travail : La classification et l'interprétation des scènes. Pour chaque approche, nous avons explicité son fonctionnement et son utilité. De plus, nous avons exposé les relations entre ces approches et la manière de combiner les unes dans les autres.

Dans le chapitre suivant, nous mettons à l'épreuve ces approches, en effectuant une multitude de tests, et en présentant les résultats obtenus, ainsi que l'application conçue.

# Chapitre 3 – Implémentation et Résultats

## 3.1 Introduction

Afin d'évaluer l'efficacité des approches proposées pour la classification et l'interprétation des scènes basées sur les objets dans le "*Chapitre 2 – Conception*", nous présentons, au cours de ce chapitre, la mise en pratique de ces approches. Plus précisément, nous introduisons les outils utilisés (software et hardware) pour l'implémentation et les tests, les différents datasets employés et les résultats obtenus. De plus, nous présentons en détails l'application conçue, entre-autres, ces fonctionnalités et les diagrammes relatifs à son fonctionnement.

Dans la section suivante, nous commençons par la présentation de l'environnement de test (langage de programmation, librairies et la plateforme utilisée).

## 3.2 Outils de test

Pour implémenter nos approches, nous avons opté pour Python [23] comme langage de programmation. Ce dernier étant simple, intuitif et le plus répandu au sein de la communauté scientifique spécialisée dans l'apprentissage automatique. De plus, nous avons utilisé *Tensorflow* [17] comme librairie présentant des fonctions prédéfinies pour l'apprentissage automatique et *Keras* [18] pour faciliter les différents traitements relatifs aux réseaux de neurones.

Nous avons utilisé *Google Colab* [19] comme plateforme de test gratuite et efficace, qui offre entre autres des GPUs et TPUs pour accélérer l'apprentissage et, par conséquent, réduire le temps d'exécution. La *figure 3.1* présente une capture d'écran de la page principale de *Google Colab*.

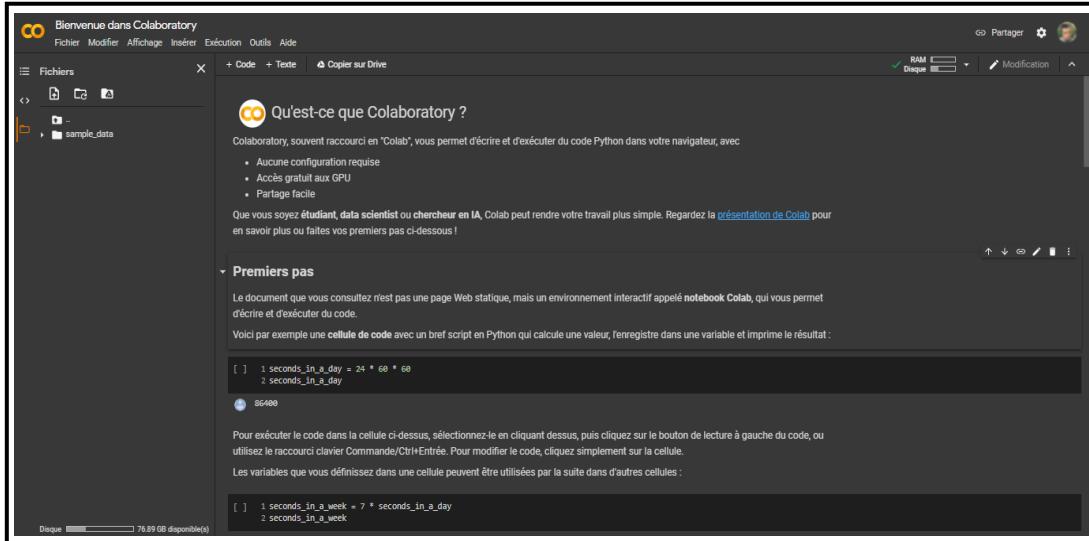


Figure 3.1 : Capture d'écran de la page principale de Google Colab

Les outils présentés seront utilisés dans la section suivante, qui sera consacrée à la partie phare de ce chapitre : Les tests effectués et les résultats obtenus.

## 3.3 Tests et résultats

Au cours de cette section, l'évaluation détaillée des approches proposées dans notre travail sera présentée. Plus précisément, nous nous concentrerons sur la description des datasets choisis avec la

motivation derrière un tel choix, le séquencement des variations des paramètres, les résultats obtenus suite aux tests effectués avec les analyses, commentaires et conclusions associés (notamment, les paramètres qui correspondent aux meilleurs résultats notés).

Pour le séquencement des tests des approches proposées, nous suivons un ordonnancement similaire que lors de leur présentation dans le "*Chapitre 2 – Conception*". Ainsi, nous commençons par la classification avec toutes ses variantes.

### 3.3.1 Tests et résultats de la classification des scènes basées sur les objets

Nous présentons, dans ce qui suit, l'environnement de test (datasets, méthode et succession des tests) et les résultats obtenus.

#### 3.3.1.1 Datasets utilisés

Les différents tests liés à la classification des scènes basées sur les objets ont été effectués sur deux datasets différents. La variation des datasets utilisés a pour but de mieux d'analyser la performance de l'approche proposée, en évaluant sa précision face aux différentes sources de données.

Pour le premier dataset, nous avons utilisé MIT-Indoor [12]. Ce dernier s'intéresse à la reconnaissance de scènes d'intérieur (par exemple : Auditorium, boulangerie, bureau, musée, salle de cours, etc.), à partir des images brutes (dataset complet) ou images annotées dans des fichiers XML (dataset partiel, un sous-ensemble du dataset complet).

La mise en pratique de notre approche de classification requiert un dataset avec scènes annotées au préalable. Pour cela, l'utilisation du MIT-Indoor complet (qui contient des images brutes) dans la classification est précédée par une étape primordiale : La détection d'objets. Pour MIT-Indoor partiel (qui contient des scènes annotées), il se présente comme suit :

- Le dataset contient 2743 scènes, réparties sur 67 catégories.
- Chaque catégorie contient en moyenne 41 scènes, avec un minimum d'une seule scène et un maximum de 350 scènes.
- Le dataset contient 3000 objets uniques (et 52 814 objets avec redondance).
- Chaque scène contient en moyenne 19 objets, avec un minimum d'un seul objet et un maximum de 111 objets.

Le deuxième dataset utilisé est Sun2012 [13] (acronyme de "*Scene Understanding 2012*"). Ce dernier définit des catégories diversifiées (d'intérieur et d'extérieur) contenant des scènes annotées (sous format XML), il se présente comme suit :

- Le dataset contient 16 873 scènes, réparties sur 1075 catégories.
- Chaque catégorie contient en moyenne 16 scènes, avec un minimum d'une seule scène et un maximum de 998 scènes.
- Le dataset contient 5479 objets uniques (et 289 008 objets avec redondance).
- Chaque scène contient en moyenne 17 objets, avec un minimum d'un seul objet et un maximum de 222 objets.

Le choix de ces deux datasets en particulier est argumenté par le fait que la différence entre le nombre de leurs catégories est très grande (67 et 1075, une différence de 1008 catégories). Ainsi, les tests seront effectués -relativement- un petit et grand dataset, et donc, l'évaluation de l'approche proposée sera plus rigoureuse et crédible.

À cause du manque de partitions prédéfinies de l'apprentissage et test, la division MIT-Indoor est standard, en prenant pour chaque catégorie, 80% de scènes (chiffre arrondi) pour l'apprentissage et le reste (20%) pour le test. Pour Sun2012, ces partitions sont prédéfinies (spécifiées au préalable par les créateurs de ce dataset) [13].

Par la suite, nous présentons l'évaluation du classifieur des scènes basées sur les objets proposés sur les deux datasets MIT-Indoor et Sun2012.

### **3.3.1.2 Évaluation du classifieur des scènes proposé**

L'évaluation du classifieur proposé est effectuée séparément, en répétant à chaque fois la même procédure de test, sur chaque dataset présenté précédemment. Dans ce qui suit, nous présentons les tests effectués sur MIT-Indoor.

#### **1) Tests sur le dataset MIT-Indoor**

Pour son évaluation, le classifieur de scènes proposé sera appliqué successivement sur les deux variantes du dataset MIT-Indoor, partiel et complet. L'intérêt de manipulation de ces deux variantes dans nos tests est pour étendre le champ de comparaisons avec les travaux similaires existants dans l'état de l'art. En effet, la comparaison entre les résultats obtenus (issus de deux travaux différents) nécessite l'opération sur deux datasets exactement similaires (mêmes instances [scènes] de l'entraînement et de test pour chaque classe [catégorie]). Ainsi, pour MIT-Indoor, les travaux sont divisés entre l'utilisation du MIT-Indoor partiel (version annotée) et MIT-Indoor complet (version non annotée). Dans ce qui suit, nous abordons les tests avec la première variante : MIT-Indoor partiel.

##### **a - Tests sur le dataset MIT-Indoor partiel**

Les tests détaillés du classifieur des scènes proposé sur le dataset MIT-Indoor partiel sont présentés dans l'*annexe A - "A.1.1.1 – 1) Tests sur le dataset MIT-Indoor partiel"*.

Nous avons, dès à présent, mis en valeur les différents tests relatifs à l'ajustement des paramètres du classifieur de scènes proposé sur le dataset MIT-Indoor partiel. Dans ce qui suit, nous présentons un récapitulatif de la meilleure configuration des valeurs des paramètres, la précision finale obtenue et la comparaison avec les travaux existants dans l'état de l'art.

##### **i / Résultat obtenu et comparaison avec les travaux de l'état de l'art**

Nous récapitulons, dans le *tableau 3.1*, les deux meilleures combinaisons des valeurs des paramètres avec la précision associée.

**Tableau 3.1 : Meilleure combinaison des valeurs des paramètres du classifieur proposé (sur MIT-Indoor partiel) avec la précision associée**

Fonction d'activation	Optimiseur	$\beta$	N	Taux de précision
Sigmoid	RMSprop	1.5	27	81.75 %
		8.5	39	

Le taux de précision atteint est comparé, dans le *tableau 3.2*, avec un autre travail abordant la même problématique (classification des scènes basées sur les objets) sur le même dataset (MIT-Indoor partiel). Le meilleur taux de précision obtenu (81.75 %) dépasse largement celui obtenu dans [14] (76.28 %) pour le même dataset, la différence (en notre faveur) est 5.47 %.

**Tableau 3.2 : Comparaison du taux de précision obtenu (sur MIT-Indoor partiel) avec [14]**

Méthode	Taux de précision
Modèles de Markov cachés (2017) [14]	76.28 %
<b>Méthode proposée</b>	<b>81.75 %</b>

La rareté des travaux relatifs au dataset MIT-Indoor partiel, et la volonté de mettre nos résultats obtenus à l'épreuve en les comparant avec davantage travaux, nous ont poussés à l'exploitation, dans la section suivante, du dataset MIT-Indoor complet.

### **b - Tests sur le dataset MIT-Indoor complet**

MIT-Indoor complet est le dataset original, contenant toutes les scènes associées à leurs catégories correspondantes (contrairement au partiel qui ne contient qu'un sous-ensemble des scènes). Ainsi, c'est la variante la plus répandue et la plus utilisée dans la littérature [23, 24, 25, 26, 27].

Cependant, les instances du MIT-Indoor complet sont sous forme des images brutes (JPG, PNG, etc.), ce qui rend impossible leur utilisation directe dans le classifieur des scènes proposé, puisque ce dernier exploite les objets contenus dans une scène (*high-level features*) et non pas la scène brute (tendance à exploiter les *low-level features*).

Pour résoudre le problème précédent, nous avons utilisé un détecteur d'objets. Ce dernier est un classifieur qui reçoit en entrée une image (matrice de pixels) et retourne en sortie l'ensemble des objets contenus dans l'image avec leurs positions. Le but étant de concevoir un dataset MIT-Indoor complet avec scènes annotées (au lieu des images brutes). Le choix du détecteur d'objets pour MIT-Indoor complet peut être fait selon trois différentes manières :

- Entraînement de notre propre détecteur d'objets : Non réalisable dans ce cas, en comparant la performance de la plateforme utilisée pour les tests (*Google Colab* [19]) avec le nombre énorme des objets disponibles (causé par la variété des scènes).
- Utilisation d'un détecteur d'objets pré-entraîné sur MIT-Indoor complet : Détecteur inexistant, d'après nos recherches sur internet. Cela est très probablement dû au cas d'utilisation pour lequel ce dataset a été conçu (le dataset est fait pour évaluer les classificateurs des scènes et non pas les détecteurs d'objets).
- Utilisation d'un détecteur d'objets pré-entraîné sur un autre dataset : Solution la plus pratique, faisable sous la contrainte de disposition du dataset (sur lequel le détecteur d'objets est entraîné) avec un nombre d'objets acceptable (pour couvrir la variété des objets contenus dans les scènes de MIT-Indoor complet).

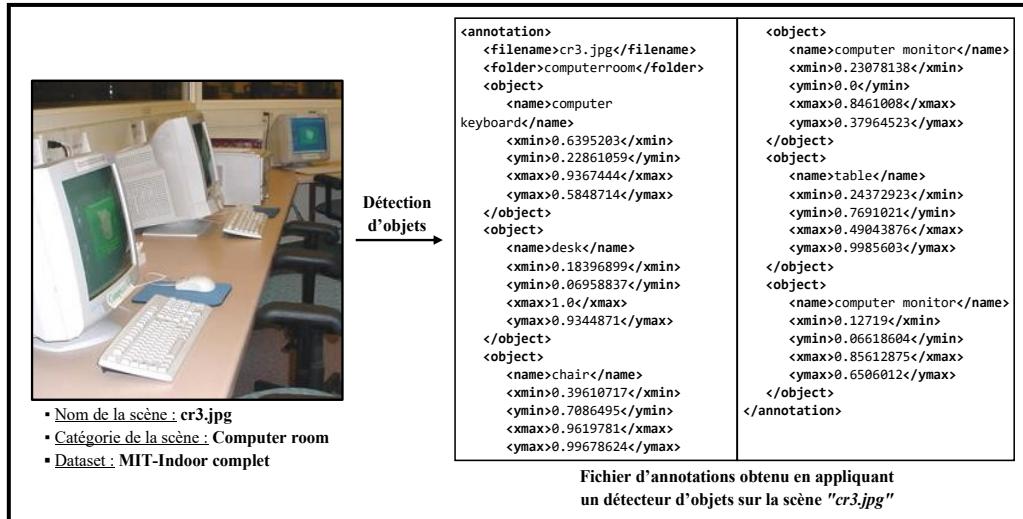
En analysant les possibilités précédentes, nous avons opté, bien-évidemment, pour la dernière possibilité "utilisation d'un détecteur d'objets pour un autre dataset". Ainsi, parmi les datasets trouvés, Open Images [10] semble le plus adéquat car il contient 600 objets différents (contrairement à, par exemple, COCO dataset [9] avec 80 objets). Le détecteur d'objets choisi est "*Faster RCNN Inception ResNet v2 atrous*" [36], il achève le meilleur taux de précision sur l'ensemble de test d'Open Images.

La création du dataset MIT-Indoor complet avec scènes annotées est faite en remplaçant chaque image brute par son fichier d'annotation XML correspondant (en passant cette image au détecteur d'objets). Le nouveau dataset (identique au MIT-Indoor original) se présente comme suit :

- Le dataset contient 6700 scènes, réparties sur 67 catégories.
- Chaque catégorie contient 100 scènes (80 pour l'entraînement et 20 pour les tests).
- Le dataset contient 275 objets uniques (et 45 079 objets avec redondance).

- Chaque scène contient en moyenne 7 objets, avec un minimum d'un seul objet et un maximum de 41 objets.

La figure 3.2 montre un exemple de fichier d'annotations résultant, après l'application du détecteur d'objets sur une scène du dataset MIT-Indoor complet.



**Figure 3.2 : Exemple de fichier d'annotations obtenu en appliquant un détecteur d'objets sur une scène du MIT-Indoor complet**

Après la présentation de l'adaptation du MIT-Indoor complet au type de données en entrée de notre classifieur de scènes proposé. Nous passons, dans la section suivante, à l'évaluation de ce classifieur (sur le dataset MIT-Indoor complet).

Les tests détaillés du classifieur des scènes proposé sur le dataset MIT-Indoor complet sont présentés dans l'*annexe A - "A.1.1.1 – 2) Tests sur le dataset MIT-Indoor complet"*.

Après la variation des paramètres du classifieur de scènes proposé, présentation des graphes obtenus, leur analyse et la récupération des meilleures valeurs pour chaque paramètre. Nous concluons l'évaluation de notre classifieur sur le dataset MIT-Indoor complet par une présentation d'un résumé récapitulant la meilleure architecture obtenue et nous comparons sa précision avec les travaux similaires existants dans la littérature.

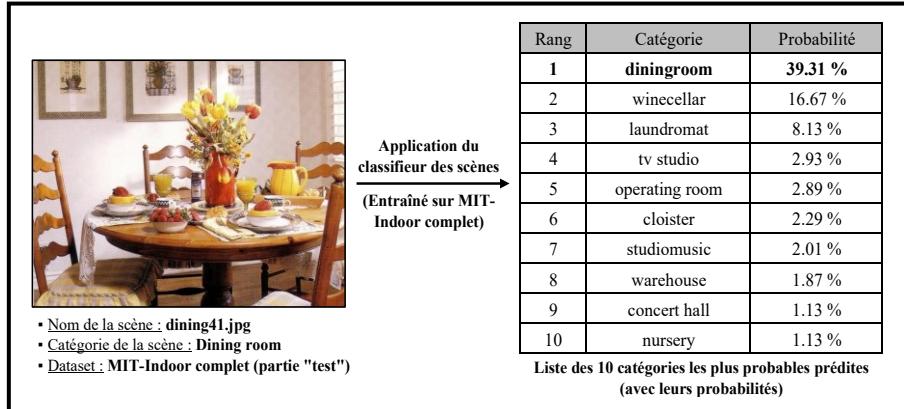
#### *i / Résultat obtenu et comparaison avec les travaux de l'état de l'art*

Les valeurs des paramètres de la meilleure architecture obtenue avec la précision associée sont présentées dans le tableau 3.3.

**Tableau 3.3 : Meilleure combinaison des valeurs des paramètres du classifieur proposé (sur MIT-Indoor complet) avec la précision associée**

Fonction d'activation	Optimiseur	$\beta$	N	Taux de précision
<b>Exponential</b>	<b>Adamax</b>	<b>14</b>	<b>35</b>	<b>43.54 %</b>

La figure 3.3 montre un exemple d'application d'un modèle de la classification en catégories standard (entraîné sur MIT-Indoor complet, avec la meilleure architecture) sur une scène donnée, les résultats sont présentés dans une liste contenant les 10 catégories les plus probables prédites.



**Figure 3.3 : Exemple d'application du classifieur des scènes (entraîné sur MIT-Indoor complet)**

Comme pour MIT-Indoor partiel, nous mettons la meilleure précision obtenue avec notre classifieur sur MIT-Indoor complet à l'épreuve, en la comparant avec les autres travaux, existants dans la littérature, opérant sur le même dataset. Le cumulatif de ces comparaisons est mis dans le *tableau 3.4* qui présente les travaux et leur précision, triés en ordre ascendant des valeurs des précisions.

**Tableau 3.4 : Comparaison du taux de précision obtenu (sur MIT-Indoor complet) avec des travaux précédents**

Méthode	Taux de précision
ROI+GIST (2009) [23]	26.50 %
MM-SCENE (2010) [24]	28.00 %
DPM (2011) [25]	30.40 %
CENTRIST (2011) [26]	36.90 %
Object Bank (2010) [27]	37.60 %
DPM+GIST-Color (2011) [25]	39.00 %
DPM+SP (2011) [25]	40.50 %
DPM+SP+GIST-Color (2011) [25]	43.10 %
<b>Méthode proposée</b>	<b>43.54 %</b>
Singh et al. (2012) [28]	49.40 %
Zuo el al. (2015) [29]	52.24 %
Method in (2014) [30]	59.50 %
Juneja et al. (2013) [31]	63.18 %
Doersch et al. (2013) [32]	66.87 %
Method in (2014) [33]	68.20 %
Fc8-FV (2015) [34]	72.86 %
MPP (2015) [35]	75.67 %

Le *tableau 3.4* obtenu montre l'emplacement de notre méthode au milieu, dépassant (en notre faveur) des anciens travaux, tout en étant dépassée par des travaux relativement nouveaux.

La faible précision obtenue par notre méthode revient essentiellement au détecteur d'objets utilisé (pour la transformation des images brutes en fichiers d'annotation). Ainsi, l'amélioration de cette précision repose sur l'utilisation d'un détecteur d'objets entraîné sur MIT-Indoor complet ou un détecteur pré-entraîné sur un dataset avec davantage d'objets.

Après l'évaluation du classifieur des scènes proposé sur le dataset MIT-Indoor (partiel et complet), en expliquant la procédure des tests, les paramètres de l'approche et les valeurs à considérer,

présentation des résultats obtenus sous forme de graphes, leur analyse et la comparaison avec les travaux similaires. Nous passons maintenant aux tests sur le deuxième dataset : Sun2012.

## 2) Tests sur le dataset Sun2012

De même que pour MIT-Indoor, l'évaluation du classifieur proposé consiste en deux étapes : Test du LSTM et test du classifieur lui-même.

Les tests détaillés du classifieur des scènes proposé sur le dataset Sun2012 sont présentés dans l'annexe A - "A.1.1.2 Tests sur le dataset Sun2012".

Une fois les tests du classifieur de scènes proposé sur le dataset Sun2012 terminés. Nous révélons, dans la section suivante, les valeurs des paramètres de la meilleure architecture obtenue avec sa précision, et nous comparons cette dernière avec les travaux similaires.

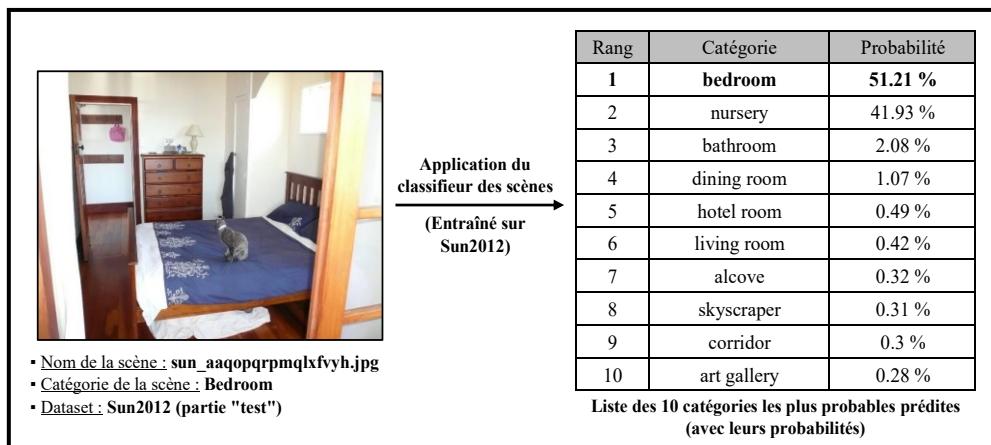
### a - Résultat obtenu et comparaison avec les travaux de l'état de l'art

Les valeurs des paramètres de la meilleure architecture obtenue avec la précision associée sont présentées dans le tableau 3.5.

**Tableau 3.5 : Meilleure combinaison des valeurs des paramètres du classifieur proposé (sur Sun2012) avec la précision associée**

Fonction d'activation	Optimiseur	$\beta$	N	Taux de précision
<b>Exponential</b>	<b>Nadam</b>	<b>0</b>	<b>37</b>	<b>46.53 %</b>

La figure 3.4 montre un exemple d'application d'un modèle de la classification en catégories standard (entraîné sur Sun2012, avec la meilleure architecture) sur une scène donnée, les résultats sont présentés dans une liste contenant les 10 catégories les plus probables prédites.



**Figure 3.4 : Exemple d'application du classifieur des scènes (entraîné sur Sun2012)**

D'après nos recherches, nous avons constaté un manque des travaux qui utilisent Sun2012 comme dataset d'évaluation des classifieurs des scènes. Ainsi, la cause précédente rend impossible la comparaison de la meilleure précision obtenue sur Sun2012 avec les travaux similaires.

Toutefois, nous justifions l'obtention d'une précision relativement basse par le grand nombre des catégories du Sun2012 et la disproportion du nombre de leurs scènes (des catégories avec une dizaine de scènes, contre d'autres avec des centaines de scènes).

Après l'évaluation du classifieur des scènes proposé sur les deux datasets MIT-Indoor et Sun2012, et suivant la même succession des approches que pour la conception, nous entamons les tests du classifieur de l'enrichissement des scènes.

### 3.3.1.3 Évaluation du classifieur de l'enrichissement des scènes

L'enrichissement des scènes est une approche qui vise à améliorer le taux de précision de la classification en catégories standard (évaluée précédemment), et cela, en ajoutant des objets qui existent, très probablement, dans la scène à classifier, mais non-visibles.

Comme présenté dans "2.2.1.2 Enrichissement des scènes", la procédure de génération de l'ensemble d'apprentissage du modèle de l'enrichissement des scènes prend deux paramètres en considération (mis à part  $\beta$  et  $N$ , comme pour la classification en catégories standard) :

- $T_i$  : un pourcentage, qui désigne le taux des objets importants d'une scène à considérer pour la  $i^{\text{ème}}$  catégorie d'un dataset.
- $NFreq_i$  : un nombre naturel positif, qui désigne la valeur de la fréquence maximale à considérer pour les objets importants retenus (lors de l'application du paramètre  $T_i$ ) d'une scène pour la  $i^{\text{ème}}$  catégorie d'un dataset.

Par la suite, des combinaisons des valeurs des fréquences obtenues sont générées, pour construire l'ensemble d'apprentissage, suivant *l'Algorithme 2.4*.

Cependant, d'après nos tests, nous avons constaté que la génération des combinaisons des fréquences est un processus gourmand, qui consomme beaucoup de la RAM, qui dépasse pour certaines valeurs des paramètres ou certains datasets (par exemple, Sun2012), les 25 Go (taille de la RAM disponible dans Google Colab). Pour cela, nous se restreignons, lors de l'évaluation de l'enrichissement des scènes, sur les datasets de taille moyenne, à savoir, MIT-Indoor partiel et complet.

De plus, à cause de la performance des GPUs de Google Colab, et dans le but de procéder à une évaluation de l'enrichissement des scènes dans un temps raisonnable, nous se limitons à une valeur commune de  $T_i$  et  $NFreq_i$  pour toutes les catégories du dataset. Ainsi, dans l'évaluation, nous notons :

- $T$  : Taux des objets importants à considérer pour une scène donnée.
- $NFreq$  : Valeur de la fréquence maximale à considérer pour les objets importants retenus (avec le paramètre  $T$ ).

Pour les autres paramètres,  $\beta$  et  $N$ , nous utilisons pour chaque dataset évalué, la meilleure combinaison des valeurs obtenus dans la classification en catégories standard.

Nous commençons, dans ce qui suit, par l'évaluation de l'enrichissement des scènes sur le dataset MIT-Indoor partiel.

#### 1) Tests sur le dataset MIT-Indoor partiel

L'évaluation de l'enrichissement des scènes suit le même séquencement que celui établi dans la classification en catégories standard, à savoir, variation de chaque paramètre séparément (en fixant les autres) afin de déterminer le meilleur ensemble des valeurs pour chaque paramètre, ensuite, combinaison des valeurs des meilleurs ensembles pour déterminer la meilleure combinaison.

La différence entre la classification en catégories standard et l'enrichissement réside dans l'étape supplémentaire requise pour ce dernier. Plus précisément, le but de l'évaluation de l'enrichissement des scènes est l'analyse de son impact sur la précision de la classification en catégories standard. Ainsi, une fois la meilleure combinaison des valeurs des paramètres du modèle de l'enrichissement sur MIT-Indoor partiel déterminée, nous procérons à l'enrichissement de toutes les scènes de l'ensemble de test du MIT-Indoor partiel, puis nous passons ces scènes enrichies au meilleur modèle de la classification en catégories standard, et nous comparons la précision obtenue "sans" et "avec" l'application de l'enrichissement.

