Vers des classifieurs ontologiquement explicables

Grégory Bourguin, Arnaud Lewandowski, Mourad Bouneffa, Adeel Ahmad

Vers des Classifieurs Ontologiquement Explicables

G. Bourguin, A. Lewandowski, M. Bouneffa, A. Ahmad

Université du Littoral Côte d'Opale, LISIC

{gregory.bourguin, arnaud.lewandowski, mourad.bouneffa, adeel.ahmad}@univ-littoral.fr

Résumé

Répondant au besoin d'explicabilité des IA qui utilisent l'Apprentissage Profond (AP), ce papier explore les apports et la faisabilité d'un processus de création de classifieurs explicables basés sur des ontologies. La démarche est illustrée par l'utilisation de l'ontologie des Pizzas pour créer un classifieur d'images qui fournit des explications visuelles impliquant une sélection de features ontologiques. Nous proposons une implémentation en complétant un modèle d'AP avec des tenseurs ontologiques générés à partir de l'ontologie exprimée avec la Logique de Description.

Mots-clés

Apprentissage automatique, ontologie, explicabilité, classifieur.

Abstract

In order to meet the explainability requirement of AI using Deep Learning (DL), this paper explores the contributions and feasibility of a process designed to create ontologically explainable classifiers while using domain ontologies. The approach is illustrated with the help of the Pizzas ontology that is used to create an image classifier that is able to provide visual explanations concerning a selection of ontological features. The approach is implemented by completing a DL model with ontological tensors that are generated from the ontology expressed in Description Logic.

Keywords

Machine learning, ontology, explainability, classifier

1 Introduction

Ces dernières années ont été marquées par une large démocratisation de solutions basées sur l'Apprentissage Automatique (AA), en particulier l'Apprentissage Profond (AP). Si la prolifération de ces nouveaux outils destinés à supporter des utilisateurs dans des tâches très diverses a démontré leur grande utilité, elle s'est aussi accompagnée de questionnements concernant la confiance que l'on peut leur accorder. De nombreux papiers de recherche ont ainsi souligné le problème de l'opacité des algorithmes d'AA, et le besoin prégnant envers de nouvelles solutions permettant d'entrouvrir ces boites noires pour faciliter leur acceptation. Cette problématique est au cœur du mouvement XAI [2] (eXplainable AI) et d'un grand nombre de travaux de recherches tournés vers l'explicabilité des IA (cf. partie 2).

Comme le souligne [12], expliquer clairement à un utilisateur final le rationnel ayant mené à une décision peut être aussi important que la décision elle-même. Pour ce faire, il est nécessaire que la solution proposée soit non seulement capable de fournir des explications quant à ses décisions, mais aussi que ces explications soient compréhensibles par l'utilisateur, c'est- à-dire qu'elles soient en adéquation avec son niveau d'abstraction. Le niveau d'abstraction auquel les explications doivent être proposées dépend donc des connaissances de l'utilisateur, de son expertise, voire de son point de vue. Le défi pour les concepteurs est alors de créer des IA explicables capables de combler l'écart sémantique entre les entités manipulées par les algorithmes, et celles permettant d'expliquer leurs décisions.

Issues du domaine de l'Ingénierie des Connaissances, les ontologies ont pour but de capturer les connaissances liées aux domaines d'expertises des utilisateurs, et de permettre aux algorithmes de les manipuler. Les ontologies sont d'ores et déjà utilisées par les chercheurs en AA dans le but d'augmenter les jeux de données utilisés pour entrainer les modèles : sélection de données basées sur leurs propriétés ontologiques, ajouts d'étiquettes déduites de l'ontologie. Cependant, l'utilisation des ontologies qui nous intéresse dans ce papier est celle qui implique des moteurs d'inférence dans le calcul du résultat : les chercheurs utilisent les algorithmes de l'AA pour identifier des entités ontologiques de « bas niveau » (ex. la présence d'objets dans une image), puis injectent ces informations dans un moteur d'inférence afin d'effectuer des déductions à haut niveau d'abstraction (ex. classification de l'activité humaine) [3]. Ce type de processus est particulièrement intéressant du point de vue de l'explicabilité du fait que l'inférence ontologique est un processus déductif qui peut être expliqué. Toutefois, la littérature souligne que l'inférence ontologique coûte cher [8], et ce type de processus n'est généralement mis en œuvre que lorsque la tâche de classification est trop complexe pour les algorithmes d'AA classiques.

Prenant acte du besoin d'explicabilité, l'objectif de ce papier est d'explorer un processus de création de classifieurs automatiques capables de fournir des explications fondées sur une ontologie. Nous ne focalisons pas ici sur les moyens permettant d'améliorer la classification, mais sur les apports d'une ontologie pour l'explicabilité. Nous explorons aussi la faisabilité d'une telle approche en utilisant des outils classiques de l'AP, et proposons une solution permettant de compléter un modèle d'AP avec des tenseurs (au sens

de Tensorflow [1]) générés à partir d'assertions exprimées en Logique de Description (DL).
Nous avons choisi d'exemplifier notre démarche en réutilisant la fameuse ontologie des Pizzas proposée par l'Université de Manchester. Notre objectif n'étant pas la performance de classification, le but de notre classifieur est d'étiqueter des images synthétiques représentant des pizzas avec les classes définies dans l'ontologie. Du point de vue de l'explicabilité, il s'agit de générer des heatmaps différenciant les garnitures (toppings) qui correspondent aux définitions ontologiques des classes.
La 2ème partie de ce papier propose un état de l'art du besoin et des solutions pour l'explicabilité des IA, ceci en focalisant sur les approches qui veulent y intégrer des connaissances grâce aux ontologies. La 3ème partie illustre les apports d'une approche fondée sur un raisonnement ontologique pour l'explicabilité, et introduit la démarche générique que nous proposons. La 4ème partie en présente l'application via l'implémentation d'un classifieur d'images ontologiquement explicable. La 5ème partie présente des réflexions qui découlent de cette expérience, avant de conclure dans la partie 6.

