applied sciences

MDPI

Article

Dataset Constrution through Ontology-Based Data Requirements Analysis

Liangru Jiang and Xi Wang *

School of Computer Engineering and Science, Shanghai University, Shanghai 200444, China; liangruuu@shu.edu.cn
* Correspondence: wangxi@t.shu.edu.cn

Abstract: Machine learning (ML) technology is rapidly evolving, and the quality of ML systems is becoming an increasingly focal point of attention. Since the ML system is shaped by the dataset it learns from, its quality largely depends on the quality of the dataset. However, the dataset is often collected in a non-standardized process and few requirements and analysis methods are given to assist in identifying the needed dataset. This leads to no guarantee for the quality of dataset, affecting the generalization ability of model and resulting in low training efficiency. To address these issues, this paper proposes an ontology-based requirement analysis method where ontology integrates domain knowledge into the process of data requirements analysis and the coverage criteria on ontology are given for specifying data requirements which can later be used to guide the high-quality construction of the dataset. We held an experiment on an image recognition system in the field of autonomous driving to validate our approach. The result shows that the ML system trained by the dataset constructed through our data requirements analysis method has a better performance.

Keywords: machine learning system; scene graph generation; data requirement analysis; ontology; coverage criteria

Citation: Jiang, L.; Wang, X. Dataset Constrution through Ontology-Based Data Requirements Analysis. *Appl. Sci.* **2024**, *14*, 2237. https://doi.org/10.3390/app14062237

Academic Editor: Chilukuri K. Mohan

Received: 26 January 2024
Revised: 27 February 2024
Accepted: 4 March 2024
Published: 7 March 2024

1. Introduction

With the rapid development of ML technology, ML systems have become widely prevalent in various fields and even have been applied in some safety-critical domains, including autonomous vehicles, medical diagnosis, and aviation, where the reliability and accuracy of systems are of utmost importance. Therefore, how to control the quality and software trustworthiness of new-generation ML systems has become an urgent problem.

The quality of the ML system itself is largely determined by the training dataset. ML software learns and adjusts its models based on data to make intelligent decisions and places a strong emphasis on data-driven processes. Traditional software operates on static data and rules, with program logic manually written by developers. In this case, the dataset is merely seen as an input and does not directly impact the quality of the software itself. In contrast, ML systems require a large amount of data to train their models, enabling them to learn and understand various patterns and rules to make accurate predictions and decisions. By analyzing the data, the system can identify potential issues in algorithms and make improvements to enhance the performance and effectiveness of the system. In the meantime, it is quite difficult to change the network architecture to improve performance on the dataset, but the problem will be addressed in a more targeted way if the dataset can be designed [1]. Therefore, ensuring the quality of the dataset is crucial for the performance of the ML system.

However, the data collection process nowadays lacks standardized methods and guidance to help users obtain the requirements they want. The development of artificial intelligence systems nowadays mostly relies on benchmark datasets of the relevant domain. Some ML systems may be trained through customized datasets, but those datasets are often

collected randomly and there exists a large amount of invalid data and noisy data. To verify the validity of data, it is necessary to conduct experiments to repeat training experiments with different datasets and determine the appropriate dataset that fits in the corresponding ML system. This approach is not only time-consuming but also resource-intensive.

Some works have paid attention to the issues mentioned above. Some researchers gave guidance from an abstract level to allow users to assess the quality of datasets like [2,3], the majority of which focus on non-functional requirements for dataset of ML systems, such as performance, transparency, privacy, and security. Some papers focused on enhancing the defective dataset to offset the defect of the dataset in the process of data collection. However, most of them lacked the requirements analysis methods related to system theory knowledge and the behavior of the system. In fact, the purpose of selecting data is to determine the behavior of the system, so the relationship between data requirements analysis and system behavior should be considered. Additionally, when constructing data requirements related to system functionality, there are challenges such as dealing with unstructured data requirements and expressing semantic information accurately in complex scenarios. Hence, it is necessary to provide domain knowledge-based semantic and structured models for machine learning prediction models to guide users in obtaining comprehensive datasets containing various conceptual instances from real scenarios.

To incorporate semantic information and domain knowledge into the process of data requirements analysis, industry and academia referred to the concept of ontology to support data analysis and mining. Ontology is a formalized knowledge representation method. In contrast to semantic models like knowledge graphs, it not only provides rich semantic expression on domain knowledge but also supports constraint rules and logical inference, making the data requirements based on ontology unambiguous. Furthermore, there are different ontology languages available to define the semantic model, which makes it convenient for generating data requirements. Once ontology is introduced, data requirements specification can be integrated into the context of the target ML system to align the process with the expected system's behavior and to facilitate the process of data collection more accurately and effectively. However, most research on data requirements analysis only considered whether the data in the dataset contain concepts in ontology, without including the potential relationships that may exist between concepts in complex scenarios.

To address the shortcomings of existing studies, we propose a data requirements analysis method based on an ontology that provides coverage criteria for guiding the specification of data requirements. These criteria consider the coverage of elements such as entities and relationships in ontology, enabling our method to obtain a training dataset with higher quality compared to previously proposed requirements analysis methods. The method mainly contains two critical techniques: *ontology-based domain model* and *criteria-based data requirements specification*. The former technique presents an ontology-based domain model that describes the knowledge of concepts and relationships between concepts in the domain of the ML system and provides a formal representation, allowing us to automatically apply and analyze the domain model. The latter guides users to specify the data requirements based on two criteria: *attribute coverage criterion* and *relation coverage criterion*, both of which can guide users in covering crucial elements to affect learning behavior and derive effective data requirements. The data requirements generated based on our method are formalized and structured, which enables us to analyze it automatically, thereby providing automated services for building datasets.

To validate the effectiveness of the proposed method in this paper, we conducted experiments on an image recognition system in traffic scenes. We trained the model using a dataset that satisfies the data requirements identified through our proposed method and separately trained the model using a candidate dataset collected randomly [4]. Subsequently, we compared the performance of the models trained with these two datasets. The results demonstrate that the model trained with the dataset collected through data requirements generated by our proposed method exhibits improved predictive performance compared to the model trained with the candidate dataset collected randomly.

The candidate dataset are available at https://github.com/csust7zhangjm/CCTSDB2021 (accessed on 24 September 2023).

It should be noted that our method does not discuss how to construct a domain model completely or use ontology to describe the complex logic of ML systems. The more complete the given domain model is, the more effective the method proposed will be. The focus of our discussion is how to utilize the semantic information through the given domain model effectively and assist users in generating effective data requirements. The more comprehensive and accurate the domain model is, the more potent the impact of our proposed method will be in guiding the generation of data requirements for ML systems.

The remainder of this article is organized as follows: Section 2 discusses the related work. Section 3 describes a data requirements analysis method and takes the traffic scene as an example. Section 4 details the design of the experiment and the experimental results. Section 5 summarizes the whole paper.