Nous débutons les tests de l'enrichissement des scènes sur MIT-Indoor partiel par l'évaluation séparée des deux paramètres : T et NFreq.

Les tests détaillés du classifieur de l'enrichissement des scènes sur le dataset MIT-Indoor partiel sont présentés dans l'*annexe A - "A.1.2 – 1) Tests sur le dataset MIT-Indoor partiel"*.

Après la variation des paramètres du classifieur de l'enrichissement des scènes, et la récupération de la meilleure combinaison des valeurs de ses paramètres. Nous terminons l'évaluation de ce classifieur par la section suivante, dans laquelle nous présentons un résumé de la meilleure architecture obtenue ainsi que nous analysons l'impact de l'application de cette architecture sur la précision de la classification en catégories standard.

#### *a - Résultat obtenu et l'impact sur la classification en catégories standard*

Nous résumons les résultats des tests précédents de l'enrichissement sur le dataset MIT-Indoor partiel dans le *tableau 3.6*, qui présente la meilleure combinaison des valeurs des paramètres qui correspond à la meilleure précision sur l'ensemble de test du dataset d'apprentissage construit.

**Tableau 3.6 : Meilleure combinaison des valeurs des paramètres du classifieur de l'enrichissement des scènes (sur MIT-Indoor partiel) avec la précision associée**

$\beta$	N	T	NFreq	Taux de précision
<b>1.5</b>	<b>27</b>	<b>20%</b>	<b>4</b>	<b>85.58 %</b>

Par la suite, nous testons l'impact de l'enrichissement sur la classification en catégories standard. Pour cela, nous appliquons un modèle de l'enrichissement (entraîné sur un dataset construit à partir des scènes de l'ensemble d'entraînement du MIT-Indoor partiel en utilisant les paramètres de la meilleure architecture) sur toutes les scènes de l'ensemble de test du MIT-Indoor partiel. Ensuite, les scènes enrichies résultantes sont passées au meilleur modèle de la classification en catégories standard afin de l'évaluer. Le *tableau 3.7* présente la précision du meilleur modèle de la classification en catégories standard, entraîné sur MIT-Indoor partiel, "sans" et "avec" l'application de l'enrichissement.

**Tableau 3.7 : Précision du meilleur modèle de la classification en catégories standard (sur MIT-Indoor partiel) avec/sans l'application de l'enrichissement des scènes**

	Sans l'enrichissement	Avec l'enrichissement
Précision du meilleur modèle de la classification en catégories standard	<b>81.75 %</b>	<b>79.03 %</b>

Le *tableau 3.7* montre une légère détérioration (2.72 %) du taux de précision en appliquant l'enrichissement, par rapport à la précision notée sans l'enrichissement des scènes.

Bien que le but de l'enrichissement soit l'amélioration du taux de précision de la classification (ce qui n'est pas le cas), nous justifions la détérioration obtenue par :

- Généralisation des paramètres propres à l'enrichissement à toutes les catégories du dataset : En utilisant des valeurs de  $T$  et  $NFreq$  communes, quelque-soit la catégorie à laquelle une scène appartient. Une telle généralisation est appliquée dans nos tests à cause de l'explosion combinatoire qui résulte lors de l'attribution d'un  $T_i$  et  $NFreq_i$  pour chaque catégorie " $i$ ".
- Valeurs des paramètres testées trop petites : Cela est dû à la capacité de la plateforme des tests (Google Colab). Ainsi, même en testant des petites valeurs, nous avons obtenu, pour certains cas, un dépassement de la taille de la RAM utilisée.

Nous tenons à noter que l'idée de l'enrichissement, telle qu'expliquée dans le "*Chapitre 2 – Conception*", est correcte. Dans le sens où, elle marchera, très probablement, convenablement en utilisant des plateformes de test plus performantes.

Une fois l'évaluation du classifieur de l'enrichissement des scènes sur MIT-Indoor partiel achevée. Nous passons, dans ce qui suit, aux tests sur le deuxième dataset : MIT-Indoor complet.

## 2) Tests sur le dataset MIT-Indoor complet

Les tests de l'enrichissement des scènes sur le dataset MIT-Indoor complet sont faits de la même manière que pour le dataset MIT-Indoor partiel. Plus précisément, nous procédons à la variation individuelle de chaque paramètre afin de déterminer le meilleur ensemble des valeurs. Ensuite, nous combinons ces meilleurs ensembles pour en tirer le meilleur. Enfin, nous résumons les paramètres de la meilleure architecture et nous analysons l'impact de cette dernière sur la performance de la classification en catégories standard.

Les tests détaillés du classifieur de l'enrichissement des scènes sur le dataset MIT-Indoor complet sont présentés dans l'*annexe A - "A.1.2 – 2) Tests sur le dataset MIT-Indoor complet"*.

Une fois la meilleure combinaison des valeurs des paramètres du classifieur de l'enrichissement des scènes sur MIT-Indoor complet récupérée. Nous passons à la dernière étape de l'évaluation de ce classifieur, dans laquelle nous présentons un résumé de la meilleure architecture obtenue ainsi que nous analysons l'impact de l'application de cette architecture sur la précision de la classification en catégories standard.

### a - Résultat obtenu et l'impact sur la classification en catégories standard

Les valeurs des paramètres de la meilleure architecture obtenue avec la précision associée sont présentées dans le *tableau 3.8*.

**Tableau 3.8 : Meilleure combinaison des valeurs des paramètres du classifieur de l'enrichissement des scènes (sur MIT-Indoor complet) avec la précision associée**

$\beta$	N	T	NFreq	Taux de précision
14	35	50%	10	52.32 %

Par la suite, nous testons l'impact de l'enrichissement sur la classification en catégories standard. En suivant la même procédure que celle pour MIT-Indoor partiel, nous présentons, dans le *tableau 3.9*, la précision du meilleur modèle de la classification en catégories standard, entraîné sur MIT-Indoor complet, "sans" et "avec" l'application de l'enrichissement.

**Tableau 3.9 : Précision du meilleur modèle de la classification en catégories standard (sur MIT-Indoor complet) avec/sans l'application de l'enrichissement des scènes**

	Sans l'enrichissement	Avec l'enrichissement
Précision du meilleur modèle de la classification en catégories standard	43.54 %	40.37 %

Le *tableau 3.9* montre une légère détérioration (3.17 %) du taux de précision en appliquant l'enrichissement, par rapport à la précision notée sans l'enrichissement des scènes.

De même que pour les tests sur le dataset MIT-Indoor partiel, nous justifions la détérioration du taux de précision de la classification en catégories standard en employant le modèle de l'enrichissement (entraîné sur MIT-Indoor complet) par la généralisation des paramètres propres à l'enrichissement à toutes les catégories du dataset (liée à l'explosion combinatoire) et l'utilisation des valeurs des paramètres testées trop petites (liée à la capacité de la plateforme des tests).

Après l'évaluation de l'enrichissement des scènes en tirant la meilleure combinaison des valeurs des paramètres pour le dataset MIT-Indoor partiel et complet, et son application sur la classification en

catégories standard. Nous passons, par la suite, à l'évaluation de la troisième variante de la classification : Classification en catégories additionnelles.

### **3.3.1.4 Évaluation des classifieurs des catégories additionnelles**

La classification en catégories additionnelles suit les mêmes étapes que celles pour la classification en catégories standard. La différence réside juste dans une étape primaire, qui consiste à la réorganisation des scènes du dataset suivant les catégories additionnelles définies. Pour cela, les meilleures valeurs des paramètres retrouvées dans la classification en catégories standard (dans les tests du LSTM et le classifieur lui-même) seront réutilisées dans la classification en catégories additionnelles.

La similarité entre la classification en catégories standard et celle en catégories additionnelles permet d'employer les mêmes datasets pour l'évaluation des classifieurs en catégories additionnelles.

Cependant, les tests sur le dataset Sun2012 ne peuvent pas être effectués dans un temps raisonnable. Cela est dû au nombre énorme des catégories originales (1075 catégories) à parcourir manuellement pour leurs attribuer les catégories additionnelles définies. Ainsi, l'évaluation des classifieurs des catégories additionnelles sera restreinte juste au premier dataset (MIT-Indoor).

Dans la section suivante, nous présentons les tests sur le dataset MIT-Indoor.

#### **1) Test sur le dataset MIT-Indoor**

Les tests des classifieurs des catégories additionnelles seront faits sur les deux variantes du dataset MIT-Indoor : Partielle et complète. Nous commençons, dans ce qui suit, par les tests sur MIT-Indoor partielle.

##### **a - Test sur le dataset MIT-Indoor partielle**

Afin d'évaluer la précision de cette variante de classification sur le dataset MIT-Indoor partielle, nous avons défini trois catégories additionnelles :

- Catégories additionnelles 1 : De type "ternaire", indiquent s'il s'agit d'une scène représentant un endroit privé (accessible par quelques personnes seulement) ou publique (accessible par tout le monde) ou rien (ambiguïté de déterminer s'il s'agit d'une scène privée ou publique).
- Catégories additionnelles 2 : De type "ternaire", indiquent s'il s'agit d'une scène qui représente un endroit de relaxation (détente) ou demandant de l'intelligence (concentration et le sérieux) ou rien (ambiguïté de déterminer s'il s'agit d'une scène de relaxation ou de l'intelligence).
- Catégories additionnelles 3 : De type "binaire", indiquent s'il s'agit d'une scène ayant relation avec le transport (contenant des moyens de transport ou représentant un endroit de transport) ou non.

Un classifieur individuel est entraîné pour chaque ensemble de catégories additionnelles, ainsi, nous aurons trois classifieurs distincts.

Comme mentionné précédemment, nous utilisons les meilleures valeurs des paramètres retrouvées dans la classification en catégories standard :

- Fonction d'activation du LSTM : *sigmoid*
- Optimiseur du LSTM : *RMSprop*
- $\beta$  (Calcul de l'importance des objets dans la scène) : 1.5
- N (Troncation de la scène) : 27

Le *tableau 3.10* présente la précision obtenue pour chaque classifieur d'un ensemble de catégories additionnelles.

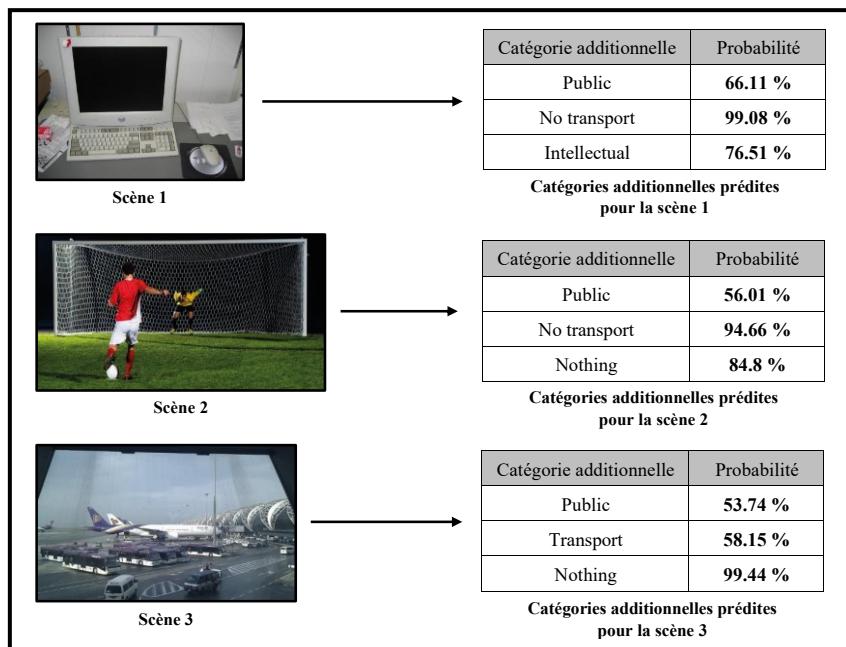
**Tableau 3.10 : Précision des classifieurs des catégories additionnelles sur MIT-Indoor partiel**

Catégories additionnelles	Taux de précision
Privé / Publique / Rien	<b>94.56 %</b>
Relaxation / Intelligence / Rien	<b>95.34 %</b>
Transport / Non-transport	<b>99.81 %</b>

Le *tableau 3.10* montre clairement l'efficacité de la classification en catégories additionnelles, la précision moyenne obtenue (en tenant compte de tous les ensembles des catégories additionnelles) est de 96.57 %.

L'obtention de tels résultats est justifiée par la méthode utilisée, qui consiste à réorganiser les scènes du dataset suivant les catégories additionnelles définies. Ces dernières étant restreintes (binaires, ternaires, etc.), ce qui contribuera à l'obtention de hautes précisions lors de l'évaluation.

La *figure 3.5* montre quelques exemples d'application des classifieurs en catégories additionnelles (entraînés sur le dataset MIT-Indoor partiel).



**Figure 3.5 : Exemple d'application des classifieurs en catégories additionnelles (entraînés sur MIT-Indoor partiel)**

Une fois les tests des classifieurs des catégories additionnelles sur MIT-Indoor partiel achevés, nous passons aux tests sur MIT-Indoor complet.

#### **b - Test sur le dataset MIT-Indoor complet**

Afin d'effectuer les tests sur MIT-Indoor complet, nous utilisons les mêmes ensembles des catégories additionnelles que pour MIT-Indoor partiel. Cela est justifié par l'existence des mêmes catégories pour les deux variantes de MIT-Indoor, la différence réside dans le nombre des scènes par catégorie et leurs objets (ce qui n'est pas pris en compte lors de la définition des catégories additionnelles).

En utilisant la meilleure combinaison des valeurs des paramètres pour MIT-Indoor complet, nous présentons dans le *tableau 3.11* la précision obtenue pour chaque classifieur d'un ensemble de catégories additionnelles.

**Tableau 3.11 : Précision des classifieurs des catégories additionnelles sur MIT-Indoor partiel**

Catégories additionnelles	Taux de précision
Privé / Publique / Rien	<b>89.53 %</b>

Relaxation / Intelligence / Rien	<b>85.39 %</b>
Transport / Non-transport	<b>92.15 %</b>

Tout comme pour la variante partielle, la classification en catégories additionnelles sur MIT-Indoor complet s'est avérée efficace, la précision moyenne obtenue est 89.02 %.

Une fois l'achèvement des tests détaillés sur les différentes valeurs des paramètres pour tous les types de la classification des scènes, la présentation des résultats obtenus, leur analyse et la comparaison avec les résultats issus des autres travaux. Nous passons, par la suite, aux tests relatifs à l'interprétation des scènes.

### 3.3.2 Tests et résultats de l'interprétation des scènes basées sur les objets

Cette partie est consacrée à l'évaluation des approches proposées pour l'interprétation. Ainsi, en suivant une approche de test similaire à celle de la classification, nous commençons par la présentation de l'environnement de test (datasets utilisés et la succession des tests).

#### 3.3.2.1 Datasets utilisés

La mise en pratique de l'interprétation des scènes nécessite un dataset adéquat, à partir duquel les relations contenues dans les scènes peuvent être identifiées et définies. Pour cela, nous avons utilisé le dataset VisualGenome [16]. À partir de ce dernier, nous avons exploité la partie qui contient des scènes avec les relations, sous format JSON, unaires : Appelées "attributs", où les différents objets de la scène, définis par leur nom et position dans la scène, se voient accordés un attribut ou plus, et binaires : La description, appelée "prédictat", reliant deux objets. Les statistiques sur le dataset VisualGenome se présentent comme suit :

- Le nombre total des scènes est 108 777.
- Le nombre des objets uniques est 3 802 374.
- Le nombre moyen des objets uniques dans une scène est 35, avec un minimum de 1 objet et un maximum de 207 objets.
- Le nombre total des attributs (relations unaires) des objets est 2 342 898 et celui des relations binaires est 2 316 104.
- Le nombre moyen des relations unaires par scène est 22, avec un minimum de 0 relations et un maximum de 208 relations.
- Le nombre moyen des relations binaires par scène est 21, avec un minimum de 0 relations et un maximum de 800 relations.

La figure 3.6 présente un exemple de l'ensemble des relations unaires et binaires contenues dans une scène de VisualGenome.

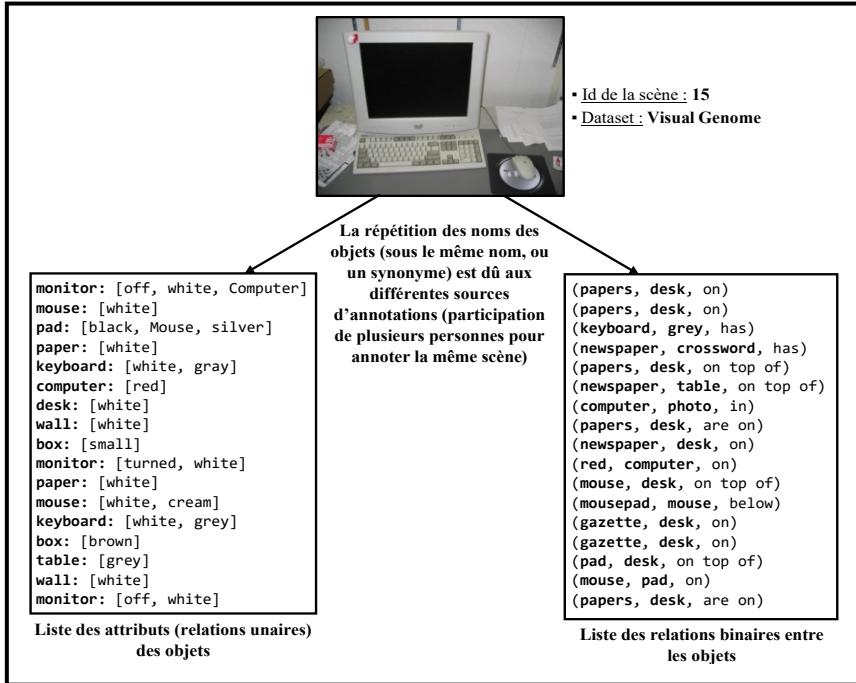


Figure 3.6 : Exemple des relations unaires et binaires d'une scène de VisualGenome

L'argument derrière le choix de VisualGenome est qu'il contient un très grand nombre des objets (3 802 374, contrairement aux datasets similaires, tel que OpenImages [10] qui ne contient que 600 objets uniques). Cette propriété permettra, très probablement, d'augmenter le nombre des objets mis en correspondance entre VisualGenome et le dataset utilisé pour l'interprétation des scènes.

En ce qui concerne l'évaluation de l'interprétation des scènes, nous avons utilisé le même dataset durant toutes les étapes de l'interprétation. Un tel choix est justifié par le fait que les étapes du processus de l'interprétation sont successives et dépendantes les unes des autres, ainsi, la sortie d'une étape sera l'entrée de la prochaine étape.

Pour le dataset de l'interprétation des scènes, nous avons opté pour COCO [9] qui contient un ensemble de scènes annotées, sous format JSON, avec leurs différentes interprétations (sous forme de phrases). Le dataset se présente comme suit :

- Le nombre total des scènes annotées est 122 218, et celui des objets uniques est 80 objets.
- Le nombre moyen des objets uniques par scène est 3 objets, avec un minimum de 1 objet et un maximum de 18 objets.
- Le nombre des interprétations pour chaque scène est 5 interprétations.

La figure 3.7 présente un exemple de l'annotation d'une scène de COCO dataset et ses interprétations associées.

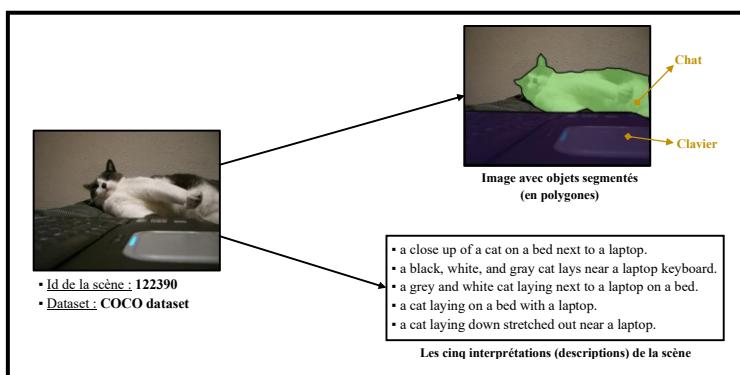


Figure 3.7 : Exemple de l'annotation d'une scène de COCO dataset et ses interprétations associées

Ce dataset (COCO) a été choisi à cause de sa nouveauté relative (dernière version réalisée en 2017) et le grand nombre de scènes interprétées (contrairement, par exemple, au PASCAL Sentences [20], réalisé en 2010 et ne contenant que 1000 scènes) et sa popularité dans le domaine de vision par ordinateur.

Dans notre travail, nous avons utilisé le dataset VisualGenome pour l'apprentissage des différents modèles, nécessaires pour la génération des interprétations, et le dataset COCO pour les tests.

L'utilisation des relations contenues dans les scènes de VisualGenome dans le dataset COCO nécessite une mise en correspondance entre les noms des objets des deux datasets. Cette dernière vise à affecter, pour chaque nom d'objet de VisualGenome, son correspondant le plus proche, s'il existe, en se basant sur les synonymes.

L'évaluation des différentes étapes (approches) de l'interprétation des scènes se déroulera en suivant le même ordre que celui présenté dans le workflow (*Figure 2.9*). Ainsi, nous procérons par les trois étapes suivantes :

- Évaluation de l'identificateur des relations.
- Évaluation du définiteur des relations (unaires et binaires).
- Génération des descriptions des scènes.

Après la présentation des datasets et le séquencement des évaluations à effectuer. Nous débutons, dans la section suivante, par l'évaluation de l'identificateur des relations.

### **3.3.2.2 Évaluation de l'identificateur des relations**

L'identificateur des relations est un classifieur permettant d'identifier les relations existantes entre les objets contenus dans une scène. L'évaluation de cet identificateur permet de déterminer les meilleures valeurs des paramètres pour lesquelles la meilleure précision possible est atteinte.

L'évaluation commencera par les tests des différentes architectures du classifieur LSTM, indépendamment des autres paramètres (ceux-ci, seront fixés tout au long de ces tests). Ensuite, avec la meilleure architecture du LSTM obtenue, nous nous intéressons aux tests relatifs aux valeurs des paramètres de notre approche.

Après la définition de l'identificateur des relations, l'intérêt de son évaluation et la procédure des tests à effectuer, nous passons au premier test : Architecture du LSTM.

Les tests détaillés de l'identificateur des relations sont présentés dans l'*annexe A - "A.2.1 Évaluation de l'identificateur des relations"*.

L'évaluation du modèle de la première étape vers l'interprétation des scènes (identificateur des relations) a permis, entre-autres, de récupérer les meilleurs paramètres de ce modèle. Ce dernier, une fois appliqué sur les instances de COCO dataset (i.e. Identifier les relations [unaires et binaires] les plus probables entre les objets de chaque scène de ce dataset), permet de passer à l'étape suivante qui consiste à définir ces relations. Ainsi, dans ce qui suit, nous nous intéressons à l'évaluation des modèles de définition des relations conçus.

### **3.3.2.3 Évaluation des définitseurs des relations**

La définition des relations est l'attribution, pour chaque relation identifiée, de l'information (appelée : description) la plus plausible caractérisant cette relation. L'intérêt de ce traitement est l'ajout de l'aspect sémantique aux relations (abstraites, lors de leur identification).

Les relations manipulées dans notre travail sont de deux types : Unaires et binaires. Ainsi, la définition de chaque type de relation implique l'utilisation des modèles différents des classifieurs.

En utilisant les scènes (avec relations identifiées) de COCO dataset, nous commençons par l'évaluation des modèles de définition des relations unaires.

### 1) Test des classifieurs de définition des relations unaires

L'entraînement des classifieurs des relations unaires passe par les étapes suivantes :

- Construction du dictionnaire global des fréquences des attributs des objets : Consiste à parcourir toutes les scènes de VisualGenome, pour chaque scène, récupérer les attributs des objets qui ont une correspondance dans COCO dataset et incrémenter les fréquences de ces attributs dans le *dictionnaire global des attributs des objets*. La structure de ce dernier est illustrée dans la *figure 3.8*.

{
"person": {
"standing": 6038,
"walking": 3750,
"sitting": 3666,
"young": 2295,
"smiling": 2091,
⋮
},
"chair": {
"wooden": 1687,
"empty": 441,
"plastic": 396,
"metal": 345,
"leather": 207,
⋮
},
⋮

**Figure 3.8 : Structure générale du dictionnaire global des attributs des objets**

- Récupération des attributs des objets les plus fréquents : Il s'agit de ne garder, pour chaque objet, que les 50 attributs les plus fréquents, le résultat étant le *dictionnaire global tronqué*. Le choix de ce nombre (50) est justifié par le fait d'avoir une marge suffisante de choix pour la prochaine étape.

- Définition des catégories (targets) des classifieurs des relations unaires : Cette étape consiste à parcourir manuellement les objets du dictionnaire tronqué précédent, pour chaque objet, regrouper les attributs de même nature (pour former les catégories d'un classifieur) en respectant deux contraintes :

- La fréquence de chaque attribut doit-être considérable. Cela assurera que l'attribut aura un nombre acceptable des instances pour l'entraînement. Pour notre cas, nous avons défini un seuil minimal de fréquence d'un attribut pour un objet à 50 (moins de cela, l'attribut ne sera pas considéré).

- Les fréquences de chaque groupe d'attributs doivent-être rapprochées. Le but étant d'avoir un classifieur neutre, non-biaisé pour un attribut (catégorie) particulier. Pour notre cas, le seuil minimal est fixé à un ratio de 40 % pour chaque paire de combinaison des attributs (catégories) d'un classifieur.

Le résultat de cette étape est des ensembles des catégories des classifieurs pour les objets où un objet peut avoir un ou plusieurs ensembles des catégories des classifieurs. La *figure 3.9* montre un exemple des résultats obtenus où l'objet "TV" a deux ensembles des catégories, tandis que "Person" n'a qu'un seul ensemble.

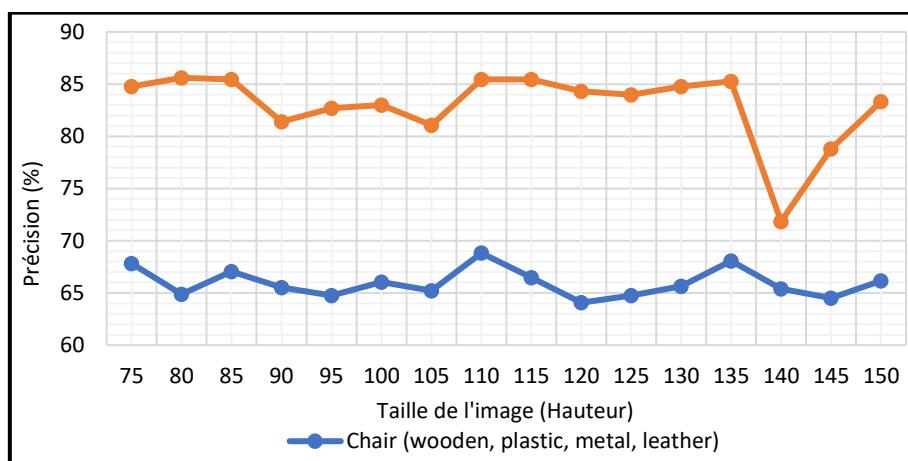
Objet	Catégories du classifieur
TV	On / Off
	Small / Large
Person	Standing / Walking / Sitting

**Figure 3.9 : Exemple des ensembles des catégories définies pour deux objets**

- Entraînement des classifieurs des relations unaires : C'est l'étape finale, elle consiste à entraîner, pour chaque ensemble de catégories d'un objet, un classifieur. La procédure d'entraînement de ce dernier comprend trois sous-étapes :

▪ Construction de l'ensemble d'entraînement pour le classifieur d'une relation unaire : Il s'agit de parcourir encore une fois toutes les scènes de VisualGenome, pour chaque scène, récupérer la partie de l'image qui désigne l'objet cible (pour lequel ce classifieur sera entraîné), ayant un attribut qui correspond à l'une des catégories (targets) du classifieur à entraîner. La taille des parties des images doit-être unique, ainsi, une fois récupérées, elles sont redimensionnées vers la même taille, cette dernière est un paramètre à ajuster.