2 Explicabilité

Il est un consensus établi sur l'importance que revêt l'utilisation d'IA dotées de capacités d'apprentissage, de raisonnement et d'adaptation pour l'accomplissement de tâches informatiques de plus en plus complexes indispensables au développement des activités humaines [26]. Les systèmes basés sur l'AP sont de plus en plus performants, mais deviennent en corollaire de plus en plus complexes et opaques. Ils apparaissent comme des boites noires [9], rendant très problématique l'intervention humaine pour la compréhension de leurs décisions, ainsi que pour le contrôle de leur exécution, de leur déploiement, et de leur évolution [2][13]. En conséquence, le besoin de transparence et surtout d'explicabilité des IA s'est révélé crucial avec des aspects liés à la confiance qu'un utilisateur peut leur accorder en matière de sureté de fonctionnement de systèmes critiques pilotés par l'IA, mais également en matière d'éthique et du respect de règles légales et sociales telles que la non ségrégation et le respect de la vie privée [11].
Dans ce papier, nous nous intéressons à l'explicabilité de l'IA en tant que justifications des décisions compréhensibles par les utilisateurs. La nécessité de fournir des explications est un besoin ancien étant apparu dès les premières implémentations de systèmes experts [25]. Les systèmes à base de règles, ainsi que les algorithmes d'AA réputés plus transparents, comme la régression linéaire et les arbres de décision, ont également besoin d'outils simplifiant, résumant, et expliquant leurs prédictions. Cela peut se traduire par une restitution de la trace d'exécution des règles appliquées pour aboutir à une décision, ou encore par la simplification d'un arbre de décision en remplaçant des nœuds et arcs de niveaux de granularité fine par des concepts plus généraux issus de la terminologie du domaine de l'utilisateur [2]. Cependant l'application de ces techniques s'avère très difficile, voire impossible, en ce qui concerne l'explicabilité des systèmes à base d'AP.

1. https ://www.tensorflow.org/guide/tensor

2.1 Les méthodes *Post hoc*

Les outils pour l'explicabilité des systèmes d'AP mettent majoritairement en œuvre des approches dites *post hoc* permettant de fournir des explications sur des modèles préexistants. La plupart de ces méthodes sont aussi appelées agnostiques du fait qu'elles sont applicables à tout algorithme d'AP.
Les techniques d'explication utilisent en majorité les facteurs d'importance des features d'entrée et reposent sur l'idée d'associer à chacune une valeur traduisant l'importance de son rôle dans la prédiction. Il est ainsi possible d'obtenir des explications sur une prédiction particulière, ou des explications plus globales exprimées par différents graphiques associant feature, facteur d'importance, et prédictions.
Un des domaines où ce type de travaux est le mieux représenté est celui de la vision par ordinateur (CV, Computer Vision) basée sur les réseaux de neurones convolutionnels (CNN). En CV, les features correspondent aux pixels d'une image fournie en entrée. Les propositions consistent à mettre en correspondance les prédictions et les pixels qui ont conduit à une classification [27]. Pour ce faire, diverses approches ont été adoptées avec en particulier les travaux consistant à explorer l'architecture des réseaux pour déterminer comment les couches intermédiaires perçoivent le monde extérieur [23]. Une des méthodes les plus représentatives de ces travaux est la méthode Grad-CAM (Gradient-Weighted Class Activation Mapping) [28] (et ses dérivées) qui utilise le gradient d'un concept cible pour produire des heatmaps identifiant les régions de l'image qui ont le plus participé à sa reconnaissance.
Les techniques utilisant une identification des features d'entrée pour expliquer un modèle ne sont pas limitées au domaine de la CV. Ainsi, des outils tels que LIME (Local Interpretable Model-Agnostic Explanations) [19] permettent aussi bien d'identifier des pixels dans une image pour un problème de CV, que d'identifier les termes participant à une prédiction dans un modèle de NLP (Natural Language Processing). La technique utilisée est l'explication par simplification qui consiste à construire des modèles linéaires sur des sous-parties du système global : il s'agit de produire des explications à partir de perturbations locales en simulant le fonctionnement du modèle boîte noire par un modèle naturellement transparent. Si dans le cas de LIME, les perturbations sont générées par le système, d'autres systèmes permettent d'explorer les prédictions à la suite de changements de valeurs de certaines features dans un processus interactif d'explication et par une analyse de type What-If [17], fournissant ainsi une sorte d'analyse contrefactuelle. Enfin, on peut aussi citer l'outil SHAP [15] (SHapely Additive exPlanation) qui s'inspire de la théorie des jeux, ou plus particulièrement de celle qui consiste à trouver la manière la plus équitable de distribuer les gains aux joueurs en

se basant sur leur taux de contribution lors de la partie. Les joueurs représentent ici les features du modèle. Une revue systématique de ce type d'approche est effectuée dans [30]. Les techniques et outils évoquées ci-avant se sont avérés très utiles dans de nombreux travaux pour fournir des explications sur le fonctionnement de modèles d'IA. Cependant, comme le souligne [14], et comme nous le verrons dans la partie 3.2, ces approches ne garantissent aucunement que les explications fournies soient compréhensibles par les utilisateurs.

2.2 Explicabilité & Ontologies

Un des objectifs principaux de l'explicabilité est de fournir aux utilisateurs, souvent des spécialistes de domaines, une description compréhensible de la manière dont le système a produit une prédiction, ou des facteurs clés qui ont conduit à cette prédiction.

L'apport des ontologies dans l'explicitation et l'axiomatisation de la sémantique d'un domaine n'est plus à démontrer. Pour de nombreux auteurs, il est apparu évident que les ontologies peuvent servir à fournir des explications adéquates et cohérentes aux conclusions d'un modèle d'AP [5]. Par exemple, dans [22], les auteurs opèrent une méthode *post-hoc* de mise en correspondance entre l'entrée d'un réseau de neurones et les classes d'une ontologie suggérée, et génèrent automatiquement des règles en Logique de Description (DL) à partir des instances classifiées pour obtenir des expressions qui opèrent comme des explications.

Dans [1] les auteurs proposent une architecture de réseaux de neurones dans laquelle des couches dites sémantiques sont introduites pour produire des explications. Ce type d'approche est formalisée dans [14] et développée plus avant dans [16] sous le concept de *semantic bottleneck* : il s'agit de construire un classifieur qui intègre dès sa conception des couches sémantiques spécifiques qui permettent au module d'AP d'extraire des features sémantiques qui sont elles-mêmes utilisées pour calculer la classification finale. La contribution pondérée des features sémantiques permet de fournir des explications quant à la prédiction, et aide à comprendre les erreurs de classification. Il faut toutefois noter que même si ces travaux parlent de sémantique, ils ne mettent pas en œuvre une approche ontologique.