2. Related Work

ML systems require a significant investment to gather sufficient data encompassing various conditions. Therefore, the majority of research teams utilize open datasets for their experiments. For example, HOTPOTQA [5], BDD100k [6], COCO [7], etc., have been successful open datasets in the field of NLP, image processing and autonomous driving for a long time. While public datasets provide researchers with the convenience of accessing data, it is essential to address their limitations in practical and engineering applications. Most open datasets aim to provide well-synchronized, denoised, and ready-to-use data but are reckless in publishing the details of their hardware configurations and open-sourcing the developing tools, which causes problems for other researchers to create the dataset they need. These researchers tend to customize the collection of datasets based on their own conditions and needs with the risk of low-quality data in the dataset. Thus, some studies focused on improving the quality of datasets in ML systems through the process of collecting. Gupta et al. [8] built a tool capable of detecting, interpreting, and remediating problems in data and automatically capturing all changes applied to the data, which can reduce turnaround time in the data preparation pipeline and simplify the data quality assessment process. Some researchers aimed to preprocess the dataset after it had been collected. Pan et al. [9] adopted a multi-strategy filtering and text augmentation algorithm based on the pre-trained model to increase the size and quality of the dataset. Refs. [10–12] performed enhancement operations such as translation, rotation, noise addition, etc., on the collected data to increase the robustness of the dataset. Yao et al. [13] formulated noisy textual queries removing and noisy images filtering as a multi-view and multi-instance learning problem separately to collect diverse and accurate images for given queries from the Web.

Data requirement analysis is an effective way to ensure the quality of the dataset as well. Refs. [14,15] attempted to analyze data from the perspectives of balance and fairness. Zhang et al. [16] presented a data-driven engineering process as a new systematic and structured requirement analysis approach for leveraging the future applications of ML in the industry. Some studies have provided some tools to supply the data requirement analysis, such as [17] which defined a model-driven engineering (MDE) method using the UML semi-formal modeling language for the analysis of dataset structural requirements. However, the above papers did not systematically model the semantics of data in complicated scenarios, failing to address the lack of depth and breadth of data.

To address these issues, knowledge engineering is increasingly being applied in the construction process of knowledge graphs and graph data. It can provide original structured knowledge data for intelligent systems and extract semantic information to enhance the datasets used for training. Such systems not only obtain domain knowledge from public data sources (e.g., Wikipedia, Freebase, etc.) but also integrate and link data from the internet (e.g., Semantic Web, Linked Data, etc.). Up to now, research on aspects such as data cleaning [18] and data profiling [19] have mainly focused on relational data,

making it difficult to be directly applied to knowledge graphs. This is caused by the issue that, in contrast to relational data, knowledge graphs can be applied without the need for strict tabular structures or predefined schemas and do not need to be strictly normalized. Therefore, the first step to ensure the quality of data is to extract the schema from knowledge graphs and check for consistency.

Ontology is a formal method of representing knowledge. There exist standards, such as OWL (https://www.w3.org/TR/owl2-overview/, accessed on 16 May 2023) and RDFS (http://www.w3.org/TR/rdf-schema/, accessed on 16 May 2023), to define the ontologies and schema constraints for the knowledge graphs modeled in RDF. Ontology provides a rigorous semantic model that supports formalized reasoning and logical inference, and it can be used to assist in requirements analysis. In the data preparation phase, ontologies can be used to improve data quality, reduce ambiguity, and achieve interoperability between different datasets. Some papers try to bring ontology into the requirements analysis phase. De Coste et al. [20] proposes a hybrid modeling framework combining Mid-winter breakups(MWBs) ontology with image recognition, which allows an ontology to define and analyze key data, events, and relations in the ice season to reduce the prediction error of time-split events. Asudeh et al. [21] provides efficient techniques for traversing all value combinations to evaluate the coverage of multiple classification attributes in the given dataset based on the ontology of diamond. However, most of the applications using ontology primarily considered whether the data in the dataset contained concepts in the ontology model, without integrating these concepts with the behavior of the system. This resulted in the inability to effectively provide semantic information for complex scenarios and satisfy corresponding requirements by leveraging ontologies. This article proposes two coverage criteria based on entity attributes and relations between entities to filter the candidate dataset by constructing a domain model.

3. Ontology-Based Approach to Data Requirements Analysis

Since the performance and accuracy of an ML system directly depend on the quality and quantity of data used, data requirements analysis is crucial for improving the quality of an ML system. Data requirements refer to the description of data needed to achieve the expected training effect of the target system. The clear definition of data requirements helps ensure that the data obtained can satisfy specific analysis, modeling, or decision-making needs. Assuming that a company aims to develop an image recognition system for identifying product defects on the factory production line, the importance of data requirements for ML systems becomes evident in the following aspects:

1. **Data Labeling**: Data requirements can help determine the types of labeled data the system needs to collect, such as the category, location, size, and other information about the products, which can guide the data labeling work, ensuring that the system receives accurate labeled data to support the training of machine learning models.
2. **Dataset Diversity**: Data requirements ensure that the system accepts diverse data, including various types of products, or videos captured under different lighting conditions, and images of products from different angles and scales, which can help ML systems better adapt to various real-world scenarios.
3. **Feature Engineering**: Data requirements can help to select and extract the most relevant features, thereby improving the performance of the system.

Due to the significant impact of data on the behavior of the system, data are closely related to the domain in which the system operates. The various complex conditions within the domain make the data possess complex semantics. Therefore, we introduce the theory of ontology to describe the complex semantics of the domain within the specific system in order to clearly articulate data requirements. The introduction of ontology to some extent alleviates the difficulty of describing semantic complexity because ontology, as a carrier of domain knowledge, provides a structured representation method to represent semantic information as a collection of concepts and relationships.

Based on ontology, this paper proposes a meta-model for the data requirement analysis pipeline process as shown in Figure 1. The meta-model consists of four layers: ontology-based domain model, criteria-based data requirements specification, dataset construction and model training and deployment. In general, data sources are processed in the following order. First, a semantic model for exploring problem space needs to be constructed at the domain modeling layer. The semantic model refers to theoretical knowledge from ontology and is used to describe entities, attributes and their relationships. Evaluation metrics for the problem space are provided, such as accuracy, RMSE, mAP, while some non-functional requirements, such as reliability, fairness, and security are also included. Then, it is determined whether the model files provided by users can be parsed by a supporting tool. The metadata files will be generated directly if possible, otherwise, they are manually created through the supporting tool and exported.

The data requirements specification layer imports the metadata files and generates corresponding data requirements based on the proposed coverage criteria and algorithms. This paper proposes two criteria *attribute coverage criterion* and *relation coverage criterion*, each of which deals with different categories of elements in the domain model, where the elements serve as the foundation for improving the dataset quality and constructing data requirements. The former aims to cover entities and the corresponding value of attributes, the latter aims to cover the relationships and entities.