Ainsi, pour déterminer la taille de l'image permettant d'achever la meilleure précision, nous allons varier la taille de 75\*75 (minimum requis par l'architecture-source de transfert des poids) à 150\*150 (limites de la plateforme de tests utilisée [Google Colab]), en l'incrémentant à chaque fois par 5 (en gardant la forme carrée, largeur = hauteur), et en répétant chaque test (d'une taille donnée) 3 fois (pour une meilleure fiabilité des tests obtenus). Les tests ont été faits sur deux classifieurs des relations unaires, à savoir "chair (wooden, plastic, metal, leather)" et "TV (open, closed)". La *figure 3.10* présente les précisions moyennes obtenues en variant la taille de l'image d'entrée sur deux classifieurs des relations unaires.



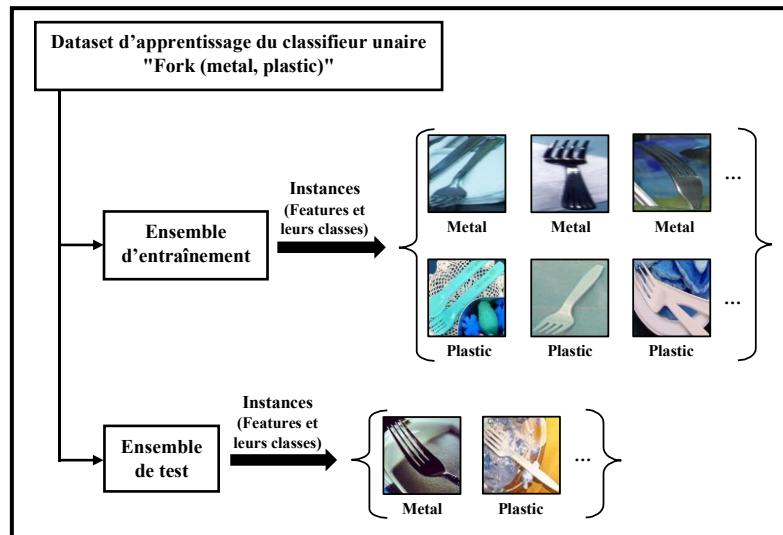
**Figure 3.10 : Précisions moyennes obtenues en fonction de la taille de l'image en entrée sur deux classifieurs des relations unaires**

Le graphe de la *figure 3.10* est constitué de deux courbes, chacune représente un définiteur d'un type de relations unaires pour un objet particulier :

- La courbe bleue concerne l'objet "chair", elle démontre une oscillation irrégulière du taux de précision. Le meilleur ensemble de valeurs des tailles des images, pour lequel l'écart entre le minimum et le maximum ne dépasse pas 1 % est {75, 110, 135}.
- La courbe orange concerne l'objet "TV", elle aussi dévoile des irrégularités des taux des précisions enregistrés. Le meilleur ensemble de valeurs des tailles des images est {75, 80, 85, 110, 115, 130, 135}, avec un écart de 0.81 %.

L'union des meilleurs ensembles pour les deux classifieurs précédents résulte par l'ensemble {75, 110, 135}. À partir de ce dernier, nous choisissons la valeur minimale (75\*75) comme étant la taille de l'image d'entrée de tous les classifieurs des relations unaires. La généralisation des résultats obtenus aux autres classifieurs se défend par l'observation du même comportement (oscillation et l'existence de la valeur 75 dans le meilleur ensemble) pour deux classifieurs distincts, et donc, ce comportement sera très probablement maintenu avec les autres définiteurs des relations unaires. Quant au choix de la valeur minimale, il est justifié par le fait que les précisions des valeurs de l'ensemble résultant sont relativement proches, ainsi, nous optons pour la taille minimale car elle prend le moins de temps et espace mémoire pendant l'apprentissage et la prédiction.

Le dataset d'apprentissage construit est divisé en parties d'entraînement et de test, selon une division standard 80% / 20%. La *figure 3.11* montre un exemple de l'ensemble d'apprentissage pour un classifieur d'une relation unaire.

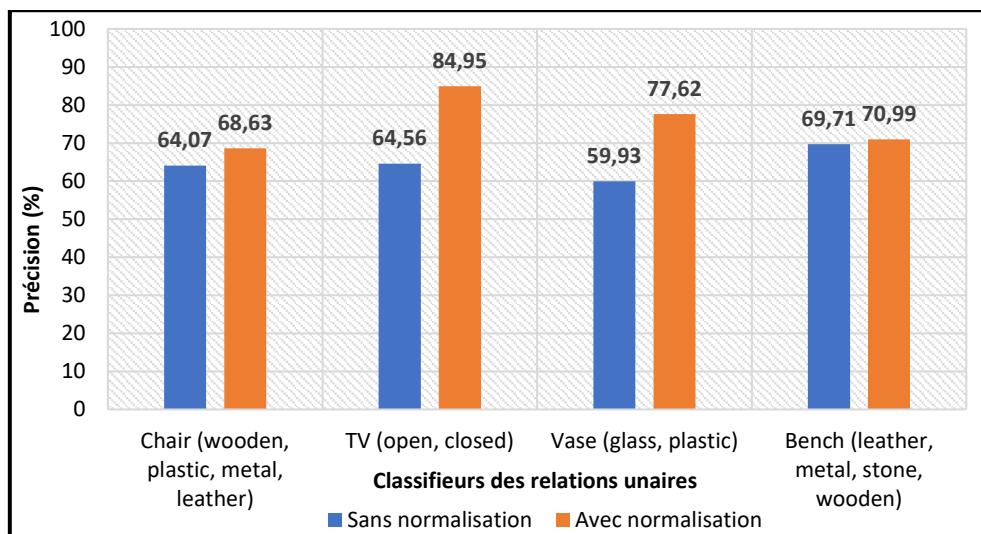


**Figure 3.11 : Exemple de l'ensemble d'apprentissage pour un classifieur d'une relation unaire**

- Normalisation des images de l'ensemble d'entraînement : Consiste à normaliser les images de l'ensemble d'apprentissage et de test pour un classifieur d'une relation unaire. Pour chaque image (matrice N \* M de pixels [\* 3, pour les trois canaux des couleurs RGB], pour notre cas N = M = 75), l'équation 3.1 est appliquée :

$$Image_{normalisée} = \frac{Image - \min(Image)}{\max(Image) - \min(Image)} \quad (3.1)$$

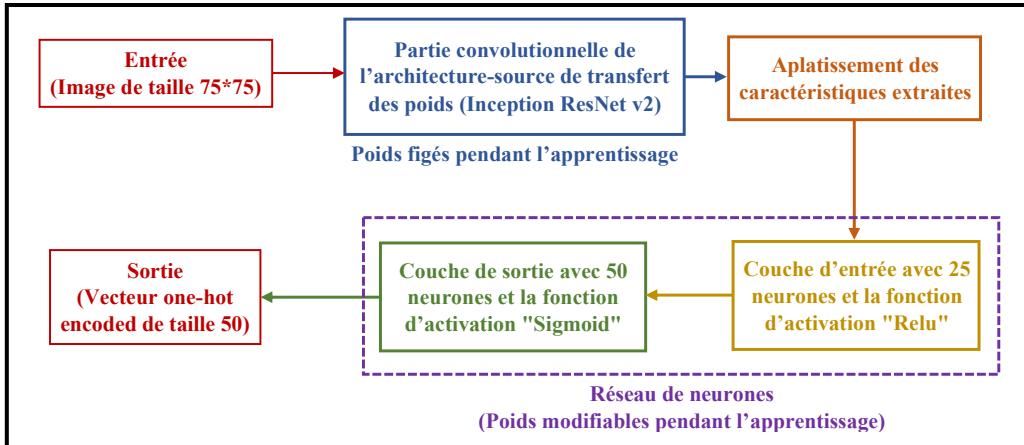
Les tests effectués montrent une amélioration considérable du taux de précision (sur l'ensemble de test) en utilisant la normalisation. La *figure 3.12* présente quelques résultats des taux des précisions de certains classificateurs de relations unaires sans / avec la normalisation.



**Figure 3.12 : Taux des précisions obtenus pour certains classificateurs des relations unaires sans / avec la normalisation**

- Entraînement du classifieur sur l'ensemble d'entraînement créé : Cela est fait en utilisant un *transfert des poids*. Ce dernier consiste à copier une partie d'un modèle pré-entraîné sur un problème similaire (avec targets différents) dans le but de bénéficier des caractéristiques déjà apprises (sans avoir le besoin de les réapprendre à nouveau). Pour notre cas, nous avons fait un transfert des poids de la partie convolutionnelle (avant l'aplatissement des matrices des convolutions et le passage à un réseau de neurones feedforward) du modèle "*Inception ResNet v2*" [21] (pré-entraîné sur le dataset *ImageNet*)

[22]). La partie transférée est figée (poids non-réajustées lors de la rétropropagation) pendant l'apprentissage du classifieur de la relation unaire courante. L'application du transfert des poids est argumentée par le fait que l'entraînement d'un CNN à partir de zéro est trop lent sur la plateforme utilisée pour les tests (Google Colab). L'architecture des classifieurs des relations unaires est illustrée dans la figure 3.13.



**Figure 3.13 : Architecture des classifieurs des relations unaires**

L'application des étapes précédentes, avec un entraînement de 100 époques, a abouti à l'obtention de 108 classifieurs des relations unaires. Ces derniers ont été épurés en ne sélectionnant que les classifieurs respectant les deux contraintes suivantes :

- Ne sélectionner que les classifieurs avec un taux de précision supérieur ou égal à 60 %.
- Dans le cas où un objet admet plus d'un classifieur, sélectionner celui avec le meilleur ratio "taille de l'ensemble d'entraînement / taux de précision".

L'application des deux contraintes précédentes sur les 108 classifieurs a résulté en 38 classifieurs (pour 38 objets de COCO dataset) de relations unaires. La figure 3.14 présente la liste exhaustive de ces classifieurs.

Objet	Catégories du classifieur	Précision
Clock	Analog / Digital	<b>100 %</b>
Person	Young / Old	<b>78.7 %</b>
Mouse	Wireless / Wired	<b>77.5 %</b>
Dining table	Wooden / Metal	<b>75.9 %</b>
Handbag	Plastic / Leather	<b>86.14 %</b>
Chair	Wooden / Plastic / Metal / Leather	<b>64.07 %</b>
Tv	Open / Closed	<b>86.41 %</b>
Apple	Red / Green / Black / Yellow	<b>62.694 %</b>
Fork	Metal / Plastic	<b>74.138 %</b>
Vase	Glass / Plastic	<b>76.895 %</b>
Bottle	Glass / Plastic	<b>78.276 %</b>
Bowl	Round / Square	<b>91.463 %</b>
Cup	Large / Small	<b>80.556 %</b>
Laptop	Off / On	<b>87.5 %</b>
Train	Moving / Stopped	<b>68.116 %</b>
Truck	Large / Small	<b>85.393 %</b>
Umbrella	Closed / Opened	<b>84.932 %</b>
Boat	Docked / Floating	<b>75.342 %</b>
Toilet	Clean / Dirty	<b>79.245 %</b>
Objet	Catégories du classifieur	Précision
Sink	Porcelain / Stainless steel / Wooden	<b>88.119 %</b>
Oven	Electric / Gas	<b>70.37 %</b>
Knife	Metal / Plastic	<b>83.333 %</b>
Bench	Leather / Metal / Stone / Wooden	<b>71.533 %</b>
Suitcase	Large / Small	<b>71.053 %</b>
Bed	Made / Unmade	<b>78.261 %</b>
Spoon	Metal / Plastic / Wooden	<b>69.412 %</b>
Banana	Ripe / Unripe	<b>84.553 %</b>
Cat	Sitting / Sleeping / Standing	<b>61.438 %</b>
Airplane	Flying / Landing / Parked / Taking off	<b>68.683 %</b>
Bird	Large / Small	<b>81.579 %</b>
Dog	Large / Small	<b>71.975 %</b>
Kite	Large / Small	<b>81.25 %</b>
Cow	Large / Small	<b>66.667 %</b>
Bear	Large / Small	<b>66.364 %</b>
Horse	Large / Small	<b>65.517 %</b>
Giraffe	Adult / Young	<b>70.588 %</b>
Sheep	Grazing / Standing / Walking	<b>60.784 %</b>
Pizza	Sliced / Whole	<b>78.495 %</b>

**Figure 3.14 : Liste des classifieurs des relations unaires entraînés**

Les classificateurs des relations unaires obtenus seront utilisés lors de la génération des descriptions des scènes. Dans ce qui suit, de même que pour les relations unaires, nous nous intéressons au classifieur des relations binaires.

## 2) Test du classifieur de définition des relations binaires

Les relations binaires concernent les paires d'objets, où la position de l'objet dans la paire (à gauche ou à droite) a une importance. Leur identification a pour but de dévoiler les relations phares existantes entre les objets d'une scène. Quant à leur définition, elle permet de décrire la nature des relations, en leur attribuant la proposition (description) la plus plausible, ou de nier une relation en détectant qu'aucune description (parmi ceux reconnaissables par le définsseur) ne peut être affecté à la relation.

Comme pour tous les classificateurs, le définsseur des relations binaires nécessite un dataset adéquat pour le processus de l'apprentissage. Ce dataset doit contenir un ensemble de scènes (matrice de pixels), chacune équipée de ses relations binaires décrites par les noms des deux objets (qui sont en relation), leur emplacement dans la scène (sous forme de bounding-box) et la description définissant la relation.

La construction d'une partie du dataset précédent inclut deux étapes :

- Récupération des scènes compatibles : Consiste à extraire un sous-ensemble de scènes (avec leurs informations) à partir de Visual Genome. Une scène n'est ajoutée à ce sous-ensemble que si tous les objets composant les relations binaires d'une scène appartiennent à COCO dataset (en utilisant le dictionnaire de correspondance entre les noms des objets de Visual Genome et COCO dataset) sur lequel l'évaluation de toutes les étapes de l'interprétation des scènes est faite. Aucune contrainte n'est posée sur les autres objets d'une scène (qui forment des relations unaires ou n'appartiennent à aucune relation), ceux-ci, peuvent ne pas appartenir à COCO dataset. La nécessité d'un tel traitement est justifiée par le fait de s'assurer que le définsseur des relations binaires va opérer sur des objets compatibles avec le dataset choisi pour l'interprétation. Pour notre cas, le nombre des scènes récupérées est 3406 (à partir d'un total de 108 777 scènes de Visual Genome).

- Construction partielle du dataset pour le définsseur des relations : À partir des scènes précédentes, le dataset d'entraînement est construit partiellement. Ce dernier contient deux types de classes (targets) :

- Descriptions des relations binaires : La description de chaque relation binaire (appartenant aux scènes récupérées) est placé comme target de cette dernière.

- Catégorie "aucune relation" : Pour chaque scène, des permutations (combinaisons dont l'ordre des éléments est important) binaires (contenant deux éléments) sont générées à partir des objets des relations binaires contenus dans la scène. Pour chaque permutation, une vérification de son existence comme relation binaire dans la scène courante est faite. Dans le cas négatif (la permutation ne représente pas une relation binaire déjà définie), le target "aucune relation" (no relation) est attribué à cette permutation d'objets. Dans le cas positif (la permutation existe déjà comme relation binaires), elle est ignorée.

Les deux types des classes précédentes sont représentés par un vecteur one-hot encoded, où chaque colonne désigne une description particulière, ou bien la catégorie "aucune relation". Le nombre des catégories (descriptions, y compris "aucune relation") reconnues est 50 catégories, présentés dans la *figure 3.15*.

| Catégories du défineur des relations binaires |
|---|---|---|---|---|
| sitting on                                    | next to                                       | riding  | inside  | herding                                       |
| waiting on                                    | behind  | under   | belonging to                                  | laying in                                     |
| <b>no_relation</b>                            | in front of                                   | by  | at  | looking at                                    |
| near  | with  | using   | waiting for                                   | eating  |
| on top of                                     | carrying                                      | sitting in                                    | sitting down                                  | inside of                                     |
| on  | in  | watching                                      | leaning on                                    | held by                                       |
| riding a                                      | sitting on a                                  | of  | walking outside                               | sits on                                       |
| standing on                                   | has   | laying on                                     | sitting on top of                             | playing                                       |
| on a  | in a  | wearing                                       | holds   | holding an                                    |
| beside  | holding                                       | of a  | attached to                                   | running on                                    |

**Figure 3.15 : Liste des catégories du défineur des relations binaires**

Cependant, même en définissant les targets, la construction du dataset ne sera pas complète que si les caractéristiques (features) récupérées des deux objets de chaque relation binaire seront définies. Ces caractéristiques dépendent de type du classifieur utilisé.

En effet, nous avons défini deux types des classifieurs pour la définition des relations binaires : Un CNN et un NN. Le but de variation des classifieurs utilisés est l'exploitation des différents modèles, leur évaluation (individuellement) et le choix du meilleur classifieur en termes de précision obtenue et ressources utilisées (nécessaires à l'apprentissage de ces classifieurs).

Après la présentation du défineur des relations binaires et la construction partielle de son dataset. Nous commençons, dans la section suivante, par l'évaluation du premier classifieur pour la définition des relations binaires : Le CNN (Convolutional Neural Network).

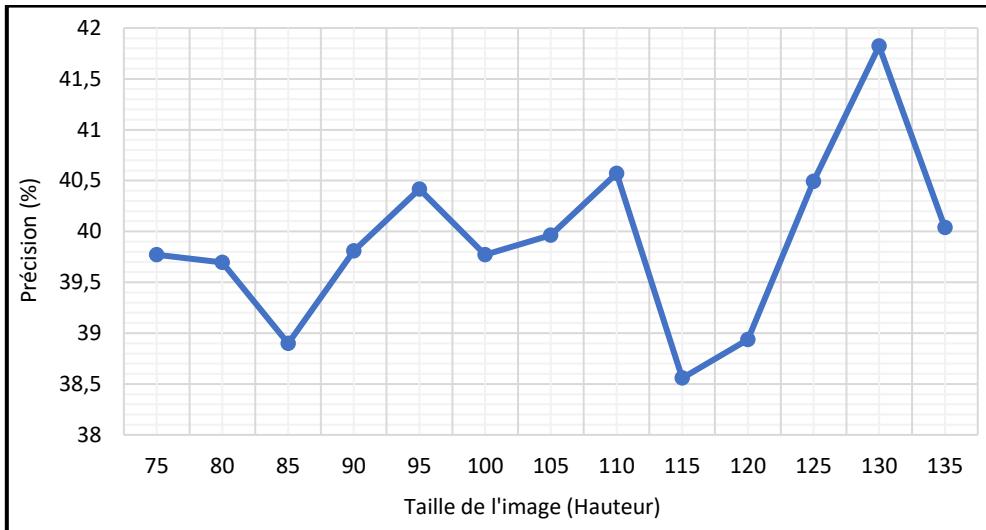
#### *a - Évaluation du CNN comme défineur des relations binaires*

Le défineur des relations binaires utilisant un CNN reçoit en entrée des images brutes (matrices de pixels). Ainsi, les caractéristiques (features) des instances du dataset sont des images. Ces dernières représentent l'union de deux parties de l'image qui contiennent les deux objets d'une relation binaire. La récupération de cette union des deux parties est faite en exploitant l'information sur le bounding-box de chacun des deux objets. Le dataset complet est divisé en parties d'entraînement et de test selon une division standard 80% / 20%.

L'apprentissage du CNN est fait en utilisant un transfert des poids. Comme pour les défineurs des relations unaires, l'intérêt de cela est pour contourner les limitations des ressources de la plateforme utilisée pour les tests (Google Colab). Les poids transférés sont récupérés à partir de l'architecture "Inception ResNet v2" [21] (pré-entraînée sur ImageNet [22]).

En s'appuyant sur le constat dans l'évaluation des défineurs des relations unaires : "La normalisation améliore considérablement la précision obtenue avec les CNNs", les features (images) des instances du dataset ont été, elles aussi, normalisées en utilisant l'équation 3.1.

L'évaluation du CNN est faite en variant la taille de l'image en entrée, du 75 \* 75 (\* 3, pour les trois canaux RGB), le minimum requis par l'architecture-source de transfert des poids, jusqu'à 150 \* 150 (limites du GPU de Google Colab), en incrémentant la taille par 5 à chaque fois. Les résultats obtenus sont présentés dans la figure 3.16.



**Figure 3.16 : Précision obtenue (avec le définitisseur des relations binaires utilisant un CNN) en fonction de la taille de l'image**

La courbe du graphe de la *figure 3.16* est irrégulière, elle démontre que la performance (en termes de la précision obtenue) du définitisseur des relations binaires utilisant un CNN est indépendante de la taille de l'image en entrée. En effet, la précision moyenne obtenue pour les petites tailles des images (entre 75 et 100, en moyenne 39.73 %) est quasi-similaire à la précision moyenne des tailles supérieures (entre 105 et 135, en moyenne 40.05 %), avec une différence (entre les deux groupes de tailles) ne dépassant pas 0.33 %. La meilleure précision (41.82 %) correspond à la taille 130 (\*130), ainsi, cette taille sera maintenue pour le définitisseur des relations binaires utilisant un CNN.

Une fois le définitisseur des relations binaires utilisant un CNN évalué, nous passons dans la section suivante, à l'évaluation du deuxième classifieur : Le NN (Neural Network).

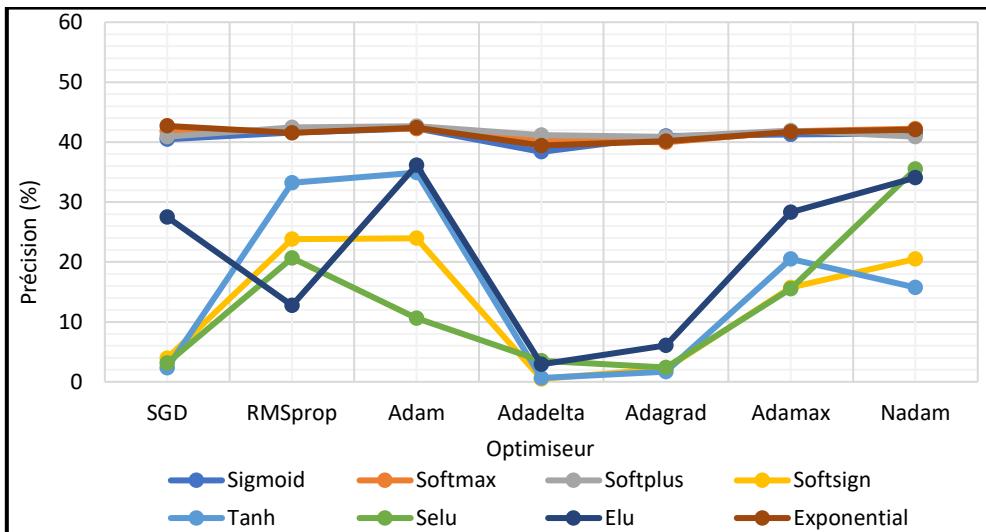
#### **b - Évaluation du NN comme définitisseur des relations binaires**

Le définitisseur des relations binaires utilisant un NN reçoit en entrée un vecteur de caractéristiques extraites des bounding-boxes des deux objets des relations binaires. Ces caractéristiques (présentées en détails dans le "*Chapitre 2 – Conception*") sont : l'importance, IOU, positions (quatre valeurs), taille, collision et la distance. Tout comme pour le CNN, le dataset complet est divisé en 80% / 20% (entraînement / test).

Puisque ce définitisseur ne possède que des paramètres standards d'un simple réseau de neurones, nous restreignons nos tests qu'à leurs variations. Plus précisément, nous varions simultanément la fonction d'activation et l'optimiseur, ainsi que simultanément le nombre de couche et neurones.

Nous commençons par la variation de la fonction d'activation et l'optimiseur. Les valeurs prises par ces deux paramètres sont similaires à ceux de la classification en catégories standard (en utilisant les LSTMs). Notons que la variation simultanée de ces deux paramètres (en évaluant chaque combinaison de valeurs) est indispensable à cause de la dépendance entre ces derniers.

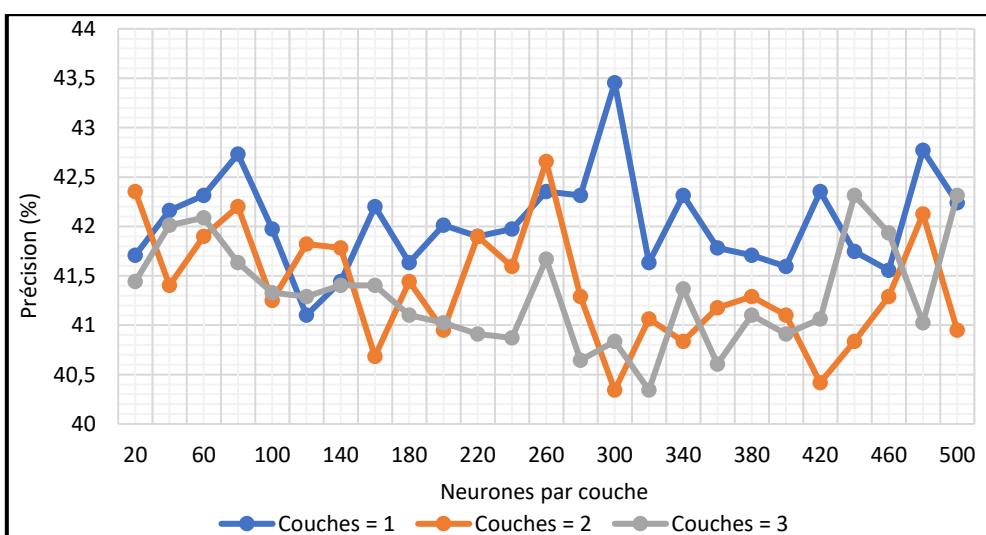
L'évaluation est faite en fixant le nombre des couches du NN à 2 : Une couche d'entrée avec 9 neurones (taille du vecteur des caractéristiques des relations binaires), aucune couche cachée, et une couche de sortie avec 50 neurones (nombre de descriptions reconnus par le modèle). Le choix d'une telle architecture est justifié par le fait de n'inclure que le minimum nécessaire pour l'apprentissage d'un classifieur NN. L'entraînement est fait sur 100 époques. La *figure 3.17* montre la précision obtenue en fonction de la *fonction d'activation* (courbes colorées) et l'*optimiseur* (les abscisses).



**Figure 3.17 : Précision obtenue (avec le définitisseur des relations binaires utilisant un NN) en fonction de la fonction d'activation et l'optimiseur**

Le graphe obtenu dans la *figure 3.17* est similaire à ceux précédents (dans la classification et l'identification des relations) en variant la fonction d'activation et l'optimiseur. En effet, deux groupes de courbes sont formés : Un groupe de basses précisions (en moyenne 15.67 %) et un groupe de hautes précisions (courbes superposées, les unes sur les autres), avec une précision moyenne de 41.35 %. Ce dernier groupe contient la meilleure combinaison (Exponential - SGD) avec une précision de 42.69 %.

Une fois la meilleure combinaison (fonction d'activation – optimiseur) déterminée, nous passons par la suite à la variation du nombre des couches cachées et leurs neurones (en utilisant la meilleure combinaison des deux paramètres précédents). Le nombre des couches cachées est varié de 1 à 3 (largement suffisant pour les 9 neurones en entrée et 50 en sortie), à chaque fois, en variant le nombre des neurones de 20 à 500 par pas de 20. En entraînant l'architecture sur 100 époques, la *figure 3.18* montre la précision obtenue en fonction du nombre des couches cachées et leurs neurones.



