Techniques neurosymboliques probabilistes pour la classification supervisée informée par la logique

Arthur Ledaguenel[1,2], Celine Hudelot[2], Mostepha Khouadjia[1]

[1] IRT SystemX, Palaiseau
[2] CentraleSupélec, MICS

arthur.ledaguenel@irt-systemx.fr

Résumé

L'IA neurosymbolique est un champ de recherche émergent dont l'objectif est de combiner les capacités d'apprentissage des réseaux de neurones avec les aptitudes de raisonnement des systèmes symboliques. Ce papier présente un formalisme pour la classification supervisée informée par la logique, décrit succinctement les principales tâches et jeux de données traitées par la littérature, puis détaille un ensemble de techniques neurosymboliques basées sur le raisonnement probabiliste et analyse leur complexité asymptotique.

Mots-clés

Neurosymbolique, classification, logique.

Abstract

Neurosymbolic AI is a growing field of research aiming to combine neural networks learning capabilities with the reasoning abilities of symbolic systems. In this paper, we introduce a formalism for supervised multi-label classification informed by propositional background knowledge, desribe the main tasks and datasets tackled in the literature, then present a set of neurosymbolic techniques based on probabilistic logics and analyze their asymptotic complexity.

Keywords

Neurosymbolic, classification, logic.

1 Introduction

L'intelligence artificielle neurosymbolique est un champ de recherche émergent dont l'objectif est de combiner les capacités d'apprentissage des réseaux de neurones avec les aptitudes de raisonnement des systèmes symboliques. Cette hybridation peut prendre de nombreuses formes en fonction de la tâche traitée et des avantages ciblés [22, 44].
Un sous-domaine important de l'IA neurosymbolique est l'apprentissage machine informé (*informed machine learning*) [39], qui étudie comment exploiter de la connaissance *a priori* pour améliorer un système d'apprentissage. De nouveau, les techniques introduites dans la littérature peuvent être de natures très diverses selon le type de tâche (eg. régression, classification, détection, génération, etc.), le formalisme utilisé pour représenter la connaissance (eg. équations mathématiques, graphes de connaissances, logiques, etc.), l'étape à laquelle la connaissance est intégrée (eg. traitement des données, design de l'architecture du réseau de neurones, procédure d'apprentissage, procédure d'inférence, etc.) ou même les avantages attendus de l'hybridation (eg. explicabilité, performance, frugalité, etc.).
Dans ce papier, nous introduisons un formalisme pour la classification multi-classes supervisée informée par la logique.
Les contributions et le plan de l'article sont les suivants. Après quelques notions préliminaires sur la classification neuronale, la logique propositionnelle et le raisonnement probabiliste en Section 2, nous introduisons en Section 3 notre nouveau formalisme pour représenter une tâche de classification multi-classes supervisée informée par la logique. La connaissance *a priori* est exprimée par une formule propositionnelle qui décrit l'ensemble des combinaisons de classes sémantiquement *valides*. Nous illustrons ce formalisme par quelques exemples de tâches et jeux de données les plus fréquemment utilisés dans la littérature neurosymbolique. Puis nous nous appuyons sur ce formalisme dans la Section 4 pour reformuler les principales techniques neurosymboliques probabilistes existantes et analysons leurs principales propriétés dans la Section 5. Nous discutons dans la Section 6 des problèmes de complexité relatifs à ces techniques. Enfin, nous exposons les travaux connexes en Section 7 et concluons en Section 8 avec des pistes de recherche pour de futurs travaux.

2 Préliminaires

2.1 Classification neuronale

En apprentissage machine supervisé, l'objectif est d'apprendre une relation fonctionnelle $f : \mathcal{X} \mapsto \mathcal{Y}$ entre un **domaine d'entrée** $\mathcal{X}$ et un **domaine de sortie** $\mathcal{Y}$ à partir d'un jeu de données annoté $\mathsf{D} := (x^i, \mathbf{y}^i)_{1 \leq i \leq d} \in (\mathcal{X} \times \mathcal{Y})^d$. Les systèmes d'apprentissage profond sont habituellement décrits en deux modules : un réseau de neurones profond (*i.e.* un graphe computationnel paramétrique et différentiable) M est conçu pour émuler au mieux la fonction f et un module de coût différentiable L est utilisé pour mesurer la distance entre les prédictions et les annotations. Les poids du réseau sont alors optimisés en utilisant la descente de gra-

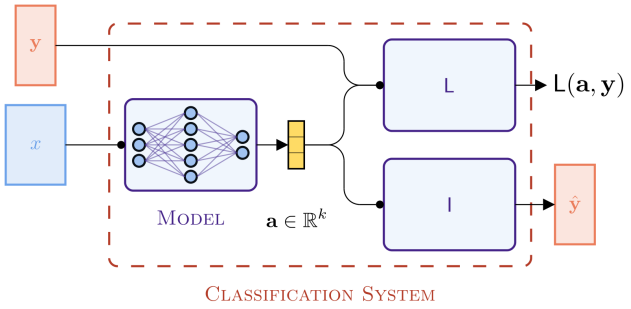

FIGURE 1 – Illustration d'un système neuronal de classification

dient afin de minimiser l'erreur empirique.
Cependant, cette description ne peut pas être appliquée telle qu'elle pour des tâches de classification. La classification multi-classes est un type de tâches d'apprentissage pour lesquelles les éléments de sortie sont des sous-ensemble d'un ensemble fini de classes $\mathbf{Y}$. On appelle un tel sous-ensemble un **état**, et le domaine de sortie (qui contient l'ensemble des états) est noté $\mathcal{Y} = \mathbb{B}^{\mathbf{Y}}$, avec $\mathbb{B} = \{0, 1\}$. Un état $\mathbf{y} \in \mathbb{B}^{\mathbf{Y}}$ peut également être vu comme une fonction qui associe à chaque variable une valeur dans $\mathbb{B}$ (*i.e.* une variable est associée à 1 par l'état si elle appartient au sous-ensemble décrit par l'état). Étant donné que l'espace de sortie est discret, un module de coût différentiable ne peut pas être défini sur $\mathcal{Y}^2$.
Ainsi, nous adoptons une description légèrement modifiée, dans laquelle un troisième module I, appelé le module d'inférence, doit être défini pour combler l'espace entre la nature continue du réseau de neurones (nécessaire à la descente de gradient) et la nature discrète du domaine de sortie. Ce troisième module, bien qu'essentiel, est rarement explicitement défini. Une illustration de cette description d'un système neuronal de classification peut être trouvée sur la Figure 1.

Définition 1. Un **système neuronal de classification** est constitué de :

- un module **paramétrique différentiable** (*i.e.* neuronal) M, appelé le **modèle**, qui prend en entrée une instance $x \in \mathcal{X}$, des paramètres $\theta \in \Theta$ et donne en sortie un vecteur de scores $\mathsf{M}(x, \theta) := \mathsf{M}_\theta(x) := \mathbf{a} \in \mathbb{R}^k$, appelé **scores pré-activation** ou **logits**.
- un module **non-paramétrique différentiable** L, appelé le module de **perte**, qui prend en entrée des logits $\mathbf{a} \in \mathbb{R}^k$ une annotation $\mathbf{y} \in \mathbb{B}^{\mathbf{Y}}$ et donne en sortie un scalaire positif $\mathsf{L}(\mathbf{a}, \mathbf{y})$.
- un module **non-paramétrique** I, appelé le module d'**inférence**, qui prend en entrée des logits $\mathbf{a} \in \mathbb{R}^k$ et donne en sortie une prédiction $\hat{\mathbf{y}} \in \mathbb{B}^{\mathbf{Y}}$.

