Ontology-Driven Cross-Domain Transfer Learning

Mattia Fumagalli, Gabor Bella, Samuele Conti and Fausto Giunchiglia

January 2020

Technical Report # DISI-20-006

Published in: FOIS 2020

January 2020

Ontology-Driven Cross-Domain Transfer Learning

Mattia FUMAGALLI, Gábor BELLA, Samuele CONTI and Fausto GIUNCHIGLIA
Department of Information Engineering and Computer Science
(DISI) - University of Trento, Italy

Abstract. The aim of transfer learning is to reuse learnt knowledge across different contexts. In the particular case of *cross-domain transfer* (also known as *domain adaptation*), reuse happens across different but related knowledge domains. While there have been promising first results in combining learning with symbolic knowledge to improve cross-domain transfer results, the singular ability of ontologies for providing classificatory knowledge has not been fully exploited so far by the machine learning community. We show that ontologies, if properly designed, are able to support transfer learning by improving *generalization* and *discrimination* across classes. We propose an architecture based on *direct attribute prediction* for combining ontologies with a transfer learning framework, as well as an ontology-based solution for cross-domain generalization based on the integration of top-level and domain ontologies. We validate the solution on an experiment over an image classification task, demonstrating the system's improved classification performance.

Keywords. transfer learning, ontology, domain adaptation, ontologies and machine learning, generalization

1. Introduction

A long-standing ambition of research in artificial intelligence has been to achieve and exceed the human ability for generalization and telling things apart. While in the last decade, statistical approaches and machine learning (ML) in AI have been vastly successful in solving complex tasks, most solutions proceed by learning stand-alone models from large amounts of raw or annotated data. *Transfer learning* has been the research area that tries to simulate generalization by applying learned models to new tasks [1,2,3].

However, despite undeniable progress in recent research on transfer learning techniques, generalization of ML knowledge is still considered an open problem, with quantitative results in real-world scenarios rarely reaching the level of practical applicability [4,5,6]. Recently, ML communities have started experimenting with the integration of resources and methods of formal knowledge representation into learning systems in the hope of surpassing the current performance plateau of pure learning-based systems. As *"formal, explicit specifications of a shared conceptualization"* [7], ontologies have been designed to enable the reusability of information, as rich semantic data structures encoding previously acquired knowledge, whether general or domain-specific.

The main focus of research in this area has so far been to address the semantic gap between symbolic and learning-based (statistical or neural) representations of knowledge. There have been significant cases of success which proved the feasibility of the reuse of symbolic knowledge in transfer learning tasks [8,9,10,11]. However, no particular attention was made as to how the ontological design of these resources affects the key abilities of the transfer learning framework to *generalize* and to *discriminate* across classes. Likewise, there have been few attempts at reusing the vast amounts of existing ontological and other (formal or semi-formal) knowledge resources.

Our paper proposes a novel solution for integrating ontologies into state-of-the-art transfer learning methods, in order to increase their ability for generalization and discrimination across classes and, ultimately, their classification performance. Our solution consists of (1) a theoretical framework that justifies the use of ontologies in the context of transfer learning and exploits them more deeply than state-of-the-art methods; (2) a practical architecture for combining learning with the generalization and discrimination ability provided by ontologies; (3) a method for combining domain and top-level ontologies within the architecture in order to increase the generalization ability in cross-domain transfer tasks; and (4) an experimental validation of the theory and architecture over a cross-domain image classification task.

The paper is structured as follows. Section 2 provides theoretical motivations for the successful use of ontologies for transfer learning. Section 3 describes the architecture and method for combining a learning framework with ontologies. Section 4 validates the solution on cross-domain image classification using both domain and top-level ontologies. Sections 5 and 6, finally, describe related work, conclusions, and perspectives.

2. The Role of Ontologies in the Transfer of Knowledge

While the range of transfer learning techniques is vast [1,2,12], their common goal is knowledge reuse: what is learnt for completing a task in a particular domain, should be reusable in other, (to a certain extent) overlapping domains. Thus, an encyclopedic text corpus annotated for the recognition of person and product names can be reused to recognize patient and drug names in medical texts, or a system trained to classify facial expressions as happy or sad could be reused for the detection of mental health problems.

The idea of our paper, in one sentence, is to inject ontological knowledge into a transfer learning system in order to increase and make explicit the overlapping knowledge, thereby improving the performance of the transfer.

We begin by defining the notions of *domain*, *(classification) task*, and *transfer* within the probabilistic learning paradigm. We then link these notions to *ontological domains*, *classes* and *properties*, and show how ontological knowledge can be used to solve problems of knowledge generalization within the learning paradigm. The contents of this section provide the motivations of the architectural solution presented in section 3.

2.1. Transfer within the Machine Learning Paradigm

In the following, we provide a standard probabilistic formalization of the notions of domain and task in ML by adopting the definitions provided in [2].

A domain $D = \{\mathscr{X}, P(X)\}$ *consists of a feature space* $\mathscr{X}$ *and a marginal probability distribution* $P(X)$ *over the feature space, namely* $X = \{x_1, ..., x_n\} \in \mathscr{X}$. *A* task $T = \{\mathscr{Y}, P(Y|X)\}$ *consists of a label space* $\mathscr{Y}$ *and a conditional probability distribution* $P(Y|X)$ *that is learned from the training data consisting of pairs* $x_i \in X$ *and* $y_i \in Y$.