**Figure 3.18 : Précision obtenue (avec le définitisseur des relations binaires utilisant un NN) en fonction du nombre des couches et leurs neurones**

L'analyse de la variation des courbes du graphe de la *figure 3.18* aboutit aux résultats suivants :

- La précision moyenne n'est pas liée au nombre des couches ou leurs neurones. En effet, la précision moyenne pour chaque couche est quasiment la même : 42.04 %, 41.39 % et 41.31 % pour une, deux et trois couches respectivement, avec une différence ne dépassant pas 0.65 %. Ce fait est maintenu pour les précisions moyennes pour chaque valeur du nombre des neurones, la moyenne totale (de toutes les moyennes) est égale à 41.58 %.

- Les précisions obtenues avec une seule couche cachée sont généralement meilleures que les précisions avec deux ou trois couches cachées. En effet, en comparant les précisions obtenues par chaque couche, pour chaque nombre de neurones individuellement, nous remarquons que les précisions dans cas d'une seule couche cachée sont supérieures (ou égales) par rapport aux deux autres cas (2 et 3 couches) pour 18 valeurs du nombre des neurones (parmi les 26 valeurs au total), les précisions pour les cas de deux et trois couches ne sont les meilleures que pour 5 et 3 valeurs respectivement.
- La meilleure précision (43.45 %) correspond à la combinaison (1, 300) [nombre des couches cachées – nombre des neurones].

L'évaluation des deux modèles de définitseurs des relations binaires (à savoir CNN et NN) et la récupération de leurs meilleures valeurs de paramètres nous conduit à la section suivante, dans laquelle, nous comparons les deux modèles pour en tirer le meilleur.

#### *c - Comparaison entre les définitseurs des relations binaires*

Une fois évalués, les deux définitseurs des relations binaires (utilisant un CNN et un NN), seront comparés. Le but de cette comparaison est de ne garder qu'un seul définitseur qui sera utilisé dans le processus de la génération des descriptions des scènes. Pour cela, nous présentons dans le *tableau 3.12* les deux définitseurs des relations binaires, chacun avec la meilleure combinaison des valeurs de ses paramètres et la précision associée.

**Tableau 3.12 : Meilleure combinaison des valeurs des paramètres pour les deux modèles des définitseurs des relations binaires (CNN et NN)**

Modèle de définitseur des relations binaires	Meilleurs paramètres		Taux de précision
CNN	Taille de l'image	<b>130*130</b>	<b>41.82 %</b>
NN	Fonction d'activation	<b>Exponential</b>	
	Optimiseur	<b>SGD</b>	
	Nombre de couches cachées	<b>1</b>	
	Nombre de neurones par couche	<b>300</b>	

La comparaison des taux des précisions des deux modèles du *tableau 3.12* montre clairement que le définitseur utilisant un NN est plus performant que celui utilisant un CNN, la différence atteint 1.63 %. Cette performance du définitseur utilisant un NN réside aussi dans la taille de la mémoire nécessaire (et par conséquence, le temps d'exécution) pour la prédiction de la description adéquat pour une relation binaire. En effet, la taille du vecteur d'entrée pour le modèle CNN est 130\*130 (\*3, pour les trois canaux RGB), soit une taille de 50 700 entiers, contre un vecteur de taille 9 pour le modèle NN.

Ainsi, suite aux deux arguments précédents (taux de précision et espace mémoire), nous maintenons le définitseur des relations binaires utilisant un NN pour le processus de génération des descriptions des scènes.

Une fois le meilleur modèle de définitseur des relations binaires choisi. Nous passons, dans la section suivante, aux tests relatifs au générateur des descriptions des scènes.

#### *3.3.2.4 Évaluation du générateur des descriptions*

Après la définition des relations unaires et binaires, et dans le but d'évaluer la qualité des interprétations des scènes résultantes en utilisant notre approche. Nous avons exécuté le générateur des interprétations conçu sur un ensemble de scènes de la partie de test du COCO dataset.

Cependant, à la différence des tests des étapes précédentes, l'évaluation des interprétations des scènes générées ne peut pas être automatisée. Cela est dû au fait de la nécessité de prendre en compte de l'aspect sémantique d'une interprétation, ce qui ne peut être fait qu'à travers une revue manuelle, par des humains, de ces interprétations générées.

Pour cela, nous avons fait appel à des questionnaires (surveys), créées à l'aide de Google Forms [89], qui offre un assistant pour la mise en forme rapide des formulaires, ainsi que des outils pratiques pour l'analyse des résultats obtenus.

Plus précisément, nous avons mis en place 5 questionnaires, chaque questionnaire contient 20 scènes avec leurs interprétations, et pour chaque scène 4 choix possibles qui désignent le niveau d'exactitude de l'interprétation associée à la scène. La *figure 3.19* montre le début d'un questionnaire sur la qualité des interprétations générées.

**Evaluation of our approach in Scene Interpretation**

As part of our Master's degree in "Automatic Scene Interpretation", we would be grateful for your participation in completing this survey. Please take a few minutes of your time and rate each automatically generated interpretation by selecting the proposed answers.

Enjoy your survey and thank you for your contribution!

NB : all the images used in this survey are from COCO dataset.

\*Obligatoire

469828 : What do you think about the description attributed to this image (its convenience)? \*

In this scene that represents probably a public place there is a young person.

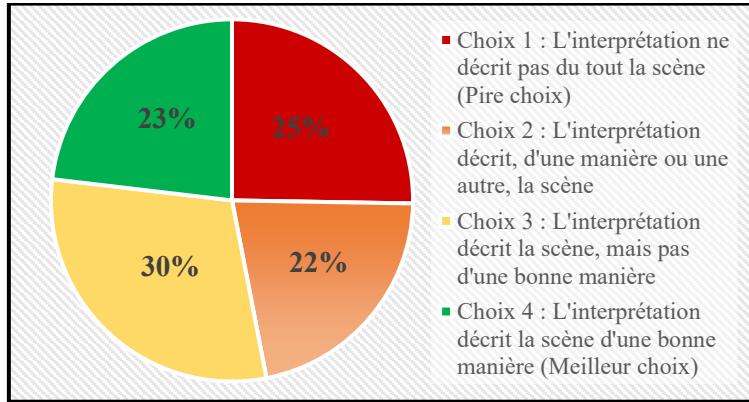
- Not describing at all the scene.
- Somehow it describes the scene.
- Describing the scene, but not really in a good manner.
- Describing the scene in a good manner.

**Figure 3.19 : Exemple d'un questionnaire sur la qualité des interprétations générées**

Une fois les questionnaires mises en place, ils sont partagés dans les différents sites et groupes publics accessibles à tout le monde (par des personnes tierces, qui ne sont en aucun cas influencés par notre travail). En attendant trois semaines, nous avons obtenu 63 réponses, réparties comme suit :

- Questionnaire 1 : 24 réponses.
- Questionnaire 2 : 11 réponses.
- Questionnaire 3 : 10 réponses.
- Questionnaire 4 : 9 réponses.
- Questionnaire 5 : 9 réponses.

Nous présentons, dans la *figure 3.20*, le résumé des résultats obtenus (en pourcentages) pour les 4 choix possibles.



**Figure 3.20 : Résumé des résultats obtenus à partir des questionnaires sur la qualité des interprétations générées**

Les résultats obtenus à partir des questionnaires sont équilibrés, nous notons un taux de 25 % (à peu près) pour chacune des 4 options. Plus particulièrement, les deux choix qui correspondent à une description acceptable (choix 3 et 4) représentent un taux de 53 %.

Les résultats détaillés des différentes interprétations des scènes sont présentés dans l'*annexe B - "B.1 Résultats détaillés des questionnaires sur la qualité des interprétations générées"*.

En présentant les résultats du questionnaire sur la qualité des interprétations des scènes générées en utilisant notre approche basée sur les règles, la partie des tests est achevée. Dans la section suivante, nous parlons sur l'application développée, qui regroupe toutes les approches conçues et les modèles entraînés dans une simple interface.

## 3.4 Application

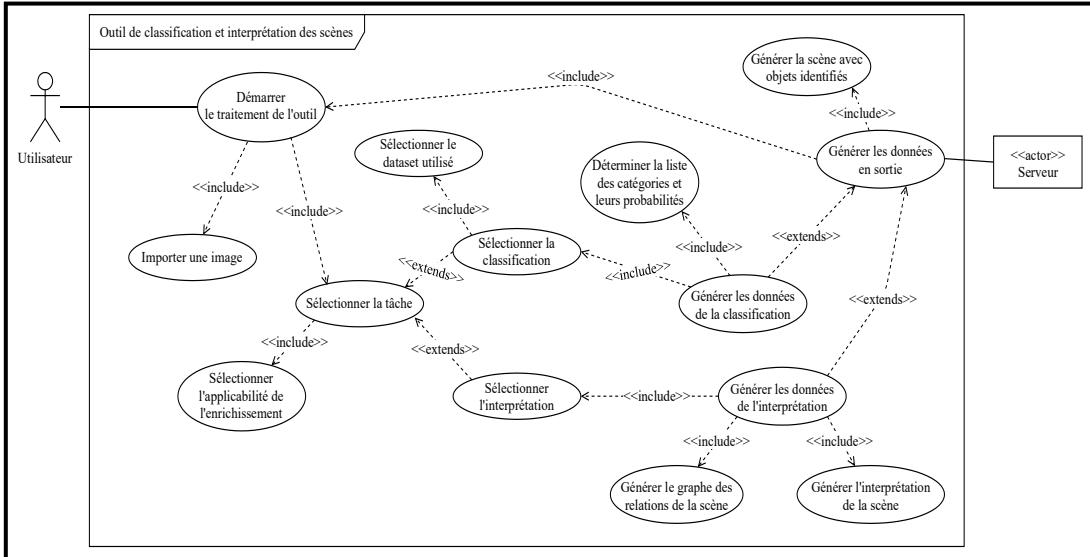
L'application implémentée regroupe tous les algorithmes proposés, développés dans notre travail. Elle offre, dans une interface simple et intuitive, aux utilisateurs le choix d'effectuer une classification ou interprétation d'une image en entrée.

Dans ce qui suit, nous détaillons les fonctionnalités de notre application à travers un diagramme de cas d'utilisation (use-case).

### 3.4.1 Diagramme de cas d'utilisation

Le diagramme use-case de l'UML [94] est un moyen pratique pour la présentation des différentes actions faites par une application, sur les deux côtés : Front-end (interaction de l'utilisateur avec l'interface de l'application) et le back-end (traitements internes du moteur de l'application).

Le diagramme de cas d'utilisation de notre application est présenté dans la *figure 3.21*. Les explications associées à cette figure seront présentées dans la "*section 3.4.3 – Fonctionnalités de l'application*".



**Figure 3.21 : Diagramme de cas d'utilisation de l'application implémentée**

L’achèvement de la brève présentation des différents traitements effectués par notre application à travers d’un diagramme use-case, nous conduit à la partie suivante, dans laquelle, nous nous intéressons aux outils utilisés pour le développement de cette application.

### 3.4.2 Outils de développement

L’application est développée sous forme d’un site web. Ce dernier forme le moyen le plus répandu pour le partage des différentes idées/solutions pratiques avec les gens. Les outils de développement sont divisés en deux catégories, selon la partie de l’application : Front-end et back-end.

#### 3.4.2.1 Partie front-end

Concerne la façade (partie visible de l’application). Elle s’intéresse à l’ensemble des interactions de l’utilisateur avec l’application, à savoir : L’entrée des données et l’affichage des résultats (sortie). Les outils utilisés sont le trio standard pour le front-end des sites web :

- HTML 5 [95] : Langage de balisage, pour l’affichage des données (texte ou médias) structurées.
- CSS 3 [96] : Pour l’attribution des styles aux différentes structures du HTML 5.
- Javascript [97] (version ECMAScript 2015, sans aucune bibliothèque ou framework externe) : Pour dynamiser les structures du HTML 5, et rendre possible l’interaction avec l’utilisateur.

#### 3.4.2.2 Partie back-end

Concerne les traitements internes (partie non-visible de l’application). Elle s’intéresse aux différentes tâches effectuées par le serveur, l’outil choisi pour cette partie est Django [98]. Ce dernier est un framework permettant, entre-autres, d’exécuter le code Python [23] dans le back-end. Le choix de ce framework est argumenté par le fait que toutes les approches proposées (dans le *Chapitre 2 – Conception*) et les modèles entraînés ont été fait, justement, avec Python (en utilisant la plateforme de Google Colab [19]).

Une fois les outils de développement de l’application proposée présentés. Nous passons, dans ce qui suit, à l’explication détaillée des différentes fonctionnalités et traitements effectués par l’application.

### 3.4.3 Fonctionnalités de l'application

Le développement de l'application nécessite une vision claire sur l'ensemble des fonctionnalités à proposer et l'ordonnancement d'exécution des différentes tâches. Pour cela, au cours de cette section, nous détaillons toutes les fonctionnalités de notre application, en commençant par l'entrée des données, en passant par les traitements du côté serveur et finissant par les données en sortie.

#### 3.4.3.1 Données en entrée de l'application

Une fois lancée, l'interface de l'application invite l'utilisateur à spécifier les données en entrée. Ces derniers se présentent comme suit :

- Image : Consiste à importer (à travers un explorateur de fichiers ou un simple "glisser-déposer") une image brute, dont l'extension fait référence à une image (JPG, PNG, Bitmap, etc.).
- Tâche à effectuer : Il s'agit de spécifier une des deux tâches possibles à effectuer sur l'image précédente : La classification en catégories standard (prédition de la catégorie la plus probable) ou bien l'interprétation (génération de description textuelle qui décrit l'image).
- Paramètres : Communes (ou non) entre les deux tâches proposées par l'application :
  - Dataset utilisé : Choix du modèle entraîné (MIT-Indoor complet ou Sun2012), utilisé pour la prédition de la catégorie de l'image. Ce paramètre est propre à la classification.
  - Enrichissement : Choisir d'enrichir ou non l'image en entrée pour la classification ou l'interprétation.

Après la spécification de toutes les données en entrée, le bouton de démarrage d'effectuation de la tâche spécifiée (classification ou interprétation) est activé. La *figure 3.22* montre un exemple des données entrées à l'application.

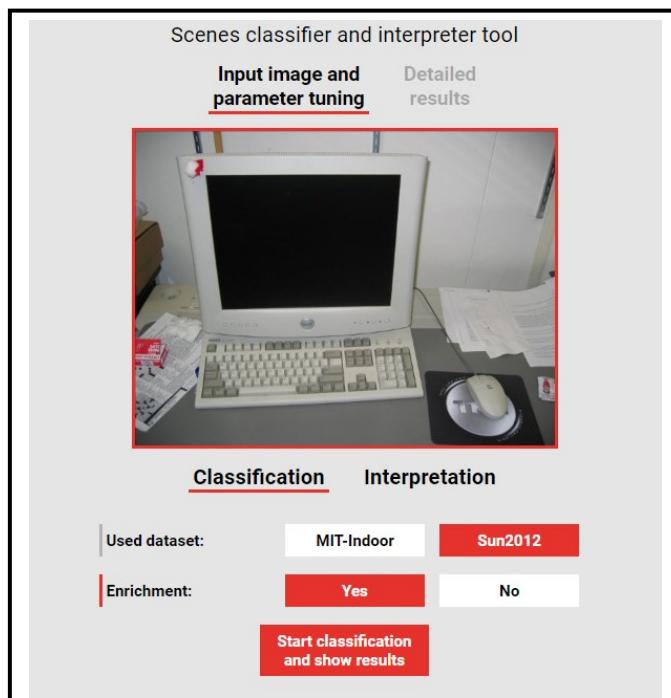


Figure 3.22 : Exemple des données en entrée de l'application implémentée

La spécification des données en entrée de l'application nous conduit à la deuxième étape : Les traitements côté serveur (back-end).

### 3.4.3.2 Traitements du côté serveur de l'application

Le côté serveur constitue le moteur de l'application développée, il reçoit en entrée les données précédentes, puis il effectue la tâche spécifiée et retourne une sortie (envoyée vers le front-end).

Au début, le serveur procède par la détection des objets et leur emplacement dans l'image entrée, en utilisant un détecteur des objets, pré-entraîné sur COCO dataset, nommé "*SSD ResNet50 v1 FPN Shared Box Predictor*" [99]. Ensuite, l'ensemble des traitements effectués dépend de la tâche spécifiée :

- Classification : La scène, représentée par la liste de ces objets, est enrichie ou non, suivant le choix de l'utilisateur. Après, la scène est classifiée, en utilisant le modèle correspondant aux meilleures valeurs des paramètres du dataset choisi). À la fin, la liste des catégories prédites et leurs probabilités est renvoyée (avec l'image et ses objets détectés) au Javascript (partie côté front-end) en format JSON.

- Interprétation : Une classification en catégories standard de la scène est lancée (en suivant la même procédure que celle pour la tâche de la classification) en utilisant le modèle entraîné sur MIT-Indoor complet. De même, une classification en trois ensembles des catégories additionnelles (du dataset MIT-Indoor partiel) est faite. De l'autre côté, les relations unaires et binaires entre les objets de la scène sont identifiées, puis définies. Ensuite, l'ensemble des résultats de la classification en catégories standard, classification en catégories additionnelles et la définition des relations est passé à la procédure de génération des descriptions des scènes. À la fin, la description générée, l'image et ses objets détectés, ainsi que le graphe des relations de la scène sont renvoyés au Javascript (côté front-end) en utilisant le format JSON.

Une fois les traitements du serveur effectués, nous entamons la dernière étape qui consiste à afficher les résultats obtenus.

### 3.4.3.3 Données en sortie de l'application

Les résultats du back-end, reçus par le front-end, sont structurés et affichés dans l'interface de l'application à l'utilisateur. L'affichage contient l'image avec les objets détectés, identifiés par des bounding-boxes. Pour les autres informations, ils dépendent, bien-évidemment, de la tâche effectuée :

- Classification : Affichage de la catégorie la plus probable, ainsi qu'un tableau trié en ordre décroissant des probabilités de chaque catégorie du dataset choisi. La figure 3.23 montre un exemple de l'affichage en sortie pour la tâche de classification.

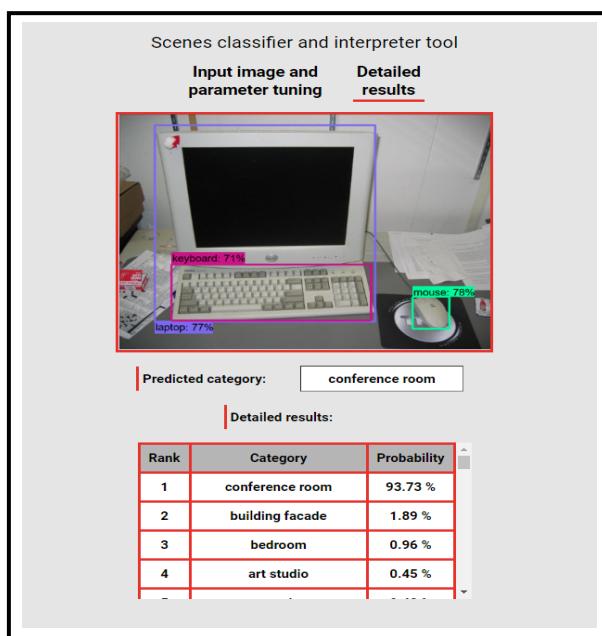
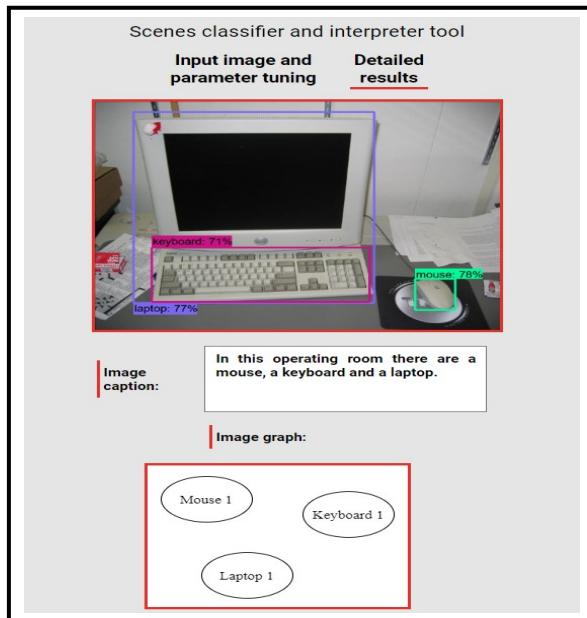


Figure 3.23 : Exemple de l'affichage en sortie pour la tâche de la classification de l'application implémentée

- Interprétation : Affichage de la description générée pour l'image, ainsi que le graphe des relations. La *figure 3.24* présente un exemple de l'affichage en sortie pour la tâche de l'interprétation.



**Figure 3.24 : Exemple de l'affichage en sortie pour la tâche de l'interprétation de l'application implémentée**

Une fois les données en sortie affichées, l'utilisateur peut revenir en arrière (en cliquant sur "Input image and parameter tuning") et refaire tout le processus à nouveau.

La présentation de l'application développée achève le chapitre courant (Implémentation et résultats). Dans ce qui suit, nous résumons, de manière brève et concise, les principaux points abordés tout au long de ce chapitre.

### 3.5 Conclusion

Ce chapitre est une mise en pratique de nos approches proposées dans la classification et l'interprétation des scènes basées sur les objets. Chaque approche est évaluée par une suite de tests (bien définis, expliqués et argumentés), dans le but de déterminer la meilleure combinaison des valeurs des paramètres. Cette meilleure combinaison obtenue est comparée avec les travaux similaires (dans la mesure du possible). De plus, les modèles (basés sur ces meilleures combinaisons) sont utilisés dans une application permettant de tester par soi-même les tâches de classification et interprétation des scènes.

La suite est une conclusion générale de ce mémoire, qui englobera tous les points abordés dans les chapitres précédents, avec un focus sur les résultats obtenus et les futures perspectives et améliorations possibles des approches proposées.

# Conclusion

## 1 Conclusion générale

Le mémoire ci-présent est une concrétisation du travail fait dans le cadre de notre projet de fin d'études, concernant la classification et l'interprétation des scènes basées sur les objets. Pour chaque thématique, nous avons proposé nos propres approches et méthodes.

La succession des chapitres de ce mémoire va du général au particulier. Plus précisément, nous avons commencé par introduire les notions phares de notre projet, leur intérêt et les éventuels obstacles à faire face. Ensuite, nous avons passé à l'état de l'art, dans lequel nous avons cerné le sens des deux thématiques abordées, en présentant leur définition formelle et les différents travaux associés, catégorisés et résumés, dans le cadre de ces thématiques. Par la suite, nous avons entamé la conception de nos approches proposées, en présentant leur conception. Après, nous avons mis ces approches à l'épreuve en effectuant une panoplie des tests (avec des graphes et leur analyse) afin de déterminer leur meilleure performance et pouvoir les comparer avec les travaux similaires et les utiliser dans l'application développée. Enfin, nous avons récapitulé tout ce que nous avons fait dans une conclusion (chapitre courant).

Pour la classification, trois variantes ont été proposées :

- Classification en catégories standard : Elle consiste à attribuer une classe (ou "catégorie") à une scène donnée. Pour cela, nous avons utilisé les LSTMs, qui reçoivent en entrée une liste triée des objets les plus importants de la scène, et retournent en sortie la classe la plus plausible. Les meilleures précisions obtenues sur les ensembles de test sont : 81.75 % pour MIT-Indoor partiel [12], 43.54 % pour MIT-Indoor complet [12] et 46.53 % pour Sun2012 [13].

- Classification avec enrichissement des scènes : Il s'agit d'ajouter les objets manquants, très probablement existants mais non-visibles (ou occlus) dans la scène, avant d'effectuer une classification en catégories standard afin de rapprocher au mieux la scène à sa vraie catégorie. Pour l'implémenter, nous avons utilisé les réseaux des neurones feedforward en tronquant la fréquence d'apparition d'un certain taux de nombre d'objets (partie : features) et en ajoutant ces fréquences tronquées des objets comme target. L'approche de l'enrichissement proposée n'est applicable que sur les datasets de taille moyenne à cause de la mémoire vive (RAM) nécessaire.

- Classification en catégories additionnelles : C'est la définition des nouveaux ensembles des catégories à valeurs restreintes (binaires, ternaires, etc.). L'assignation de ces catégories à un dataset donné est faite via un parcours des catégories originales de ce dataset et leur remplacement par la catégorie la plus adéquate de l'ensemble des catégories additionnelles défini. Ensuite, le nouveau dataset suit le même processus que celui pour la classification en catégories standard. La précision moyenne obtenue en définissant trois ensembles des catégories additionnelles (un binaire, et deux ternaires) est : 96.57 % pour MIT-Indoor partiel et 89.02 % pour MIT-Indoor complet. Cette approche n'est applicable que sur les datasets de taille moyenne (ce qui n'est pas le cas pour Sun2012) à cause du temps nécessaire pour le parcours des catégories du dataset.

Les variantes de la classification sont utilisées dans la génération des interprétations des scènes, qui est composée de :

- Identification des relations : C'est la détermination des relations unaires (un seul objet) et les relations binaires (entre deux objets) à partir de tous les objets qui composent une scène.

- Définition des relations : Il s'agit d'attribuer un attribut à chaque relation unaire et une préposition à chaque relation binaire identifiées précédemment. Pour les relations unaires, nous avons utilisé les

réseaux de neurones convolutionnels (CNNs), en achevant un taux moyen de 76.4 % sur 38 objets de COCO dataset [9]. Et pour les relations binaires, nous avons utilisé les réseaux de neurones feedforward, en obtenant un taux de 43.45 % sur un ensemble de scènes de VisualGenome [16] compatibles avec COCO dataset, pour 50 types des relations binaires les plus fréquentes.

- Génération des interprétations : C'est la construction d'un texte descriptif d'une scène donnée. Pour cela, nous avons utilisé une approche basée sur des règles, qui exploitent un graphe des relations - définies- d'une scène pour générer des sous-phrases. Ces dernières, et avec les catégories prédictes avec les différentes variantes de la classification, sont regroupées pour former une description textuelle complète d'une scène. L'évaluation des descriptions générées est faite sur les scènes de COCO dataset, via un questionnaire public avec Google Forms [89], avec une notation de chaque scène sur 4 (qualité de la description), nous avons obtenu un taux des interprétations acceptables de 53 %.