Enfin, nous souhaitons citer les travaux tels [3][6] qui s'intéressent à l'interprétation d'images fondée sur des ontologies. Dans ces travaux, un processus d'AP comme la détection d'objets ou la segmentation sémantique est utilisé pour identifier des features qui correspondent aux concepts de l'ontologie, et qui servent ensuite à inférer des déductions de plus haut niveau d'abstraction. Ces travaux impliquent un raisonnement à base de DL dans le but de rendre possible ou d'améliorer des tâches de classification complexes. Même s'ils ne focalisent pas explicitement sur la problématique d'explicabilité, une classification basée sur la DL est intrinsèquement explicable, et ces solutions peuvent de fait fournir des explications au niveau d'abstraction de l'ontologie impliquée.

2.3 Positionnement

Notre but est de fournir des explications tout en mettant en exergue, dans les données d'entrée, les features qui ont participé aux prédictions. De ce point de vue, nous nous inspirons des outils tels que Grad-CAM ou LIME. Cependant, à la différence de ces outils, notre approche n'est pas agnostique : même si notre démarche se veut générique, les explications que nous voulons fournir sont intimement liées au domaine visé et les features que nous voulons mettre en exergue se doivent d'être au niveau d'abstraction des utilisateurs, c.à.d, de notre point de vue, des features ontologiques.

L'approche que nous développons n'est pas non plus *post-hoc* puisque nous verrons que l'ontologie est ici directement impliquée dans le processus même de création du classifieur. Elle est similaire à celles utilisant des *semantic bottlenecks*, à la différence près que la sémantique est ici fournie par une ontologie qui, de plus, sert directement au calcul de la prédiction.

De ce point de vue, nous sommes fortement inspirés par les travaux impliquant ontologies et DL pour l'interprétation d'images à haut niveau d'abstraction. Notre démarche est cependant aussi quelque peu différente en focalisant résolument sur le problème d'explicabilité, et en proposant des classifieurs explicables qui impliquent des ontologies y compris dans des tâches de classification qui n'en auraient *a priori* pas besoin. Enfin, l'interprétation à haut niveau d'abstraction peut impliquer des moteurs d'inférence externes qui s'intègrent difficilement dans un pipeline classique d'AA : comme le souligne [8], l'inférence ontologique est un mécanisme coûteux. C'est pourquoi nous proposons une mise en œuvre qui n'utilise pas de moteur d'inférence ontologique « classique » comme Jena, Hermit ou Pellet, mais qui repose sur des modules de raisonnement spécifiques que nous générons automatiquement à partir des définitions de l'ontologie, et qui utilisent les mêmes technologies d'implémentation que les modèles d'AP (cf. 4.2).

3 Explicabilité Ontologique

3.1 Domaine d'illustration : les pizzas

Pour illustrer notre démarche, nous avons choisi de réutiliser l'ontologie des Pizzas (Université de Manchester). Les raisons sont multiples, mais la principale est le fait que cette ontologie est accessible et possède une très grande notoriété.

L'ontologie des pizzas définit un ensemble de classes de pizzas (ex. *Napoletana*), sous-classes de la classe *NamedPizza* (elle-même sous-classe de *Pizza*). Les définitions utilisent principalement la propriété d'objet *hasTopping* dont le domaine est la classe *Pizza*, et l'image est la classe *PizzaTopping* qui est la superclasse de garnitures telles que les anchois (*AnchoviesTopping*), etc.

L'ontologie définit 22 sous-classes de *NamedPizza* à partir de 36 sous-classes de *PizzaTopping*. Pour simplifier la construction du jeu de données (les images de pizzas), nous avons choisi de focaliser sur 14 sous-classes de *NamedPizza* en impliquant 16 sous-classes de *PizzaTopping*. On

trouvera ainsi par exemple la *Napoletana* définie par :

$$\begin{aligned} Napoletana \equiv & Pizza \\ & \sqcap (\exists\, hasTopping.AnchoviesTopping) \\ & \sqcap (\exists\, hasTopping.OliveTopping) \\ & \sqcap (\forall\, hasTopping \\ & .(AnchoviesTopping \sqcup OliveTopping)) \end{aligned} \quad (1)$$

Notre objectif étant de mettre en relation les résultats d'un classifieur d'images avec les définitions de l'ontologie, nous avons constitué un jeu de données dont les exemples sont étiquetés avec les sous-classes de *NamedPizza*, et dont les images correspondent à l'ontologie, c'est-à-dire que les toppings apparaissant dans l'image étiquetée correspondent à la définition ontologique de l'étiquette.
D'autres chercheurs ont déjà constitué des jeux de données contenant des images de pizzas [18] : aucun ne correspond aux définitions fournies par l'ontologie. De plus, notre but dans ces expérimentations n'est pas d'optimiser la classification (en termes de précision, etc.), mais d'étudier les apports et la faisabilité d'une démarche impliquant une ontologie pour améliorer l'explicabilité d'un classifieur. Inspirés par les travaux de [18] qui génèrent des images de pizzas synthétiques pour obtenir un jeu de données contrôlé, nous avons mis en œuvre une méthode similaire et créé un module Pizzaïolo qui génère des images synthétiques de pizzas à partir de l'ontologie en combinant des cliparts de toppings (cf. Figure 1). Les images utilisent volontairement la même base (seule la répartition des toppings varie en nombre, position et orientation) de manière à forcer un quelconque classifieur (non ontologique) à focaliser sur les toppings pour différencier les pizzas.

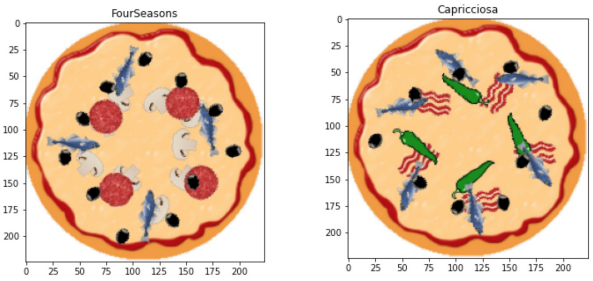

FIGURE 1 – Pizzas synthétiques ontologiques.

La tâche de classification de ces images étant assez simple, nous n'avons généré qu'un «petit» jeu de données totalement équilibré de 200 pizzas par classe.