Given a source domain D_S, a corresponding source task T_S, as well as a target domain D_T and a target task T_T, the objective of transfer learning *is to improve the learning of a conditional probability distribution $P(\mathscr{Y}_T|\mathscr{X}_T)$ in D_T with the information gained from D_S and T_S where $D_S \neq D_T$ or $T_S \neq T_T$.*

We will call *cross-task transfer learning* the case when $T_S \neq T_T$, and *cross-domain transfer learning* when $D_S \neq D_T$. In the practice of transfer learning, the cross-domain case most often means $D_S \subset D_T$ and $\mathscr{Y}_S \subset \mathscr{Y}_T$, i.e. that the source domain and label space are *extended* (or with a more widely used term: *adapted*) by fusing them with the target domain and label space. In other words, the system learns to recognize new classes without "forgetting" about previous ones.

The work discussed in [2] identifies four main scenarios where transfer learning is needed: (1) $\mathscr{X}_S \neq \mathscr{X}_T$: difference in the feature spaces; (2) $P(X_S) \neq P(X_T)$: difference in the marginal probability distributions; (3) $\mathscr{Y}_S \neq \mathscr{Y}_T$: difference in the label spaces; and (4) $P(Y_S|X_S) \neq P(Y_T|X_T)$: difference in the conditional probability distributions. The first two cases are considered as cross-domain transfer while the last two cases are cross-task transfer within the same domain. In general, all transfer learning setups solve a problem of mapping unseen data (features, labels) to what has already been learnt.

2.2. Transfer within the Ontological Paradigm

To understand how ontologies can support such a mapping, we draw a parallel between the scenarios above and ontology-based classification. For the scope of our work, we consider only the taxonomical and class–property relationships encoded in ontologies.

We model an ontology taxonomy *(that we abbreviate simply as "ontology" in the rest of the paper*[1]*) as $\mathscr{O} = \langle C, P, I, \Phi, \Psi \rangle$. We take $C = \{c_1, ..., c_n\}$ to be the set of* classes *of $\mathscr{O}$, $P = \{p_1, ..., p_n\}$ the set of* properties *of $\mathscr{O}$, I a binary relation such that $I \subseteq C \times P$, which expresses which classes* are associated to *which properties. Φ and Ψ represent the "class hierarchy" and the "property hierarchy", respectively: a set of directed edges in the form of $c_i \overset{is-a}{\rightarrow} c_j$ (resp. $p_i \overset{is-a}{\rightarrow} p_j$) where $c_i \in C$ (resp. $p_i \in P$) is the child and c_j (resp. p_j) the parent of the directed edge.*

We say that a class is *associated to* a property when the latter is used to describe the former, and that a property is *associated to* a class with the dual meaning. We also talk of class c being in the domain of a property p, $c \in \mathrm{Dom}(p)$, when c is associated with p.

According to [13], *domain ontologies "describe the vocabulary related to a generic domain (such as medicine, or automobiles)"*. A domain ontology $\mathscr{O}_D$ is then an ontology with classes, properties, hierarchies, and instances that are specific to—though not necessarily exclusive of—the domain. Within the ontological paradigm, cross-domain classification means classifying against a different domain ontology $\mathscr{O}_{D_T}$. The classes that form the source and target label spaces, $C_S \subseteq C_{D_S}$ and $C_T \subseteq C_{D_T}$, resp., are classes of two distinct ontologies, with possible heterogeneity in naming, hierarchical structure, granularity, etc. These major divergences are analogous to the distinct feature spaces and marginal probabilities in the probabilistic paradigm.

---

[1] For brevity, we allow ourselves this simplification in terminology with respect to the canonical definition of the term.

2.3. Ontology Design for Transfer Learning

The core idea underlying our paper is that symbolic knowledge resources, when properly organized according to ontological principles, provide two major tools for cross-domain transfer: *generalization* and *discrimination*. While generalization uncovers the overlap between the semantic spaces across domains, discrimination provides distinctive properties of classes as key features to the transfer learning process.

By discrimination *we understand the requirement that, in an ontology, classes are used to represent in an abstract but formal way the properties that are definitional of a group of instances in any possible world* [7].

While the idea that classes are characterized by their properties is in itself almost trivial, the stronger formal ontological requirement that sibling classes must be distinguished between each other by at least one discriminating property is far from being universally applied in real-world knowledge resources.[2] The use of such ontology-derived discriminating properties as features lends more discriminating power to ML-based classification methods.

As a theoretical background for *generalization*, we reuse the *theory of abstraction* from [14], further explored in [15]. We adopt as a typology of generalization the three major kinds of abstraction defined there:

- *predicate abstraction* where a predicate (in our case, a class) is mapped to a more general one, e.g. $Car(X) \rightarrow Vehicle(X)$;
- *domain abstraction* where a constant or function (in our case, a property) is mapped to a more general one, e.g. $fatherOf(X) \rightarrow parentOf(X)$;
- *propositional abstraction* where one or more arguments of a predicate are dropped, e.g. $isSiblingOf(X,Y) \rightarrow Sibling(X)$.

Thus, by generalization *we understand the application of predicate, domain, or propositional abstraction operations to any class or property of an ontology.*

Generalization helps the transfer learning method take advantage of the hierarchical relatedness of classes and properties beyond mere equivalence based on their labels. For example, in a *named entity recognition* task, a general learning model trained on *organization* classes can be reused in the medical domain to recognize names of *clinics* (predicate abstraction). Likewise, a model that recognizes the property of *having legs* can be used to identify *having limbs* (domain abstraction).

In a cross-domain scenario, however, the lack of a unifying theory for the two domains may prevent abstraction from providing results. In this case, the integration of two domain ontologies with a "cross-domain" ontology (e.g. a *core* or *top-level ontology*) may enable abstraction. The use of top-level ontologies is thus a crucial element of our solution for supporting cross-domain transfer.