## 2 Améliorations et perspectives

Les approches proposées dans notre travail ne sont pas parfaites, elles peuvent être améliorées ou remplacées par d'autres approches plus performantes. Pour cela, nous listons quelques améliorations possibles pour les différentes étapes de la classification et l'interprétation :

- Utilisation des datasets plus riches et variés : L'un des facteurs les plus importants pour l'entraînement des modèles est la qualité du dataset. Et comme la plupart de nos approches consistent en entraînement des modèles, l'utilisation des datasets plus riches et variés dont les scènes sont mieux et correctement annotées, les catégories sont variées et le nombre des scènes par catégorie est suffisant, pourra être bénéfique sur la précision finale obtenue.
- Utilisation d'autres architectures ou classifieurs : En modifiant la configuration de l'architecture utilisée (encodage des données en entrée / sortie, variation du nombre des couches et neurones, etc.) ou carrément, en employant des classifieurs non testés dans notre travail (arbres de décision, SVMs, etc.).
- Utilisation des plateformes de développement plus performantes : Pour entraîner nos modèles, nous avons utilisé Google Colab [19], qui offre une taille acceptable de la RAM et GPUs gratuits. Cependant, les modèles de ces GPUs sont loin d'être les plus performants, et souvent partagés avec les autres utilisateurs de cette plateforme. Ainsi, dans le but d'accélérer l'entraînement des modèles, et par conséquence, faire davantage des essais et tests, l'utilisation des plateformes plus performantes, souvent payantes, telles que : Google Cloud [90], Microsoft Azure [91] ou Amazon Web Services [92] pourraient envisagé.
- Conception d'autres approches pour l'enrichissement des scènes : L'impact de notre approche de l'enrichissement sur la classification est minime. Ainsi, il sera intéressant de concevoir d'autres approches afin de tirer le meilleur de l'enrichissement, qui est une idée prometteuse. Notons que nous avons essayé d'utiliser les règles d'association (avec l'algorithme Apriori [93]), mais sans souci (nous avons noté une détérioration du taux de précision de la classification).
- Utilisation des meilleurs détecteurs d'objets : Le point de départ de la classification et l'interprétation est les objets identifiés (contenus dans une scène). Ainsi, l'utilisation des meilleurs détecteurs des objets, pré-entraînés ou entraînés à partir de zéro par soi-même, pourra-être avantageux sur la qualité des résultats obtenus. Cette suggestion est liée, bien-évidemment, à la performance de la plateforme de développement utilisée.
- Définition davantage des règles dans la génération des interprétations : En traitant, par exemple, minutieusement des tailles des sous-graphes isolés supérieures à 3 (contenant plus de trois noeuds).
- Développement des techniques basées sur l'apprentissage automatique pour la génération des interprétations des scènes : Une alternative de l'approche, basée sur les règles, proposée dans notre travail. Notons que nous avons essayé d'utiliser un modèle exploitant le graphe des relations d'une scène, nommé GCN (Graph Convolutional Network, implémentation de StellarGraph [100]), en le

fusionnant avec un LSTM (architecture nommée GCN-LSTM, telle que proposée dans [101]), mais nous n'avons pas obtenu des résultats satisfaisants par souci de l'encodage des données en entrée du GCN.

- Exploitation des low-level features d'une scène : Au lieu, comme dans notre travail, les high-level features (objets d'une scène). L'utilisation des classifieurs basés sur les low-level features peuvent, peut-être, extraire des caractéristiques qui définissent une scène mieux qu'une simple liste d'objets.

# Bibliographie

- [1] R. Girshick. "Fast r-cnn." Proceedings of the IEEE international conference on computer vision. 2015.
- [2] W. Liu, D. Anguelov, D. Erhan et al. "Ssd: Single shot multibox detector." European conference on computer vision. Springer, Cham, 2016.
- [3] S. Ren, K. He, R. Girshick, et al. "Faster r-cnn: Towards real-time object detection with region proposal networks." Advances in neural information processing systems. 2015.
- [4] T. Lin, P. Dollár, R. Girshick, et al. "Feature pyramid networks for object detection." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.
- [5] A. Van Etten. "You only look twice: Rapid multi-scale object detection in satellite imagery." arXiv preprint arXiv:1805.09512 (2018).
- [6] Y. Chen, W. Li, C. Sakaridis, et al. "Domain adaptive faster r-cnn for object detection in the wild." Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.
- [7] C. Zhu, Y. He, M. Savvides. "Feature selective anchor-free module for single-shot object detection." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2019.
- [8] A. Bochkovskiy, C. Y. Wang, H. Y. M. Liao. "YOLOv4: Optimal Speed and Accuracy of Object Detection." arXiv preprint arXiv:2004.10934 (2020).
- [9] T. Y. Lin, M. Maire, S. Belongie, et al. "Microsoft coco: Common objects in context." European conference on computer vision. Springer, Cham, 2014.
- [10] A. Kuznetsova, H. Rom, N. Alldrin, et al. "The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale." arXiv preprint arXiv:1811.00982 (2018).
- [11] B. C. Russell, A. Torralba, K. P. Murphy, et al. "LabelMe: a database and web-based tool for image annotation." International journal of computer vision 77.1-3 (2008): 157-173.
- [12] A. Quattoni, A. Torralba. "Recognizing indoor scenes." 2009 IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 2009.
- [13] J. Xiao, J. Hays, K. A. Ehinger, et al. "Sun database: Large-scale scene recognition from abbey to zoo." 2010 IEEE computer society conference on computer vision and pattern recognition. IEEE, 2010.
- [14] B. Lamine, B. Nadia. "Scene Classification Using Hidden Markov Models." (2017).
- [15] <https://colah.github.io/posts/2015-08-Understanding-LSTMs/> (Last checked: 05/09/2020).
- [16] R. Krishna, Y. Zhu, O. Groth, et al. "Visual genome: Connecting language and vision using crowdsourced dense image annotations." International journal of computer vision 123.1 (2017): 32-73.
- [17] <https://www.tensorflow.org/> (Last checked: 05/09/2020).
- [18] <https://keras.io/> (Last checked: 05/09/2020).
- [19] <https://colab.research.google.com/> (Last checked: 05/09/2020).

- [20] C. Rashtchian, P. Young, M. Hodosh, et al. "Collecting image annotations using amazon's mechanical turk." Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk. 2010.
- [21] [https://www.tensorflow.org/api\\_docs/python/tf/keras/applications/InceptionResNetV2](https://www.tensorflow.org/api_docs/python/tf/keras/applications/InceptionResNetV2) (Last checked: 05/09/2020).
- [22] J. Deng, W. Dong, R. Socher, et al. "Imagenet: A large-scale hierarchical image database." 2009 IEEE conference on computer vision and pattern recognition. Ieee, 2009.
- [23] Python Software Foundation. Python Language Reference, version 3.7. Available at "<https://www.python.org/>".
- [24] J. Zhu, L. J. Li, L. Fei-Fei, et al. "Large margin learning of upstream scene understanding models." Advances in neural information processing systems. 2010.
- [25] M. Pandey, S. Lazebnik. "Scene recognition and weakly supervised object localization with deformable part-based models." 2011 International Conference on Computer Vision. IEEE, 2011.
- [26] J. Wu, J. M. Rehg. "Centrist: A visual descriptor for scene categorization." IEEE transactions on pattern analysis and machine intelligence 33.8 (2010): 1489-1501.
- [27] L. J. Li, H. Su, L. Fei-Fei, et al. "Object bank: A high-level image representation for scene classification & semantic feature sparsification." Advances in neural information processing systems. 2010.
- [28] S. Singh, A. Gupta, A. A. Efros. "Unsupervised discovery of mid-level discriminative patches." European Conference on Computer Vision. Springer, Berlin, Heidelberg, 2012.
- [29] Z. Zuo, G. Wang, B. Shuai, et al. "Exemplar based deep discriminative and shareable feature learning for scene image classification." Pattern Recognition 48.10 (2015): 3004-3015.
- [30] J. Donahue, Y. Jia, O. Vinyals, et al. "Decaf: A deep convolutional activation feature for generic visual recognition." International conference on machine learning. 2014.
- [31] M. Juneja, A. Vedaldi, C. V. Jawahar, et al. "Blocks that shout: Distinctive parts for scene classification." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2013.
- [32] C. Doersch, A. Gupta, A. A. Efros. "Mid-level visual element discovery as discriminative mode seeking." Advances in neural information processing systems. 2013.
- [33] L. Liu, C. Shen, L. Wang, et al. "Encoding high dimensional local features by sparse coding based fisher vectors." Advances in neural information processing systems. 2014.
- [34] M. Dixit, S. Chen, D. Gao, et al. "Scene classification with semantic fisher vectors." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.
- [35] D. Yoo, S. Park, J. Y. Lee, et al. "Multi-scale pyramid pooling for deep convolutional representation." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. 2015.
- [36] [http://download.tensorflow.org/models/object\\_detection/faster\\_rcnn\\_inception\\_resnet\\_v2\\_atrous\\_oid\\_v4\\_2018\\_12\\_12.tar.gz](http://download.tensorflow.org/models/object_detection/faster_rcnn_inception_resnet_v2_atrous_oid_v4_2018_12_12.tar.gz) (Last checked: 05/09/2020).
- [37] G. Akila, R. Gayathri. "Advances in scene classification of remotely sensed high resolution images and the existing datasets." International Journal of Innovative Technology and Exploring Engineering (IJITEE). Volume-8 Issue-10, August 2019.

- [38] L. Fei-Fei, P. Perona. "A bayesian hierarchical model for learning natural scene categories." 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). Vol. 2. IEEE, 2005.
- [39] G. Heitz, S. Gould, A. Saxena, et al. "Cascaded classification models: Combining models for holistic scene understanding." Advances in Neural Information Processing Systems. 2009.
- [40] S. Lazebnik, C. Schmid, J. Ponce. "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories." 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06). Vol. 2. IEEE, 2006.
- [41] Z. Niu, G. Hua, X. Gao, et al. "Context aware topic model for scene recognition." 2012 IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 2012.
- [42] F. Zhang, B. Du, L. Zhang. "Saliency-guided unsupervised feature learning for scene classification." IEEE Transactions on Geoscience and Remote Sensing 53.4 (2014): 2175-2184.
- [43] A. Romero, C. Gatta, G. Camps-Valls. "Unsupervised deep feature extraction for remote sensing image classification." IEEE Transactions on Geoscience and Remote Sensing 54.3 (2015): 1349-1362.
- [44] A. Krizhevsky, I. Sutskever, G. E. Hinton. "Imagenet classification with deep convolutional neural networks." Advances in neural information processing systems. 2012.
- [45] K. Simonyan, A. Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).
- [46] M. D. Zeiler, R. Fergus. "Visualizing and understanding convolutional networks." European conference on computer vision. Springer, Cham, 2014.
- [47] M. Lin, Q. Chen, S. Yan. "Network in network." arXiv preprint arXiv:1312.4400 (2013).
- [48] R. Girshick, J. Donahue, T. Darrell, et al. "Rich feature hierarchies for accurate object detection and semantic segmentation." Proceedings of the IEEE conference on computer vision and pattern recognition. 2014.
- [49] C. Szegedy, W. Liu, Y. Jia, et al. "Going deeper with convolutions." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.
- [50] B. Zoph, V. Vasudevan, J. Shlens, et al. "Learning transferable architectures for scalable image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.
- [51] G. K. Langat, H. D. Jun, A. Oad, et al. "Object and Feature-Based Scene Classification." (2016).
- [52] Z. Li, X. Cai, Y. Liu, et al. "A Novel Gaussian–Bernoulli Based Convolutional Deep Belief Networks for Image Feature Extraction." Neural Processing Letters 49.1 (2019): 305-319.
- [53] G. Cheng, J. Han, X. Lu. "Remote sensing image scene classification: Benchmark and state of the art." Proceedings of the IEEE 105.10 (2017): 1865-1883.
- [54] G. Cheng, C. Yang, X. Yao, et al. "When deep learning meets metric learning: Remote sensing image scene classification via learning discriminative CNNs." IEEE transactions on geoscience and remote sensing 56.5 (2018): 2811-2821.
- [55] C. Shorten, T. M. Khoshgoftaar. "A survey on image data augmentation for deep learning." Journal of Big Data 6.1 (2019): 60.
- [56] A. Galdran, A. Alvarez-Gila, M. I. Meyer, et al. "Data-driven color augmentation techniques for deep skin image analysis." arXiv preprint arXiv:1703.03702 (2017).
- [57] A. J. Ratner, H. Ehrenberg, Z. Hussain, et al. "Learning to compose domain-specific transformations for data augmentation." Advances in neural information processing systems. 2017.

- [58] A. Jurio, M. Pagola, M. Galar, et al. "A comparison study of different color spaces in clustering based image segmentation." International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems. Springer, Berlin, Heidelberg, 2010.
- [59] R. Takahashi, T. Matsubara, K. Uehara. "Ricap: Random image cropping and patching data augmentation for deep cnns." Asian Conference on Machine Learning. 2018.
- [60] C. Summers, M. J. Dinneen. "Improved mixed-example data augmentation." 2019 IEEE Winter Conference on Applications of Computer Vision (WACV). IEEE, 2019.
- [61] Z. Zhong, L. Zheng, G. Kang, et al. "Random Erasing Data Augmentation." AAAI. 2020.
- [62] J. Vogel, B. Schiele. "Natural scene retrieval based on a semantic modeling step." International Conference on Image and Video Retrieval. Springer, Berlin, Heidelberg, 2004.
- [63] A. Bosch, A. Zisserman, X. Muñoz. "Scene classification via pLSA." European conference on computer vision. Springer, Berlin, Heidelberg, 2006.
- [64] L. J. Li, H. Su, Y. Lim, et al. "Objects as attributes for scene classification." European conference on computer vision. Springer, Berlin, Heidelberg, 2010.
- [65] X. Luo, J. Xu. "Object-based representation for scene classification." Canadian Conference on Artificial Intelligence. Springer, Cham, 2016.
- [66] S. Li, Z. Tao, K. Li, et al. "Visual to text: Survey of image and video captioning." IEEE Transactions on Emerging Topics in Computational Intelligence 3.4 (2019): 297-312.
- [67] P. Héde, P. A. Moëllic, J. Bourgeoys, et al. "Automatic generation of natural language description for images." RIAO. 2004.
- [68] A. Farhadi, M. Hejrati, M. A. Sadeghi, et al. "Every picture tells a story: Generating sentences from images." European conference on computer vision. Springer, Berlin, Heidelberg, 2010.
- [69] S. Li, G. Kulkarni, T. Berg, et al. "Composing simple image descriptions using web-scale n-grams." Proceedings of the Fifteenth Conference on Computational Natural Language Learning. 2011.
- [70] G. Kulkarni, V. Premraj, V. Ordonez, et al. "Babytalk: Understanding and generating simple image descriptions." IEEE Transactions on Pattern Analysis and Machine Intelligence 35.12 (2013): 2891-2903.
- [71] A. Aker, R. Gaizauskas. "Generating image descriptions using dependency relational patterns." Proceedings of the 48th annual meeting of the association for computational linguistics. Association for Computational Linguistics, 2010.
- [72] B. Z. Yao, X. Yang, L. Lin, et al. "I2t: Image parsing to text description." Proceedings of the IEEE 98.8 (2010): 1485-1508.
- [73] Y. Feng, M. Lapata. "How many words is a picture worth? automatic caption generation for news images." Proceedings of the 48th annual meeting of the Association for Computational Linguistics. 2010.
- [74] Y. Yang, C. Teo, H. Daumé III, et al. "Corpus-guided sentence generation of natural images." Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing. 2011.
- [75] A. Gupta, P. Mannem. "From image annotation to image description." International Conference on Neural Information Processing. Springer, Berlin, Heidelberg, 2012.
- [76] D. Elliott, F. Keller. "Image description using visual dependency representations." Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing. 2013.

[77] J. Malmaud, E. Wagner, N. Chang, et al. "Cooking with semantics." Proceedings of the ACL 2014 Workshop on Semantic Parsing. 2014.

[78] V. Ordonez, G. Kulkarni, T. L. Berg. "Im2text: Describing images using 1 million captioned photographs." Advances in neural information processing systems. 2011.

[79] P. Kuznetsova, V. Ordonez, A. Berg, et al. "Collective generation of natural image descriptions." Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2012.

[80] R. Socher, A. Karpathy, Q. V. Le, et al. "Grounded compositional semantics for finding and describing images with sentences." Transactions of the Association for Computational Linguistics 2 (2014): 207-218.

[81] H. Fang, S. Gupta, F. Iandola, et al. "From captions to visual concepts and back." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.

[82] J. Mao, W. Xu, Y. Yang, et al. "Deep captioning with multimodal recurrent neural networks (m-rnn)." arXiv preprint arXiv:1412.6632 (2014).

[83] A. Karpathy, L. Fei-Fei. "Deep visual-semantic alignments for generating image descriptions." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.

[84] K. Xu, J. Ba, R. Kiros, et al. "Show, attend and tell: Neural image caption generation with visual attention." International conference on machine learning. 2015.

[85] O. Vinyals, A. Toshev, S. Bengio, et al. "Show and tell: A neural image caption generator." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.

[86] M. M. A. Baig, M. I. Shah, M. A. Wajahat, et al. "Image caption generator with novel object injection." 2018 Digital Image Computing: Techniques and Applications (DICTA). IEEE, 2018.

[87] J. Redmon, A. Farhadi. "YOLO9000: better, faster, stronger." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.

[88] M. D. Z. Hossain, F. Sohel, M. F. Shiratuddin, et al. "A comprehensive survey of deep learning for image captioning." ACM Computing Surveys (CSUR) 51.6 (2019): 1-36.

[89] <https://www.google.com/intl/fr/forms/about/> (Last checked: 05/09/2020).

[90] <https://cloud.google.com/> (Last checked: 05/09/2020).

[91] <https://azure.microsoft.com/> (Last checked: 05/09/2020).

[92] <https://aws.amazon.com/> (Last checked: 05/09/2020).

[93] R. Agrawal, H. Mannila, R. Srikant, et al. "Fast discovery of association rules." Advances in knowledge discovery and data mining 12.1 (1996): 307-328.

[94] UML, OMG. "OMG (2017) Unified Modeling Language (OMG UML) Version 2.5.1 <https://www.omg.org/spec/>" UML, 2017.

[95] <https://developer.mozilla.org/fr/docs/Web/HTML> (Last checked: 05/09/2020).

[96] <https://developer.mozilla.org/fr/docs/Web/CSS> (Last checked: 05/09/2020).

[97] <https://developer.mozilla.org/fr/docs/Web/JavaScript> (Last checked: 05/09/2020).

[98] <https://www.djangoproject.com/> (Last checked: 05/09/2020).

[99] [http://download.tensorflow.org/models/object\\_detection/ssd\\_resnet50\\_v1\\_fpn\\_shared\\_box\\_predictor\\_640x640\\_coco14\\_sync\\_2018\\_07\\_03.tar.gz](http://download.tensorflow.org/models/object_detection/ssd_resnet50_v1_fpn_shared_box_predictor_640x640_coco14_sync_2018_07_03.tar.gz) (Last checked: 05/09/2020).

[100] <https://www.stellargraph.io/> (Last checked: 05/09/2020).

[101] T. Yao, Y. Pan, Y. Li, et al. "Exploring visual relationship for image captioning." Proceedings of the European conference on computer vision (ECCV). 2018.

# Annexe A

## A.1 Tests et résultats de la classification des scènes basées sur les objets

### A.1.1 Évaluation du classifieur des scènes proposé

#### A.1.1.1 Tests sur le dataset MIT-Indoor

##### 1) Tests sur le dataset MIT-Indoor partiel

L’application du classifieur proposé sur le dataset MIT-Indoor partiel passe par deux évaluations successives : Test des paramètres du LSTM et test des paramètres propres au classifieur que nous avons mis en place. Le but étant de récupérer (et garder) les meilleures valeurs des paramètres à chaque étape de l’évaluation.

###### a - Test du LSTM

Nous commençons par tester les LSTMs. La procédure de test consiste à varier ses deux paramètres les plus importants :

- Fonction d’activation [18] : Valeurs possibles : sigmoid, softmax, softplus, softsign, tanh, selu, elu et exponential.
- Optimiseur [18] : Valeurs possibles : SGD, RMSprop, Adam, Adadelta, Adagrad, Adamax et Nadam.

Pour chaque combinaison des valeurs des paramètres, un apprentissage avec 100 époques est fait, le résultat étant la précision sur l’ensemble de test de la meilleure époque. Notons que les paramètres propres à l’approche elle-même ont été fixés à des valeurs quelconques durant tout le test ( $\beta = 0.6$  et  $N$  [nombre d’objets maximal à garder dans la troncation] = 17).

Notons que la combinaison des valeurs des paramètres est indispensable pour une procédure de test rigoureuse. Cela est dû au fait que ces paramètres sont dépendants les uns des autres. Ainsi, la variation d’un paramètre nécessite la variation de l’autre.

La figure A.1 présente la précision obtenue en fonction de l’optimiseur (les abscisses) et la fonction d’activation (les courbes coloriées).

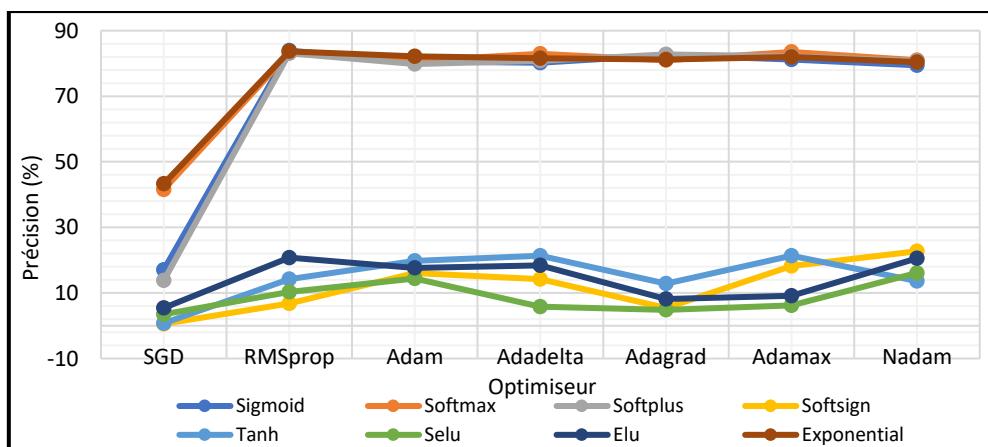


Figure A.1 : Précision du LSTM obtenue (sur MIT-Indoor partiel) en fonction de l’optimiseur et la fonction d’activation

La figure A.1 montre le taux de précision (en pourcentage) obtenu pour chaque paire de combinaison d'une valeur de l'optimiseur et une valeur de la fonction d'activation. L'observation du graphe indique clairement que l'optimiseur *SGD* enregistre le plus bas taux de précision (par rapport aux autres optimiseurs) pour toutes les fonctions d'activation. De plus, nous remarquons la formation de deux clusters de fonctions d'activations : Un cluster de faible précision (pour *softsign*, *tanh*, *selu* et *elu*, avec des taux de précision moyens de 12.01%, 14.87%, 8.74% et 14.31% respectivement) et un cluster de forte précision (pour *sigmoid*, *softmax*, *softplus* et *exponential*, avec des taux de précision moyens de 72.12%, 76.26%, 71.87% et 76.31% respectivement). Ainsi, pour ce deuxième cluster, nous notons "*sigmoid - RMSprop*" comme étant la meilleure combinaison de "fonction d'activation - optimiseur" avec un taux de précision de 83.88%.

Les meilleurs paramètres de l'architecture du LSTM (*sigmoid* [fonction d'activation] et *RMSprop* [optimiseur]) seront utilisés lors du test du classifieur proposé.

### **b - Test du classifieur proposé**

Les tests du classifieur proposé sont effectués en deux phases successives : Récupération du meilleur ensemble des valeurs pour chaque paramètre et la composition des valeurs des ensembles de chaque paramètre. Nous commençons, dans ce qui suit, par la première phase : Récupération du meilleur ensemble des valeurs pour chaque paramètre.

#### **i / Récupération du meilleur ensemble des valeurs pour chaque paramètre**

Cette phase consiste à varier chaque paramètre de l'approche proposée individuellement (un paramètre à la fois, en fixant les autres paramètres), en récupérant à chaque fois la meilleure valeur pour le paramètre testé. Le but étant de déterminer expérimentalement, l'ensemble rapproché où les meilleures précisions sont notées pour chaque paramètre. L'intérêt de cette phase est la diminution du nombre des valeurs possibles pour chaque paramètre (pour la prochaine phase), et par conséquence, avoir une meilleure présentation, lisibilité et analyse des résultats obtenus.

Le classifieur proposé contient deux paramètres :

- Calcul de l'importance des objets dans la scène (paramètre  $\beta$ ).
- Troncation de la scène (paramètre N).

Nous commençons par varier le paramètre  $\beta$ , l'échantillon de valeurs pris est représentatif et suffisant. Les variations sont entre 0 et 20 par pas de 0.5. Le test est effectué en fixant N à 17, et en entraînant le LSTM en 100 époques avec la meilleure architecture obtenue précédemment ("sigmoid - RMSprop"). La figure A.2 présente la précision obtenue en fonction de  $\beta$ .

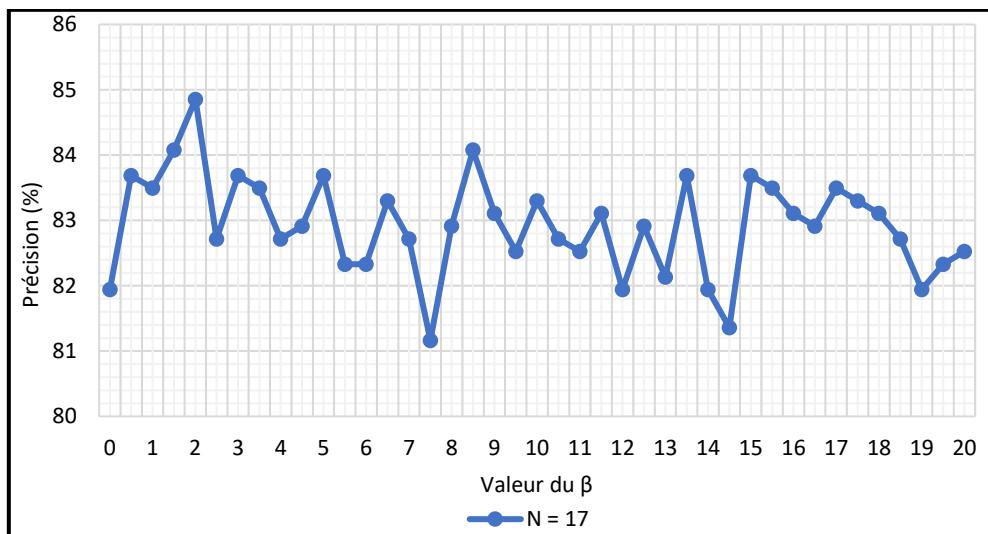
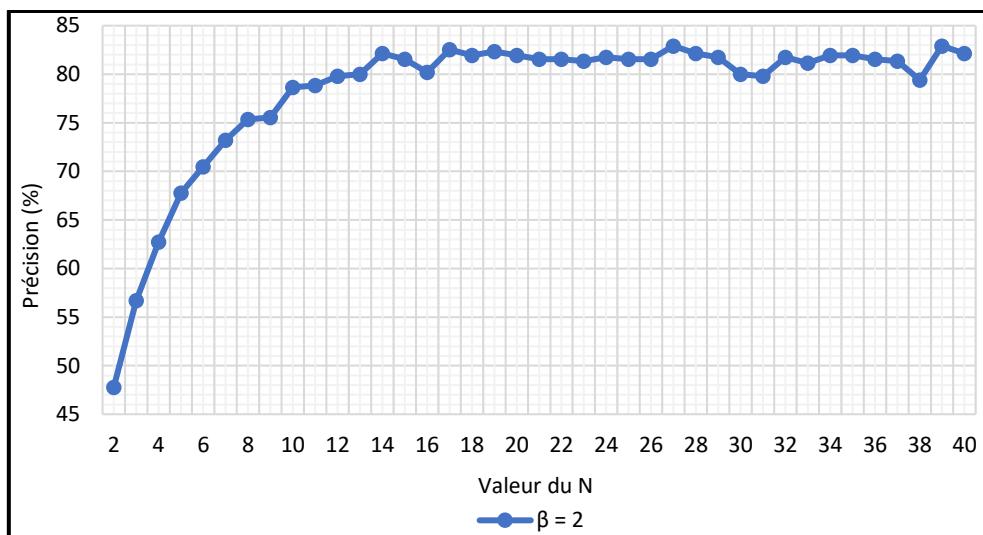


Figure A.2 : Précision obtenue (de classifieur de scènes, sur MIT-Indoor partiel) en fonction de  $\beta$

Le graphe de la *figure A.2* montre une oscillation irrégulière du taux de précision en variant les valeurs de  $\beta$ , la précision minimale notée est 81.17 % (pour  $\beta = 7.5$ ) et celle maximale est 84.85 % (pour  $\beta = 2$ ). Le meilleur ensemble de valeurs du paramètre  $\beta$  est celui pour lequel une précision supérieure ou égale à 84 % est notée, et donc, l'ensemble {1.5, 2, 8.5}.