Une approche classique pour définir un système neuronal de classification découle d'une interprétation probabiliste. Les logits produits par le réseau de neurones sont vus comme des paramètres d'une distribution conditionnelle de probabilité sur le domaine de sortie en fonction de l'entrée $\mathcal{P}(\cdot|\mathsf{M}_\theta(x))$, le module de perte calcule l'entropie croisée de cette distribution avec les annotations tandis que le module d'inférence donne la prédiction la plus probable étant donné la distribution apprise.
Quand aucune connaissance *a priori* n'est disponible (cas non-informé), il est classique de faire l'hypothèse que les variables de sorties sont indépendantes. On illustre ci-dessous comment cette hypothèse peut se traduire dans un système neuronal de classification.

Indépendante Pour la classification indépendante multi-classes (*cim*), on applique une couche *sigmoïde* par dessus le réseau pour transformer les logits en scores de probabilité. Le module de perte calcule l'entropie croisée entre ces scores et l'annotation, tandis que le module d'inférence prédit 1 pour toutes les variables dont la probabilité dépasse 0.5 et 0 pour les autres. Cela se traduit par les modules suivants :

$$\mathsf{L}_{cim}(\mathbf{a}, \mathbf{y}) := \mathtt{BCE}(\mathsf{s}(\mathbf{a}), \mathbf{y})$$
$$= -\sum_j y_j . \log(\mathsf{s}(a_j)) + (1 - y_j) . \log(1 - \mathsf{s}(a_j)) \quad (1)$$

$$\mathsf{I}_{cim}(\mathbf{a}) := \mathbb{1}[\mathbf{a} \geq 0] \quad (2)$$

où $\mathsf{s}(a_i) = \frac{e^{a_i}}{1+e^{a_i}}$ est la fonction sigmoïde, $\mathtt{BCE}$ est l'entropie croisée binaire et $\mathbb{1}[z] := \begin{cases} 1 & \text{si z vrai} \\ 0 & \text{sinon} \end{cases}$ est la fonction indicatrice.

2.2 Logique propositionnelle

Une signature propositionnelle est un ensemble $\mathbf{Y}$ de symboles appelés **variables** (*e.g.* $\mathbf{Y} = \{a, b\}$). Une **formule propositionnelle** est formée de manière inductive en combinant variables et autres formules par des connecteurs unaires ($\neg$, qui exprime la négation) et binaires ($\vee, \wedge$, qui expriment la disjonction et la conjonction respectivement). Par exemple, la formule $\kappa = a \wedge b$ exprime la conjonction des deux variables a et b. Un **état** de $\mathbf{Y}$ est une application $\mathbf{y} : \mathbf{Y} \mapsto \mathbb{B}$, où $\mathbb{B} := \{0, 1\}$. Un état $\mathbf{y}$ peut être prolongé en une **valuation** $\mathbf{y}^*$ sur l'ensemble des formules en suivant la sémantique usuelle (*e.g.* $\mathbf{y}^*(a \wedge b) = \mathbf{y}(a) \times \mathbf{y}(b)$). On dit qu'un état $\mathbf{y}$ **satisfait** une formule κ, noté $\mathbf{y} \models \kappa$, ssi $\mathbf{y}^*(\kappa) = 1$. Une formule est dite **satisfiable** s'il existe un état qui la satisfait. Deux formules sont dites équivalentes, noté $\kappa \equiv \gamma$, ssi elles sont satisfaites par les mêmes états. Dans la suite du papier et sauf mention contraire on supposera que l'ensemble de variables est fini et on notera $\mathbf{Y} = \{Y_j\}_{1 \leq j \leq k}$. On se réfère à [41] pour plus de détails sur la logique propositionnelle.

2.3 Raisonnement probabiliste

Un défi pour l'IA neurosymbolique est de combler l'écart entre la nature discrète de la logique et la nature continue des réseaux de neurones. Dans cette section, on définit les notion de distributions discrètes et de raisonnement probabiliste afin de fournir une interface entre ces deux univers.
Une **distribution de probabilité** sur un ensemble fini de variables $\mathbf{Y}$ est une application $\mathcal{P} : \mathbb{B}^{\mathbf{Y}} \mapsto \mathbb{R}^+$ qui associe un réel positif $\mathcal{P}(\mathbf{y})$ à chaque état $\mathbf{y} \in \mathbb{B}^{\mathbf{Y}}$, de sorte que la somme des valeurs donne $\sum_{\mathbf{y} \in \mathbb{B}^{\mathbf{Y}}} \mathcal{P}(\mathbf{y}) = 1$. Pour pouvoir

définir des opérations internes, comme la multiplication de deux distributions, nous étendons cette définition à des distributions non-normalisées $\mathcal{E} : \mathbb{B}^{\mathbf{Y}} \mapsto \mathbb{R}^+$. La distribution nulle associe tous les états à 0. La fonction de **partition** $\mathsf{Z} : \mathcal{E} \mapsto \sum_{\mathbf{y} \in \mathbb{B}^{\mathbf{Y}}} \mathcal{E}(\mathbf{y})$ donne la somme de la distribution sur tous ses états. On note $\overline{\mathcal{E}} := \frac{\mathcal{E}}{\mathsf{Z}(\mathcal{E})}$ la distribution **normalisée** (lorsque $\mathcal{E}$ est non nulle). Le **mode** d'une distribution $\mathcal{E}$ est son état le plus probable, *i.e.* $\operatorname{argmax}_{\mathbf{y} \in \mathbb{B}^{\mathbf{Y}}} \mathcal{E}(\mathbf{y})$.
Une distribution classique dans la littérature est la distribution exponentielle, qui est au coeur des modules de perte et d'inférence utilisés pour la classification indépendante multi-classes.

Définition 2. Étant donné un vecteur $\mathbf{a} \in \mathbb{R}^k$, on définit la **distribution exponentielle** comme :

$$\mathcal{E}(\cdot|\mathbf{a}) : \mathbf{y} \mapsto \prod_{1 \leq i \leq k} e^{a_i \cdot y_i}$$

On note également $\mathcal{P}(\cdot|\mathbf{a}) = \overline{\mathcal{E}(\cdot|\mathbf{a})}$ la distribution de probabilité correspondante.

Remarque 1. La distribution exponentielle est la distribution jointe de variables indépendantes de Bernouilli $\mathcal{B}(p_i)_{1 \leq i \leq k}$ avec $p_i = \mathsf{s}(a_i)$, où $\mathsf{s}(\mathbf{a}) = (\frac{e^{a_j}}{1+e^{a_j}})_{1 \leq j \leq k}$ est la fonction sigmoid.

Exemple 1. *La Table 1 représente la distribution exponentielle sur deux variables Y_1 et Y_2 paramétrée par le vecteur $\mathbf{a} := (2, -1)$. La fonction de partition est $\mathsf{Z}(\mathcal{E}(\cdot|\mathbf{a})) = 11.5$. Le mode de la distribution $\mathcal{P}(\mathbf{y}|\mathbf{a})$ est $\hat{\mathbf{y}} = (1, 0)$ avec une probabilité de 0.64.*

y_1	y_2	$\mathcal{E}(\mathbf{y}\|\mathbf{a})$	$\mathcal{P}(\mathbf{y}\|\mathbf{a})$
0	0	e^0	0.09
0	1	e^{-1}	0.03
1	0	e^2	**0.64**
1	1	e^1	0.24
		11.5	1

TABLE 1 – Représentation tabulaire d'une distribution.