Generalization and discrimination are used in combination for efficient transfer. For example, generalization allows the discovery that *Neurologist* (from the source domain of *Neurology*) and *Surgeon* (from the target domain of *Surgery*) are both *Doctors*. Then, discrimination via the property *operates-on-brain* allows the distinction within the target domain between *Neurosurgeon* and, say, *Vascular surgeon*.

---

[2] For instance, `schema.org` has many classes defined that only differ in their names.

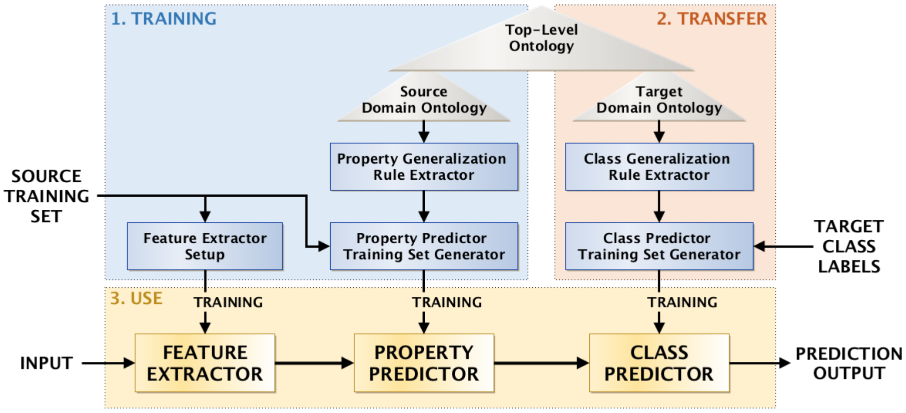

Figure 1. Architecture of the ontology-based cross-domain transfer learning method.

3. Architecture and Process

Figure 1 shows the high-level architecture of our solution. It is divided into three main phases: *(1) training*, *(2) transfer*, and *(3) use*. The cross-domain classification tasks, which are the goal of the architecture, are executed in the *use* phase by a three-step pipeline inspired by the *Direct Attribute Prediction* (DAP) technique.

DAP is a state-of-the-art transfer learning method described, among others, in [16]. The main principle behind DAP is, rather than performing a direct classification on the input, to start by predicting the properties (attributes) of the object in input and then predict the class based on the properties. The mapping of the learning-based (neural, probabilistic, etc.) and ontology-based representations is achieved through the use of ontology class and property labels as the label spaces of both property and class prediction.

The strengths of DAP are its ability to *predict unseen classes* and to *simplify the prediction task* (for sensory input such as images, it is often easier to predict a small set of properties as an intermediate step rather than to predict classes directly). Property prediction is also *more generic:* while the class definitions are often specific to the domain of use, it is less the case for properties that encode lower-level information, therefore property predictors are “transferred” more easily. Accordingly, the pipeline consists of:

1. a *Feature Extractor* $f_E : \mathscr{I} \mapsto \mathscr{X}_S$ that takes the input data $\mathscr{I}$ to be classified (text, images, etc.) and extracts a high number of low-level patterns or *features*;
2. a *Property Predictor* $f_P : \mathscr{X}_S \mapsto P_{D_S}$ that takes low-level features as input and predicts a set of higher-level *properties* from them;
3. a *Class Predictor* $f_C : P_{D_S} \mapsto \mathscr{Y}_T$ that emits a class label based on the properties previously predicted.

The purpose of the *training* and *transfer* phases preceding the use of DAP is to train the pipeline for cross-domain classification. The *training* phase is run initially and *only once*, in order to set up and train the feature extractor and the property predictor based on an initial *source training set*. The *transfer* phase is run every time the system needs to be adapted to a new domain.

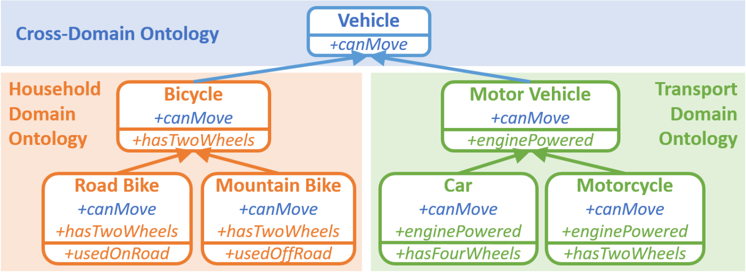

Figure 2. Example (toy) ontologies demonstrating ontology design principles that result in efficient prediction.

3.1. Ontology Design Principles for Transfer Learning

The difference of our architecture with respect to state-of-the-art DAP pipelines resides in its use of ontologies—designed by expert communities according to ontological principles—as sources of class labels, property labels, and class–property associations.

In a cross-domain scenario it is typically not possible to find a single ontology that describes the classes and properties of both the source and the target domain adequately. We therefore apply a standard ontology engineering approach—so far unexplored in the context of transfer learning—where two distinct *source* and *target domain ontologies* are interconnected with a *top-level ontology* (in the following: TLO, as depicted in Figure 1). The TLO can be a genuine top-level ontology such as DOLCE or PROTON but, in order not to be too restrictive, we also allow other kinds of higher-level ontologies capable of subsuming both domain ontologies. The root nodes of the domain ontologies need to be connected by *is-a* relations to the nodes of the TLO.

While a full experimental account on the influence of all ontology design principles on transfer learning performance is beyond the scope of this paper (and is foreseen as future work), we provide a simple illustration of how an ontology design that is aware of the generalization and discrimination principles helps improve transfer results. As an example, let us consider an image classification task that uses the very simple ontologies in Figure 2. Suppose we have a Property Predictor trained on different kinds of family vehicles and other objects from the *household domain*. We would like to reuse this predictor to find cars on a different set of images related to the *travel domain*. In order to improve transfer, we reuse the tiny domain ontologies and TLO from Figure 2.