3.2 Problèmes d'approche non ontologique

Pour illustrer les problèmes liés aux outils pour l'explicabilité, nous avons construit et entrainé un classifieur « classique » basé sur un CNN dont la base est formée par une architecture VGG19 [24] pré-entrainé sur Imagenet [20], à laquelle nous avons simplement ajouté une couche Dense (256) puis un SoftMax (14 classes de pizzas). Les données étant simples, le classifieur a pu être entrainé pour atteindre une précision de 100% sur un ensemble de test constitué de 20% des échantillons.
Nous avons ensuite utilisé les outils dérivés de LIME [19] et Grad-CAM [23] qui expliquent le résultat d'une classification en générant une heatmap mettant en valeur les pixels de l'image qui ont principalement participé à la prédiction. Nos images étant constituées de manière à ce que les seuls éléments qui différencient les pizzas soient les toppings présents sur l'image, on peut espérer que les heatmaps focalisent sur les pixels qui y correspondent.

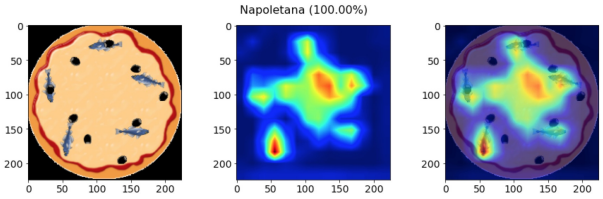

FIGURE 2 – Explication fournie par Grad-CAM.

La Figure 2 montre les résultats obtenus avec Grad-CAM pour la classification d'une *Napoletana* qui est constituée uniquement d'olives et d'anchois (cf. définition partie 3.1). Les résultats issus de LIME et Grad-CAM sont similaires. On peut constater que le CNN focalise bien sur les anchois. Cependant, il ignore les olives, tout en focalisant aussi sur une partie de la base (vide de toppings). On peut alors considérer que pour le CNN, cette pizza est une *Napoletana*, du fait qu'elle a des anchois et du vide : ce qui ne correspond bien entendu pas à la définition que l'on attendait.

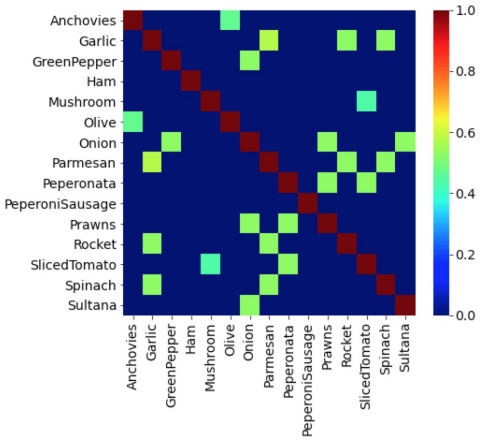

FIGURE 3 – Corrélation ontologique des toppings.

Toutefois, l'ontologie des pizzas permet d'expliquer ce phénomène si on l'utilise pour générer une matrice de corrélation ontologique des toppings (Figure 3). Cette matrice révèle en quelle mesure les toppings sont corrélés dans les définitions ontologiques des pizzas : on peut constater que les anchois (*AnchoviesTopping*) apparaissent systématiquement avec des olives (*OliveTopping*). Par contre, les olives apparaissent fréquemment avec d'autres toppings. De fait, pour le CNN, sur une *Napoletana* constituée uniquement

d'anchois et d'olives, le discriminant est la présence d'anchois. De plus, l'étude des définitions de toutes les classes de l'ontologie (non illustrée ici) révèle que la *Napoletana* est la pizza qui possède le moins de toppings, ce qui peut expliquer pourquoi le CNN a aussi considéré la zone vide comme un discriminant.

Ces remarques n'ont aucunement pour but de discréditer les outils tels que Grad-CAM et LIME. Comme nous venons de le montrer, ils se révèlent très utiles pour expliquer comment fonctionne un classifieur. Cependant, ces explications ne peuvent généralement être interprétées que par des spécialistes en IA et, comme l'ont souligné les travaux tentant d'associer une sémantique aux filtres des CNN [10], cet exemple démontre que le niveau d'abstraction des discriminants qui émergent de l'entraînement d'un CNN n'est pas en adéquation avec celui d'un expert des pizzas. De fait, ces outils ne paraissent pas les plus indiqués pour fournir des explications facilement interprétables par des experts du domaine.

3.3 Approche proposée

La démarche que nous proposons a pour but de créer un classifieur et de fournir des explications au niveau d'abstraction des experts du domaine d'application, c.à.d., de notre point de vue, dans les termes d'une ontologie. Dans notre exemple, il s'agit de classifier des images de pizzas avec les classes de l'ontologie, et de générer des heatmaps qui correspondent aux définitions ontologiques de ces classes.

Les étapes de la réalisation consistent à :

(a) Construire un ensemble C constitué des classes de l'ontologie prédites en sortie du classifieur.

(b) Soit D l'ensemble des axiomes de l'ontologie qui définissent les concepts de C :
D = {d | $\exists$ c $\in$ C, d $\equiv$ c est un axiome de l'ontologie}.
Soit P l'ensemble des propriétés de l'ontologie impliquées dans D.
Soit R l'ensemble des images de P tel que :
R = {r | r=range(p), p $\in$ P}.
Construire l'ensemble F des features ontologiques f $\in$ F, c.à.d. des triplets (c, p, r) impliqués dans D et qui seront utilisés dans l'explication d'une classification.

(c) Mettre en œuvre une technique d'AA permettant de construire l'ensemble FI $\subseteq$ F des features ontologiques identifiées (assertions satisfaites) dans une donnée envoyée au classifieur et qui sont de la forme FI = {fi $\in$ F | fi $\equiv$ $\exists$ p.r}

(d) Mettre en œuvre un raisonnement ontologique qui utilise D et FI pour calculer CI $\subseteq$ C, l'ensemble des classes ci identifiées pour une donnée.

(e) Utiliser l'ensemble des axiomes DI $\subseteq$ D tel que DI = {di $\equiv$ ci} et l'ensemble FI pour expliquer la classification CI.