Une fois le meilleur ensemble du  $\beta$  déduit, nous passons à la variation du paramètre N. Les valeurs prises sont successives, de 2 à 40, en effectuant à chaque fois une incrémentation de 1. Moins de 2 étant trop (effet destructif sur la sémantique de la scène, car la plupart des objets seront éliminés) et plus de 40 étant insuffisant (effet de troncation minime, car la plupart des objets seront gardés, sachant que le nombre moyen des objets par une scène de test est 19 objets, de plus, le fait de garder des objets non discriminants peut être nuisible au bon fonctionnement du classifieur).

De même que pour les tests de  $\beta$ , nous fixons ce dernier à la meilleure valeur obtenue précédemment ( $\beta = 2$ ), et en entraînant le LSTM en 100 époques avec la meilleure architecture ("sigmoid - RMSprop"). La *figure A.3* présente la précision obtenue en fonction de N.



**Figure A.3 : Précision obtenue (de classifieur de scènes, sur MIT-Indoor partiel) en fonction de N**

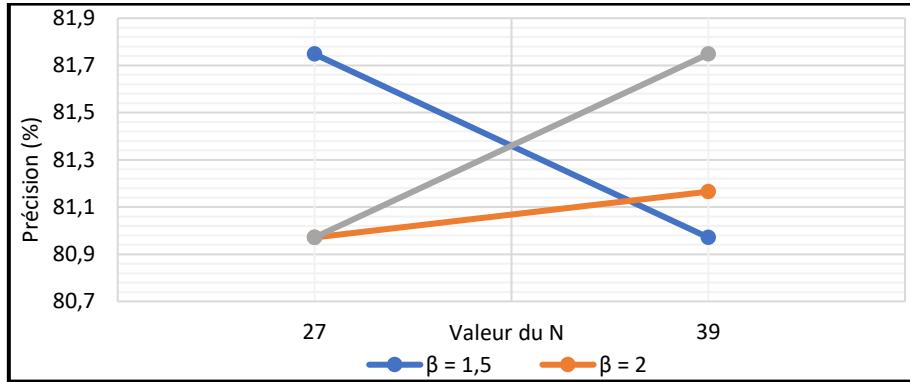
La *figure A.3* montre la variation du taux de la précision (en pourcentage) en fonction des valeurs de N. Le graphe résultant indique une amélioration progressive de la précision (en moyenne 2.86 %) pour valeurs de N allant de 2 jusqu'à 14. De plus, à partir de cette dernière valeur (N = 14), des fluctuations sont notées jusqu'à la dernière valeur testée (N = 40), avec un écart entre la meilleure et la mauvaise précision de 3.5 %. Au cours de ces fluctuations, l'ensemble des meilleures valeurs de N est déduit, il s'agit de {27, 39}, avec une précision (similaire pour les deux valeurs) de 82.91 %.

La définition de l'ensemble des valeurs pour lesquelles les meilleures précisions sont notées pour chaque paramètre, permet de passer à la phase suivante qui consiste à la génération des compositions entre ces valeurs.

### *ii / Composition des valeurs des ensembles de chaque paramètre*

Cette phase consiste à composer les valeurs restreintes obtenues, une par une, pour chaque paramètre et évaluer la précision obtenue. Cette phase est indispensable pour une procédure de test rigoureuse. Cela est dû au fait que ces paramètres sont dépendants les uns aux autres.

Pour cela, nous procédons à la génération de toutes les combinaisons possibles entre le meilleur ensemble de  $\beta$  ({1.5, 2, 8.5}) et celui de N ({27, 39}). Pour chaque combinaison, nous évaluons le taux de la précision en utilisant, bien-évidemment, le LSTM avec la meilleure architecture ("sigmoid - RMSprop") entraîné sur 100 époques. La *figure A.4* présente la précision du classifieur proposé en fonction du  $\beta$  et N.



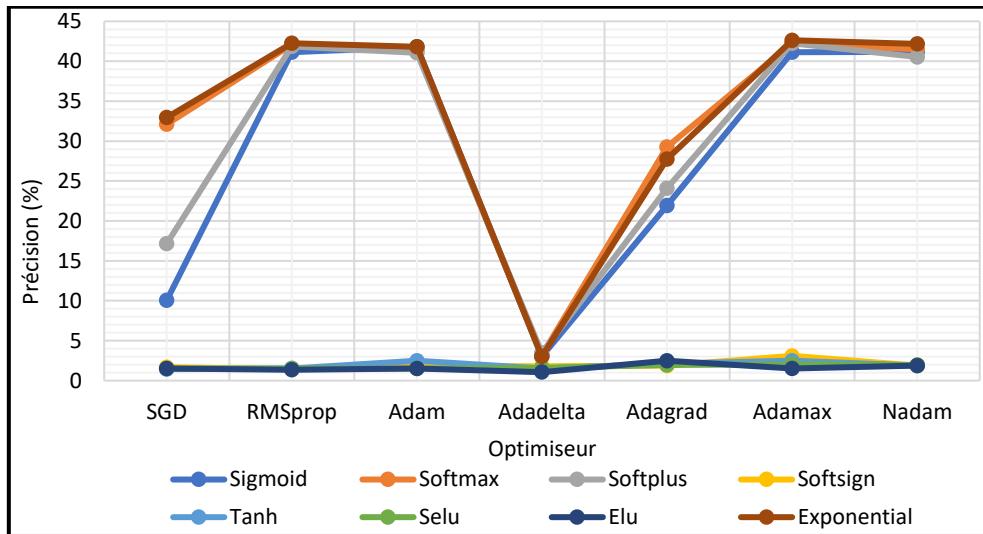
**Figure A.4 : Précision du classifieur proposé (sur MIT-Indoor partiel) en fonction des meilleures ensembles du  $\beta$  et N**

La figure A.4 montre le taux de précision (en pourcentage) pour chaque paire de combinaison d'une valeur de N et une valeur de  $\beta$ . Les résultats obtenus sont très rapprochés, avec un écart, entre la plus grande et la plus petite précision, de 0.78 %. La meilleure précision notée (81.75 %) correspond aux deux tuples ( $\beta = 1.5$ , N = 27) et ( $\beta = 8.5$ , N = 39).

## 2) Tests sur le dataset MIT-Indoor complet

### a - Test du LSTM

Le test du LSTM du classifieur de scènes proposé sur MIT-Indoor complet est fait de manière similaire au MIT-Indoor partiel. Ainsi, nous testons toutes les combinaisons des valeurs de la fonction d'activation et l'optimiseur. Chaque combinaison passe par un entraînement de 100 époques, en fixant  $\beta$  à 0.6 et N à 17 (valeurs quelconques). La figure A.5 présente la précision obtenue en fonction de l'optimiseur (les abscisses) et la fonction d'activation (les courbes colorées).



**Figure A.5 : Précision du LSTM obtenu (sur MIT-Indoor complet) en fonction de l'optimiseur et la fonction d'activation**

Le graphe de la figure A.5 montre la formation de deux ensembles des résultats (illustrés par des courbes superposées) :

- Ensemble de très basses précisions : Pour les fonctions d'activation "Softsign", "Tanh", "Selu" et "Elu", avec une précision ne dépassant pas 3.1 % (en moyenne : 1.8 %).
- Ensemble de hautes précisions : Pour les fonctions d'activation "Sigmoid", "Softmax", "Softplus", et "Exponential". Les résultats sont nettement meilleurs que le premier ensemble (basses précisions), bien qu'ils soient eux aussi relativement faibles (ne dépassant pas 45 % de précision). De plus, nous remarquons que l'optimiseur "Adadelta", suivi par "SGD" enregistrent une basse précision (par rapport

à la moyenne de cet ensemble : 31.25 %), 3.17 % et 23.07 % respectivement (en moyenne). La meilleure combinaison des valeurs "fonction d'activation - optimiseur" est ("Exponential", "Adamax"), avec une précision de 42.63 %.

Les meilleurs paramètres de l'architecture du LSTM (*Exponential* [fonction d'activation] et *Adamax* [optimiseur]) seront utilisés lors du test du classifieur proposé.

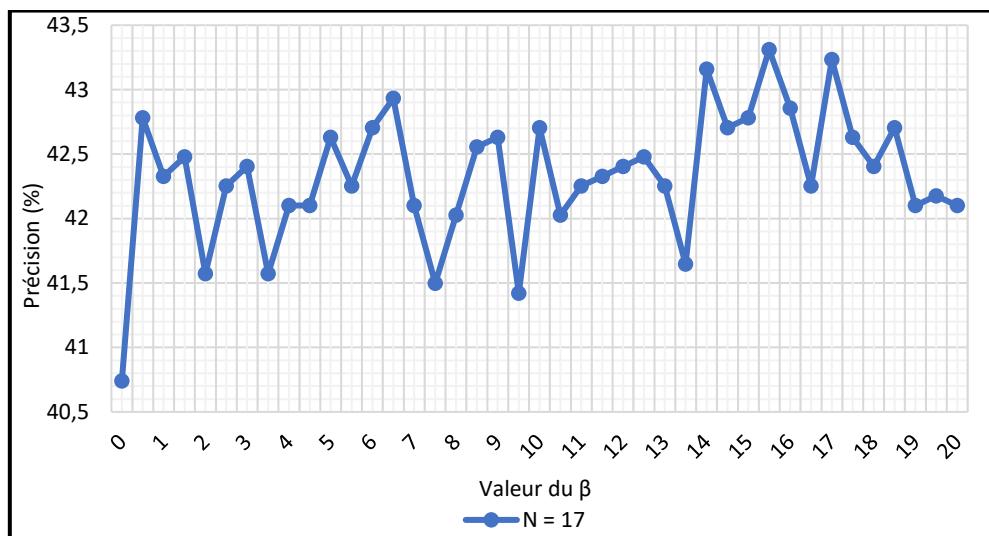
### **b - Test du classifieur proposé**

Suivant la même stratégie de l'évaluation que pour MIT-Indoor partiel, nous nous intéressons maintenant au test du classifieur proposé. Pour cela, nous passons par deux étapes.

#### **i / Récupération du meilleur ensemble des valeurs pour chaque paramètre**

Le but de cette étape est la diminution du nombre des combinaisons générées pour la prochaine étape. Elle consiste à varier individuellement les deux paramètres ( $\beta$  et N) en récupérant à chaque fois le meilleur ensemble des valeurs.

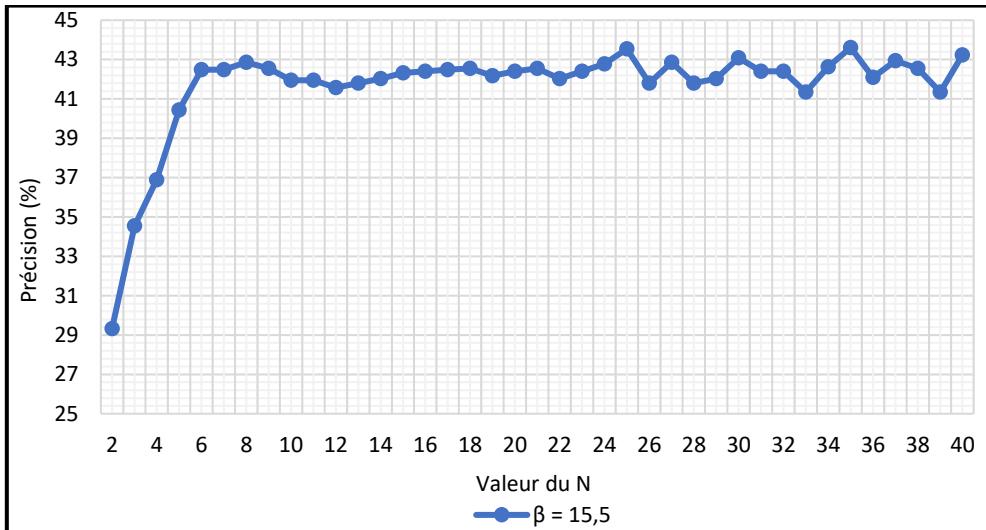
En suivant le séquencement des étapes de déroulement de notre classifieur, nous commençons par la variation de  $\beta$ , de 0 à 20 par pas de 0.5, en fixant N à 17, et en entraînant le LSTM en 100 époques avec la meilleure architecture ("Exponential - Adamax"). La *figure A.6* présente la précision obtenue en fonction de  $\beta$ .



**Figure A.6 : Précision obtenue (de classifieur de scènes, sur MIT-Indoor complet) en fonction de  $\beta$**

Le graphe de la *figure A.6* montre des fluctuations (montées et descentes, d'une valeur à autre) du taux de précision en variant les valeurs de  $\beta$ , la précision minimale notée est 40.47 % (pour  $\beta = 0$ ) et celle maximale est 43.31 % (pour  $\beta = 15.5$ ). Le meilleur ensemble de valeurs du paramètre  $\beta$  est celui pour lequel une précision supérieure ou égale à 43 % est notée, et donc, l'ensemble {14, 15.5, 17}.

La détermination du meilleur ensemble pour  $\beta$  nous conduit à la variation du paramètre N. Les valeurs prises pour ce dernier sont successives, de 2 à 40. L'entraînement est fait en fixant  $\beta$  à sa meilleure valeur ( $\beta = 15.5$ ) et en entraînant le LSTM en 100 époques avec la meilleure architecture ("Exponential - Adamax"). La *figure A.7* présente la précision obtenue en fonction de N.



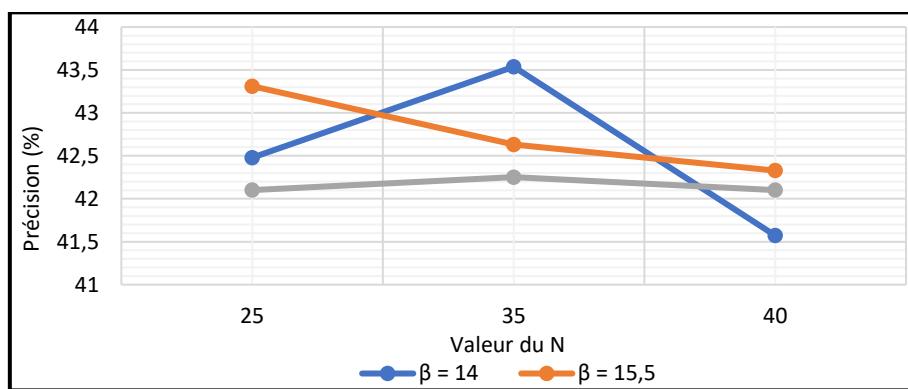
**Figure A.7 : Précision obtenue (de classifieur de scènes, sur MIT-Indoor complet) en fonction de N**

Le graphe de la *figure A.7* montre une amélioration progressive de la précision (en moyenne 2.26 %) pour valeurs de N allant de 2 jusqu'à 8. De plus, à partir de cette dernière valeur ( $N = 8$ ), des fluctuations sont notées jusqu'à la dernière valeur testée ( $N = 40$ ), avec un écart entre la meilleure et la mauvaise précision de 2.27 %. Le meilleur ensemble pour N est {25, 35, 40}, avec une précision maximale de 43.61 % (en moyenne 43.46 %).

Après la définition du meilleur ensemble des valeurs pour  $\beta$  et N, nous passons à la prochaine étape qui consiste à la génération des compositions entre ces valeurs.

#### *ii / Composition des valeurs des ensembles de chaque paramètre*

Le test de chaque combinaison du meilleur ensemble de  $\beta$  ({14, 15.5, 17}) et de N ({25, 35, 40}) est fait en entraînant le LSTM avec la meilleure architecture ("Exponential - Adamax") en 100 époques. La *figure A.8* présente la précision du classifieur proposé en fonction du  $\beta$  et N.



**Figure A.8 : Précision du classifieur proposé (sur MIT-Indoor complet) en fonction des meilleurs ensembles du  $\beta$  et N**

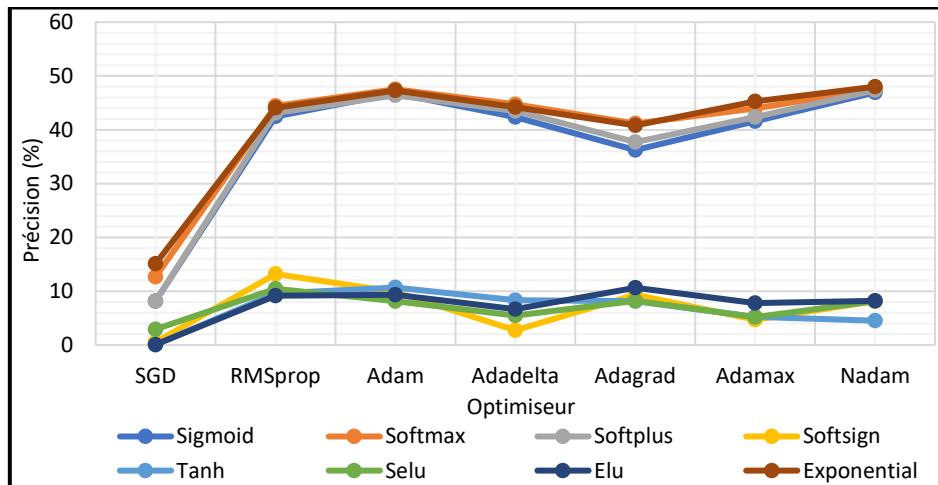
À partir du graphe présenté dans la *figure A.8*, nous remarquons aisément que la meilleure combinaison correspond au tuple ( $\beta = 14$ ,  $N = 35$ ) avec la précision 43.54 %. Les valeurs de ce tuple sont les mêmes que la meilleure valeur obtenue lors de la variation individuelle de chaque paramètre (pour déterminer le meilleur ensemble).

#### A.1.1.2 Tests sur le dataset Sun2012

##### 1) Test du LSTM

Le test du LSTM consiste à varier les deux paramètres définissant son architecture : La fonction d'activation et l'optimiseur, avec les mêmes valeurs que pour le test sur MIT-Indoor.

La combinaison des valeurs des paramètres étant nécessaire à cause de la dépendance entre eux. Pour chaque combinaison, un apprentissage de 100 époques est fait, tout en fixant les paramètres propres à l'approche proposée à des valeurs quelconques durant tout le test ( $\beta = 5$  et  $N = 20$ ). La figure A.9 présente la précision obtenue en fonction de l'*optimiseur* (les abscisses) et la *fonction d'activation* (les courbes colorées).



**Figure A.9 : Précision du LSTM obtenue (sur Sun2012) en fonction de l'optimiseur et la fonction d'activation**

Tout comme pour le test du LSTM sur le dataset MIT-Indoor, le graphe de la figure A.9 montre une basse précision pour l'optimiseur SGD (en moyenne 5.98 %), quelque-soit la fonction d'activation. De plus, la formation de deux clusters est notée : Celui des faibles précisions (en moyenne 7 %) et celui des fortes précisions (en moyenne 39.28 %). Ce dernier cluster contient, bien-évidemment, la meilleure combinaison de "fonction d'activation – optimiseur" ("exponential - Nadam") avec un taux de précision de 48.01 %.

Les meilleurs paramètres de l'architecture du LSTM (exponential [fonction d'activation] et Nadam [optimiseur]) seront utilisés lors du test du classifieur proposé.

##### 2) Test du classifieur proposé

Tout comme pour le dataset MIT-Indoor, les tests du classifieur proposé sur Sun2012 passent par deux étapes : Récupération du meilleur ensemble des valeurs pour chaque paramètre et la composition des valeurs des ensembles pour chaque paramètre. Nous commençons, dans ce qui suit, par la première étape.

###### a - Récupération du meilleur ensemble des valeurs pour chaque paramètre

L'intérêt de cette phase est la diminution du nombre des valeurs possibles pour les paramètres de l'approche proposée. Ces derniers étant deux :  $\beta$  et  $N$ .

Nous commençons par varier le paramètre  $\beta$ , les valeurs prises sont entre 0 et 20 par pas de 0.5. Le test est effectué en fixant  $N$  à 20, et en entraînant le LSTM en 100 époques avec la meilleure architecture obtenue précédemment ("exponential - Nadam"). La figure A.10 présente la précision obtenue en fonction de  $\beta$ .

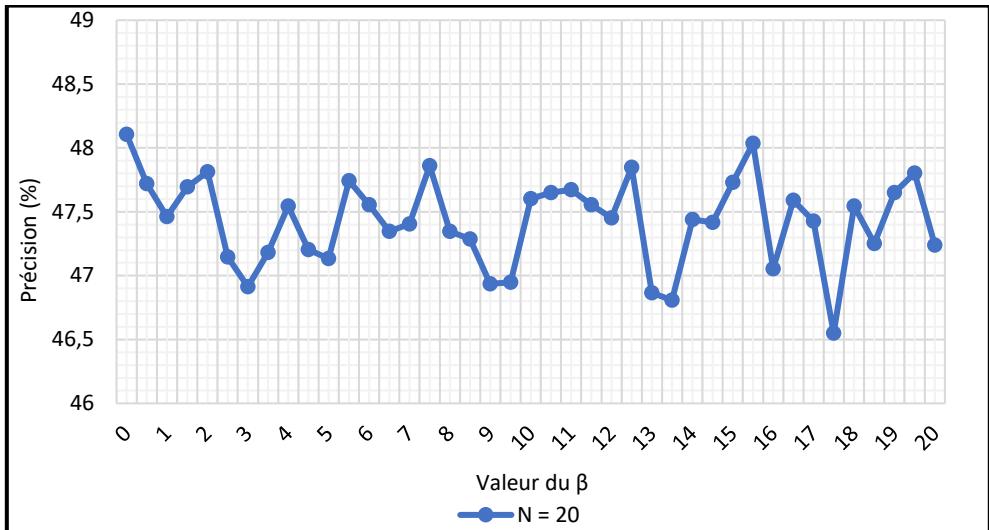


Figure A.10 : Précision obtenue (de classifieur de scènes, sur Sun2012) en fonction de  $\beta$

Le graphe de la *figure A.10* présente une oscillation du taux de précision en variant les valeurs de  $\beta$ . En effet, le point minimal est aisément identifiable (46.55 %, pour  $\beta = 17.5$ ). Pour le maximum (meilleure précision), nous remarquons deux valeurs très rapprochées : 48.11 % et 48.04 % pour  $\beta = 0$  et  $\beta = 15.5$  respectivement. Ainsi, ces deux dernières valeurs forment le meilleur ensemble pour le paramètre  $\beta$  ( $\{0, 15.5\}$ ).

Une fois le meilleur ensemble du  $\beta$  déduit, nous passons à la variation du paramètre N. Les valeurs prises sont successives, par pas de 1, de 2 à 40.

Pour cela, nous fixons  $\beta$  à la meilleure valeur obtenue précédemment ( $\beta = 0$ ), et en entraînant le LSTM en 100 époques avec la meilleure architecture ("exponential - Nadam"). La *figure A.11* présente la précision obtenue en fonction de N.

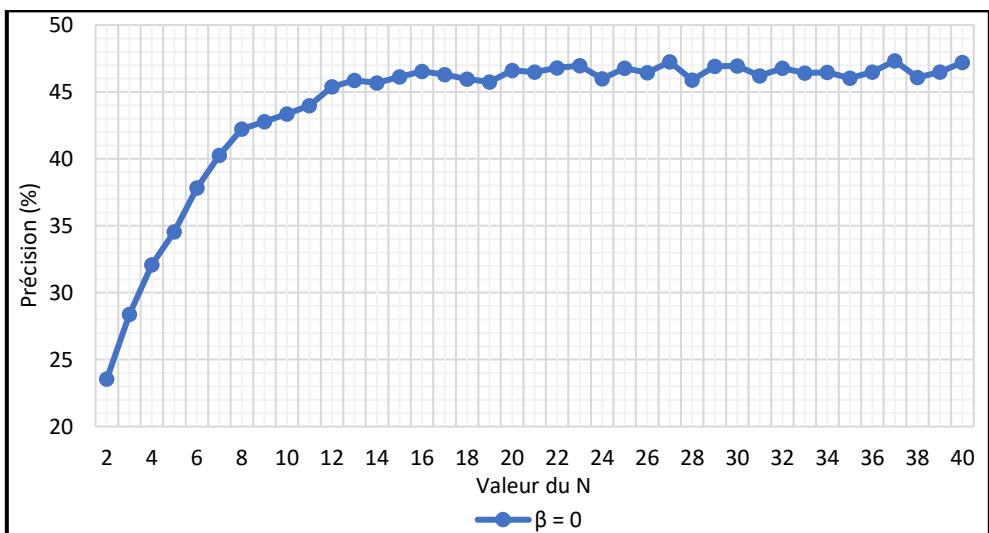


Figure A.11 : Précision obtenue (de classifieur de scènes, sur Sun2012) en fonction de N

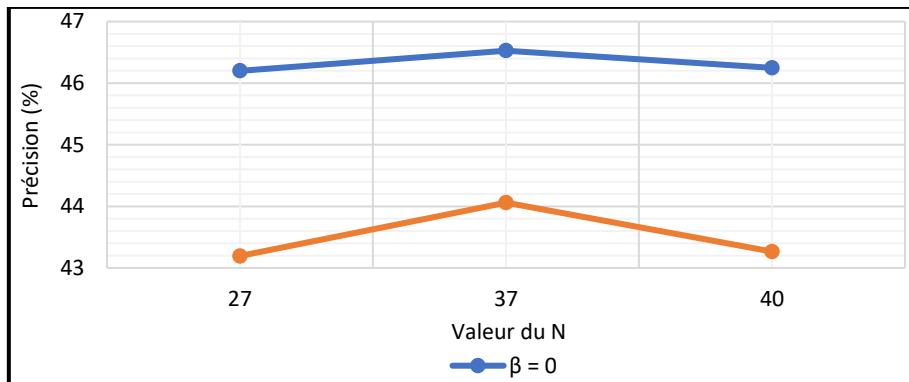
Le graphe de la *figure A.11* présente un comportement similaire que celui pour le dataset MIT-Indoor (complet et partiel). Plus précisément, une amélioration régulière de la précision (en moyenne 1.64 %) pour  $N \in [2 ; 16]$  est notée. De plus, des fluctuations sont notées, pour  $N \in [16 ; 40]$  avec un écart (de la précision) de 1.57 %. Au cours de ces fluctuations, nous remarquons trois valeurs, pour lesquels, les meilleures précisions sont notées, il s'agit de l'ensemble  $\{27, 37, 40\}$  avec une précision moyenne de 47.27 %. Cet ensemble est le meilleur ensemble pour le paramètre N.

Après la définition du meilleur ensemble pour chaque paramètre, nous passons à la phase suivante qui consiste à la génération des compositions entre les valeurs de ces ensembles.

### *b - Composition des valeurs des ensembles de chaque paramètre*

Cette phase consiste à combiner les meilleurs ensembles des deux paramètres  $\beta$  et  $N$ . La génération de ces combinaisons est nécessaire à cause de la dépendance entre ces paramètres.

Pour cela, nous procédons à la génération de toutes les combinaisons entre le meilleur ensemble de  $\beta$  ( $\{0, 15.5\}$ ) et celui de  $N$  ( $\{27, 37, 40\}$ ) en utilisant le LSTM avec la meilleure architecture ("exponential - Nadam") entraîné sur 100 époques. La *figure A.12* présente la précision du classifieur proposé en fonction du  $\beta$  et  $N$ .