Discrimination is supported by the domain ontologies by always associating distinct property sets to non-equivalent classes, e.g. *Cars* and *Motorcycles* are distinguishable by the number of wheels (let us not consider the *Piaggio Ape* or the *Reliant Robin* this time). At the same time, when comparing the property set $\{hasFourWheels\}$ specific to *Car* with $\{humanPowered\}$ specific to *Car*, no overlap is found, making property-based cross-domain transfer impossible without relying on the class the hierarchy.

Generalization helps in this case: through property inheritence from domain and top-level superclasses, the property sets are extended to $\{canMove, enginePowered, hasWheels\}$ and $\{canMove, enginePowered, hasFourWheels\}$, respectively. *Car* and *Automobile* have now two properties out of three in common, which the Class Predictor can use to infer the right class label output. At the same time, the *canMove* property gained from the TLO also provides discriminative power, e.g. for the distinction between vehicles and other household objects.

In case a property hierarchy is available, domain abstraction can also be applied as further generalization. Let us suppose that the TLO defines $\Psi_{TLO} = \{hasFourWheels \overset{is\text{-}a}{\rightarrow} hasWheels, hasTwoWheels \overset{is\text{-}a}{\rightarrow} hasWheels\}$. Then classification results are further improved as the similarity of the property sets of *Automobile* and *Car* are further increased, as the property set of *Automobile* will be extended to $\{$*canMove*, *enginePowered*, *hasFourWheels*, *hasWheels*$\}$.

3.2. Training Phase

In this first phase, a *Feature Extractor* f_E and a *Property Predictor* f_P are set up and trained. This is done only once for any given type of input, although the two components may need to be re-adapted if the input data representation changes considerably.

Feature extraction is a classic early step of machine learning processes that deals with heterogeneity in the input. Even if in more recent deep learning architectures f_E can itself be trained as early layers in the neural network, even such architectures need to preprocess the input to some extent. The design of feature extractors is outside of the scope of our work (although we do implement it as part of our experiment in section 4).

The Property Predictor is typically a machine learning component that maps the features $\mathscr{X}_S$ extracted from the input to higher-level *properties*. This supposes the existence of a training set (marked as *source training set* in Figure 1) that needs to be labeled with property names. Therefore, the two challenges are: (1) the choice of properties to use for annotation; (2) the annotation task itself.

The properties used for annotation are provided by the *Property Generalization Rule Extractor* component. All properties P_S are extracted from the *source domain ontology* $\mathscr{O}_{D_S}$ and the TLO $\mathscr{O}_{D_{TLO}}$ (Figure 1): true to the nature of transfer learning, at the moment of training, the target domains are not yet known. If a property hierarchy Ψ_{D_S} is available in $\mathscr{O}_{D_S}$ then, according to *domain abstraction* (see section 2.3), each property is enriched into a property set with its ancestor properties. If $X_i \subset \mathscr{X}_S$ is the set of features corresponding to the i^{th} input data element observed, and $\mathrm{Ann}(X_i, p_j)$ its annotation by a property p_j, then we model domain abstraction as:

$$\mathrm{Ann}(X_i, p_j) \wedge (p_j \overset{is\text{-}a}{\rightarrow} p_k) \Rightarrow \mathrm{Ann}(X_i, p_k) \qquad (1)$$

For example, if an image of a person was annotated by a property *"has-leg"* then a new super-property *"has-limb"* will be added as annotation, provided that such a super-property exists in the property hierarchy Ψ_{D_S}.

The once-and-for-all annotation task itself is performed by the *Property Predictor Training Set Generator* component. In practice, this component can be implemented in multiple ways depending on the data available: supervised or unsupervised, automated or manual. In another DAP setup for image classification, [17] uses an unsupervised NLP-based method to extract property names from captions associated to images. In our experiment in section 4, we manually extended the annotations of an existing dataset, over a 12.500-image corpus.

3.3. Transfer Phase

The goal of the transfer phase is to adapt the DAP pipeline to specific classification domains and tasks. Adaptation is achieved by automatically training the *Class Predictor* to determine the most likely class(es) based on the properties predicted and the classification labels $\mathscr{Y}_T$ of the target domain and task. While using a logical inference engine as Class Predictor is theoretically possible, for increased robustness we implement it—as do other DAP-based solutions—as a machine learning component. The source of both the predicted class labels $\mathscr{Y}_T$ and the property–class associations used for training is the *target domain ontology*; it can, however, also be the TLO if the target classes are very general (Figure 1). The TLO also plays an important role by connecting the source and target domain ontologies. Having a common TLO increases the amount of properties shared between source and target classes, improving cross-domain transfer.

The role of the *Class Generalization Rule Extractor* component is to extract class–property associations (in the form of $\mathrm{Dom}(p_i, c_j)$, meaning that the class c_j is in the domain of the property p_i, using the term "domain" in the mathematical sense here) from the entire ontology (i.e. the combination of the three ontologies). The associations are *generalized* because *predicate abstraction* (formula 2) and *domain abstraction* (formula 3) are taken into account:

$$\mathrm{Dom}(p_i, c_k) \wedge (c_j \overset{\textit{is-a}}{\rightarrow} c_k) \Rightarrow \mathrm{Dom}(p_i, c_j) \tag{2}$$

$$\mathrm{Dom}(p_i, c_k) \wedge (p_i \overset{\textit{is-a}}{\rightarrow} p_j) \Rightarrow \mathrm{Dom}(p_j, c_k) \tag{3}$$

meaning, respectively, that subclasses "inherit" the properties of their superclasses and that if a property is associated to a class then its superproperties are also associated to it.