Traditionnellement, lorsqu'une croyance (*belief*) a propos de variables aléatoires est exprimée par une distribution de probabilité et que de nouvelles informations sont collectées sous la forme d'observations (*evidence*), deux choses nous intéressent : calculer la probabilité de ces observations et mettre à jour nos croyances en utilisant la règle de Bayes, en conditionnant la distribution sur les observations. Le raisonnement probabiliste nous permet d'effectuer les mêmes opérations avec de la connaissance logique à la place d'observations. Prenons une distribution de probabilité $\mathcal{P}$ sur un ensemble de variables $\mathbf{Y} := \{Y_j\}_{1 \leq j \leq k}$ et une formule propositionnelle **satisfiable** κ sur ce même ensemble de variables. Notons $\mathbb{1}_\kappa$ la fonction indicatrice de κ qui associe 1 aux états qui satisfont κ et 0 aux autres :

$$\mathbb{1}_\kappa(\mathbf{y}) = \begin{cases} 1 & \text{si } \mathbf{y} \models \kappa \\ 0 & \text{sinon} \end{cases}$$

Définition 3. La **probabilité** de κ sous $\mathcal{P}$ est la somme des probabilités des états qui satisfont κ, *i.e.* :

$$\mathcal{P}(\kappa) := \mathsf{Z}(\mathcal{P} \cdot \mathbb{1}_\kappa) = \sum_{\mathbf{y} \in \mathbb{B}^{\mathbf{Y}}} \mathcal{P}(\mathbf{y}) \cdot \mathbb{1}_\kappa(\mathbf{y}) \tag{3}$$

La distribution $\mathcal{P}$ **conditionée par** κ, notée $\mathcal{P}(\cdot|\kappa)$, est :

$$\mathcal{P}(\cdot|\kappa) := \overline{\mathcal{P} \cdot \mathbb{1}_\kappa} \tag{4}$$

Remarque 2. Les deux définitions données ci-dessus sont sémantiques et non syntaxiques : elles s'appuient uniquement sur l'ensemble des états qui satisfont la formule et pas sur la syntaxe de la formule, ce qui signifie que deux formules équivalentes auront la même probabilité et la même distribution conditionnée.

Lorsque la distribution utilisée est une distribution de probabilité exponentielle $\mathcal{P}(\cdot|\mathbf{a})$, on note :

$$\mathcal{P}(\kappa|\mathbf{a}) := \mathsf{Z}(\mathcal{P}(\cdot|\mathbf{a}).\mathbb{1}_\kappa) \tag{5}$$

$$\mathcal{P}(\cdot|\mathbf{a}, \kappa) := \frac{\mathcal{P}(\cdot|\mathbf{a}).\mathbb{1}_\kappa}{\mathcal{P}(\kappa|\mathbf{a})} \tag{6}$$

Étant donné que la distribution $\mathcal{P}(\cdot|\mathbf{a})$ est strictement positive (pour tous $\mathbf{a}$), si κ est satisfiable, alors $\mathcal{P}(\kappa|\mathbf{a}) > 0$. Calculer $\mathcal{P}(\kappa|\mathbf{a})$ est un problème de comptage appelé **Probabilistic Query Estimation** (PQE). Calculer le mode de $\mathcal{P}(\cdot|\mathbf{a}, \kappa)$ est un problème d'optimisation appelé **Most Probable Explanation** (MPE). Résoudre ces deux problèmes se révélera au coeur de plusieurs techniques neurosymboliques que l'on introduira (voir Section 4).

Exemple 2. *La Table 2 reprend la distribution de l'Exemple 1 et illustre le raisonnement probabiliste sur la formule $\kappa = \neg Y_1 \vee Y_2$. La probabilité de κ est $\mathcal{P}(\kappa|\mathbf{a}) = 0.36$. Le mode de la distribution $\mathcal{P}(\cdot|\mathbf{a}, \kappa)$ est $\hat{\mathbf{y}} = (1, 1)$ avec une probabilité de 0.67.*

y_1	y_2	$\mathcal{P}(\mathbf{y}\|\mathbf{a})$	$\mathcal{P}(\mathbf{y}\|\mathbf{a}) \cdot \mathbb{1}_\kappa(\mathbf{y})$	$\mathcal{P}(\mathbf{y}\|\mathbf{a}, \kappa)$
0	0	0.09	0.09	0.24
0	1	0.03	0.03	0.09
1	0	0.64	0	0
1	1	0.24	0.24	**0.67**
		1	**0.36**	1

TABLE 2 – Représentation tabulaire d'une distribution.

3 Classification supervisée informée par la logique propositionnelle

On dit qu'une tâche de classification multi-classes supervisée est **informée** lorsque lui est attachée de la connaissance *a priori*, exprimée par une formule propositionnelle κ satisfiable, qui spécifie quels états du domaine de sortie $\mathcal{Y}$ sont **sémantiquement valides**.
Un jeu de données supervisé D est **cohérent** avec la formule κ si toutes les annotations la satisfont (*i.e.* $\forall 1 \leq i \leq$

$n, \mathbf{y}^i \models \kappa$). Dans ce papier, nous faisons l'hypothèse que les jeux de données d'entraînement et de test sont cohérents à la connaissance *a priori*. Cependant, certaines techniques permettent d'assouplir cette hypothèse et de travailler avec des jeux de données contenant des incohérences.

Les techniques évoquées dans ce papier ne s'intéressent pas à l'architecture du réseau en elle-même (eg. perceptron multi-couches, réseau convolutif, réseau récurent, transformer, etc.), qui dépend principalement de la structure du domaine d'entrée (eg. images, textes, etc.), mais se focalisent sur les deux autres modules afin d'intégrer notre connaissance *a priori* sur le domaine de sortie. On donne ci-dessous quelques exemples du type de structures qui peuvent être exprimées en logique propositionnelle, ainsi que de la manière dont on peut intégrer cette connaissance dans les modules de perte et d'inférence.

Catégorique Dans une tâche de classification **catégorique**, une et une seule variable est classifiée comme *vraie* dans chaque état valide. Cette contrainte peut être facilement exprimée en logique propositionnelle :

$$\kappa_{\odot_k} := \left(\bigvee_{1 \leq j \leq k} Y_j \right) \wedge \left(\bigwedge_{1 \leq j < l \leq k} (\neg Y_j \vee \neg Y_l) \right) \quad (7)$$

où la première partie impose qu'une variable soit classifiée comme *vraie* et la seconde partie empêche deux variables d'être *vraies* en même temps.

Pour la classification catégorique, la couche *sigmoid* est habituellement remplacée par une couche *softmax*, et la variable avec le score de probabilité maximum est prédite, ce qui donne les modules suivants :

$$\mathsf{L}_{\odot_k}(\mathbf{a}, \mathbf{y}) := \mathtt{CE}(\mathtt{s}(\mathbf{a}), \mathbf{y}) = -\log(\langle \sigma(\mathbf{a}), \odot_k(j) \rangle) \quad (8)$$

$$\mathsf{I}_{\odot_k}(\mathbf{a}) := \odot_k(\mathrm{argmax}(\mathbf{a})) \quad (9)$$

où $\mathtt{CE}$ est l'entropie croisée, $\sigma(\mathbf{a}) = (\frac{e^{a_j}}{\sum_l e^{a_l}})_{1 \leq j \leq k}$ et $\odot_k$ donne le *one-hot encoding* (en commençant à 1) de $j \in [\![1, k]\!]$, *e.g.* $\odot_4(2) = (0, 1, 0, 0)$.

Exemple 3 (MNIST)**.** *MNIST [28] est l'un des jeux de données les plus vieux de la vision par ordinateur et consiste en des petites images de chiffres manuscrits (e.g.* *ou* *). La tâche de classification catégorique a pour domaine d'entrée l'espace des images* 28×28 *en niveaux de gris, et une classe pour chaque chiffre. Comme décrit plus haut, la connaissance a priori sur la tâche est simplement exprimée par la formule* $\kappa_{\odot_{0:9}}$.