Dans notre exemple :

(a) C = {c $\sqsubseteq$ Pizza}
ex. Napoletana

(b) D = {d $\equiv$ c}
ex. ($\exists$ hasTopping . AnchoviesTopping) $\sqcap$
($\exists$ hasTopping . OliveTopping) $\sqcap$
($\forall$ hasTopping .
(AnchoviesTopping $\sqcup$ OliveTopping))
P = { hasTopping }
R = {r $\sqsubseteq$ PizzaTopping }
ex. AnchoviesTopping
F = {(c $\sqsubseteq$ Pizza, hasTopping, r $\sqsubseteq$ PizzaTopping)}
ex. (Napoletana, hasTopping, AnchoviesTopping)

(c) Module de segmentation sémantique (cf. 4.1)

(d) Module OntoClassifier (cf. 4.2)

(e) Projection des assertions OWL (cf. 4.3)

Il est à noter que les étapes (b) et (c) sont fortement liées du fait qu'il serait vain de construire F avec des features ontologiques qui ne pourraient pas être extraites des données. Dans notre exemple, nous avons focalisé sur la propriété *hasTopping* du fait qu'elle est en adéquation avec les définitions des pizzas, mais aussi parce que la présence des toppings (éléments de R) peut être déduite de l'image. Le fait qu'il n'y ait ici qu'une seule propriété dans P (*hasTopping*) est lié à l'exemple : il serait tout à fait possible de considérer plusieurs propriétés différentes à extraire des données d'entrée pour inférer une classification.

On peut aussi souligner que le niveau d'abstraction des explications est intimement lié au niveau d'abstraction des features ontologiques. En effet, si dans notre exemple il sera possible d'expliquer qu'une image représente une *Napoletana* du fait qu'elle est constituée d'anchois et d'olives, le classifieur n'aura pas d'explication à fournir sur la manière dont il a décidé qu'une zone de l'image représente un anchois. Pour pouvoir le faire, il faudrait raffiner l'ontologie en donnant une définition des toppings eux-mêmes à partir de features ontologiques de plus bas niveau. Néanmoins, il faut rappeler que toute démarche pour l'explicabilité est confrontée au fait qu'à un certain niveau d'abstraction, on considère ne plus devoir fournir d'explications. On peut par exemple citer [12] qui propose un classifieur d'espèces d'oiseau mêlant CNN et NLP pour proposer des explications : l'outil peut expliquer qu'une image représente un Albatros du fait que l'oiseau possède un bec jaune, etc., mais il ne tente pas de démontrer ce qu'est un bec jaune.

Enfin, on peut remarquer que cette démarche engendre la création d'un pipeline de classification possédant 2 entités principales : un module utilisant une technique d'AP pour extraire les features ontologiques, suivi d'un module de raisonnement ontologique. Comme nous l'avons souligné, cette décomposition est similaire à celle que l'on peut trouver dans les solutions dédiées à identifier des classes de haut niveau d'abstraction comme les activités humaines [3][6]. Cependant, dans ces travaux, l'ontologie est principalement mise en œuvre pour aider la classification. La démarche que nous proposons est de partir de l'ontologie dans le but explicite de fournir des explications, y compris pour des problèmes de classification qui n'auraient *a priori* pas besoin d'ontologie. Nous verrons aussi dans la suite du papier que nous proposons une solution originale nommée

OntoClassifier pour la partie raisonnement de notre pipeline de classification.

4 Classifieur Ontologique

Cette partie présente l'implémentation de notre démarche sur l'exemple des pizzas présenté précédemment. Cette implémentation est constituée de 2 principaux modules : un module d'AP de segmentation sémantique destiné à identifier les features ontologiques (FI) présentes dans une image, et un module ontologique nommé OntoClassifier destiné à calculer les classes CI qui peuvent être déduites de FI, tout en étant capable de fournir des explications. Ces 2 modules sont implémentés et associés en Tensorflow 2. Le pipeline général de notre classifieur est présenté dans la Figure 4. Une image fournie en entrée de ce pipeline est traitée séquentiellement (flèches vers la droite) par le module d'AP (cf. 4.1), puis par le module ontologique (cf. 4.2) pour obtenir en sortie l'ensemble CI des classes identifiées pour cette image. Le mécanisme d'introspection du module ontologique permet ensuite de fournir des explications à propos de chaque classe identifiée ci ∈ CI (flèches vers la gauche) en utilisant sa définition ontologique di ∈ DI et les features fi ∈ FI correspondantes qui peuvent de plus être projetées sur (mises en exergue dans) l'image de départ (cf. 4.3).

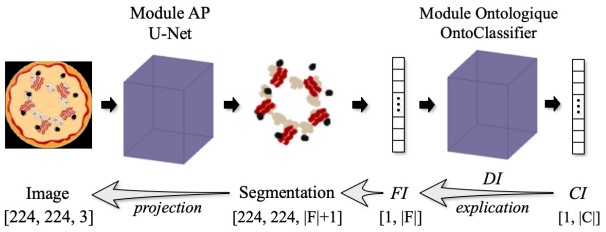

FIGURE 4 – Classifieur ontologiquement explicable.

4.1 Module AP : Segmentation Sémantique

La première partie du pipeline de classification a pour mission d'identifier les features ontologiques fi ∈ FI, c.à.d. la satisfiabilité des assertions correspondantes aux triplets de F sachant qu'ici :

$$F = \{(c \sqsubseteq \text{Pizza}, \text{hasTopping}, r \sqsubseteq \text{PizzaTopping})\}$$

Nous avons choisi d'utiliser une technique de segmentation sémantique dont l'objectif est d'étiqueter chaque pixel d'une image avec les classes de {r ⊑ PizzaTopping}. Notre jeu de données étant simple et totalement contrôlé, nous avons mis en œuvre une architecture de modèle basée sur U-Net [20], et généré les masques de segmentation nécessaires à l'entraînement du modèle en même temps que nos images. Cette implémentation de U-Net (basée sur MobileNetV2 [21] avec les poids d'Imagenet [7]) reçoit en entrée des images de pizzas constituées de 3 canaux (RGB) (224x224x3) pour obtenir en sortie une segmentation de l'image en 17 canaux (224x224x17). Chaque canal correspond à une des 16 sous-classes de *PizzaTopping*, excepté 1 canal destiné à recevoir les pixels qui ne correspondent à aucun topping.

La partie centrale de la Figure 4 montre comment une image fournie en entrée de l'U-Net est segmentée : pour représenter cette segmentation, nous avons ici superposé les différents canaux en leur attribuant chacun une couleur différente. Chaque couleur/canal correspond à une classe ontologique de topping (r ∈ R). Dans la suite du papier, cette représentation sera nommée *masque ontologique* dans le sens où les pixels de ce masque permettent (par projection) d'identifier la classe de topping à laquelle correspondent les pixels de l'image d'entrée.