From the set of class–property associations, the *Class Predictor Training Set Generator* component automatically generates a training corpus for the Class Predictor, taking as input the subset of class labels $\mathscr{Y}_T$ that need to be predicted (it is highly unlikely that prediction should cover *all* domain ontology and TLO classes). The Class Predictor is then retrained on this corpus to be adapted to the new classification task or domain. Thus, after plugging in all necessary ontologies, the actual adaptation is fully automatic and is practically of a push-button simplicity.

3.4. Use Phase

In the use phase, the DAP pipeline, including the adapted Class Predictor, is used to solve the cross-domain classification problem.

The Feature Extractor $f_E : \mathscr{S} \mapsto \mathscr{X}_S$ component takes sensory inputs $s_i \in \mathscr{S}$ and outputs the associated features $x_i \in \mathscr{X}_S$. The output of the Property Predictor $f_P : \mathscr{X}_S \mapsto P_S \times \mathbb{R}[0;1]$ consists of a list of confidence values (between 0 and 1) for each property, associated to each input sensory observation. This output is then given as input to the Class Predictor $f_C : P_S \times \mathbb{R}[0;1] \mapsto \mathscr{Y}_T \times \mathbb{R}[0;1]$. It runs a prediction for each class in $\mathscr{Y}_T$ and for each class outputs a confidence value again between 0 and 1.

4. Evaluation on Image Classification

The goal of our evaluations was to understand the impact of ontology-based knowledge on transfer learning in general and on the results obtained from our DAP-based architecture in particular. For comparability with state-of-the-art results, we opted for an image classification problem, reusing and extending the *aPascal-aYahoo* dataset and annotations[3] introduced for the first time in [18] and also reused in in [16]. This dataset consists of the union of a subset of the *PASCAL VOC 2008* dataset and of images that were collected using the Yahoo image search engine, popular for image classification tasks.[4] The dataset covers a wide range of people, animals, as well as artefacts such as vehicles, furniture, etc. We manually sampled this dataset for over 12,000 images relating to either of the two domains used in our experiment: the *household domain* with 7,490 images about everyday objects around the house, serving as our training set, and the *travel domain* with 5,203 images about holidays and travel-related objects, serving as our test set. As part of cross-domain transfer, we evaluated both *cross-task* performance with the prediction of *unseen classes* (i.e. not used in training), and *same-task* performance where the same classes are predicted across domains.

The evaluations covered three setups:

- a *SoA "Flat" Classifier* that implements the state-of-the-art DAP pipeline as described in [16], based on flat class–property lists without a hierarchy nor ontological design;
- a *Domain Ontology Classifier* that uses a source domain ontology for training and a target domain ontology for testing;
- a *Domain+TLO Classifier* that, in addition to using the same domain ontologies, also uses a top-level ontology.

Our goal with introducing the second setup—that lacks the TLO—was to get an idea of the effect of the TLO itself on the results.

4.1. Ontology Design

As basis for our classification task, we used the *PASCAL Visual Object Classes*[5] resource. While it is not itself an ontology, it is widely used in the image classification literature, and for cross-comparability with SoA results we decided to adopt it.

From this database we extracted 20 *base classes* that best describe the images from the source and target domains. Seven of the 20 classes belonged to the household domain, eight to the travel domain, and five were shared between the two domains. These base classes correspond to *basic-level categories* [19], such as *bicycle*, *cat*, or *boat*.

We then organized these domain-related classes and properties into two domain ontologies, by introducing superclass domain hierarchies following the principles in section 3.1. As reference material, we used the linked data and ontological analysis service *Ontobee*, and in particular ontologies such as FOODON and NCIT[6].

Based on these resources, we proceeded to build the two taxonomies, shown in Figure 3, as follows. As leaves, we created *base classes* (*Cat*, *Dog*, *Train*, etc., in green) that have a one-to-one correspondence with the original aPascal classes (that we call *aPascal-Cat*, *aPascal-Dog*, etc., in white). As the aPascal classes did not respect the principle of discrimination (for example, the property set of *aPascal-Train* was a subset of that of *aPascal-Boat*), wherever it was necessary we extended the "legacy" aPascal attribute sets

---

[3] `http://vision.cs.uiuc.edu/attributes/`

[4] `http://www.pascal-network.org/challenges/VOC`, `https://images.search.yahoo.com`

[5] `http://host.robots.ox.ac.uk/pascal/VOC/voc2005/index.html`

[6] `http://purl.obolibrary.org/obo/FOODON_03490214`, `www.ontobee.org/ontology/NCIT`

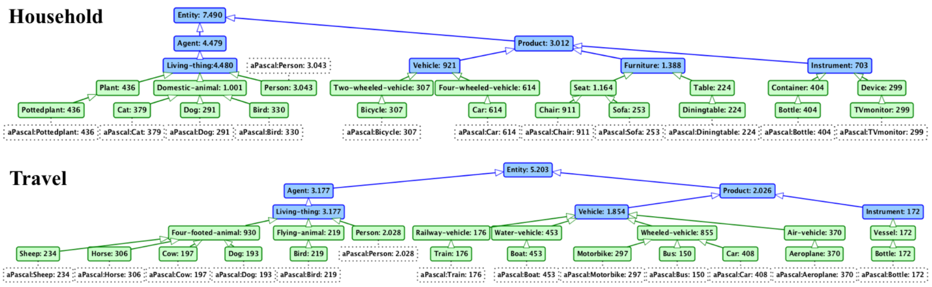

Figure 3. Selected source (training) and target (test) domains, with the number of corresponding images.

with definitional properties[7] on the base class level (e.g. *Train* had all properties from *aPascal-Train* as well as the new definitional property *has-locomotive*). Note that the property sets of base classes shared across the two domains (e.g. *Car* or *Dog*) typically overlap only partially, as it would be the case with real-world domain ontologies.