Ce type de tâches peut être élargi pour inclure les tâches **multi-catégoriques** pour lesquelles l'ensemble de variables peut être partitionné en plusieurs groupes de variables catégoriques. Le formule propositionnelle serait alors une conjonction de formule catégoriques sur des ensembles de variables disjoints.

Exemple 4 (Leptograpsus)**.** *Le jeu de données Leptograpsus [9] décrit* 5 *mesures morphologiques effectuées sur* 200 *crabes de couleurs et de sexes différents. L'objectif de la tâche de classification multi-catégorique associée est de prédire la couleur et le sexe d'un crabe à partir de ces mesures. Le domaine d'entrée est l'espace des mesures morphologiques* $\mathcal{X} = \mathbb{R}^5$ *et le domaine de sortie est l'ensemble des états des variables* $\mathbf{Y} = \{\mathsf{r}, \mathsf{b}, \mathsf{f}, \mathsf{m}\}$ *pour les classes* ***red****,* ***blue****,* ***female*** *et* ***male*** *respectivement. La connaissance a priori sur la tâche impose que chaque crabe soit d'une et une seule couleur, et d'un et un seul sexe, i.e. :*

$$\kappa := (\mathsf{r} \vee \mathsf{b}) \wedge (\neg \mathsf{r} \vee \neg \mathsf{b}) \wedge (\mathsf{f} \vee \mathsf{m}) \wedge (\neg \mathsf{f} \vee \neg \mathsf{m})$$

Hiérarchique La classification **hiérarchique** sur un ensemble de variables $\mathbf{Y}$ est usuellement représentée par un graphe dirigé acyclique $G = (\mathbf{Y}, E_h)$ où les noeuds sont les variables et les arrêtes E_h exprime l'inclusion d'une classe dans une autre (*e.g.* un chien est un animal). Lorsque le graphe est un arbre (ou une forêt) on parle de classification **taxonomique**. Ce formalisme peut également être enrichi par des arrêtes d'exclusion $H = (\mathbf{Y}, E_h, E_e)$ (*e.g.* une instance ne peut pas être classifiée comme chien et chat simultanément), comme dans les HEX-graphs [15]. Là encore, la traduction en logique propositionnelle vient naturellement :

$$\kappa_H := \left(\bigwedge_{(i,j) \in E_h} Y_i \vee \neg Y_j \right) \wedge \left(\bigwedge_{(i,j) \in E_e} (\neg Y_i \vee \neg Y_j) \right) \quad (10)$$

où la première partie s'assure qu'une classe fille ne peut être *vraie* que si sa classe mère l'est également et la seconde partie empêche deux classes mutuellement exclusives d'être *vraies* en même temps.

Dans le cas hiérarchique il n'y a pas de consensus dans la littérature sur les modules de pertes et d'inférence à utiliser : [36] définit un module de perte hiérarchique pour plus pénaliser les erreurs faîtes sur les plus hautes classes de la hiérarchies (en conservant le module d'inférence de *cim*), [21] affine les logits en fonction de la hiérarchie tandis que [15] conditionne la distribution exponentielle par la connaissance hiérarchique.

Exemple 5 (Cifar-100)**.** *Cifar-100 dataset [25] est un jeu de données composé de 60,000 images classifiées en 20 macro-classes (e.g. reptile), chacune divisée en 5 micro-classes (e.g. crocodile, dinosaur, lizard, turtle, et snake). Le domaine d'entrée est l'espace des images RGB de taille* 32×32*. Typiquement utilisé dans un cadre catégorique sur les 100 micro-classes, il peut également être utilisé dans le cadre d'une tâche de classification hiérarchique en incluant les macro-classes. Les relations hiérarchiques d'une classe macro vers ses classes micro, ainsi que les relations d'exclusion entre les macro-classes et entre les micro-classes peuvent être encodées dans un HEX-graphe* $H = (Y, E_h, E_e)$ *et donc exprimées par une formule propositionnelle* κ_H.

Exemple 6 (ImageNet)**.** *Le ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [40] est un challenge de classification d'images, qui a eu lieu annuellement entre 2010 et 2017, et s'est imposé comme un benchmark de référence en vision par ordinateur pour comparer les performances des systèmes neuronaux de classification. En août*

(a) MNIST-Sudoku [4]

(b) Warcraft Shortest Path [3]

FIGURE 2 – Exemples d'instances provenant de jeux de données structurés

2014, le jeu de données ImageNet contenait 14,197,122 images annotées classifiées selon 21,841 synsets de la hiérarchie WordNet [35]. Comme pour Cifar-100 (voire Exemple 5), le jeu de données est le plus souvent utilisé dans un cadre catégorique, mais peut être également utilisé dans un cadre hiérarchique en mobilisant les relations d'inclusion entre synsets définies dans WordNet.

Au delà de ces exemples, la logique propositionnelle peut être utilisée pour définir des domaines de sortie structurés très divers : addition de chiffres manuscrits (*i.e.* k-Add-MNIST) [24], solutions d'un Sudoku [4, 14], chemins dans un graphe dirigé [38, 45, 3], classements [45], matching parfait dans un graphe [38, 1], etc.

4 Techniques neurosymboliques probabilistes

L'objectif d'une technique neurosymbolique est de construire un système neuronal de classification automatiquement à partir de la connaissance *a priori* disponible sur le domaine de sortie, généralisant ainsi les travaux menés sur les cas particuliers de la classification multi-classes indépendante, catégorique et hiérarchique au cas général d'une formule propositionnelle.

Comme précisé plus haut, on ne s'intéresse qu'à la spécification des modules de perte et d'inférence. En particulier, nous étudierons un ensemble de techniques qui s'appuient sur le raisonnement probabiliste pour définir leurs modules. Nous utilisons (abusivement) l'appellation *techniques probabilistes* pour les décrire, même si ces techniques ne disposent pas nécessairement d'une interprétation probabiliste.

Régularisation sémantique Une première technique neurosymbolique utilise une approche multi-objectifs : un terme de régularisation mesurant la cohérence des sorties du modèle avec la connaissance *a priori* est ajouté à l'entropie croisée afin d'optimiser les paramètres sur un double objectif de performance et de cohérence. Il existe plusieurs manières de calculer ce terme de régularisation. Introduit dans un premier temps en utilisant de la logique floue [17, 23, 5], une version basée sur le raisonnement probabiliste est présentée par [45].

Définition 4. Un système neuronal performe la **régularisation sémantique** (*rs*) sur κ ssi ses modules de perte et d'inférence sont :

$$\mathsf{L}^{\lambda}_{rs}(\mathbf{a},\mathbf{y}) = \mathsf{L}_{cim}(\mathbf{a},\mathbf{y}) - \lambda.\log(\mathcal{P}(\kappa|\mathbf{a})) \quad (11)$$

$$\mathsf{I}_{cim}(\mathbf{a}) = \underset{\mathbf{y}\in\mathbb{B}^{\mathbf{Y}}}{\operatorname{argmax}}\mathcal{P}(\mathbf{y}|\mathbf{a}) \quad (12)$$

Conditionnement sémantique Suivant l'interprétation probabiliste introduite en Section 2.1, une manière naturelle d'intégrer de la connaissance *a priori* κ dans un système neuronal de classification est de conditionner la distribution $\mathcal{P}(\cdot|\mathsf{M}_\theta)$ par κ. Ce conditionnement affecte les modules de perte et d'inférence, qui reposent tous deux sur la distribution de probabilité paramétrée par la sortie du réseau de neurones, et conduit à la technique neurosymbolique suivante.