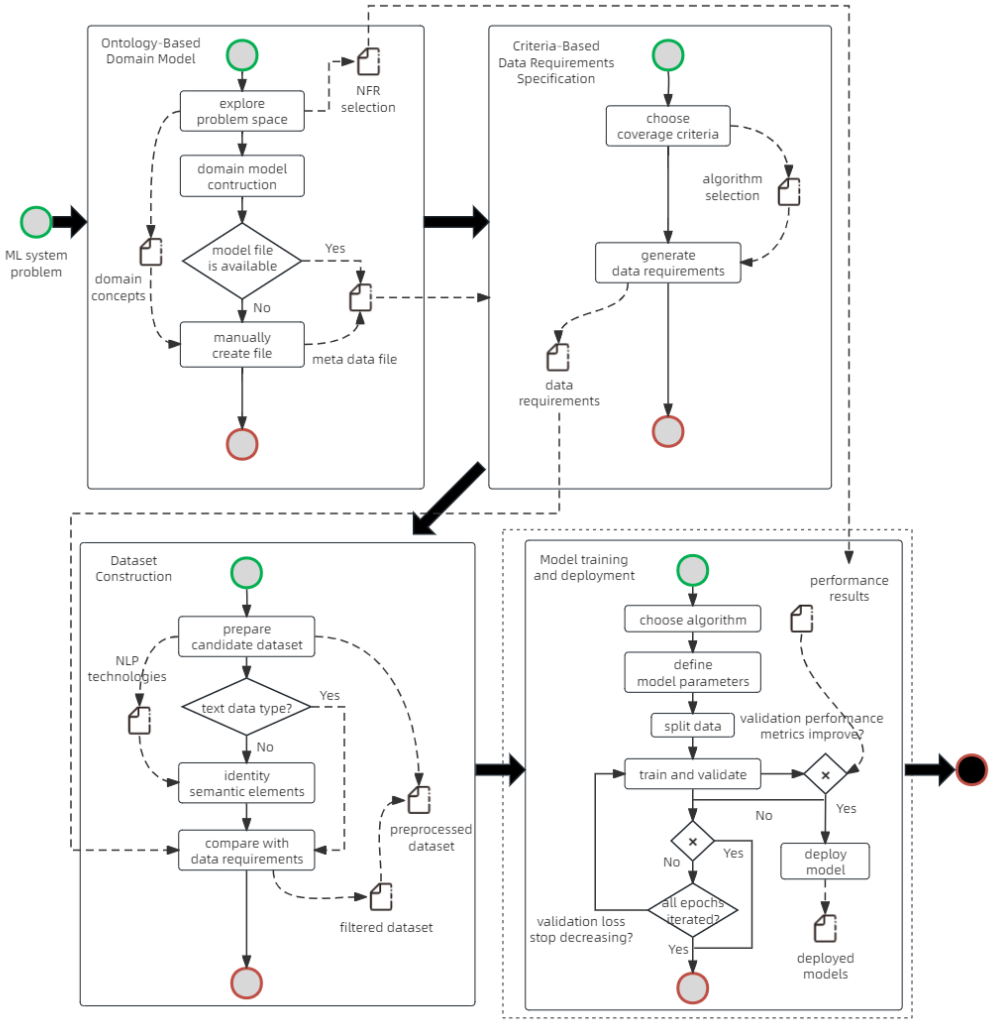


Figure 1. The process meta-model of ontology-based data requirements analysis approach.

The dataset construction layer first obtains the user's initial data set and determines whether the type of data is text. If so, the candidate dataset is directly compared with the data requirements obtained from the previous layer. If not, the data need to be processed into triples using NLP techniques before being compared with the data requirements. The data satisfying the requirements will be filtered into the preprocessed dataset and output for the model training and deployment phase. The stage of model training and deployment involves combining analysis with non-functional requirements (*NFR*) scores and weights and selecting appropriate algorithms and parameters for each ML model based on NFR. The model will be trained using a preprocessed dataset to generate learning models and deployed.

3.1. Ontology-Based Domain Model

The behavior of ML systems is closely related to domain knowledge and data. Therefore, specific data requirements can only be determined after modeling domain knowledge. Ontology is a domain modeling method we used, it can structure domain knowledge represented in natural language into a series of symbols to provide the basis for requirements analysis and present it in a graphical form. In order to obtain requirements for the dataset automatically and conduct requirements analysis, we need to first give formal definitions of the semantic model within domain knowledge. Our domain model draws on the concepts of entities and relationships in ontology, and its formal definition is as follows:

Definition 1 (Domain Model). *A domain ontolgoy is a tuple* $(C, Label, \varphi)$*, such that:*

- *C denotes a set of concepts*
- *Label: {is, has} denotes a set of relation names*
- φ*: C* $\times$ *Label* $\rightarrow$ $P(C)$ *denotes the relations among concepts where* $\varphi(c, l)$ *indicates a set of concepts related with concepts c through label l.*

A domain model reflects the domain knowledge in the physical world, where each node represents a concept in the domain, and each pair of nodes is connected by an edge, representing the connection between concepts in a physical scene. In contrast to considering the edge as a relationship between two nodes in ontology, we establish a distinct type of node for the concept of relations due to the critical impact of relationships between entities on the behavior of ML systems. The relation node simultaneously connects to two nodes representing the entities in the physical world. The edges between nodes are then classified into two labels based on the subordinate relationship between the nodes on either side of the edge. The label *has* indicates a subordination relation between concepts, while the label *is* indicates an implementation relation between concepts.

According to the above definitions, we establish a domain model with the example of the recognition or detection system in an autonomous driving system. Figure 2 shows the traffic participants, environment, and other concepts in the traffic road scene, along with their corresponding value of attributes and relationships. Each concept is connected by a labeled edge. We take the node of *Tree* as an example, the *Tree* is connected with the nodes $leaf$ and *position* by an edge labeled *has*, because the concepts $leaf$ and *position* have subordination relations with the concept *Tree*. The nodes *floush and green* and *withered and yellow* are connected with the node $leaf$ by an edge labeled by *is* because of the implementation relations between them and the concept $leaf$. The set $\varphi(Tree, has)$ contains nodes *position* and *leaf*, while the set $\varphi(Tree, is)$ is empty.

Although they are all considered individual physical nodes in the graph, the semantic information they represent may differ, and there may even be semantic relationships such as subordination and implementation between them. Therefore, based on the different characteristics of concepts, the node in the domain model is divided into four categories, including *entity*, *relation*, *attribute*, and *value*. These categories can be real-world entities, as well as abstract concepts, properties, or relationships. Next, we will provide detailed explanations of these four different categories.

Nodes of *entity* denoted as E_O, represent the physical entities in the real world. Each element e has only an outdegree and no indegree in the domain model and has at least one edge labeled as *has* started with itself. Mathematically, *entity* nodes need to satisfy the following expression:

$$\forall e(deg^+(e) = 0 \wedge deg^-(e) \geq 1) \tag{1}$$

where $deg^+(n)$ represents the indegree of node, and $deg^-(n)$ represents the outdegree of node. The node *trafficSign* shown in Figure 2 is directly related to the nodes *position*, *shape* and *color* with edges labeled by *has*, while $deg^+(trafficSign) = 0$ and $deg^-(trafficSign) = 3$. In addition to the node of *trafficSign*, all of the nodes include *pedestrian*, *trafficLight*, *Tree*, *Vehicle* and *Enviroment* in the domain model satisfy the expression provided.