Then, continuing in a bottom-up manner, we created new levels of domain-specific superclasses based on the domain hierarchies retrieved using *Ontobee*, each time introducing discriminative properties as necessary. We call these *mid-level classes*. Finally, we introduced three layers of *top-level classes* shared across the two domains (in blue in Figure 3): the root class *Entity*, then *Agent* and *Product*, and finally *Vehicle*, *Instrument*, *Living-thing*, and *Furniture*. The ontologies were built *a priori* and then unchanged during the experiments.

Thus, the final *household* domain ontology contains 27 classes, the *travel* ontology 26 classes, and the shared top-level ontology seven classes. The 63 properties reused from Pascal VOC were completed by 20 domain and 8 top-level properties, the total number of properties reaching 91. We did not organize the properties into a hierarchy and thus did not rely on domain abstraction for our results, which we leave as future work.

4.2. *Predictor Setup*

We have implemented all three components of the DAP pipeline (Figure 1) for all three setups using the *RapidMiner*[8] machine learning framework.

Feature Extractor. For this component, in all three setups we relied on the process described in [18]. For each image, we reused the precomputed *color*, *texture*, *edge orientation*, and *HoG* features that the authors extracted from the bounding boxes of the objects (as provided by the PASCAL VOC annotation) and released as part of the dataset. We deliberately used the same features for cross-comparability with works such as [16] that reuse the same datasets. The feature output was then fed into the Property Predictor.

Property Predictor. For the SoA Flat Classifier, we trained the Property Predictor on the original 63 *aPascal* properties that characterize shape, material, and the parts of the visible object. (For instance, images about motorbikes were annotated with properties such as *"plastic"*, *"metal"*, or *"engine"*.) For the Domain Classifier, this set was com-

---

[7] As a theoretical reference we followed [20]; in particular, we took inspiration from the notion of *rigid properties*. However, we did not carry out a deep ontological analysis for each class. Our major objective was to identify, for each class, properties that were *essential*, i.e. always present for all instances in the dataset.

[8] `http://www.rapidminer.com`

pleted by 20 discriminative domain properties, and for the Domain+TLO classifier with the extra eight TLO properties as well. As one of the contributions of this paper, we publish the annotated dataset free for research purposes.[9] The annotated dataset was then divided according to the two domains: the *household* domain data set (7.490 images) served as training data and the *travel* domain (5.203 images) served as test, allowing us to examine cross-domain behavior. As seen in Figure 3, the two domain classes overlap only partially: seven and eight classes are specific to the household and the travel domain, respectively. Using the annotations, we trained a *k-Nearest-Neighbor* classifier for each property (this method provided a good balance between performance and speed).

Class Predictor. We implemented the Class Predictor as a multi-label classification task. We again used *k-Nearest-Neighbor* classifiers in all three setups, which we trained on class–attribute associations extracted from the two ontologies using the *Class Generalization Rule Extractor*. For the SoA Flat classes, only the original aPascal classes and properties were used. The Domain Classifier was trained on all source and target domain ontology classes (base and mid-level) and all inherited properties. The Domain+TLO Classifier, in turn, was trained on all of the above plus the top-level classes and properties.

4.3. Experiments and Results

Our experiments measured classification performance with respect to two transfer learning scenarios: a *same-task cross-domain* and a *cross-task cross-domain* scenario. The test set of images was run through the three DAP pipeline setups using their respective property and class predictors.

Same-task cross-domain classification. Here, the goal was to classify unseen images into *seen* classes, i.e. that are defined in both source and target domains. The results, in terms of F-measure, are shown in Figure 4 for all five base classes shared between both domains: *Person*, *Car*, *Bird*, *Bottle*, and *Dog*. We also present results for the shared mid-level and top-level classes *Living-thing*, *Agent*, *Product*, *Vehicle*, and *Instrument*. The SoA Flat Classifier obviously could not provide any results for top-level and mid-level classes, as it does not use the entire ontologies. Similarly, the Domain Classifier did not have results for the two top-level classes as it does not use the TLO. In the case of base classes, where all three setups could be tested, the two ontology-based pipelines consistently reach higher scores, the difference with respect to the SoA running from moderate (4% for *Person*, 7% for *Car*), to very high (17% for *Bird*, 35% for *Bottle*). The exception is *Dog*, which all pipelines consistently misclassified as *Cat*. With respect to the TLO, only in the case of *Bottle* do we observe a major positive effect. It plays a more important role, however, for mid-level classes: the prediction of *Vehicle* and *Instrument* is improved by the TLO in a major way (by 15% and 45%, resp.). This also explains the improvement of the TLO on *Bottle*: it is the only *Instrument* class in the target ontology, and the latter is discriminated by a top-level attribute. For more general classes, the extra discriminative properties provided by the TLO have a clearly positive effect.