**Figure A.12 : Précision du classifieur proposé (sur Sun2012) en fonction des meilleurs ensembles du  $\beta$  et  $N$**

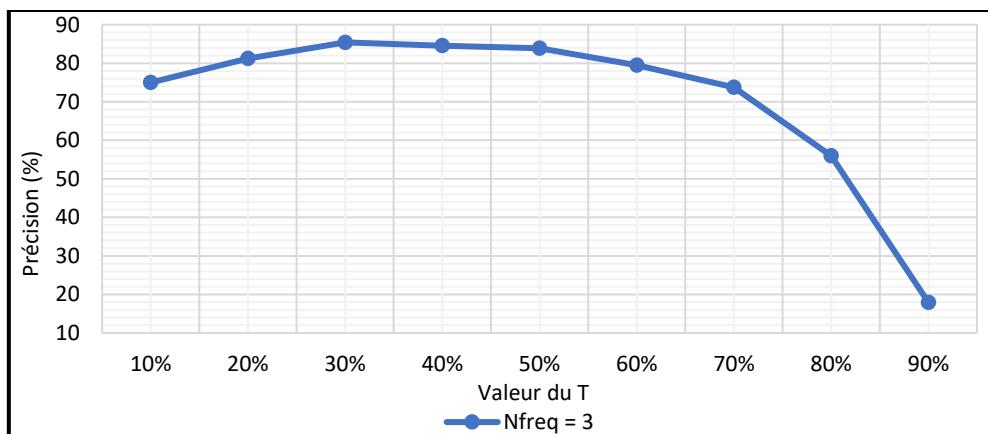
Le graphe de la *figure A.12* montre deux courbes séparées pour les deux valeurs de  $\beta$  (0 et 15,5) avec un comportement similaire (et une différence minimale de précision de 2.19 %). En effet, les faibles précisions sont notées pour  $N = 27$  (46.2 % et 43.2 % pour des valeurs de  $\beta$  égales à 0 et 15.5 respectivement). Quant aux meilleures précisions, elles correspondent à  $N = 37$  (46.53 % et 44.06 % pour  $\beta$  égale à 0 et 15.5 respectivement). Ainsi, la meilleure combinaison correspond, bien-évidemment, au tuple ( $\beta = 0$ ,  $N = 37$ ) dont la précision est 46.53 %.

#### A.1.2 Évaluation du classifieur de l'enrichissement des scènes

##### *1) Tests sur le dataset MIT-Indoor partiel*

###### *a - Récupération du meilleur ensemble des valeurs pour chaque paramètre*

Nous commençons par varier le paramètre  $T$ , de 10% à 90%, en augmentant le pourcentage à chaque fois par 10%. Nous fixons  $\beta$  à 1.5 et  $N$  à 27 (meilleure combinaison des valeurs pour la classification en catégories standard sur MIT-Indoor partiel). Ainsi, en fixant l'autre paramètre de l'enrichissement, NFreq à 3, et en entraînant le classifieur en 1000 époques, nous présentons, dans la *figure A.13*, la précision obtenue en fonction de  $T$ .



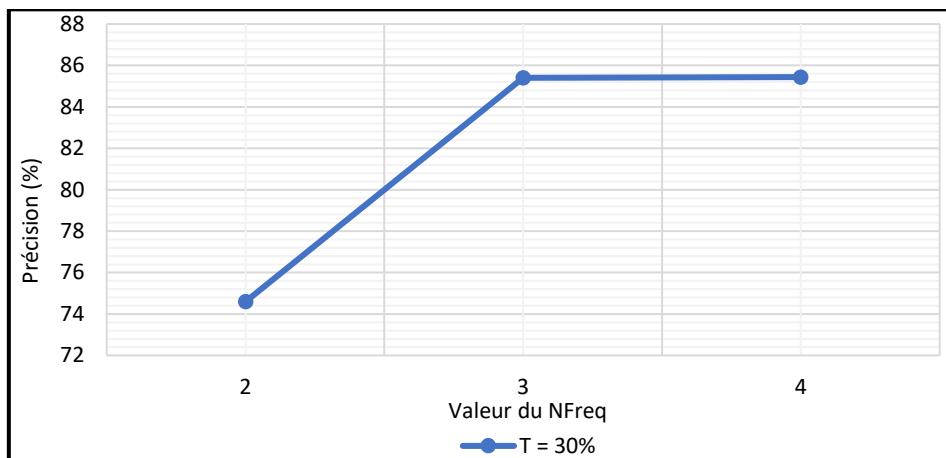
**Figure A.13 : Précision obtenue (de l'enrichissement des scènes, sur MIT-Indoor partiel) en fonction de  $T$**

La courbe du graphe de la *figure A.13* est concave, elle montre une augmentation progressive du taux de précision, de 75 % (pour  $T = 10\%$ ) jusqu'à 85.4 % (meilleure précision, pour  $T = 30\%$ ). Ensuite, cette précision diminue jusqu'à 17.95 % (pire précision, pour  $T = 90\%$ ). Le meilleur ensemble pour  $T$  contient des valeurs dont la précision minimale est 80%, et donc, {20%, 30%, 40%, 50%}.

Une fois le meilleur ensemble pour  $T$  déterminé, nous tournons vers la variation du deuxième paramètre de l'enrichissement : NFreq. Les valeurs prises pour ce dernier sont successives, de 2 à 10. L'entraînement est fait en fixant  $T$  à sa meilleure valeur notée précédemment ( $T = 30\%$ ) et en entraînant le classifieur en 1000 époques (avec la meilleure combinaison des valeurs de  $\beta$  et  $N$  notée dans la classification en catégories standard).

Notons que pendant les tests, un dépassement de la RAM disponible a été signalé pour la valeur 5 de NFreq. Ainsi, l'effectuation des tests à partir de cette valeur n'est pas possible. Et donc, les valeurs de NFreq testées se limitent à l'ensemble {2, 3, 4}.

La *figure A.14* présente la précision obtenue en fonction de NFreq.



**Figure A.14 : Précision obtenue (de l'enrichissement des scènes, sur MIT-Indoor partiel) en fonction de NFreq**

La *figure A.14* montre clairement que le meilleur ensemble pour NFreq est {3, 4}, dont la précision est presque similaire, en moyenne 85.42 %.

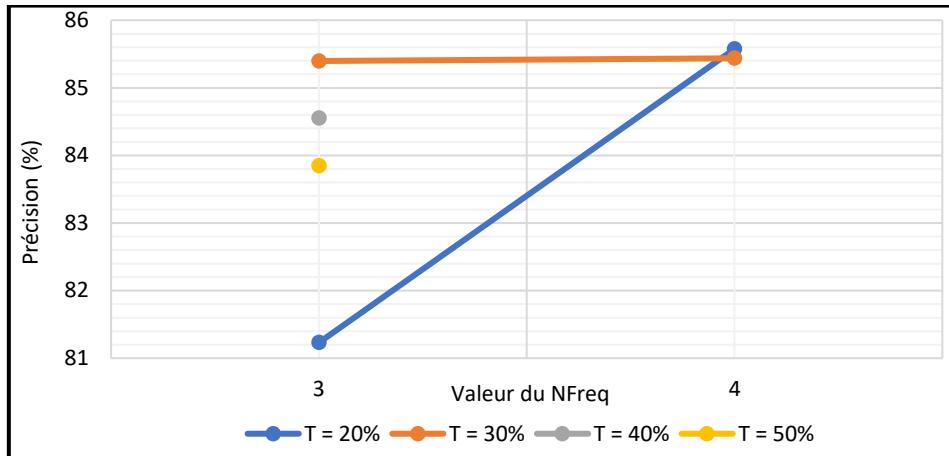
Après la définition du meilleur ensemble des valeurs pour  $T$  et NFreq, nous passons à la prochaine étape qui consiste à la génération des compositions entre ces valeurs.

#### **b - Composition des valeurs des ensembles de chaque paramètre**

Le test de chaque combinaison du meilleur ensemble de  $T$  ({20%, 30%, 40%, 50%}) et de NFreq ({3, 4}) est fait en entraînant le classifieur en 1000 époques.

De même que pour les tests que NFreq (individuellement), un dépassement de la RAM a été signalé pour les deux combinaisons : ( $T = 40\%$ , NFreq = 4) et ( $T = 50\%$ , NFreq = 4).

La *figure A.15* présente la précision du classifieur de l'enrichissement des scènes sur MIT-Indoor partiel en fonction du  $T$  et NFreq.



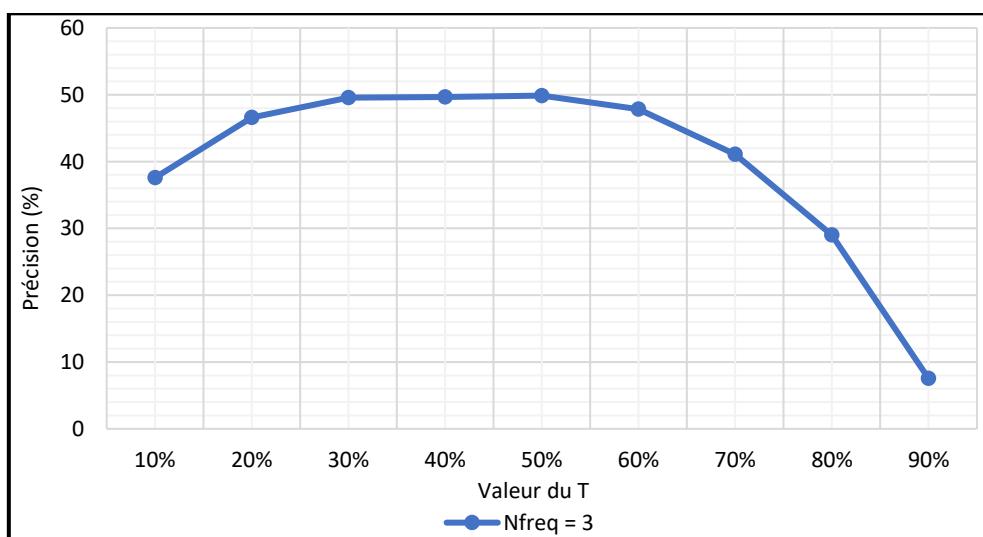
**Figure A.15 : Précision du classifieur de l'enrichissement des scènes (sur MIT-Indoor partiel) en fonction des meilleurs ensembles du T et NFreq**

En analysant le graphe de la *figure A.15*, nous remarquons aisément que la meilleure combinaison des valeurs des paramètres N et NFreq correspond au tuple ( $T = 20\%$ ,  $NFreq = 4$ ), avec une précision de 85.58 %.

## 2) Tests sur le dataset MIT-Indoor complet

### a - Récupération du meilleur ensemble des valeurs pour chaque paramètre

Comme pour les tests sur MIT-Indoor partiel, nous varions le paramètre T, de 10% à 90%, par pas de 10%, en fixant  $\beta$  à 14 et N à 35 (meilleure combinaison des valeurs pour la classification en catégories standard sur MIT-Indoor complet). Ainsi, en fixant l'autre paramètre de l'enrichissement, NFreq à 3, et en entraînant le classifieur en 1000 époques, nous présentons, dans la *figure A.16*, la précision obtenue en fonction de T.



**Figure A.16 : Précision obtenue (de l'enrichissement des scènes, sur MIT-Indoor complet) en fonction de T**

La courbe du graphe de la *figure A.16* est concave, elle montre une augmentation progressive du taux de précision, de 37.59 % (pour  $T = 10\%$ ) jusqu'à 49.88 % (meilleure précision, pour  $T = 50\%$ ). Ensuite, cette précision diminue jusqu'à 7.53 % (pire précision, pour  $T = 90\%$ ). Le meilleur ensemble pour T est construit des valeurs dont la précision tourne autour 50 %, et donc, {30%, 40%, 50%}.

Une fois le meilleur ensemble pour T déterminé, nous passons à la variation de NFreq. Les valeurs prises pour ce dernier sont successives, de 2 à 10. L'entraînement est fait en fixant T à sa meilleure valeur notée précédemment ( $T = 50\%$ ) et en entraînant le classifieur en 1000 époques (avec la meilleure

combinaison des valeurs de  $\beta$  et N notée dans la classification en catégories standard). La figure A.17 présente la précision obtenue en fonction de NFreq.

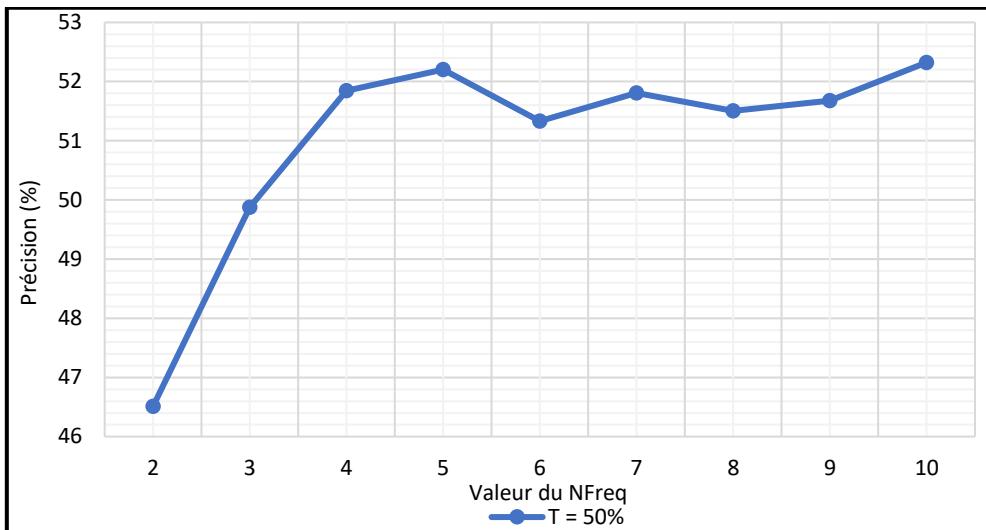


Figure A.17 : Précision obtenue (de l'enrichissement des scènes, sur MIT-Indoor complet) en fonction de NFreq

La courbe de la figure A.17 passe par deux phases :

- Incrémentale : Pour les valeurs de NFreq  $\in [2; 5]$ , où la précision est en corrélation avec la valeur de NFreq (i.e. La précision s'améliore en augmentant NFreq). La précision moyenne pour cette phase est : 50.11 % (avec un minimum de 46.51 %, pour NFreq = 2, et un maximum de 52.2 % pour NFreq = 5).
- Irrégulière : Pour les valeurs de NFreq  $\in [6; 10]$ , où des variations de la précision sont notées. La précision moyenne est : 51.73 % (avec un minimum de 51.33 %, pour NFreq = 6, et un maximum de 52.32 %, pour NFreq = 10).

Le meilleur ensemble des valeurs pour le paramètre NFreq est  $\{4, 5, \dots, 10\}$ .

Après la définition du meilleur ensemble des valeurs pour T et NFreq, nous passons à la prochaine étape qui consiste à la génération des compositions entre ces valeurs.

#### b - Composition des valeurs des ensembles de chaque paramètre

Le test de chaque combinaison du meilleur ensemble de T ( $\{30\%, 40\%, 50\%\}$ ) et de NFreq ( $\{4, 5, \dots, 10\}$ ) est fait en entraînant le classifieur en 1000 époques. La figure A.18 présente la précision du classifieur de l'enrichissement des scènes sur MIT-Indoor complet en fonction du T et NFreq.

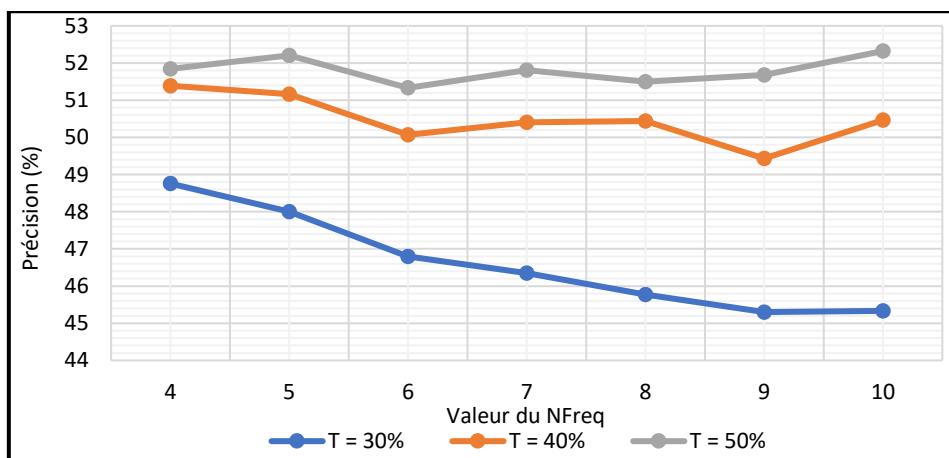


Figure A.18 : Précision du classifieur de l'enrichissement des scènes (sur MIT-Indoor complet) en fonction des meilleurs ensembles du T et NFreq

La meilleure combinaison des valeurs notée dans la *figure A.18* est ( $T = 50\%$ ,  $N_{Freq} = 10$ ), avec une précision de 52.32 %.

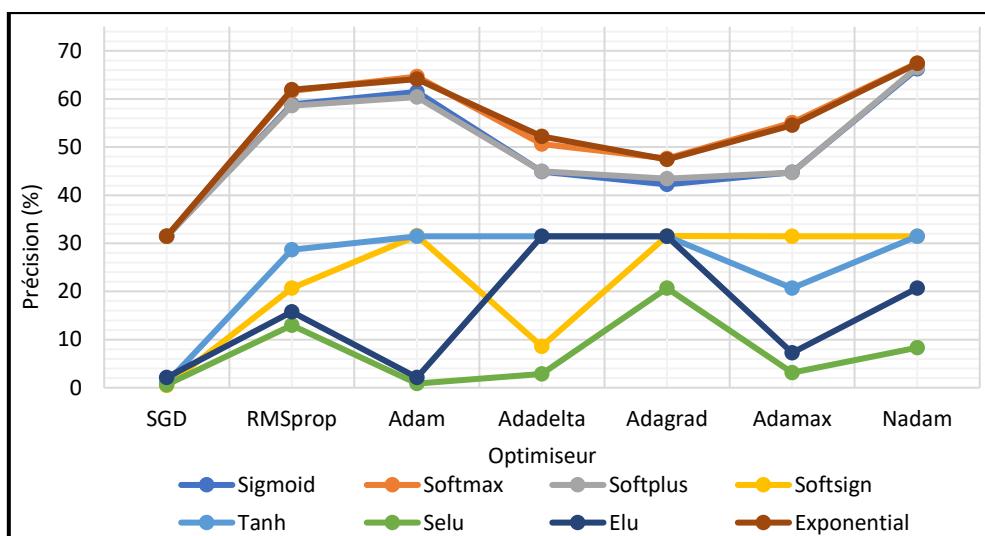
## A.2 Tests et résultats de l'interprétation des scènes basées sur les objets

### A.2.1 Évaluation de l'identificateur des relations

#### A.2.1.1 Test du LSTM

Le test du LSTM consiste à varier ses deux paramètres les plus importants (lesquels affecteront le plus le taux de précision obtenu) : La fonction d'activation et l'optimiseur. Les valeurs de ces paramètres étant similaires aux celles du LSTM du classifieur de scènes proposé.

Tout comme pour la classification, la combinaison des valeurs des paramètres est nécessaire à cause de la dépendance entre eux. Pour chaque combinaison, un apprentissage de 100 époques est fait. Notons que les paramètres propres à l'approche elle-même ont été fixés à des valeurs quelconques durant tout le test ( $\beta = 0$  et  $N$  [nombre d'objets maximal à garder dans la troncation] = 10), avec le nombre maximal des relations en sortie (paramètre non-évaluable) égal à 7. La *figure A.19* présente la précision obtenue en fonction de l'*optimiseur* (les abscisses) et la *fonction d'activation* (les courbes colorées).



**Figure A.19 : Précision du LSTM (de l'identificateur des relations) obtenue en fonction de l'optimiseur et la fonction d'activation**

La *figure A.19* montre le taux de précision (en pourcentage) obtenu pour chaque paire de combinaison d'une valeur de l'optimiseur et une valeur de la fonction d'activation. Similairement à la classification, l'observation du graphe résulte en deux :

- L'optimiseur SGD enregistre le plus bas taux de précision (par rapport aux autres optimiseurs) pour toutes les fonctions d'activation.
- La formation de deux clusters de fonctions d'activations : Un cluster de faible précision (avec une précision moyenne de 17.58%) et un cluster de forte précision (avec une précision moyenne de 52.06%). Ainsi, pour ce deuxième cluster, nous notons "softmax - Nadam" comme étant la meilleure combinaison de "fonction d'activation - optimiseur" avec un taux de précision de 67.46%.

Les meilleurs paramètres de l'architecture du LSTM (softmax [fonction d'activation] et Nadam [optimiseur]) seront utilisés lors du test du l'identificateur des relations.

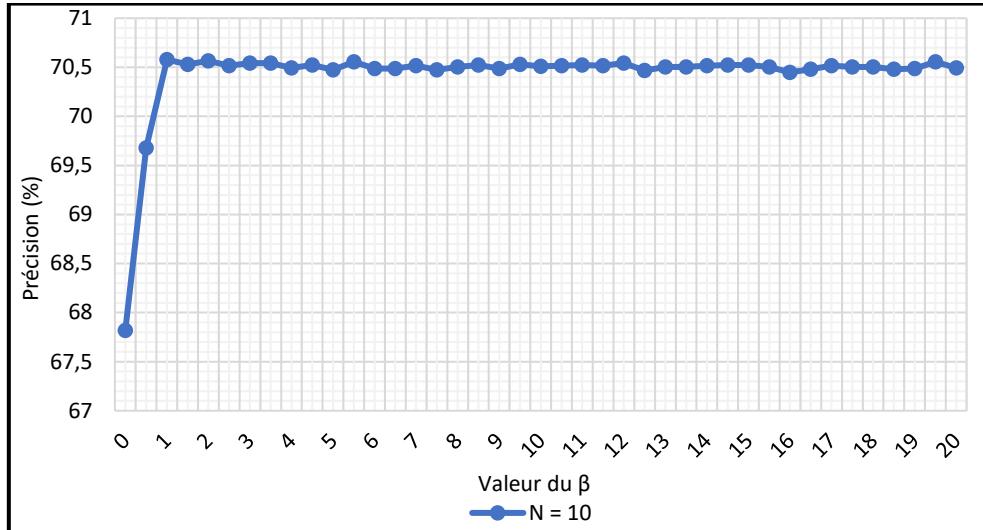
#### A.2.1.2 Test de l'identificateur des relations

Les tests de l'identificateur des relations sont effectués en deux phases successives :

### 1) Récupération du meilleur ensemble des valeurs pour chaque paramètre

Tout comme pour les tests effectués dans la partie classification, nous commençons tout d'abord par la récupération du meilleur ensemble pour chaque paramètre de l'identificateur des relations ( $\beta$  et  $N$ ).

La variation de  $\beta$  de 0 à 20 par pas de 0.5 est mise en place, en fixant  $N$  à 10 et en entraînant le classifieur LSTM en 100 époques avec la meilleure architecture obtenue précédemment ("softmax - Nadam"). La *figure A.20* présente la précision obtenue en fonction de  $\beta$ .

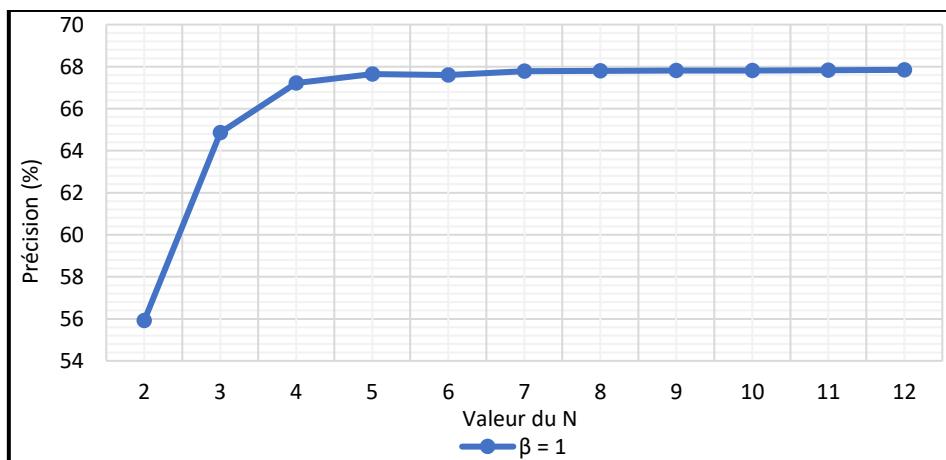


**Figure A.20 : Précision obtenue (de l'identificateur des relations) en fonction de  $\beta$**

Le graphe de la *figure A.20* passe par deux états :

- Une augmentation significative du taux de précision (du 67.82 % à 70.58 %, en moyenne 1.38 %) pour les trois valeurs de  $\beta$ , à savoir 0, 0.5 et 1.
- Une stagnation (avec quelques fluctuations négligeables, ne dépassant pas 0.08 % entre deux précisions successives) pour des valeurs de  $\beta$  entre 1 et 20. Cette stagnation rend impossible la détermination d'un ensemble restreint de valeurs de  $\beta$ , car toutes ces valeurs, une fois combinées avec le meilleur ensemble  $N$ , peuvent atteindre la meilleure précision. Pour cela, le meilleur ensemble de valeurs pour  $\beta$  est {1, 1.5, 2, ..., 19, 19.5, 20}.

Une fois le meilleur ensemble de  $\beta$  déterminé, nous refaisons la même chose pour  $N$  en le variant de 2 à 12 (nombre maximal les objets uniques dans une scène de COCO dataset), en fixant  $\beta$  à 1 (la meilleure valeur obtenue précédemment), et en entraînant le classifieur LSTM en 100 époques avec la meilleure architecture obtenue précédemment ("softmax - Nadam"). La *figure A.21* présente la précision obtenue en fonction de  $N$ .



**Figure A.21 : Précision obtenue (de l'identificateur des relations) en fonction de  $N$**

La figure A.21 montre le taux de précision (en pourcentage) obtenu pour chaque valeur N. Nous remarquons dans le graphe une augmentation du taux de précision entre les valeurs 2 et 3 de N, avec une différence (avantageuse) de 8.94%. De plus, en augmentant la valeur de N, nous notons une diminution de l'écart entre les précisions obtenues (2.36%, 0.44%, ...), cet écart étant négligeable à partir de la valeur 7 de N. La meilleure précision notée (67.85%) correspond à la borne supérieure de l'intervalle des valeurs testées pour N, et donc, N = 12.

Le meilleur ensemble pour N contient les valeurs supérieures ou égales à la valeur, à partir laquelle, l'écart entre les précisions notées est négligeable, et donc, l'ensemble {7, 8, 9, 10, 11, 12}.

Une fois les meilleurs ensembles des valeurs de  $\beta$  et N récupérés, nous procédons à leur composition.

## 2) Composition des valeurs des ensembles de chaque paramètre

Cette phase consiste à composer les ensembles des deux paramètres ( $\beta$  et N) obtenus. La nécessité de cette phase est due au fait que ces deux paramètres sont dépendants entre eux.

Ainsi, nous procédons à la génération de toutes les combinaisons entre le meilleur ensemble de  $\beta$  ({1, 1.5, 2, ..., 19, 19.5, 20}) et celui de N ({7, 8, 9, 10, 11, 12}) en utilisant le LSTM avec la meilleure architecture ("softmax - Nadam") entraîné sur 100 époques.

L'explosion combinatoire (234 combinaisons) causée par le grand nombre de valeurs du meilleur ensemble de  $\beta$  (39 valeurs) et N (6 valeurs), a rendu nécessaire l'application d'une étape supplémentaire dans le but d'alléger le nombre des combinaisons, et donc, améliorer la lisibilité du graphe obtenu et son analyse. Plus précisément, en analysant les taux des précisions obtenus, nous avons défini un ensemble de contraintes de telle sorte qu'une valeur de  $\beta$  ou N n'est gardée que si sa précision satisfait toutes les contraintes :

- Pour  $\beta$ , trois contraintes ont été définies : La précision moyenne  $\geq 68\%$ , la précision maximale  $\geq 68.5\%$  et le "nombre des précisions (en variant N) supérieures à 68%"  $\geq 4$ . L'application de ces contraintes a permis de diminuer le nombre des valeurs de  $\beta$  de 39 à 12.
- Pour N, deux contraintes ont été définies : La précision moyenne  $\geq 68\%$  et le "nombre des précisions (en variant  $\beta$ ) supérieures à 68%"  $\geq 30$ . Ces contraintes ont permis de diminuer le nombre des valeurs de N de 6 à 2.