Définition 5. Un système neuronal performe le **conditionnement sémantique** (*cs*) sur κ si ses modules de perte et d'inférence sont :

$$\mathsf{L}_{|\kappa}(\mathbf{a},\mathbf{y}) = -\log(\mathcal{P}(\mathbf{y}|\mathbf{a},\kappa)) \quad (13)$$

$$\mathsf{I}_{|\kappa}(\mathbf{a}) = \underset{\mathbf{y}\in\mathbb{B}^{\mathbf{Y}}}{\operatorname{argmax}}\mathcal{P}(\mathbf{y}|\mathbf{a},\kappa) \quad (14)$$

Les modules de pertes et d'inférence présentés pour la classification indépendante et la classification catégorique sont des cas particuliers du conditionnement sémantique. De même, [15] définit le conditionnement sémantique pour les tâches de classification hiérarchiques. Cette technique est généralisée par [3] à un ensemble de distributions de probabilités définies sur des circuits arithmétiques.

Conditionnement sémantique à l'inférence Le conditionnement sémantique à l'inférence est dérivé du conditionnement sémantique, mais applique le conditionnement uniquement sur le module d'inférence (*i.e.* prédit la sortie la plus probable qui satisfait κ) et conserve le module de perte indépendant.

Définition 6. Un système neuronal performe le **conditionnement sémantique à l'inférence** (*csi*) sur κ ssi ses modules de perte et d'inférence sont :

$$\mathsf{L}_{cim}(\mathbf{a},\mathbf{y}) = -\log(\mathcal{P}(\mathbf{y}|\mathbf{a})) \quad (15)$$

$$\mathsf{I}_{|\kappa}(\mathbf{a}) = \underset{\mathbf{y}\in\mathbb{B}^{\mathbf{Y}}}{\operatorname{argmax}}\mathcal{P}(\mathbf{y}|\mathbf{a},\kappa) \quad (16)$$

En plus de conserver des propriétés clé du conditionnement sémantique (voir Section 5), le conditionnement sémantique à l'inférence se démarque des deux autres techniques en n'intégrant la connaissance que dans le module d'inférence, sans impact sur l'entraînement, ce qui présente deux avantages majeurs. Premièrement, tandis que *cs* et *rs* nécessitent de résoudre PQE pour calculer leur module de perte, *csi* ne nécessite que de résoudre MPE pour son module d'inférence, ce qui est plus facile pour certaines classes de formules (*i.e.* matchings parfaits). Deuxièmement, intégrer la connaissance uniquement à l'inférence offre plus de flexibilité. Par exemple, *csi* peut être utilisé dans le cas

où la connaissance n'est pas disponible pendant l'entraînement. C'est une propriété particulièrement importante à l'ère des **modèles sur étagère** et **modèles de fondations** [8], qui sont pré-entraînés sur des grands volumes de données généralistes avant d'être affinés et combinés sur une multitude de tâches hétérogènes, puisque la connaissance spécifique aux différentes tâches ne peut pas être intégrée pendant la majeure partie de l'entraînement.

5 Propriétés

Nous détaillons ci-dessous quelques propriétés théoriques des techniques présentées. Nous renvoyons le lecteur à [30] pour trouver les définitions et les preuves formelles des propriétés énoncées.

Insensibilité syntaxique Une technique neurosymbolique est **insensible à la syntaxe** (*invariant to syntax*) si deux formules équivalentes aboutissent à des modules de pertes et d'inférence identiques. Toutes les techniques qui s'appuient sur le raisonnement probabiliste sont insensibles à la syntaxe (voir Remarque 2), mais ce n'est pas le cas des techniques qui utilisent la logique floue [5].

Consistance Une technique neurosymbolique est consistante (*consistent*) si les prédictions du module d'inférence satisfont nécessairement la connaissance *a priori* κ. Le conditionnement sémantique ainsi que le conditionnement sémantique à l'inférence sont toutes deux consistantes. En revanche, les techniques neurosymboliques qui n'intègrent la connaissance que dans le module de perte (comme la régularisation sémantique) ne peuvent pas garantir la consistance.

Supérieur à l'inférence Un module d'inférence L_1 est dit supérieur à un module d'inférence L_2 ssi, quel que soit les scores d'entrée du module, si la prédiction de L_2 est consistante avec la connaissance a priori, alors la prédiction de L_1 est identique à celle de L_2. Cette propriété est intéressante car elle garantit, sous l'hypothèse de consistance du jeu de données, que la performance du module L_1 sera nécessairement meilleure (au sens large) que celle du module L_2 si on les évalue sur le même réseau de neurones (avec les mêmes poids entraînés). En particulier, on peut montrer que le module d'inférence conditionné $\mathsf{I}_{|\kappa}$ est supérieur au module d'inférence indépendant I_{cim}. Cette garantie théorique se traduit expérimentalement, comme montré dans [30].

6 Algorithmes et complexité

Après avoir introduit les principales techniques neurosymboliques probabilistes dans la section précédente, nous nous attaquons dans cette section à la question de leur implémentation et de leur complexité. Alors que de nombreux papiers pointent les problèmes de passage à l'échelle des ces techniques, leur complexité asymptotique n'est pas étudiée de manière systématique dans la littérature neurosymbolique. Ceci mène à plusieurs déboires : certaines techniques sont illustrées sur des tâches pour lesquelles le passage à l'échelle n'est pas possible, tandis que certaines tâches tractables sont considérées intractables.

On remarque premièrement que toutes les techniques mentionnées précédemment s'appuient sur la résolution de problèmes de PQE et MPE sur des distributions exponentielles. Ces problèmes sont malheureusement #P-complet et NP-complet (par réduction de #SAT et SAT respectivement) et sont donc **intractables** en général. Il devient alors naturel de se demander pour quelles familles de formules ces problèmes peuvent être résolus en temps polynomial. Nous appelons ces familles des **familles tractables** et étudions certains cas dans la Section 6.2.

Pour ce faire, nous présentons d'abord deux approches de résolution des problèmes de PQE et MPE, basées sur les modèles graphiques et la compilation de connaissance respectivement. On identifie ensuite quelques familles de formules tractables et intractables.

6.1 Algorithmes

Modèles graphiques Les modèles graphiques [27, 43] permettent de spécifier une famille de distributions de probabilité sur un ensemble de variables à travers une représentation graphique. Le graphe encode un ensemble de propriété (*e.g.* factorisation, indépendance, etc.) que les distributions qui appartiennent à la famille doivent respecter. Ces propriétés peuvent alors être exploitées afin de produire une représentation compressée des distributions et d'exécuter des algorithmes d'inférence efficaces [26]. Dans le contexte des logiques propositionnelles probabilistes, le graphe primaire d'une formule κ indique le modèle graphique auquel appartiennent les distributions exponentielles conditionnées par κ. En particulier, les algorithmes classiques de passage de messages sur un arbre de jonction peuvent être utilisés pour résoudre les problèmes de PQE et MPE sur ces distributions, avec une complexité en temps $\mathcal{O}(k2^{\tau(\kappa)})$ où k est le nombre de variables et $\tau(\kappa)$ est la largeur d'arbre du graphe primaire de κ.

Compilation de connaissances La compilation de connaissances [11] consiste à traduire une formule propositionnelle dans un langage de représentation qui permet d'effectuer certaines opérations de manière tractable. Les Sentential Decision Diagrams (SDD) [12] (voir Figure 3) est un langage de représentation qui permet la négation, la disjonction et la conjonction en temps polynomial (en la taille du circuit), ainsi que le calcul de PQE et MPE dans un temps linéaire (en la taille du circuit). De plus, il a été montré qu'une formule κ en forme normale conjonctive de largeur d'arbre $\tau(\kappa)$ peut être traduite dans un *trimmed and compressed* SDD de taille $\mathcal{O}(k2^{\tau(\kappa)})$ [12]. En raison de ces propriétés, les SDD sont devenus un langage standard de représentation des connaissances pour les systèmes neurosymboliques probabilistes [45, 3].