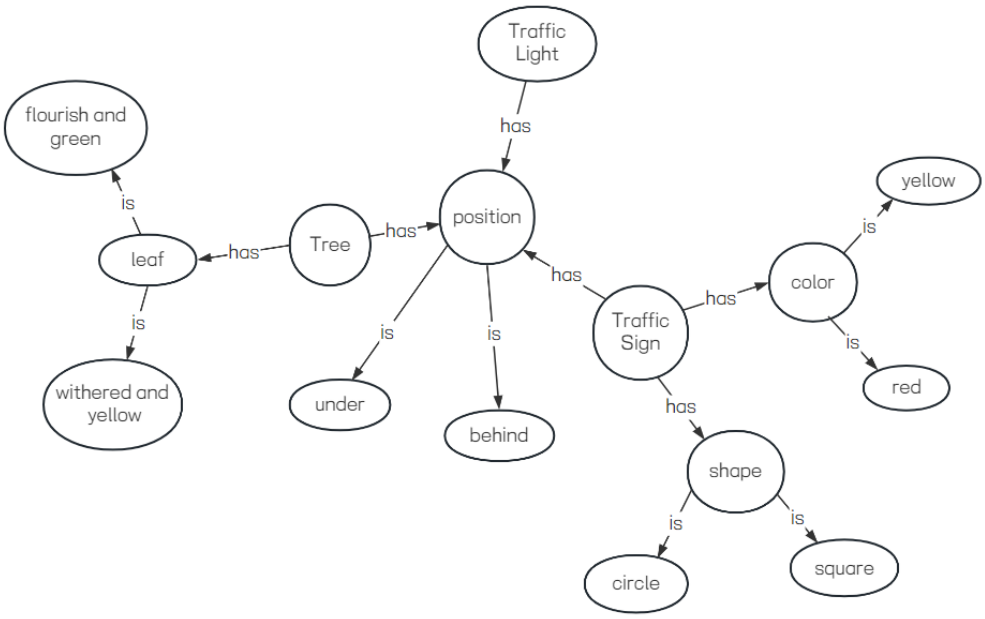


Figure 2. Part of the ontology-based domain model towards traffic scene.

The *relation* nodes represent the relations between two entities, which can be denoted as R_O in the domain model, *relation* nodes are a list of nodes, each of which has more than one edge labeled as *has* ended with itself and multiple edges labeled as *is* starting from itself, each relation ontology node r in R_O is connected to two *entity* nodes through two edges and should satisfy the expression as follows:

$$\forall r(r \in \varphi(c, has) \wedge deg^+(r) > 1 \wedge deg^-(r) \geq 1) \tag{2}$$

where c represents an element of C. There are two *relation* nodes shown in the domain model including *position* and *do*. The indegree of both are more than one and the outdegree of both are at least one. For example, there exist several edges labeled by *has* among five different *entity* nodes including *pedestrian*, *trafficLight*, *Tree*, *Vehicle* and *trafficSign*, which leads to the indegree being more than one. There also exist two edges labeled by *is* among node *position*, node *behind* and node *under*.

The attribute nodes represent the features of a physical entity which can be denoted as A_O in the domain model, each of which has only an edge labeled as *has* starting from itself and multiple edges labeled as *is* ending with itself, an entity node is linked to several attribute nodes. Each attribute node has several mutually exclusive value nodes V_O; the value node represents the value of the elements, which has an edge labeled as *is* started

from an attribute node or a relation node. Mathematically, the attribute nodes and value nodes should satisfy the expression as follows:

$$\begin{aligned} &\forall a(a \in \varphi(c, has) \wedge deg^{+}(a) = 1 \wedge deg^{-}(a) \geq 1), \\ &\forall v(v \in \varphi(c, is) \wedge deg^{+}(v) = 1 \wedge deg^{-}(v) = 0). \end{aligned} \tag{3}$$

just as in Figure 2, the entity node *trafficSign* is linked with attribute nodes *color*, *content* and *shape*, which are the attributes of *trafficSign*. For attribute node *shape*, there exist value nodes *circle* and *square* linked with it. The relation node *position* is linked with the value nodes *under* and *behind*. It is important to note that the examples shown in the figure cannot fully explain the complete road traffic scene. We have simplified the entities, attributes, and values for the convenience of explaining the criteria we propose and the data requirements generated from them.

The strength of relationships between concepts in domain knowledge is an important factor to consider in requirements analysis. Entities that are closer to the current entity have a greater influence on the current entity, while entities that are further in relation have less impact on the prediction of the current entity. Therefore, we assess the strength of the relation between two nodes by measuring the distance between them in the domain model, which indicates whether the nodes are directly related.

From a semantic perspective, two entity nodes are directly related if and only if they are jointly connected to a relation node. For example, entity node *pedestrian* is directly related to relation node *position* because there exists an edge between them. Similarly, entity node *trafficSign* and entity node *Tree* are directly related to each other because they are both connected to relation node *position*. To facilitate the automation, we introduce the definition as follows:

Definition 2 (Directly Related). *Given two entity nodes e*1 *and e*2*, they are directly related if and only if both of them connect to the same relation node. The expression is given as follows:*

$$\forall (e1, e2) \in E_O,\ \varphi(e1, has) \cap \varphi(e2, has) \neq \emptyset \tag{4}$$

In addition, we use $F(c)$ to represent the set of concepts connected with concept c. Especially, the entity nodes directly related to e are denoted as $F_{entity}(e)$ and the relation nodes connected to e are denoted as $F_{rel}(e)$, both of which need to satisfy the following equations.

$$\begin{aligned} F_{rel}(e) &= \varphi(F(e), has) \\ F_{entity}(e) &= F(e) - F_{rel}(e) \end{aligned} \tag{5}$$

3.2. Criteria-Based Data Requirements Specification

Data requirements are a series of conditions that a dataset needs to satisfy. They serve as the standard for validating the dataset and a guide for establishing the dataset. Given a set of data requirements *Req* derived from criteria provided subsequently, these requirements can offer standards to cover crucial elements when we collect datasets. To facilitate future automated comparison operations, we can build the dataset D by satisfying the condition specified as follows:

$$\forall req \in Req,\ \exists d \in D(d \doteq req) \tag{6}$$

where $\doteq$ means that the data should satisfy the given data requirements.

The generation of data requirements requires consideration of many factors. The behavior of a ML system is influenced by the characteristics of entities, including their inherent attributes and relations with other entities. Our approach is based on ontology which contains many elements. Therefore, it is necessary to cover the crucial elements in the ontology when constructing the data requirements for collecting the dataset for training. To standardize the data requirements generating process, we propose coverage criteria

based on the structure of the domain model; each criterion will generate corresponding data requirements. The criteria serve as guiding principles for generating requirements that satisfy specific coverage standards. Specifically, we have provided two types of coverage criteria to cover attribute elements and relation elements, respectively.

Attribute Coverage Criterion. Given a domain ontology O and a set of crucial entity concepts $E_O^{'}$, for each value combination of attributes of entities in $E_O^{'}$ and the entity nodes directly related to an entity in $E_O^{'}$, there must exist at least one data point in the dataset to cover it.

The attributes of the data directly influence the training processes of the ML system. The attributes and different combinations of attributes of an entity can provide different information to affect the system's understanding of data. Additionally, the attributes of entities directly related to the target entity will also affect the learning results of the system. *Attribute Coverage Criterion* aims to determine data requirements by covering the combinations of values of attributes in the target entity and entities directly related to it. The requirements generated based on the *Attribute Coverage Criterion* are formulated as follows:

$$req_o^{attr} = \{(a, v) | v \in F_{val}(a) \wedge a \in F_{attr}(c) \wedge c \in E_O^{'} \cap F_{entity}(e_o)\} \quad (7)$$

where Fval(a) represents all the value nodes directly related with attribute node a; Fattr(c) represents all the attribute nodes directly related with concepts c. The criterion-based data requirement is a collection of pairs, where each pair consists of a and v, which represent the semantic information of the attribute and its value. To illustrate the attribute coverage criterion, we are going to analyze the data for the recognition system in the context of autonomous driving to generate the data requirements shown in Table 1 and assume that we want to recognize traffic signs. This criterion ensures that all semantic entities in each data can be described by a series of attribute coverage lists.