La figure A.22 présente la précision de l'identificateur des relations en fonction du  $\beta$  et N.

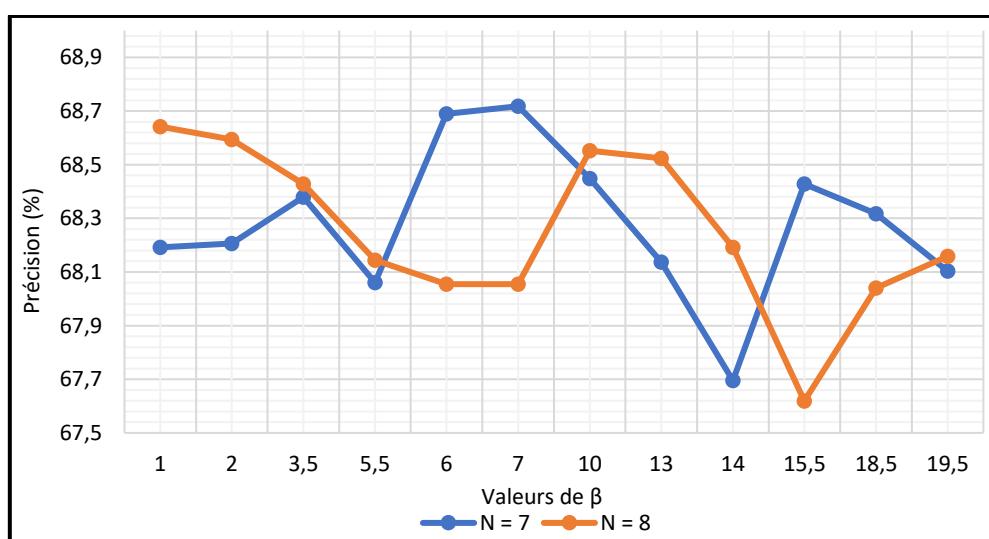


Figure A.22 : Précision de l'identificateur des relations en fonction des meilleurs ensembles du  $\beta$  et N

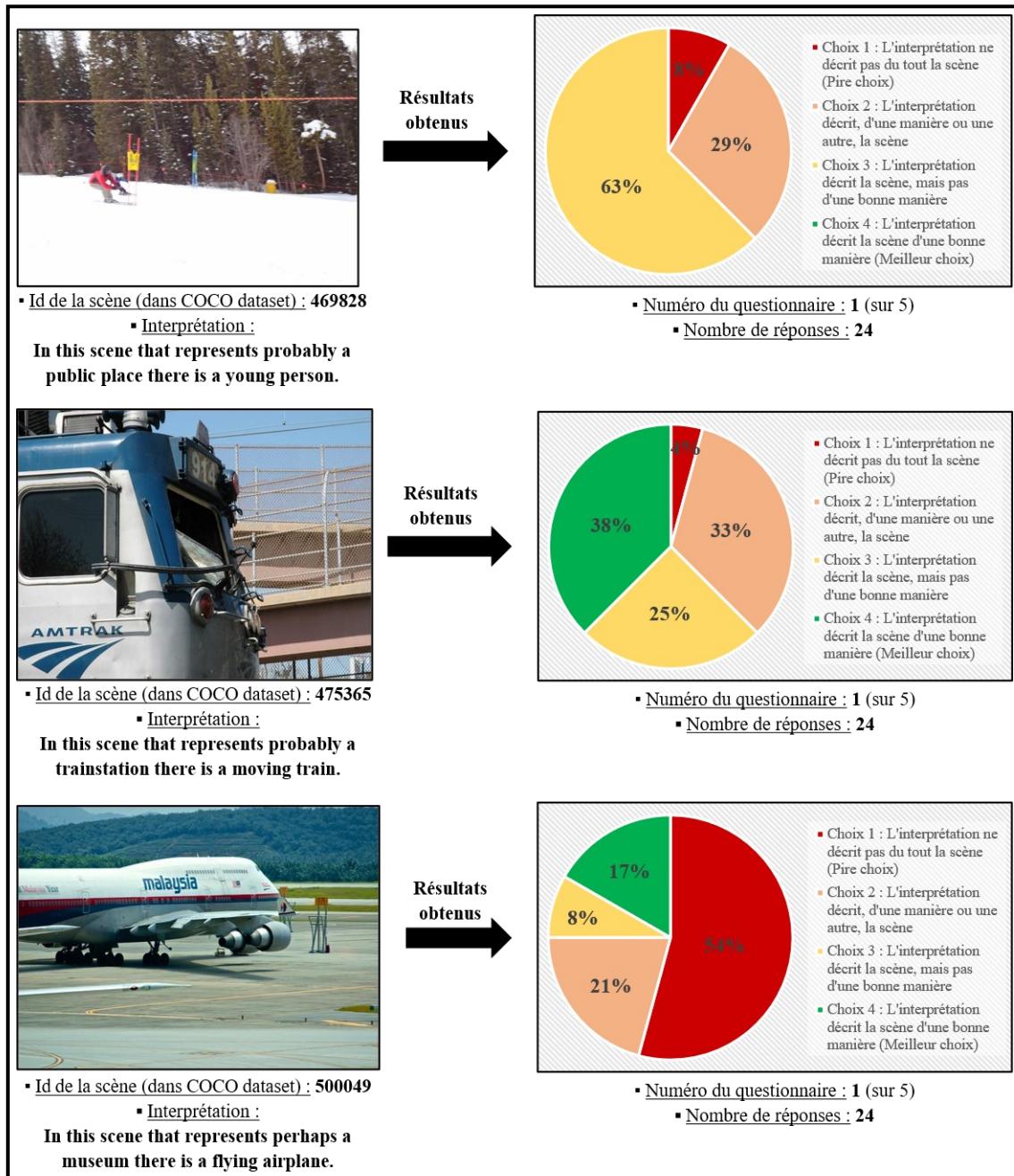
Bien que la variation des deux courbes (pour les deux valeurs de N, 7 et 8) du graphe de la figure A.22 soit différente, leurs paramètres sont similaires : Précisions moyennes de 68.28 % pour N = 7 et

68.25 pour N = 8, et un écart (entre le minimum et le maximum) de 1.02 % pour les deux valeurs. La meilleure combinaison est ( $\beta = 7$ , N = 7) avec un taux de précision égal à 68.72 %.

## Annexe B

### B.1 Résultats détaillés des questionnaires sur la qualité des interprétations générées

Nous présentons dans la *figure B.1*, pour chacun des 5 questionnaires, 5 scènes (donc, un total de 25 scènes) avec leurs interprétations, nombre de réponses et les statistiques des résultats obtenus.

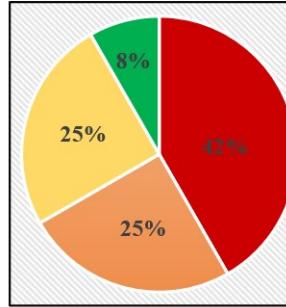
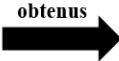




- Id de la scène (dans COCO dataset) : 501523
- Interprétation :

**In this private place there are a porcelain sink and two glass bottles.**

Résultats obtenus



- Choix 1 : L'interprétation ne décrit pas du tout la scène (Pire choix)
- Choix 2 : L'interprétation décrit, d'une manière ou une autre, la scène
- Choix 3 : L'interprétation décrit la scène, mais pas d'une bonne manière
- Choix 4 : L'interprétation décrit la scène d'une bonne manière (Meilleur choix)

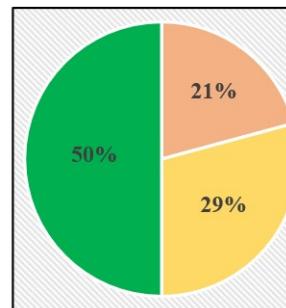
- Numéro du questionnaire : 1 (sur 5)
- Nombre de réponses : 24



- Id de la scène (dans COCO dataset) : 502599
- Interprétation :

**In this scene that represents perhaps a museum there are more than two flying airplanes.**

Résultats obtenus



- Choix 1 : L'interprétation ne décrit pas du tout la scène (Pire choix)
- Choix 2 : L'interprétation décrit, d'une manière ou une autre, la scène
- Choix 3 : L'interprétation décrit la scène, mais pas d'une bonne manière
- Choix 4 : L'interprétation décrit la scène d'une bonne manière (Meilleur choix)

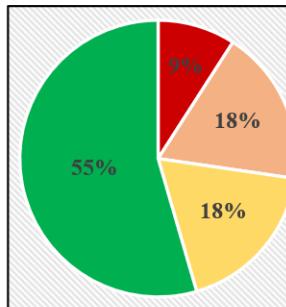
- Numéro du questionnaire : 1 (sur 5)
- Nombre de réponses : 24



- Id de la scène (dans COCO dataset) : 336265
- Interprétation :

**In this scene that represents probably a public place there is a young person.**

Résultats obtenus



- Choix 1 : L'interprétation ne décrit pas du tout la scène (Pire choix)
- Choix 2 : L'interprétation décrit, d'une manière ou une autre, la scène
- Choix 3 : L'interprétation décrit la scène, mais pas d'une bonne manière
- Choix 4 : L'interprétation décrit la scène d'une bonne manière (Meilleur choix)

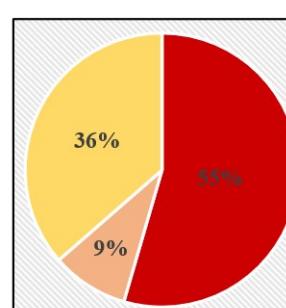
- Numéro du questionnaire : 2 (sur 5)
- Nombre de réponses : 11



- Id de la scène (dans COCO dataset) : 360097
- Interprétation :

**In this scene that represents perhaps a public place there is a person on a large suitcase.**

Résultats obtenus



- Choix 1 : L'interprétation ne décrit pas du tout la scène (Pire choix)
- Choix 2 : L'interprétation décrit, d'une manière ou une autre, la scène
- Choix 3 : L'interprétation décrit la scène, mais pas d'une bonne manière
- Choix 4 : L'interprétation décrit la scène d'une bonne manière (Meilleur choix)

- Numéro du questionnaire : 2 (sur 5)
- Nombre de réponses : 11

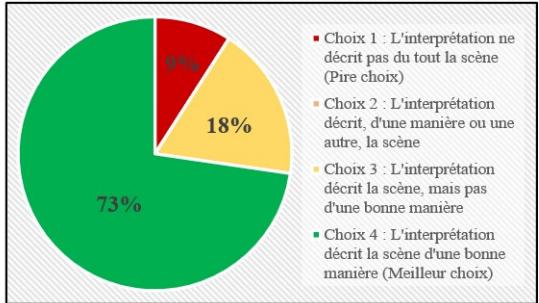


▪ Id de la scène (dans COCO dataset) : 361621

▪ Interprétation :

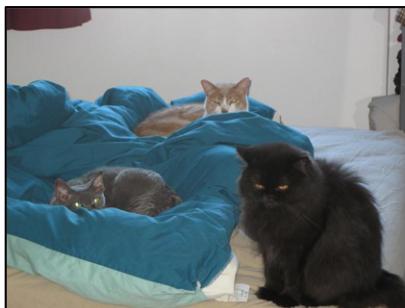
**In this bathroom there are a porcelain sink and a sitting cat.**

Résultats obtenus →



▪ Numéro du questionnaire : 2 (sur 5)

▪ Nombre de réponses : 11

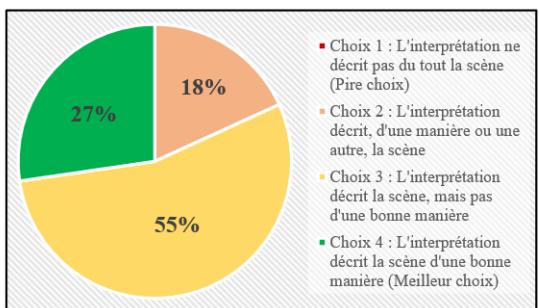


▪ Id de la scène (dans COCO dataset) : 387383

▪ Interprétation :

**In this private place there is a sitting cat in a made bed.**

Résultats obtenus →



▪ Numéro du questionnaire : 2 (sur 5)

▪ Nombre de réponses : 11

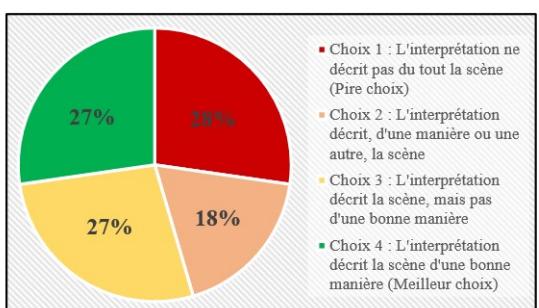


▪ Id de la scène (dans COCO dataset) : 437898

▪ Interprétation :

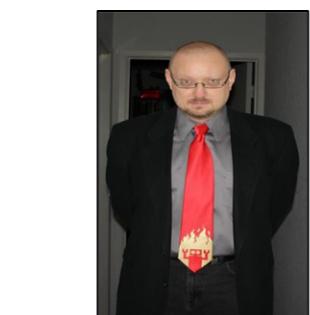
**In this kitchen there are two metal spoons.**

Résultats obtenus →



▪ Numéro du questionnaire : 2 (sur 5)

▪ Nombre de réponses : 11

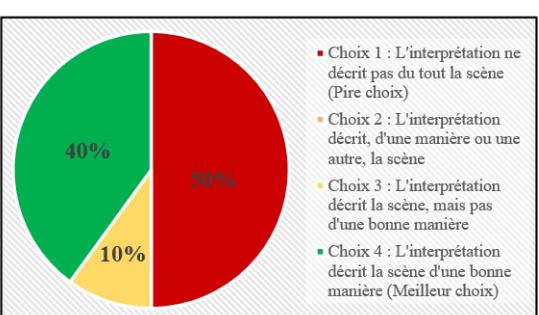


▪ Id de la scène (dans COCO dataset) : 244496

▪ Interprétation :

**In this scene that represents probably a public place there is a young person.**

Résultats obtenus →

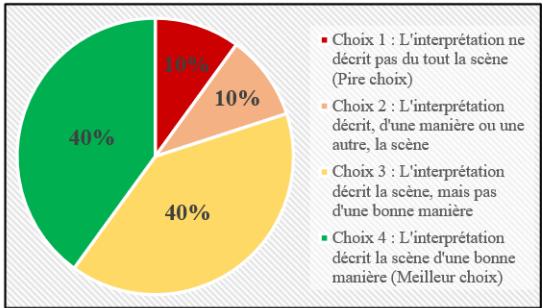


▪ Numéro du questionnaire : 3 (sur 5)

▪ Nombre de réponses : 10



Résultats obtenus →



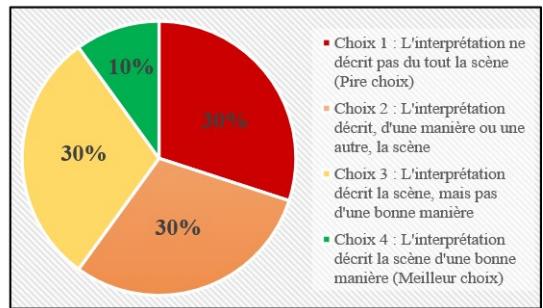
▪ Id de la scène (dans COCO dataset) : 264335

▪ Interprétation :

In this scene that represents perhaps a private place there is a large bird.



Résultats obtenus →



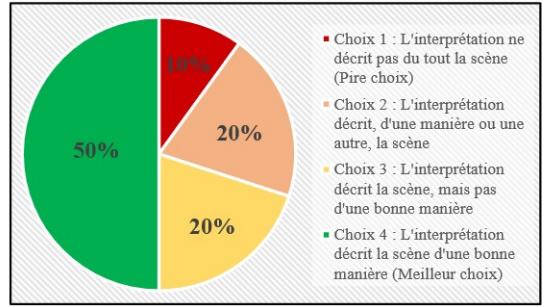
▪ Id de la scène (dans COCO dataset) : 265518

▪ Interprétation :

In this scene that represents probably a private place there is a metal fork.



Résultats obtenus →



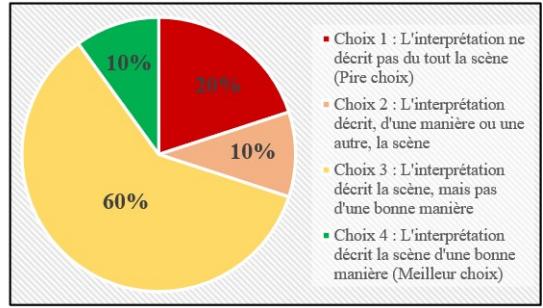
▪ Id de la scène (dans COCO dataset) : 272049

▪ Interprétation :

In this scene that represents probably a museum there is a large truck.



Résultats obtenus →



▪ Id de la scène (dans COCO dataset) : 284279

▪ Interprétation :

In this scene that represents perhaps a private place there is a large bird.

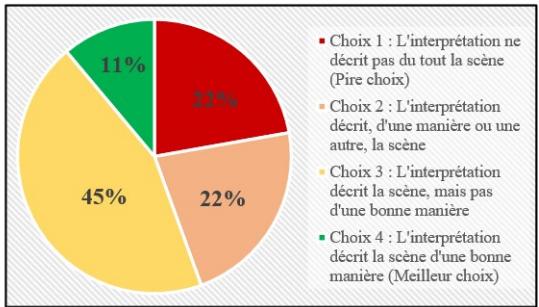


▪ Id de la scène (dans COCO dataset) : 116589

▪ Interprétation :

**In this scene that represents perhaps a private place there is a zebra on a zebra.**

Résultats obtenus



▪ Numéro du questionnaire : 4 (sur 5)

▪ Nombre de réponses : 9

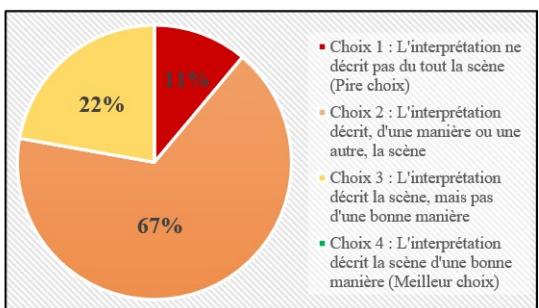


▪ Id de la scène (dans COCO dataset) : 124659

▪ Interprétation :

**In this livingroom there are books on other books and more than two wooden chairs.**

Résultats obtenus



▪ Numéro du questionnaire : 4 (sur 5)

▪ Nombre de réponses : 9

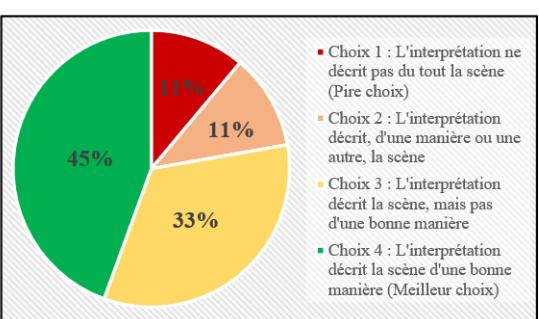


▪ Id de la scène (dans COCO dataset) : 125472

▪ Interprétation :

**In this scene that represents probably a public place there is a young person.**

Résultats obtenus



▪ Numéro du questionnaire : 4 (sur 5)

▪ Nombre de réponses : 9

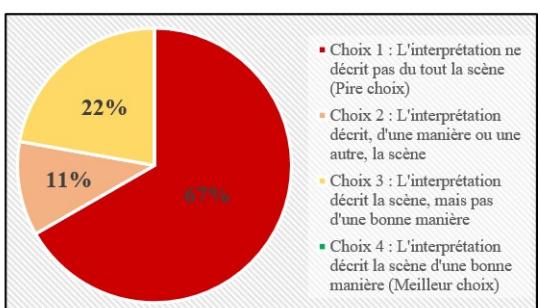


▪ Id de la scène (dans COCO dataset) : 140270

▪ Interprétation :

**In this scene that represents probably a public place there are a young person and a large horse.**

Résultats obtenus

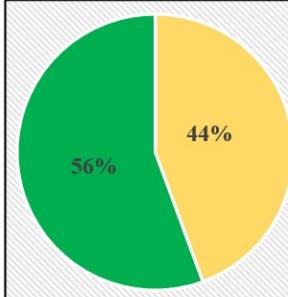


▪ Numéro du questionnaire : 4 (sur 5)

▪ Nombre de réponses : 9



Résultats obtenus →



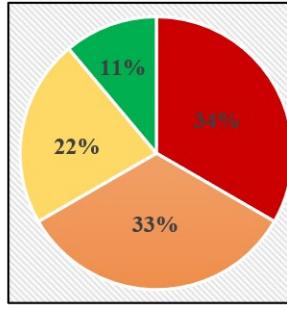
- Choix 1 : L'interprétation ne décrit pas du tout la scène (Pire choix)
- Choix 2 : L'interprétation décrit, d'une manière ou une autre, la scène
- Choix 3 : L'interprétation décrit la scène, mais pas d'une bonne manière
- Choix 4 : L'interprétation décrit la scène d'une bonne manière (Meilleur choix)

- Id de la scène (dans COCO dataset) : 167067
- Interprétation :

In this scene that represents probably a public place there is a young person.



Résultats obtenus →



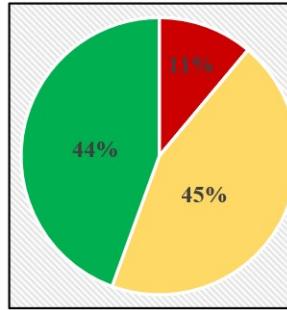
- Numéro du questionnaire : 4 (sur 5)
- Nombre de réponses : 9

- Id de la scène (dans COCO dataset) : 6012
- Interprétation :

In this scene that represents probably a private place there are two ripe bananas.



Résultats obtenus →



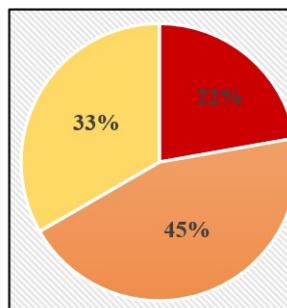
- Numéro du questionnaire : 5 (sur 5)
- Nombre de réponses : 9

- Id de la scène (dans COCO dataset) : 9772
- Interprétation :

In this private place there are two porcelain sinks and a young person.



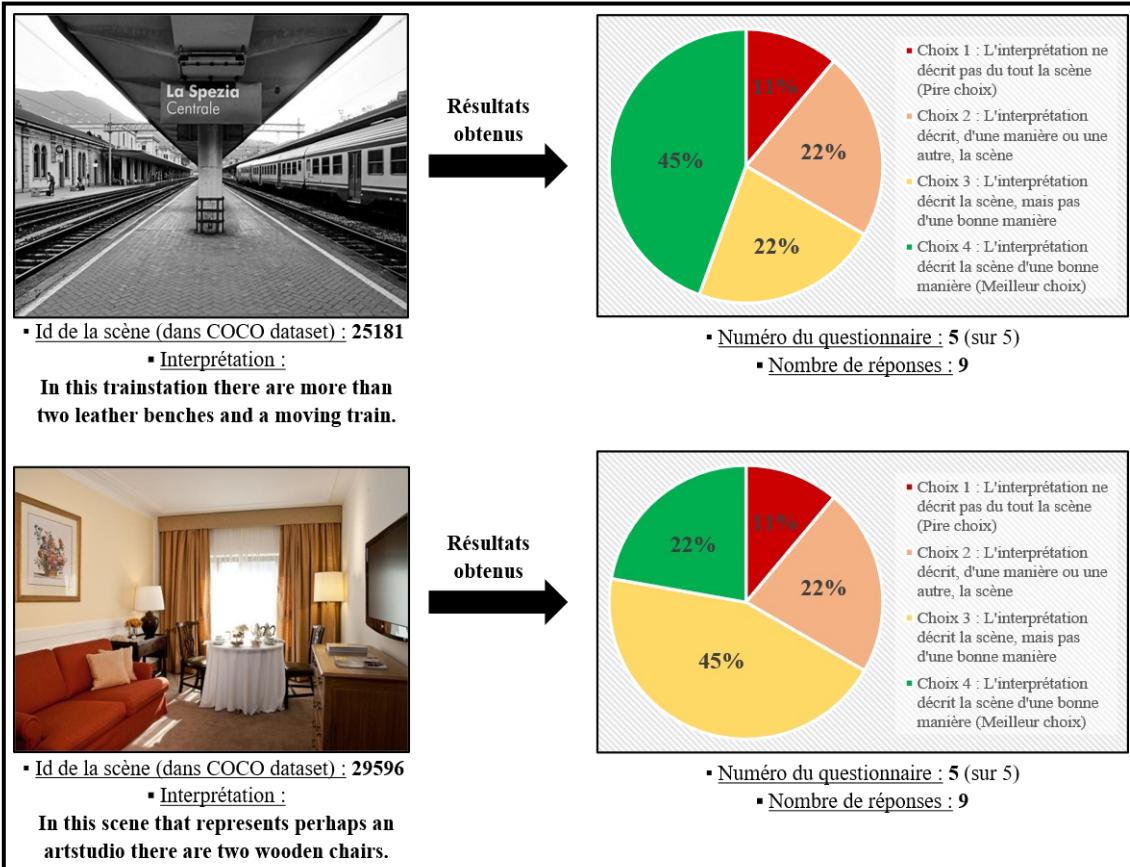
Résultats obtenus →



- Numéro du questionnaire : 5 (sur 5)
- Nombre de réponses : 9

- Id de la scène (dans COCO dataset) : 40471
- Interprétation :

In this private place there are more than two small cups and an electric oven.



**Figure B.1 : Résultats détaillés des questionnaires sur la qualité des interprétations générées**

# **Classification et interprétation de scènes basées sur les objets**

Présenté par :

**BENAMARA Soufian Przemyslaw et DERARDJA Mohamed Elamine**

Proposé et encadré par :

**Mme N. BAHA et Mr Lamine BENRAIS**

---

## **Résumé**

Le traitement des images est une importante branche de l'informatique qui permet de comprendre mieux le monde qui nous entoure grâce à ses diverses applications. Dans ce mémoire, nous nous intéressons à deux applications complémentaires : La classification et l'interprétation des scènes basées sur les objets.

Notre contribution dans la classification consiste en trois variantes : Classification en catégories standard (attribution d'une catégorie à une scène), enrichissement des scènes (ajout des objets manquants aux scènes, pour améliorer la classification) et la classification en catégories additionnelles (un ensemble de sous-catégories à valeurs restreintes, auxquelles une scène appartient).

L'interprétation des scènes est faite en exploitant les relations visuelles et en ajoutant les résultats de la classification afin de générer des descriptions textuelles.

Pour chaque approche développée, nous utilisons des techniques hybrides, entre la définition des règles et l'apprentissage automatique pour exploiter une variété de datasets.

**Mots-clés :** Traitement des images, classification, interprétation, catégories standard, enrichissement des scènes, catégories additionnelles, relations visuelles, description, définition des règles, apprentissage automatique, datasets.

---

## **Abstract**

Image processing is an important branch of computer science that allows us to better understand the world around us through its various applications. In this thesis, we focus on two complementary applications: Classification and interpretation of scenes based on objects.

Our contribution to the classification consists of three variants: Classification into standard categories (attribution of a category to a scene), scenes enrichment (addition of missing objects to scenes, to improve the classification) and classification into additional categories (a set of sub-categories with restricted values, to which a scene belongs).

Scene interpretation is done by exploiting visual relationships and adding classification results to generate textual descriptions.

For each developed approach, we use hybrid techniques, between rule definition and machine learning to exploit a variety of datasets.

**Keywords :** Image processing, classification, interpretation, standard categories, scenes enrichment, additional categories, visual relationships, description, rule definition, machine learning, datasets.

---