Programmation Linéaire Binaire Comme souligné dans [37], une formule propositionnelle en forme normale conjonctive peut être facilement compilée dans un programme linéaire équivalent (*i.e.* qui est satisfait par les mêmes états). Ainsi, la tâche de MPE sur cette formule peut alors être résolue en utilisant les nombreux algorithmes combinatoires qui ont été développés pour les problèmes de programmation linéaire binaire ou mixte [7].

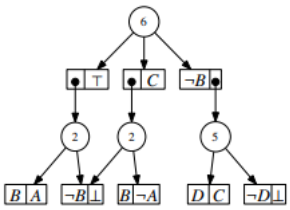

FIGURE 3 – Représentation graphique d'un SDD dans [12]

6.2 Familles tractables

Une familles de formules est dite **tractable** si et seulement si pour toute formule κ de la famille il est possible d'effectuer PQE et MPE sur n'importe quelle distribution exponentielle conditionnée par κ en un temps polynomial (en le nombre de variables de la formule).

Largeur d'arbre bornée Comme énoncé ci-dessus, une condition suffisante de tractabilité d'une famille de formules (et commune aux modèles graphiques et à la compilation de connaissances) est la possibilité de construire en temps polynomial (dans la taille de la signature) une décomposition en arbre de largeur bornée pour toutes les formules de la famille. Ceci implique directement que la famille des formules **taxonomiques** est tractable, puisque de largeur d'arbre 1. Le cas de k-Add-MNIST est intéressant à cet égard : longtemps jugé intractable par la communauté [24], il s'est avéré qu'une représentation légèrement différent du problème (incluant les retenues successives des additions de chiffres) aboutit à une famille de formules de largeur d'arbre bornée (avec une décomposition constructible en temps linéaire) [32].

Énumérable en temps polynomial Un autre type de famille tractable est celui des familles énumérables en temps polynomial, *i.e.* dont on peut énumérer les solutions de chaque formule dans un temps polynomial (en le nombre de variables de la formule). Il est facile de voir pourquoi MPE et PQE sont tous deux calculables en temps polynomial pour une famille énumérable : il suffit de lister chaque état valide et sa probabilité pour compter la probabilité de la formule et trouver son état le plus probable. Cette famille recouvre par exemple le cas des formules **catégoriques** et celui des formules **hiérarchiques** sur un **HEX-graphe en forme d'arbre** et **saturé**.

Cependant, ces deux conditions n'épuisent pas l'ensemble des familles tractable, comme le montre les exemples suivants.

Formules multi-catégoriques La famille des formules multi-catégoriques est un bon exemple des limites des critères de largeur d'arbre et d'énumérabilité pris séparément. En effet, il est facile de voir qu'il s'agit d'une famille de largeur d'arbre non bornée (la largeur d'arbre d'une formule catégorique est son nombre de variables) et non énumérable en temps polynomial (le nombre d'états valides est potentiellement exponentiel). Cependant, il est possible de décomposer chaque formule en composantes indépendantes et énumérables, ce qui rend la famille tractable.

Chemins simples La famille des formules qui encodent les chemins (simples) dans un graphe dirigé acyclique est elle aussi de largeur d'arbre non bornée et non énumérable. Cependant, en suivant un ordre topologique des arrêtes du graphe, il est possible de compiler cette formule en un OBDD [11] de taille polynomiale (dans le nombre d'arêtes) [29]. Cette propriété permet d'assurer la tractabilité de MPE et PQE pour cette famille. Il est de plus intéressant de remarquer que le calcul de MPE pour cette famille peut se ramener facilement à un calcul de plus court chemin, et donc être résolu avec des algorithmes combinatoires classiques (*e.g.* Bellman-Ford [6] ou Dijkstra [16]) : les probabilités sur les arrêtes sont transformés en scores réels par une sigmoïde inverse et multipliés par -1. Cette équivalence avec le problème de plus court chemin nous indique également qu'une famille de formules qui encodent les chemins simples dans un graphe dirigé (potentiellement cyclique) est **intractable**. En effet, le calcul de plus court chemin (avec poids négatifs) est un problème NP-complet, par réduction du problème d'existence d'un cycle Hamiltonien [19]. De plus, le calcul de PQE est #P-complet pour cette famille, par réduction au problème de comptage des chemins simples dans un graphe dirigé [42].

7 Travaux connexes

Logiques alternatives Il existe d'autres langages et sémantiques que la logique propositionnelle pour exprimer de la connaissance *a priori* sur une tâche de classification. Si certains correspondent à un fragment de la logique propositionnelle, comme les HEX-graphs dans [15], d'autres lui sont incommensurables, comme la programmation logique avec sémantique des modèles stables dans [46] ou la programmation par contraintes linéaires dans [37], voire passent à l'ordre supérieur, comme le langage de programmation logique Prolog [13] dans [33] ou la logique de premier ordre dans [5]. Les compromis pour ces différents langages sont principalement entre leur concision, leur expressivité et leur tractabilité. Les conséquences du choix de langage en termes de complexité de calculs ne sont pas encore bien comprises.

Logiques floues De nombreux travaux utilise les **logiques floues** [20, 23, 5] à la place du raisonnement probabiliste comme moyen de faire le pont entre la nature discrète de la connaissance et la nature continue du réseau de neurones.

Logiques pondérées Dans notre formalisme, nous avons fait l'hypothèse que la connaissance *a priori* est une contrainte dure, systématiquement satisfaite dans les jeux d'entraînement et de test. Certains formalismes permettent un langage plus souple qui permettent de représenter de l'incertitude sur la connaissance *a priori*. Des logiques pondérées permettent par exemple d'exprimer quelles formules ont *le plus de chances* d'être satisfaites et peuvent être intégrées dans des systèmes neurosymboliques [10, 34]. Les travaux de la *theory of evidence* [31] ou des *probability kinematics* [18] offrent des outils théoriques qui permettent de combiner de l'information provenant de deux sources incertaines (*e.g.* un réseau de neurones et de la connaissance

a priori probabiliste).

Apprentissage non supervisé De nombreux travaux explorent le potentiel de l'IA neurosymbolique dans un cadre d'apprentissage non *pleinement* supervisé (*i.e*. chaque instance du jeu de données est annotée sur l'ensemble des classes), comme l'apprentissage **faiblement supervisé** (*i.e*. certaines instances ne pas annotées sur toutes les classes) ou **semi-supervisé** (*i.e*. certaines instances ne sont pas annotées). Les techniques de régularisation [2, 45, 23, 5, 17] en particulier sont particulièrement indiquées pour l'apprentissage semi-supervisé et montrent que l'intégration de connaissances *a priori* dans le processus d'apprentissage peut grandement diminuer le besoin en instances annotées tout en augmentant les performances du système. Les techniques de conditionnement sémantique [3, 33, 46] permettent de traiter certaines variables comme des variables latentes en marginalisant la distribution apprise, et ne nécessitent donc pas de labels sur les variables latentes pendant l'apprentissage.