Cross-task cross-domain classification. Here, the goal was to classify unseen images into unseen classes, i.e. that are not part of the training household domain but are defined within the target travel domain ontology. The results on all eight unseen classes can be found in the right-hand side of Figure 4. Although to greatly varying degrees, all three

---

[9] `https://github.com/Matt-81/ontology-driven-TL`

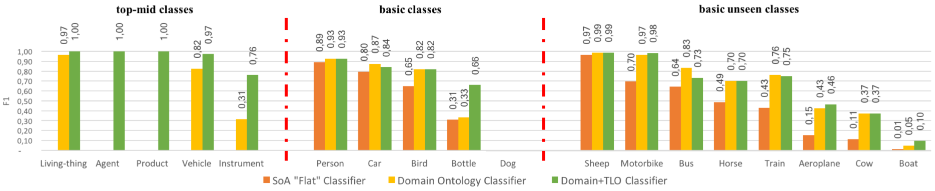

Figure 4. Prediction results (F1): same-task on seen top/mid (left) and base classes (middle), cross-task on unseen base classes (right), for the three classifier setups. Top-level results could only be obtained by the Domain+TLO classifier. Mid-level results were obtained by the two ontology-based classifiers.

Table 1. Overall accuracy over the entire dataset.

Classes	*SoA Flat Classifier*	*Domain Classifier*	*Domain+TLO Classifier*
base	**58.50%**	**72.84%**	**74.59%**
mid-level	–	**85.32%**	**95.46%**
top-level	–	–	**100%**

classifiers were able to detect all unseen classes, demonstrating the efficiency of Direct Attribute Prediction. As in the same-task scenario, improvements with respect to the SoA classifier were consistent, from 2% (for *Sheep*) up to 33% (for *Train*). When comparing the Domain and the Domain+TLO classifiers, the differences between the two are minor, with slight improvements for certain classes and slight deteriorations for others. Overall, we do not observe a major benefit from the TLO.

Overall accuracy. Finally, in Table 1 we report global accuracy results across the entire test dataset, for the three setups and on the three class levels (top, mid, base). On base classes, a major difference (+14.34%) is observed for the domain-ontology-based setup with respect to the SoA. Further improvement due to the TLO is moderate (+1.75%). The positive effect of the TLO is more obvious on the top and mid-level, where the TLO is clearly enabling high-precision and high-recall prediction. The TLO also seems to ensure a fully accurate coarse-grained distinction between *Agents* and *Products* (i.e. animate and inanimate objects).

Discussion. While we are careful not to draw overly general conclusions from experimenting with a single set of ontologies, we still observe a few salient phenomena. First of all, the application of the principle of discrimination, by making sure that every class is distinguished from its siblings by at least one property, had a major positive effect on the results, as shown by the improvements with respect to the SoA Flat Classifier. The effect of generalization through the TLO, on the other hand, was minor on base classes but of a remarkably high quality (95–100%) on mid-level and top-level classes. We attribute these results to the way our Class Predictor works: in the property sets of leaf classes, the few properties inherited from the top level had a relatively low weight with respect to the number of properties introduced on lower levels. A more efficient setup, e.g. based on *hierarchical learning* [21], would take into account the high-quality predictions on top-level and mid-level classes as input for base class prediction (so that, e.g. chairs are not classified as *Person* if they were given the top-level class of *Product*). We foresee this improvement of our setup as immediate future work.

5. Related Work

Our work is a follow-up to recent efforts that introduce formal or semi-formal knowledge into transfer learning setups and, in particular, *Zero-shot learning* [10,16]. These works, and the ones cited in the introduction, use simple knowledge organization schemes, such as flat class–attribute associations designed bottom-up, without following any theoretical principles. More recent work [22,9] also used class hierarchies, effectively relying on generalization, but still in an ad-hoc informal manner, without paying attention to how the hierarchy is organized. Some works focus on how to generalize information by reusing already existing *unstructured* domain information, usually extracted from natural language text such as image captions. [11] generates a taxonomy generalizing the information codified in the data populating the training set.

While in our experiments we also used custom built ontologies, our approach and motivations were different: our goal was to explore and demonstrate the effect of ontology design principles—generalization and discrimination—on transfer learning results. Exploiting well-established principles from *knowledge representation* and *ontological analysis* [23,7,13], we propose guidelines by which existing top-down ontological resources can be efficiently reused as knowledge sources for cross-domain transfer. In our experiment, the semantic information are not extracted from a text corpus and the resulting ontology is reusable for future tasks. A further contribution of our paper is the use of various abstraction operations (proposed earlier in [14] and further explored in [15]) in the aim of more fully exploiting the generalization ability of ontologies. Works on the role of discrimination and the subsumption relation in order properly to exploit a hierarchical class structure, such as [24,25], were also considered.

Finally, works focusing on bridging the gap between "high-level" knowledge and "low-level" (i.e. perceptual) information also overlap with our efforts [26]. The role of ontologies on image annotation and management has been studied in [27,28,29]. [30] and [31] explore the different yet complementary functions of knowledge-based *classification* and perception-based *identification*. While we share motivations with these works insomuch as we aim to integrate different forms of knowledge into a single system, our focus in this paper is specifically the transfer learning problem.

6. Conclusion and Perspectives

While our work is inscribed in the general line of recent efforts that aim at combining symbolic (top-down) and learning-based (bottom-up) knowledge, we also consider it as a starting point for the investigation of the effect of formal knowledge (e.g. ontology) design on the performance of the overall combined system. Based on our first encouraging results, we plan to extend our experiments to the evaluation of existing top-level and domain ontologies in terms of their reusability in the context of prediction tasks. This involves a theoretical research line that examines the impact of various aspects of formal ontology characteristics on prediction, as well as a very practical investigation towards an actual service that evaluates existing ontologies for their prediction ability.

Acknowledgement. This paper was supported by the *WeNet* project, funded by the European Union (EU) Horizon 2020 programme under grant number 823783.

References

[1] K. Weiss, T.M. Khoshgoftaar and D. Wang, A survey of transfer learning, *Journal of Big data* **3**(1) (2016), 9.