4.2 Module Ontologique : OntoClassifier

La présence de pixels dans une couche de segmentation peut être interprétée comme la présence d'une feature ontologique (∃ hasTopping . topping), topping ∈ R, ce qui permet de déduire l'ensemble FI des assertions satisfaites pour chaque image traitée par le modèle. Il reste alors à raisonner à partir de FI en utilisant l'ensemble des définitions D pour en déduire les classes CI qui sont applicables à l'image.
Ce processus de raisonnement à partir de propriétés extraites de l'image est en partie similaire à celui qu'on peut trouver dans divers travaux mêlant AP et ontologies pour effectuer des interprétations à haut niveau d'abstraction. L'approche classique est de peupler l'ontologie avec des instances représentant les exemples à classifier, puis d'effectuer des déductions avec un raisonneur ontologique comme Jena, Hermit ou encore Pellet. Cependant, comme souligné dans [8], ce processus est coûteux du fait qu'il faille compléter le modèle d'AP par des outils externes, et que ces outils destinés à raisonner sur la globalité d'une ontologie sont bien plus lents que les pipelines d'AP utilisés pour créer des classifieurs. Dans la démarche que nous proposons, nous n'avons pas besoin de toute la puissance d'un raisonneur ontologique pour déduire CI à partir de FI et D. Nous avons donc créé le module OntoClassifier dont le constructeur génère un ensemble de tenseurs à partir de l'ontologie, en particulier de C, D et F.

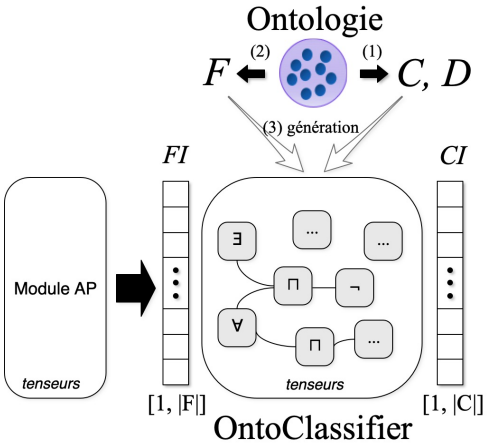

FIGURE 5 – Génération d'un OntoClassifier.

Ce processus est illustré dans la Figure 5 : après la sélection des classes et définitions visées pour construire C et D (1), et des features ontologiques qui constituent F (2),

un ensemble de tenseurs est automatiquement généré (3) : ces tenseurs sont typés et interconnectés grâce à la décomposition des expressions OWL contenues dans D. Les opérateurs de construction considérés sont la conjonction ($\sqcap$), la disjonction ($\sqcup$), la négation ($\neg$), les restrictions existentielles ($\exists r.c$), universelles ($\forall r.c$), et de cardinalité. L'OntoClassifier se base également sur l'hypothèse que les classes de R (range des features ontologiques) sont disjointes. Rappelons enfin que si dans l'exemple nous n'avons utilisé qu'une seule propriété ontologique (*hasTopping*), l'OntoClassifier est tout à fait capable de gérer un ensemble F contenant plusieurs propriétés (cf. 3.3).
L'OntoClassifier résultant est alors prêt à compléter le pipeline de classification en sortie du module d'AP (Figure 4, partie droite). Cet assemblage permet de calculer la satisfiabilité d'assertions ontologiques complexes comme la définition d'une *Napoletana* (cf. 3.1) ou encore des expressions impliquant des super-classes de toppings telles :

$$\text{CheesyPizza} \equiv \exists\ \text{hasTopping}\ .\ \text{CheeseTopping}$$

$$\text{VegetarianPizza} \equiv\ \neg\ (\exists\ \text{hasTopping}\ .\ \text{FishTopping}) \sqcap \neg\ (\exists\ \text{hasTopping}\ .\ \text{MeatTopping})$$

Enfin, l'OntoClassifier étant composé d'un graphe de tenseurs qui représente la décomposition des éléments de D, ce module permet de remonter le graphe des assertions pour identifier les éléments de FI – ou l'absence d'éléments – qui les ont satisfaites dans un exemple donné.

4.3 Résultats

Le pipeline de classification étant en place, il ne reste plus qu'à lui envoyer des images pour obtenir une classification. Pour commencer, nous aimerions souligner que la génération de l'OntoClassifier sous forme de tenseurs permet l'intégration directe de la dimension ontologique dans le pipeline de classification. Le modèle global résultant est alors bien plus rapide que dans le cas où le raisonnement ontologique est délégué à un moteur d'inférence externe. À titre d'exemple, en utilisant notre module AP de segmentation sémantique couplé à une instance de raisonneur Hermit sur nos machines (I9- 10850K à 3.6 GHz, 32 Go DDR4 3200MHz, GPU RTX 3080), il faut en moyenne 130s pour classifier 100 images de pizzas. Avec l'OntoClassifier, sur les mêmes machines, la classification des mêmes données prend en moyenne 1,6s. Il faut bien entendu relativiser cette différence car, comme nous l'avons souligné dans la partie 4.2, un module comme Hermit est destiné à raisonner sur une ontologie dans sa globalité en considérant de manière exhaustive l'ensemble des relations qui peuvent être inférées, alors que le raisonnement réalisé par un OntoClassifier focalise uniquement sur les relations ontologiques qui servent à calculer les classes de C, c'est à dire celles explicitement visées par le classifieur. Comme dans la partie 3.2, et sur le même jeu de données assez simples d'images synthétiques de pizzas, ce pipeline a pu être entraîné pour atteindre une précision de 100% sur un ensemble de test constitué de 20% des échantillons.
La Figure 6 montre l'exemple d'une image de *Fiorentina*. Le classifieur fournit la liste des classes qui ont été détectées (*Fiorentina*, *Cheesy*, *Vegetarian*). Le masque ontologique (résultant de la segmentation) fournit la liste des ingrédients. Ce masque peut être projeté sur l'image « à la Grad-CAM », mais nous avons choisi ici de l'afficher séparément pour plus de lisibilité. On peut noter que contrairement à ce que nous avions montré en 3.2, le niveau d'abstraction est ici en adéquation avec celui de l'ontologie : aucun topping n'est ignoré par le classifieur et dans le masque, les entités mises en valeur correspondent à la définition ontologique des classes, et chaque topping est différencié et identifiable grâce à un code couleur.