8 Conclusion

Après avoir présenté un formalisme pour la classification supervisée informée par la logique propositionnelle, ce papier réalise une revue de littérature des jeux de données structurés ainsi que des techniques neurosymboliques probabilistes. Enfin, nous détaillons quelques résultats relatifs à la complexité asymptotique des techniques probabilistes, qui manque d'une étude systématique dans la littérature.
Les pistes de recherche pour nos futurs travaux incluent, entre autres : une meilleure cartographie des familles tractables pour les différentes techniques probabilistes, l'utilisation de logiques alternatives, une étude des cadres d'apprentissages non-supervisés en IA neurosymbolique, ou encore l'apprentissage simultané du modèle neuronal et de la structure de la tâche.

Remerciements

Ce travail a été soutenu par le gouvernement français dans le cadre du programme "France 2O30" au sein de l'Institut de Recherche Technologique SystemX.

Références

[1] Kareem AHMED, Kai-Wei CHANG et Guy VAN DEN BROECK. « Semantic Strengthening of Neuro-Symbolic Learning ». In : *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*. Sous la dir. de Francisco RUIZ, Jennifer DY et Jan-Willem van de MEENT. T. 206. Proceedings of Machine Learning Research. PMLR, 25–27 Apr 2023, p. 10252-10261. URL : https://proceedings.mlr.press/v206/ahmed23a.html.

[2] Kareem AHMED et al. « Neuro-symbolic entropy regularization ». In : *Proceedings of the Thirty-Eighth Conference on Uncertainty in Artificial Intelligence*. Sous la dir. de James CUSSENS et Kun ZHANG. T. 180. Proceedings of Machine Learning Research. PMLR, jan. 2022, p. 43-53. URL : https://proceedings.mlr.press/v180/ahmed22a.html.

[3] Kareem AHMED et al. « Semantic Probabilistic Layers for Neuro-Symbolic Learning ». In : *Advances in Neural Information Processing Systems*. Sous la dir. de S KOYEJO et al. T. 35. Curran Associates, Inc., 2022, p. 29944-29959.

[4] Eriq AUGUSTINE et al. « Visual Sudoku Puzzle Classification : A Suite of Collective Neuro-Symbolic Tasks ». In : *International Workshop on Neural-Symbolic Learning and Reasoning*. 2022.

[5] Samy BADREDDINE et al. « Logic Tensor Networks ». In : *Artificial Intelligence* 303 (2022), p. 103649. ISSN : 0004-3702. DOI : https://doi.org/10.1016/j.artint.2021.103649. URL : https://www.sciencedirect.com/science/article/pii/S0004370221002009.

[6] Richard BELLMAN. « On a Routing Problem ». In : *Quarterly of Applied Mathematics* 16.1 (1958). Publisher : Brown University, p. 87-90. ISSN : 0033-569X. URL : https://www.jstor.org/stable/43634538 (visité le 23/04/2024).

[7] M. BENICHOU et al. « Experiments in mixed-integer linear programming ». In : *Mathematical Programming* 1.1 (1er déc. 1971), p. 76-94. ISSN : 1436-4646. DOI : 10.1007/BF01584074. URL : https://doi.org/10.1007/BF01584074 (visité le 02/05/2024).

[8] Rishi BOMMASANI et al. « On the Opportunities and Risks of Foundation Models ». In : *CoRR* abs/2108.07258 (2021). arXiv : 2108.07258. URL : https://arxiv.org/abs/2108.07258.

[9] N. A. CAMPBELL et R. J. MAHON. « A multivariate study of variation in two species of rock crab of the genus Leptograpsus ». In : *Australian Journal of Zoology* 22.3 (1974). Publisher : CSIRO PUBLISHING, p. 417-425. ISSN : 1446-5698. DOI : 10.1071/zo9740417. URL : https://www.publish.csiro.au/zo/zo9740417 (visité le 02/05/2024).

[10] Alessandro DANIELE et Luciano SERAFINI. « Knowledge Enhanced Neural Networks ». In : *PRICAI 2019 : Trends in Artificial Intelligence : 16th Pacific Rim International Conference on Artificial Intelligence, Cuvu, Yanuca Island, Fiji, August 26–30, 2019, Proceedings, Part I*. Berlin, Heidelberg : Springer-Verlag, 26 août 2019, p. 542-554. ISBN : 978-3-030-29907-1. DOI : 10.1007/978-3-030-29908-8_43. URL : https://doi.org/10.1007/978-3-030-29908-8_43 (visité le 19/04/2024).

[11] Adnan DARWICHE. *Modeling and Reasoning with Bayesian Networks*. Cambridge : Cambridge University Press, 2009. ISBN : 978-0-521-88438-9. DOI : 10.1017/CBO9780511811357. URL : https://www.cambridge.org/core/books/modeling-and-reasoning-with-bayesian-networks/8A3769B81540EA93B525C4C2700C9DE6 (visité le 07/08/2023).

[12] Adnan DARWICHE. « SDD : A New Canonical Representation of Propositional Knowledge Bases ». In : *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence*. 2011.

[13] Luc DE RAEDT, Angelika KIMMIG et Hannu TOIVONEN. « ProbLog : a probabilistic prolog and its application in link discovery ». In : *Proceedings of the 20th international joint conference on Artifical intelligence*. IJCAI'07. San Francisco, CA, USA : Morgan Kaufmann Publishers Inc., jan. 2007, p. 2468-2473. (Visité le 07/08/2023).

[14] Marianne DEFRESNE, Sophie BARBE et Thomas SCHIEX. « Scalable coupling of deep learning with logical reasoning ». In : *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*. IJCAI '23. <conf-loc>, <city>Macao</city>, <country>P.R.China</country>, </conf-loc>, 19 août 2023, p. 3615-3623. ISBN : 978-1-956792-03-4. DOI : 10.24963/ijcai.2023/402. URL : https://doi.org/10.24963/ijcai.2023/402 (visité le 23/02/2024).

[15] Jia DENG et al. « Large-Scale Object Classification Using Label Relation Graphs ». In : *Computer Vision – ECCV 2014*. Sous la dir. de David FLEET et al. Springer International Publishing, 2014, p. 48-64. ISBN : 978-3-319-10590-1.

[16] E. W. DIJKSTRA. « A note on two problems in connexion with graphs ». In : *Numerische Mathematik* 1.1 (1er déc. 1959), p. 269-271. ISSN : 0945-3245. DOI : 10.1007/BF01386390. URL : https://doi.org/10.1007/BF01386390 (visité le 21/02/2024).

[17] Michelangelo DILIGENTI, Marco GORI et Claudio SACCÀ. « Semantic-based regularization for learning and inference ». In : *Artificial Intelligence* 244 (mars 2017), p. 143-165. ISSN : 00043702. DOI : 10.1016/j.artint.2015.08.011.

[18] Zoltan DOMOTOR, Mario ZANOTTI et Henson GRAVES. « Probability Kinematics ». In : *Synthese* 44.3 (1980). Publisher : Springer, p. 421-442. ISSN : 0039-7857. URL : https://www.jstor.org/stable/20115538 (visité le 02/05/2024).

[19] Michael R. GAREY et David S. JOHNSON. *Computers and Intractability : A Guide to the Theory of NP-Completeness*. USA : W. H. Freeman & Co., 1979. 338 p. ISBN : 978-0-7167-1044-8.

[20] Francesco GIANNINI et al. « T-norms driven loss functions for machine learning ». In : *Applied Intelligence* 53.15 (fév. 2023), p. 18775-18789. ISSN : 0924-669X. DOI : 10.1007/s10489-022-04383-6. URL : https://doi.org/10.1007/s10489-022-04383-6 (visité le 07/08/2023).

[21] Eleonora GIUNCHIGLIA et Thomas LUKASIEWICZ. « Coherent Hierarchical Multi-Label Classification Networks ». In : *Advances in Neural Information Processing Systems*. T. 33. Curran Associates, Inc., 2020, p. 9662-9673. URL : https://proceedings.neurips.cc/paper/2020/hash/6dd4e10e3296fa63738371ec0d5df818-Abstract.html (visité le 13/09/2023).