Based on the definition of directly related, we can see in Figure 2 that the entity nodes directly related to *TrafficSign* include *Tree*, *Vehicle*, *Pedestrian* and *TrafficLight*. The *Tree* has an attribute called *leaf* with two values *withered and yellow)* and *flourish and green*, which semantically represent leaves in spring and summer, and leaves in autumn and winter, respectively. The traffic sign has two attributes *shape* and *color*, each with its corresponding values, which semantically represent different types of signs and the characteristics exhibited by signs in complex scenes. We combine the corresponding values of each attribute to form the pair of (a, v), such as $\{(color, \text{red}), (shape, \text{circle})\}$. After obtaining a series of entities directly related to $trafficSign$ and their attributes with values, we can obtain all the data requirements based on req_o^{attr} as shown in Table 1.

Table 1. Data requirements based on *Attribute Coverage Criterion* towards ontology of traffic scene.

Crucial Entity	Attribute	Value	Req_o^{attr}
TrafficSign	shape	circle	{{(*shape*, circle), (*color*, red), (*leaf*, flourish)},
		square	{(*shape*, circle), (*color*, red), (*leaf*, withered)},
	color	red	{(*shape*, circle), (*color*, blue), (*leaf*, flourish)},
		yellow	{(*shape*, circle), (*color*, blue), (*leaf*, withered)},
Tree	leaf	flourish	{(*shape*, square), (*color*, red), (*leaf*, flourish)},
		withered	{(*shape*, square), (*color*, red), (*leaf*, withered)},
			{(*shape*, square), (*color*, blue), (*leaf*, flourish)},
			{(*shape*, square), (*color*, blue), (*leaf*, withered)}}

Relation Coverage Criterion. Given a domain ontology O, a set of crucial entity concepts $E_O^{'}$ and the entity nodes directly related to an entity in $E_O^{'}$, for each possible entity combination of pair (e_{o1}, e_{o2}) and the value of relation concept r related to e_{o1} and e_{o2}, there must exist at least one data point in the dataset to cover it.

Relation Coverage Criterion aims to determine data requirements by covering multiple entities with their relations. The data requirements generated based on the Relation Coverage Criterion need to be formulated as follows:

$$req_o^{rel} = \{(e_{o1}, v, e_{o2}) | v \in F_{val}(r) \wedge r \in F_{rel}(e_o) \wedge (e_{o1}, e_{o2}) \in E_O^{'} \cap F_{entity}(e)\} \quad (8)$$

where $F_{val}(r)$ represents all the value nodes directly related with relation node r and the combination is denoted as a triple (e_o, v, e_o). As shown in Figure 2, we can see that the relation node *Position* is linked with the target entity node *TrafficSign* and entity nodes linked with *Position*, such as *Tree*, *Vehicle*, *Pedestrian*, and *TrafficLight*. We can obtain all the data requirements in the form of three-triples about the entity *trafficSign* based on req_o^{rel} as shown in Table 2.

Table 2. Data requirements based on *Relation Coverage Criterion* towards ontology of traffic scene.

Critial Relation	Value	Critial Entity	Req_o^{rel}
position	under behind	*TrafficSign*	{{(Tree, under, Traffic Sign)}, {(Tree, behind, Traffic Sign)},
		Tree	{{(Vehicle, under, Traffic Sign)},
		Vehicle	{{(Vehicle, behind, Traffic Sign)},
		Pedestrian	{{(Pedestrian, under, Traffic Sign)},
		TrafficLight	{{(Pedestrian, behind, Traffic Sign)},
			{{(Traffic Light, behind, Traffic Sign)},
			{{(Traffic Light, under, Traffic Sign)}}

It should be noted that the shape of a red-colored sign is generally circular in reality, but it may also appear in different combinations due to factors such as lighting, weather, and the occlusion of obstacles. During the phase of data requirement construction, we only need to combine the values of attributes. Whether the combination is in line with the actual situation needs the participation of domain experts.

Since the criteria are formalized, the algorithm for generating corresponding data requirements based on criteria is summarized in Algorithm 1.

The provided example semantic model is only partial. As the relevant field evolves, new domain-related elements may need to be introduced. Analysts and domain experts should maintain these semantic models to ensure that they provide up-to-date and advanced information for specific model construction. Next, we will conduct experiments based on criteria to illustrate the mapping relation between the recognition accuracy of semantic objects in images and their corresponding system. The research aims to provide engineers with a foundation for describing the learning behavior of ML systems. Specifically, it aims to identify which features of data lead to the improvement or decline in the capability of the system. This enables the construction of data requirements based on elements that can enhance the system's capability and facilitates the matching and acquisition of high-quality datasets based on these data requirements.

Algorithm 1: Data Requirements Generate.

Input : domain ontology O, entity nodes $\{e_{o1}, e_{o2}, \ldots, e_{on}\}$
Output: the data requirements Req_o^{attr} and Req_o^{rel};

$GenerateGraph(O)$;
$Req_o^{attr} \leftarrow []$;
$Req_o^{rel} \leftarrow []$;
foreach e_o *in* $\{e_{o1}, e_{o2}, \ldots, e_{on}\}$ **do**
  /*Step I: requirements based on Attribute Coverage Criteria*/
  $A_O \leftarrow F_{attr}(e_o)$;
  $req_o^{attr} \leftarrow []$;
  $GenReqBasedACC(A_O, req_o^{attr}, 1)$;
  /*Step II: requirements based on Relation Coverage Criteria*/
  $req_o^{rel} \leftarrow []$;
  for $i = 1$ *to* $len(F_{rel}(e_o))$ **do**
    $val \leftarrow F_{rel}(e_o)[i]$; **for** $j = 1$ *to* $len(F_{entity}(e_o))$ **do**
      $req_o^{rel}.append([e_o, val, F_{entity}(e_o)[j]])$;
    end
  end
  $Req_o^{rel}.append(req_o^{rel})$
  return $[Req_o^{attr}, Req_o^{rel}]$;
end

Function `GenReqBasedACC`$(A_O, req_o^{attr}, start)$**:**
  if $len(req_o^{attr}) = len(A_O)$ **then**
    $Req_o^{attr}.append(req_o^{attr})$;
  else
    for $i = start$ *to* $len(A_O)$ **do**
      $V_O \leftarrow F_{attr}(A_O[i])$;
      for v_o *in* V_O **do**
        $req_o^{attr}.append(v_o)$;
        $generate(A_O, req_o^{attr}, start + 1)$;
        $req_o^{attr}.removeLast(v_o)$;
      end
    end
  end
End Function

4. The Supporting Tool

It is essential to determine the candidate dataset and the data types, which can be numerical, textual, or image-based. Additionally, it is important to identify the fields that need to be extracted from the dataset. If the data type is image-based, the semantic information in images needs to be converted into text or numerical form for comparison with the constructed data requirements. Based on the data requirements, we can analyze and filter the existing candidate dataset. Items in the candidate dataset that do not satisfy the data requirements will be documented in a missing report. This report will serve as a reference for future stages, where these missing items can be supplemented and compensated for. This process ensures that the dataset is comprehensive and accurate, providing a solid foundation for analysis and decision-making.