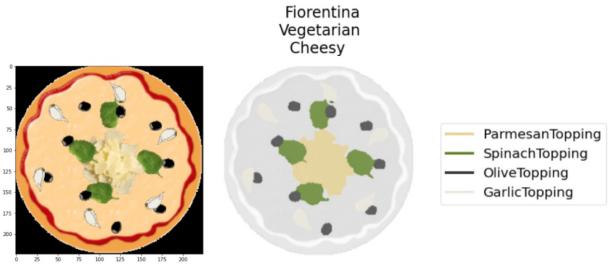

FIGURE 6 – Classification et segmentation ontologique.

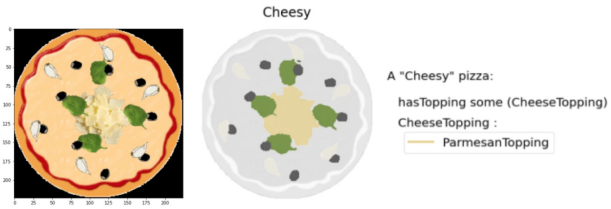

FIGURE 7 – Explications visuelles ontologiques.

Le système permet de plus de focaliser sur une des classes identifiées et d'utiliser l'introspection de l'OntoClassifier pour expliquer cette classification. Ainsi, la Figure 7 illustre une focalisation sur la classe *Cheesy* identifiée pour l'image de la Figure 6. Ce focus reprend le masque ontologique, et y ajoute (en partie droite) une explication qui met en correspondance la définition en OWL de la classe avec les pixels de l'image qui ont participé, en tant que features ontologiques, à la satisfiabilité des assertions qui la composent. Dans la même idée, la Figure 8 illustre le fait que l'OntoClassifier peut aussi expliquer le résultat d'une classification du fait de l'absence de features.

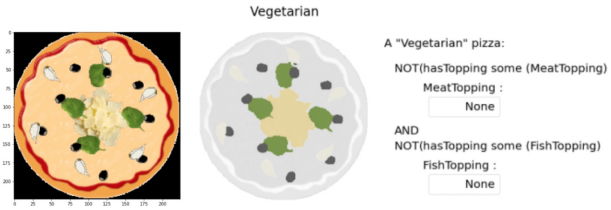

FIGURE 8 – Explications par des features absentes.

5 Discussion

Il est vrai que la démarche présentée ici demande un travail préparatoire supplémentaire autour de la création d'une ontologie pour réifier le niveau d'abstraction des utilisateurs, de la construction des ensembles C, D et F, ainsi que la mise en œuvre d'une technique d'AP plus complexe que pour une « simple » classification. On peut cependant aussi noter que cette approche, en plus de résulter en un classifieur ontologiquement explicable, possède d'autres avantages.
Tant que l'ensemble des features ontologiques (F) ne change pas, il est facilement possible d'ajouter de nouvelles classes dans C, d'en supprimer, ou de les modifier, et d'intégrer ces évolutions au pipeline de classification sans avoir à ré-entraîner le modèle. Il suffit de (re-)générer l'OntoClassifier pour que la nouvelle version intègre les nouvelles étiquettes, et soit capable d'expliquer pourquoi une image y correspond.
Comme le soulignent [29] et [4], la notion de point de vue est importante, y compris dans le domaine des ontologies : le sens des choses est pluriel, elles ont d'ailleurs souvent des définitions différentes selon les points de vue. La démarche et les outils que nous proposons peuvent apporter des éléments de réponse au besoin de multi points de vue puisqu'il est possible de générer plusieurs OntoClassifiers dédiés à des points de vue différents sur le même domaine. Pour ce faire, il est possible de composer différents ensembles C (un par point de vue), et de générer les différents OntoClassifiers qui viendront compléter le même module d'AP, proposant ainsi des pipelines de classification et d'explicabilité dédiés à chaque point de vue.
Enfin, notre démarche permet aussi d'introduire la notion de point de vue au niveau de l'explicabilité elle-même. En effet, pour un même ensemble de classes C, il est possible de créer différents ensembles de définitions D, et donc de générer des classifieurs explicables selon diverses définitions ontologiques. Dans l'ontologie des Pizzas, une *Vegetetarian* peut ainsi être définie de plusieurs manières :

(1) VegetarianPizza $\equiv \neg$ ($\exists$ hasTopping . FishTopping) $\sqcup$ $\neg$ ($\exists$ hasTopping . MeatTopping)

(2) VegetarianPizza $\equiv \forall$ hasTopping . VegetarianTopping

Si ces 2 définitions aboutissent à la même classification, elles permettent de créer des OntoClassifiers qui fourniront des explications visuelles avec des points de vue différents, focalisant sur (mettant en valeur) la présence ou l'absence soit (1) des toppings sous-classes de poissons et viandes (cf. Figure 8), soit (2) des toppings sous-classes d'ingrédients végétariens.

6 Conclusion

Répondant au besoin d'explicabilité des IA, nous avons exploré dans ce papier les apports et la faisabilité d'un processus de création de classifieurs ontologiquement explicables du point de vue des utilisateurs du domaine. Nous avons proposé une démarche générique inspirée de diverses approches de l'état de l'art qui permettent l'explicabilité, et qui résulte en une architecture basée sur 2 modules, l'un dédié à l'extraction de features ontologiques par des techniques d'AP, l'autre dédié au raisonnement ontologique et nommé OntoClassifier.
De manière à intégrer la dimension ontologique au cœur même du classifieur sans trop alourdir le pipeline de classification résultant, nous avons introduit un outil permettant la génération de l'OntoClassifier, ce module étant implémenté automatiquement sous la forme d'un graphe de tenseurs ontologiques construit directement à partir des définitions fournies par l'ontologie.
Nous avons exemplifié notre démarche par la création d'un classifieur d'images dédié au domaine de la fameuse ontologie des Pizzas, et illustré les possibilités offertes par l'OntoClassifier, aussi bien du point de vue de la classification, que de celui de l'explicabilité visuelle des prédictions en utilisant les définitions OWL de l'ontologie. Comme nous l'avons souligné, le jeu de données mis en oeuvre dans cet exemple était simple et totalement contrôlé, ceci dans le but d'aider à la formalisation et à l'illustration de la démarche que nous proposons. Nos travaux en cours impliquent d'ores et déjà cette démarche et ces outils dans un projet d'envergure dédié à un autre domaine, sur des données réelles, et pour une tâche de classification d'images à grain fin.
Les éléments que nous avons présentés restent cependant à être étoffés. Nous avons en particulier pour l'instant principalement focalisé sur les possibilités offertes par un classifieur ontologiquement explicable et n'avons pas encore travaillé sur l'ergonomie des interfaces homme-machine qui permettront aux utilisateurs finaux de manipuler et d'explorer les explications fournies par le système : ce point fera l'objet de futurs travaux. De plus, les explications visuelles proposées ici utilisent directement les expressions OWL pour les relier à l'image : cette représentation demande aussi à être améliorée, puis à être évaluée avec des utilisateurs.
Les éléments exposés dans ce papier fournissent néanmoins des bases solides nous permettant d'entamer de nouveaux projets ayant besoin de classifieurs explicables au niveau d'abstraction de leurs utilisateurs. Il permettent de plus d'imaginer de nouvelles fonctionnalités prometteuses, voire nécessaires dans le cadre de projets impliquant des acteurs aux cultures différentes, comme par exemple le fait d'envisager l'explicabilité des IA tout en considérant le besoin de multi points de vue.