[2] S.J. Pan and Q. Yang, A survey on transfer learning, *IEEE Transactions on knowledge and data engineering* **22**(10) (2009), 1345–1359.

[3] W. Wang, V.W. Zheng, H. Yu and C. Miao, A survey of zero-shot learning: Settings, methods, and applications, *ACM Transactions on Intelligent Systems and Technology (TIST)* **10**(2) (2019), 1–37.

[4] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong and Q. He, A Comprehensive Survey on Transfer Learning, *arXiv preprint arXiv:1911.02685* (2019).

[5] M. Wang and W. Deng, Deep visual domain adaptation: a survey, *Neurocomputing* **312** (2018).

[6] G. Glavas, R. Litschko, S. Ruder and I. Vulic, How to (properly) evaluate cross-lingual word embeddings, *arXiv preprint arXiv:1902.00508* (2019).

[7] N. Guarino, D. Oberle and S. Staab, What is an ontology?, in: *Handbook on ontologies*, Springer, 2009, pp. 1–17.

[8] J. Chen, F. Lécué, J.Z. Pan, I. Horrocks and H. Chen, Knowledge-based transfer learning explanation, in: *Sixteenth International Conference on Principles of Knowledge Representation and Reasoning*, 2018.

[9] H. Kono, A. Kamimura, K. Tomita and T. Suzuki, Transfer learning method using ontology for heterogeneous multi-agent reinforcement learning, *International Journal of Advanced Computer Science & Applications* **5**(10) (2014).

[10] M. Palatucci, D. Pomerleau, G.E. Hinton and T.M. Mitchell, Zero-shot learning with semantic output codes, in: *Advances in neural information processing systems*, 2009, pp. 1410–1418.

[11] B. Liu, L. Yao, Z. Ding, J. Xu and J. Wu, Combining ontology and reinforcement learning for zero-shot classification, *Knowledge-Based Systems* **144** (2018), 42–50.

[12] Y. Xian, C.H. Lampert, B. Schiele and Z. Akata, Zero-shot learning—A comprehensive evaluation of the good, the bad and the ugly, *IEEE transactions on pattern analysis and machine intelligence* **41**(9) (2018), 2251–2265.

[13] N. Guarino, Understanding, building and using ontologies, *International Journal of Human-Computer Studies* **46**(2–3) (1997), 293–310.

[14] F. Giunchiglia and T. Walsh, A theory of abstraction, *Artificial intelligence* **57**(2–3) (1992), 323–389.

[15] F. Giunchiglia and M. Fumagalli, Teleologies: Objects, actions and functions, in: *International conference on conceptual modeling*, Springer, 2017, pp. 520–534.

[16] C.H. Lampert, H. Nickisch and S. Harmeling, Attribute-based classification for zero-shot visual object categorization, *IEEE transactions on pattern analysis and machine intelligence* **36**(3) (2013), 453–465.

[17] C.H. Lampert, H. Nickisch and S. Harmeling, Learning to detect unseen object classes by between-class attribute transfer, in: *2009 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, 2009, pp. 951–958.

[18] A. Farhadi, I. Endres, D. Hoiem and D. Forsyth, Describing objects by their attributes, in: *2009 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, 2009, pp. 1778–1785.

[19] E. Rosch, C.B. Mervis, W.D. Gray, D.M. Johnson and P. Boyes-Braem, Basic objects in natural categories, *Cognitive psychology* **8**(3) (1976), 382–439.

[20] N. Guarino and C. Welty, A formal ontology of properties, in: *International Conference on Knowledge Engineering and Knowledge Management*, Springer, 2000, pp. 97–112.

[21] J.V. Bouvrie, Hierarchical learning: Theory with applications in speech and vision, PhD thesis, Massachusetts Institute of Technology, 2009.

[22] Z. Al-Halah and R. Stiefelhagen, How to transfer? zero-shot object recognition via hierarchical transfer of semantic attributes, in: *2015 IEEE Winter Conference on Applications of Computer Vision*, IEEE, 2015, pp. 837–843.

[23] G. Guizzardi, On ontology, ontologies, conceptualizations, modeling languages, and (meta) models, *Frontiers in artificial intelligence and applications* **155** (2007), 18.

[24] N. Guarino and C. Welty, Identity and subsumption, in: *The Semantics of Relationships*, Springer, 2002, pp. 111–126.

[25] I. Johansson, Four Kinds of ‘Is A’ Relation, *Applied Ontology: An Introduction, Frankfurt: ontos* (2008), 235–254.

[26] S. Coradeschi, A. Loutfi and B. Wrede, A short review of symbol grounding in robotic and intelligent systems, *KI-Künstliche Intelligenz* **27**(2) (2013), 129–136.

[27] Z. Kuang, J. Yu, Z. Li, B. Zhang and J. Fan, Integrating multi-level deep learning and concept ontology for large-scale visual recognition, *Pattern Recognition* **78** (2018), 198–214.

[28] I. Donadello and L. Serafini, Mixing low-level and semantic features for image interpretation, in: *European Conference on Computer Vision*, Springer, 2014, pp. 283–298.

[29] D. Porello, M. Cristani and R. Ferrario, Integrating ontologies and computer vision for classification of objects in images, in: *Proceedings of the Workshop on Neural-Cognitive Integration in German Conference on Artificial Intelligence*, 2013, pp. 1–15.

[30] F. Giunchiglia and M. Fumagalli, Concepts as (Recognition) Abilities., in: *FOIS*, 2016, pp. 153–166.

[31] M. Fumagalli, G. Bella and F. Giunchiglia, Towards understanding classification and identification, in: *Pacific Rim International Conference on Artificial Intelligence*, Springer, 2019, pp. 71–84.