Our supporting tool primarily provides the function of image-based dataset construction, which is mainly divided into three components, as shown in Figure 3.

First is the domain model analysis module, which can standardize the domain model files into a row data list composed of multiple concepts. The second is the data requirements specification module, which generates a series of data requirements based on criteria. The third is the dataset construction module, where the candidate dataset is filtered to determine if it satisfies the output of the data requirement specification modules. The result of filtering

and data requirement construction can guide the generation of missing reports to address any deficiencies in the dataset within this module. This module is implemented differently depending on the system task, and we have specifically constructed an image dataset for image recognition tasks in traffic scenarios. The user interface is mainly developed based on these three modules and provides image display functionality, which consists of a main image and several thumbnails. Users can switch to display different images by clicking on the thumbnails, and they can also switch between displaying the original image and the annotated image by clicking on buttons.

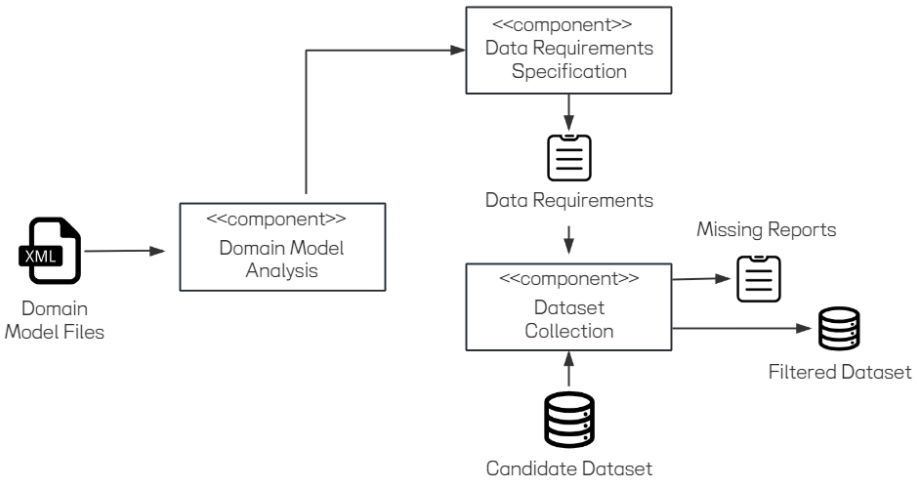


Figure 3. The design of data requirement analysis supporting tool.

To extract semantic information from image data, we use an anchor-based object detection technique proposed in the field of computer vision. The detector calculates the probability of each anchor box generated at the beginning of the procedure. By using standard metrics, such as Intersection over Union (IoU), which measures the overlap between the anchor box and the ground truth bounding box the model continuously adjusts the initial anchor boxes to finally obtain the optimal predicted bounding box. For instance, Figure 4 illustrates a set of detected objects and their predicted bounding boxes for label prediction. Finally, the recognition accuracy towards objects is calculated based on metrics such as mean average precision (mAP) score, which is considered as a predefined IoU threshold.

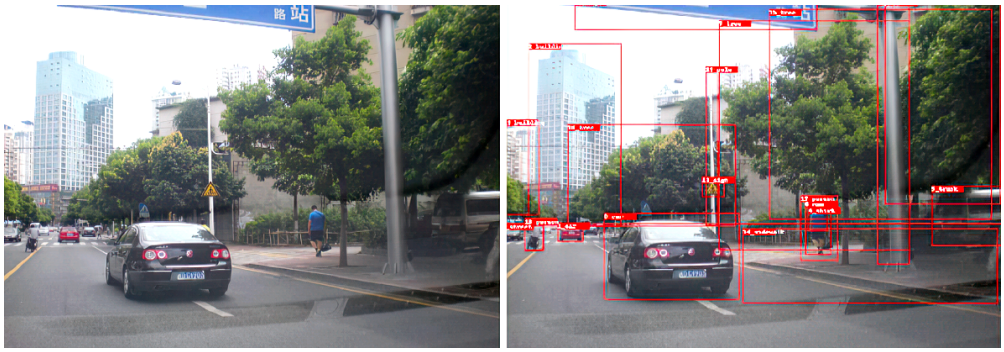


Figure 4. The pre-trained Faster R-CNN detected objects in an image of a traffic scene from the CCTSDB dataset.

To generate scene graphs, we introduced the technique of Scene Graph Generation (*SGG*) and selected a state-of-the-art unbiased scene graph generation (*USGG*) framework [22] that used causal inference. Scene graph generation is a task in computer vision that involves creating a structured representation of a visual scene, capturing the relationships between objects and their attributes. The scene graph represents the objects in the scene as nodes

and each directed edge represents the corresponding predicate between the two objects, each scene graph can also be formulated as a set of visual relation triplets (i.e., <sub, pred, obj>), providing a rich and structured understanding of the visual content. Existing scene graph generation is often biased, mainly due to the severe long-tail distribution issue of the dataset, e.g., collapsing diverse humans walk on/sit on/lay on the beach into humans on the beach. Given such SGG, the semantic extraction task can only identify structures in the scene as a series of simple objects. Biased predictions prevent further use of scene graphs in real-world applications while Unbiased scene graph generation aims to address the above-mentioned issues and improve the quality of image label extraction.

The USGG framework we used first builds a causal graph based on the extracted objects in an image and deletes the incorrect cause-result relationship caused by counterfactual causality. By distinguishing between primary effects and side effects, the model generates relatively less biased results. Figure 5 illustrates a partial scene graph derived from the image data shown in Figure 4 by USGG, which can be represented by individual triples. The USGG model performs two main tasks: object detection and relation extraction. It first detects the objects appearing in the image and then extracts the potential relations between the detected objects in the form of *Entity1-Relation-Entity2* triples. In the following paper, we will conduct experiments based on the criteria to elucidate the mapping relation between the recognition accuracy of semantic objects in images and their corresponding systems.

The domain model file is an XML file that contains several concepts, with different namespaces distinguishing each concept to represent different semantic nodes. It can be divided into <*concept*:*entity*>, <*concept*:*relation*>, <*concept*:*attribute*>, and <*concept*:*value*> to represent the entity node, relation node, attribute node and value node, respectively. Each concept tag has its corresponding keyword attribute. Different concept tags can be nested with each other using the label tag, which has attribute values of *has* and *is*. Figure 6 shows a part of the content of the domain model file, where the entity *sign* has the attribute *shape* and *size* connected through the label *has*, and the attribute *shape* connects with a series of values through the label *is*. The relation node *positon* connects with the entity node *sign* through the label *has* as well.

After importing the XML file, the tool depicted in Figure 7 will automatically generate a concept tree and various requirements based on different criteria to enhance the user experience. Utilizing the functionality offered by the USGG framework, the tool will present triples comprising entities and their relations. It will then determine whether the triples in the image meet the data requirements derived from the criteria, filtering them accordingly. The user interface will exhibit the triples that satisfy the criteria, while any triples that fail to satisfy the criteria will be recorded and a comprehensive report will be generated to guide the user in improving the image dataset.