Références

[1] P. Angelov and E. Soares. Towards explainable deep neural networks (xDNN). *Neural networks : the official journal of the International Neural Network Society*, 130 :185–194, 2020.

[2] A. Arrieta, N. Díaz-Rodríguez, J. Ser, Adrien Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-López, D. Molina, R. Benjamins, R. Chatila, and F. Herrera. Explainable artificial intelligence (XAI) : Concepts, taxonomies, opportunities and challenges toward responsible AI. *ArXiv*, abs/1910.10045, 2020.

[3] J. Atif, C. Hudelot, and I. Bloch. Explanatory reasoning for image understanding using formal concept analysis and description logics. *IEEE Transactions on Systems, Man, and Cybernetics : Systems*, 44 :552–570, 2014.

[4] A. Bénel and C. Lejeune. Humanities 2.0 : documents, interpretation and intersubjectivity in the digital age. *Int. J. Web Based Communities*, 5 :562–576, 2009.

[5] R. Confalonieri and Tarek R. Besold. Trepan reloaded : A knowledge-driven approach to explaining black-box models. In *ECAI*, 2020.

[6] D. Conigliaro, R. Ferrario, C. Hudelot, and D. Porello. Integrating computer vision algorithms and ontologies for spectator crowd behavior analysis. In *Group and Crowd Behavior for Computer Vision*, 2017.

[7] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and L. Fei-Fei. ImageNet : A large-scale hierarchical image database. In *CVPR*, 2009.

[8] Z. Ding, L. Yao, B. Liu, and J. Wu. Review of the application of ontology in the field of image object recognition. In *ICCMS 2019*, 2019.

[9] F. K. Dosilovic, M. Brčič, and N. Hlupic. Explainable artificial intelligence : A survey. *41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pages 0210–0215, 2018.

[10] A. Gonzalez-Garcia, D. Modolo, and V. Ferrari. Do semantic parts emerge in convolutional neural networks ? *International Journal of Computer Vision*, 126 :476–494, 2017.

[11] B. Goodman and S. Flaxman. European union regulations on algorithmic decision-making and a "right to explanation". *AI Mag.*, 38 :50–57, 2017.

[12] L. A. Hendricks, Z. Akata, M. Rohrbach, J. Donahue, B. Schiele, and T. Darrell. Generating visual explanations. In *ECCV*, 2016.

[13] Z. C. Lipton. The mythos of model interpretability. *Queue*, 16 :31 – 57, 2018.

[14] M. Losch, M. Fritz, and B. Schiele. Interpretability beyond classification output : Semantic bottleneck networks. *ArXiv*, abs/1907.10882, 2019.

[15] S. M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *NIPS*, 2017.

[16] D. Marcos, S. Lobry, and D. Tuia. Semantically interpretable activation maps : what-where-how explanations within CNNs. *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 4207–4215, 2019.

[17] D. Martens and F. Provost. Explaining data-driven document classifications. *MIS Q.*, 38 :73–99, 2014.

[18] D. P. Papadopoulos, Y. Tamaazousti, F. Ofli, I. Weber, and A. Torralba. How to make a pizza : Learning a compositional layer-based gan model. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7994–8003, 2019.

[19] M. Tulio Ribeiro, S. Singh, and C. Guestrin. "why should i trust you ?" : Explaining the predictions of any classifier. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.

[20] O. Ronneberger, P. Fischer, and T. Brox. U-Net : Convolutional networks for biomedical image segmentation. *ArXiv*, abs/1505.04597, 2015.

[21] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L-C. Chen. MobileNetV2 : Inverted residuals and linear bottlenecks. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4510–4520, 2018.

[22] Md. Kamruzzaman Sarker, N. Xie, D. Doran, M. Raymer, and P. Hitzler. Explaining trained neural networks with semantic web technologies : First steps. *ArXiv*, abs/1710.04324, 2017.

[23] R. R. Selvaraju, A. Das, R. Vedantam, M. Cogswell, D. Parikh, and D. Batra. Grad-CAM : Visual explanations from deep networks via gradient-based localization. *International Journal of Computer Vision*, 128 :336–359, 2019.

[24] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2015.

[25] W. Swartout, C. Paris, and J. Moore. Explanations in knowledge systems : design for explainable expert systems. *IEEE Expert*, 6 :58–64, 1991.

[26] D. West. The future of work : Robots, AI, and automation. 2018.

[27] M. D. Zeiler, G. W. Taylor, and R. Fergus. Adaptive deconvolutional networks for mid and high level feature learning. *2011 International Conference on Computer Vision*, pages 2018–2025, 2011.

[28] Q. Zhang, Y. Wu, and S. Zhu. Interpretable convolutional neural networks. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8827–8836, 2018.

[29] M. Zhitomirsky-Geffet, E. S. Erez, and J. Bar-Ilan. Toward multiviewpoint ontology construction by collaboration of non-experts and crowdsourcing : The case of the effect of diet on health. *Journal of the Association for Information Science and Technology*, 68, 2017.

[30] J. Zhu, A. Liapis, S. Risi, R. Bidarra, and G. Youngblood. Explainable AI for designers : A human-centered perspective on mixed-initiative co-creation. *2018 IEEE Conference on Computational Intelligence and Games (CIG)*, pages 1–8, 2018.