[22] Henry A. KAUTZ. « The Third AI Summer : AAAI Robert S. Engelmore Memorial Lecture ». In : *AI Mag.* 43 (2022), p. 93-104.

[23] Emile van KRIEKEN, Erman ACAR et Frank van HARMELEN. « Analyzing Differentiable Fuzzy Logic Operators ». en. In : *Artificial Intelligence* 302 (jan. 2022), p. 103602. ISSN : 0004-3702. DOI : 10.1016/j.artint.2021.103602. URL : https://www.sciencedirect.com/science/article/pii/S0004370221001533 (visité le 07/08/2023).

[24] Emile van KRIEKEN et al. « A-NeSI : A Scalable Approximate Method for Probabilistic Neurosymbolic Inference ». In : *Advances in Neural Information Processing Systems* 36 (13 fév. 2024). URL : https://proceedings.neurips.cc/paper_files/paper/2023/hash/4d9944ab3330fe6af8efb9260aa9f307-Abstract-Conference.html (visité le 21/02/2024).

[25] Alex KRIZHEVSKY. *Learning Multiple Layers of Features from Tiny Images*. 2009.

[26] Frank R. KSCHISCHANG, Brendan J. FREY et Hans Andrea LOELIGER. « Factor graphs and the sum-product algorithm ». In : *IEEE Transactions on Information Theory* 47.2 (2001), p. 498-519. ISSN : 00189448. DOI : 10.1109/18.910572. (Visité le 28/03/2022).

[27] Steffen L. LAURITZEN. *Graphical Models*. Clarendon Press, 1996. ISBN : 978-0-19-852219-5.

[28] Yann LECUN et al. « Gradient-based learning applied to document recognition ». In : *Proceedings of the IEEE* 86 (11 1998), p. 2278-2323. ISSN : 00189219. DOI : 10.1109/5.726791.

[29] Arthur LEDAGUENEL, Céline HUDELOT et Mostepha KHOUADJIA. *Complexity of Probabilistic Reasoning for Neurosymbolic Classification Techniques*. 2024. arXiv : 2404.08404 [cs.AI].

[30] Arthur LEDAGUENEL, Céline HUDELOT et Mostepha KHOUADJIA. *Improving Neural-based Classification with Logical Background Knowledge*. 2024. arXiv : 2402.13019 [cs.AI].

[31] Jianbing MA et al. « Bridging jeffrey's rule, agm revision and dempster conditioning in the theory of evidence ». In : *International Journal on Artificial Intelligence Tools* 20.4 (août 2011). Publisher : World Scientific Publishing Co., p. 691-720. ISSN : 0218-2130. DOI : 10.1142/S0218213011000401. URL : https://www.worldscientific.com/doi/10.1142/S0218213011000401 (visité le 02/05/2024).

[32] Jaron MAENE et Luc DE RAEDT. « Soft-Unification in Deep Probabilistic Logic ». In : *Advances in Neural Information Processing Systems* 36 (15 déc. 2023). URL : https://papers.nips.cc/paper_files/paper/2023/hash/bf215fa7fe70a38c5e967e59c44a99d0-Abstract-Conference.html (visité le 21/02/2024).

[33] Robin MANHAEVE et al. « Neural probabilistic logic programming in DeepProbLog ». en. In : *Artificial Intelligence* 298 (sept. 2021), p. 103504. ISSN : 0004-3702. DOI : 10.1016/j.artint.2021.103504. URL : https://www.sciencedirect.com/science/article/pii/S0004370221000552 (visité le 07/08/2023).

[34] Giuseppe MARRA et Ondřej KUŽELKA. « Neural markov logic networks ». In : *Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence*. Sous la dir. de Cassio de CAMPOS et Marloes H. MAATHUIS. T. 161. Proceedings of Machine Learning Research. PMLR, 27–30 Jul 2021, p. 908-917. URL : https://proceedings.mlr.press/v161/marra21a.html.

[35] George A. MILLER. « WordNet ». In : *Communications of the ACM* 38 (11 nov. 1995), p. 39-41. ISSN : 15577317. DOI : 10.1145/219717.219748. URL : https://dl.acm.org/doi/10.1145/219717.219748.

[36] Bruce R. MULLER et W. SMITH. « A Hierarchical Loss for Semantic Segmentation ». In : *VISIGRAPP*. 2020. URL : https://api.semanticscholar.org/CorpusID:215791996.

[37] Mathias NIEPERT, Pasquale MINERVINI et Luca FRANCESCHI. « Implicit MLE : Backpropagating Through Discrete Exponential Family Distributions ». In : *Advances in Neural Information Processing Systems*. T. 34. Curran Associates, Inc., 2021, p. 14567-14579. URL : https://proceedings.neurips.cc/paper_files/paper/2021/hash/7a430339c10c642c4b2251756fd1b484-Abstract.html (visité le 17/01/2024).

[38] Marin Vlastelica POGANČIĆ et al. « Differentiation of Blackbox Combinatorial Solvers ». en. In : sept. 2019. URL : https://openreview.net/forum?id=BkevoJSYPB (visité le 27/10/2023).

[39] Laura von RUEDEN et al. « Informed Machine Learning – A Taxonomy and Survey of Integrating Prior Knowledge into Learning Systems ». In : *IEEE Transactions on Knowledge and Data Engineering* 35.1 (jan. 2023). Conference Name : IEEE Transactions on Knowledge and Data Engineering, p. 614-633. ISSN : 1558-2191. DOI : 10.1109/TKDE.2021.3079836.

[40] Olga RUSSAKOVSKY et al. « ImageNet Large Scale Visual Recognition Challenge ». In : *International Journal of Computer Vision* 115 (3 déc. 2015), p. 211-252. ISSN : 15731405. DOI : 10.1007/s11263-015-0816-y.

[41] Stuart RUSSELL et Peter NORVIG. *Artificial Intelligence A Modern Approach (4th Edition)*. Pearson Higher Ed, 2021. Chap. 7, p. 208-250.

[42] Leslie G. VALIANT. « The Complexity of Enumeration and Reliability Problems ». In : *SIAM Journal on Computing* 8.3 (1er août 1979), p. 410-421. ISSN : 0097-5397. DOI : 10.1137/0208032. URL : https://doi.org/10.1137/0208032 (visité le 21/02/2024).

[43] M J WAINWRIGHT et al. « Graphical Models, Exponential Families, and Variational Inference ». In : *Foundations and Trends R in Machine Learning* 1.2 (2008), p. 1-305. DOI : 10.1561/2200000001. (Visité le 07/04/2022).

[44] Wenguan WANG, Yi YANG et Fei WU. *Towards Data-and Knowledge-Driven Artificial Intelligence : A Survey on Neuro-Symbolic Computing*. 2023. arXiv : 2210.15889 [cs.AI]. URL : https://arxiv.org/abs/2210.15889.

[45] Jingyi XU et al. « A Semantic Loss Function for Deep Learning with Symbolic Knowledge ». In : *35th International Conference on Machine Learning, ICML 2018*. T. 12. International Machine Learning Society (IMLS), 2018, p. 8752-8760. ISBN : 9781510867963.

[46] Zhun YANG, Adam ISHAY et Joohyung LEE. « NeurASP : Embracing Neural Networks into Answer Set Programming ». In : *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*. California : International Joint Conferences on Artificial Intelligence Organization, juill. 2020, p. 1755-1762. ISBN : 978-0-9992411-6-5. DOI : 10.24963/ijcai.2020/243.