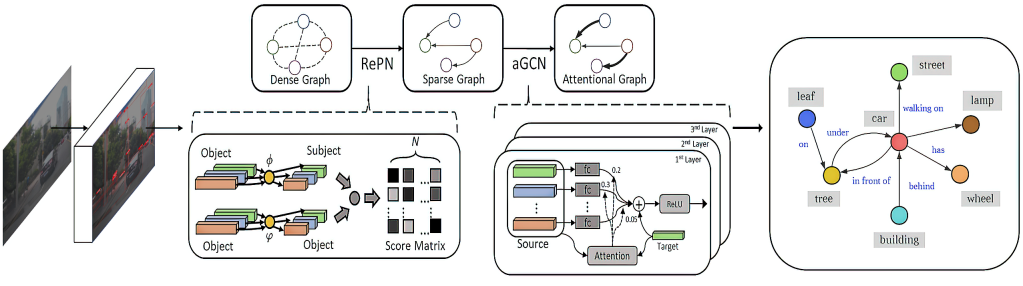


Figure 5. Overall framework of Graph RCNN which is a type of USGG model.

```
<?xml version="1.0" encoding='utf-8'?>
<xmlparse:XP xmlns:xmlparse="http://datarequirement.com" xmlns:concept="http://datarequirement.com">
    <concepts>
        <concept:entity keywords="sign">
            <label value="has">
                <concept:relation keywords="position"/>
                <concept:attribute keywords="shape">
                    <label value="is">
                        <concept:value keywords="square"/>
                        <concept:value keywords="circle"/>
                        <concept:value keywords="triangle"/>
                    </label>
                </concept:attribute>
                <concept:attribute keywords="size">
                    <label value="is">
                        <concept:value keywords="small"/>
                        <concept:value keywords="medium"/>
                        <concept:value keywords="large"/>
                    </label>
                </concept:attribute>
                <concept:attribute keywords="color">
                    <label value="is">
                        <concept:value keywords="red"/>
```

Figure 6. The structure of XML file of domain model.

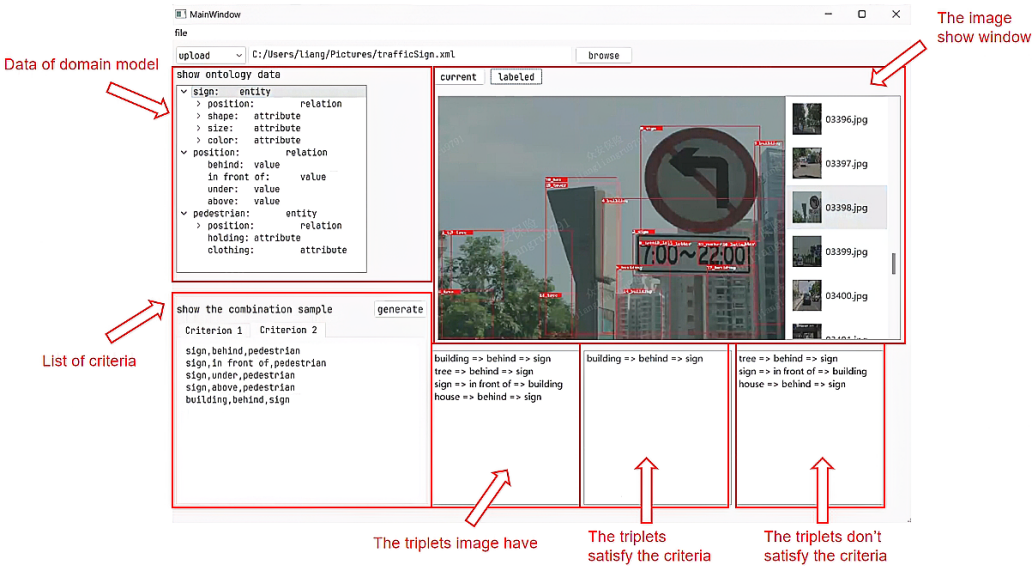


Figure 7. The user interface of data requirements analysis tool.

5. Experiments and Analysis

We have conducted the evaluation to assess the effectiveness of our data requirement analysis method in improving the model's recognition ability in ML systems using real-world datasets.

5.1. Experimental Setup

We conduct a series of experiments on the traffic scene dataset [4] to explore the potential impact of semantic information on predicting behavior and validate the effectiveness of our proposed method. In this section, we first describe the dataset used in our experiments. Then we extract entities and attributes from the images and utilize the relations extraction technique SGG to extract triples. Finally, we perform some check experiments to compare the performance of the system trained with different datasets.

We focus on detecting traffic signs that have the potential to cause traffic accidents in road driving scenarios. For this purpose, we utilize the CCTSDB2021 (https://github.com/csust7zhangjm/CCTSDB2021, accessed on 24 September 2023) China Road traffic data list, which consists of 16,365 training pictures and 2000 test pictures. To categorize the dataset, we consider three dimensions: category meanings, weather conditions, and sign

sizes. Because of the computational limitations, we selected 2000 pictures as training data and 500 pictures as testing data.

The dataset used for experimentation is targeted at traffic scenes, and we mainly focus on various traffic signs on the road. Therefore, the characteristics of the dataset generally include the factors of environment and weather which can be reflected based on the light, color and other features in the images at the time of the photo, as well as various entities that may affect the recognition of signs in the image, such as vehicles, pedestrians, trees, and so on. Different entities have different interactions and spatial relationships, which may affect the model's recognition ability. Therefore, we are trying to use existing frameworks to extract these factors.

In Section 3, only partial information was modeled, which did not fully contain the various semantic information within the traffic scenario. In this part, we attempt to utilize the method realized by [23] to obtain as much semantic information related to the traffic scenario as possible. The paper developed a script that utilized the Stanford CoreNLP parser to parse and tokenize 1,326,488 tweets. The terms in each sentence of the retrieved tweets were annotated as triplets, consisting of a subject, predicate, and object. After filtering and merging redundant information, the tweets were processed to 333 records. Among the subject-object pairs with over 20 predicates, they used the Word2vec model and the Google News corpus as references to filter the predicates based on the similarity scores. Ultimately, they selected the top 10 predicates with the highest similarity scores, which correspond to the different pairs of subjects and objects. Table 3 shows a few subject-object-predicate pairs to describe traffic scenarios.

Table 3. A few subject-object-predicates pairs to describe traffic scenario.

Subject	Object	Predicates
car	footpath	['intervened', 'walked', 'dumped', ...]
pedestrian	walker	['hit', 'travels', 'make', 'walking', 'gets', ...]
pedestrian	vehicle	['identified', 'say', 'closed', 'charged', 'rendering', ...]
motorist	intersection	['improve', 'kills', ...]
bus	bridge	['add', 'run', 'stop', 'delivering', 'work', 'see', ...]
...	...	...

We can extract semantic information related to traffic scenes by using the SGG framework mentioned in Section 4. The SGG framework provides a total of 51 relations and 151 detection entities. In Table 4, we present several relations, entities, and corresponding triples derived from SGG. For the check experiment, we select three entities and four relations as the basis of our analysis. We abstract these four relations into the keyword *position*. Therefore, the data requirements generated using our proposed method mainly include entities extracted by the SGG framework and relations elements between entities.

Table 4. Entities and Relations The framework USGG supplies.

Entities	Relations	E-R-E Triples	Entities We Choose	Relations We Choose
building	above	tree behind sign	building	above
sign	holding	street has sign	tree	in front of
tree	has	sign above car	sign	behind
car	behind	building in front of sign		under
street	walking on	woman walking on street		
woman				

5.2. Experiments Results

During the ontology semantic modeling process for a specific scene, we obtain different types of semantic nodes. To generate different coverage combinations, specific nodes in the ontology graph are traversed to satisfy the criteria. These criteria include attribute nodes,

entity ontology nodes, relation ontology nodes, and value nodes. By traversing these nodes, diverse coverage combinations can be obtained to capture different aspects of the scene.

We generate several coverage combinations based on a more complete domain model shown in Figure 8 as data requirements for different cases, where the coverage combinations can be categorized into two types. The first type of coverage combination is used to cover the entity and the value of attributes presented in images. The second type of coverage combination is used to cover three-triples between entity nodes and relation nodes in images. Table 5 shows a part of value combinations based on Algorithm 1, while *req*1–*req*10 represents the data requirements derived from attribute criterion and *req*11–*req*18 represents the data requirements derived from relation criterion.

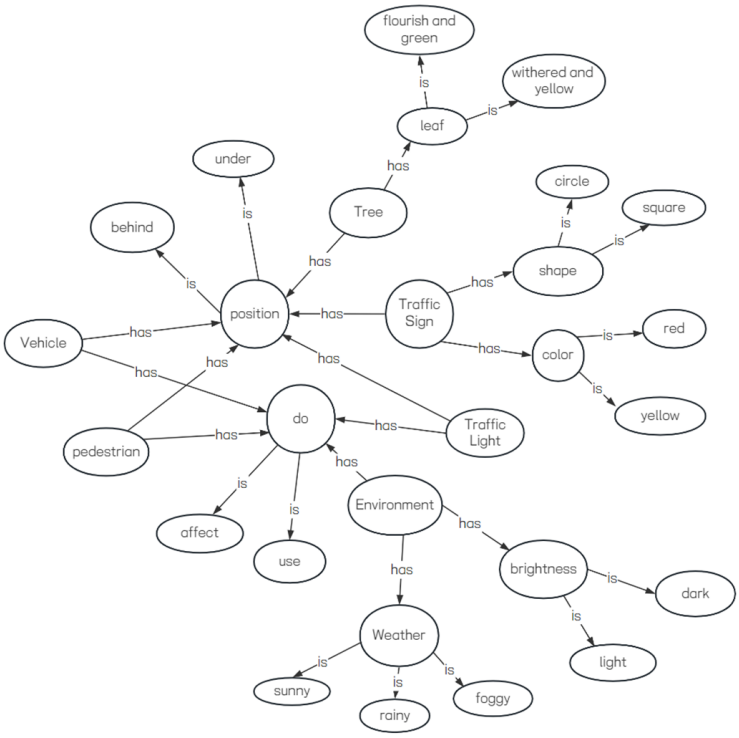


Figure 8. The ontology-based domain model towards traffic scene.

We selected seven data requirements from Table 5 as our data collection standards. Each data requirement represents a training dataset with distinct characteristics, while the contrast dataset lacks the current set of characteristics. When collecting data, we base our collection not only on individual data requirements, but also on combinations of multiple data requirements. Table 6 displays the accuracy of a model trained by different datasets in identifying traffic signs, where experiments match their baseline with the same color.

- **Experiment 1a and 1b:** In experiments 1a and 1b, we evaluated YOLO's performance in detecting the label 'sign'. We filtered the candidate dataset to obtain 321 data points that do not satisfy the data requirements and an equal number of data points that satisfy the data requirements as the training dataset, respectively. It is important to note that the dataset in Experiment 1a does not satisfy the characteristics mentioned in the data requirements, while the dataset in Experiment 1b is constructed by satisfying the data requirements derived from the attribute criterion. When images in the training dataset satisfy the characteristics described by $(red, square, medium)$, the recognition accuracy of the image recognition system is higher compared to datasets that do not satisfy these characteristics.
- **Experiment 2a and 2b:** In experiments 2a and 2b, similar to the previous experiment, we assessed YOLO's performance in detecting the label *Sign* for images. We filtered the candidate datasets to obtain 1316 data that do not satisfy the data requirements

and an equal number of data that satisfy the data requirements derived from relation criterion as training datasets, respectively. If the training dataset satisfies the data requirement $(building, behind, sign)$, the recognition accuracy of the image recognition system for traffic signs will decrease compared to the system trained on the dataset that satisfies the triple $(tree, behind, sign)$.

- **Experiment 3a and 3b:** In experiments 3a and 3b, we filtered the candidate dataset to obtain 1768 data that do not satisfy the data requirements and an equal number of data that satisfy the data requirements derived from relation criterion as training dataset, respectively. The result shows that if the training dataset satisfies $(tree, above, sign)$, $(building, above, sign)$ and $(car, above, sign)$ at the same time, the recognition accuracy of the detection system for traffic signs is higher than in dataset without the data requirements.
- **Experiment 4a and 4b:** In experiments 4a and 4b, we filtered candidate datasets to obtain 3260 data that do not satisfy the data requirements and an equal number of data that satisfy the data requirements derived from relation criterion as the training dataset, respectively. The result shows that if the training dataset satisfies $(car, behind, sign)$ and $(car, under, sign)$ at the same time, the recognition accuracy of the detection system for traffic signs is higher than in the dataset without the data requirements.

Table 5. Data Requirements Derived From Attribute Relation Criterion.

Id	Data Requirements
$req1$	{{(color, *red*)), (shape, *circle*)), (size, *medium*))}
$req2$	{(color, *red*)), (shape, *circle*), (size, *large*)}
$req3$	{(color, *red*), (shape, *square*), (size, *small*)}
$req4$	{(color, *red*), (shape, *square*), (size, *medium*)}
$req5$	{(color, *red*), (shape, *square*), (size, *large*)}
$req6$	{(color, *red*), (shape, *triangle*), (size, *small*)}
$req7$	{(color, *red*), (shape, *triangle*), (size, *medium*)}
$req8$	{(color, *red*), (shape, *triangle*), (size, *large*)}
$req9$	{(color, *blue*), (shape, *circle*), (size, *small*)}
$req10$	{(color, *yellow*), (shape, *square*), (size, *large*)}
$req11$	$(building, behind, sign)$
$req12$	$(tree, above, sign)$
$req13$	$(building, above, sign)$
$req14$	$(sign, above, street)$
$req15$	$(pedestrian, behind, sign)$
$req16$	$(car, above, sign)$
$req17$	$(car, under, sign)$
$req18$	$(car, behind, sign)$
...	...

We trained the model using datasets of the same size but in different experimental conditions. The results show that the model trained by a dataset that satisfies the data requirements has higher predictive ability compared to the model trained by a dataset that does not satisfy the data requirements. Next, we conducted another experiment to demonstrate the difference in the model's performance, where the sizes of training datasets are different.

Table 7 shows the different sizes of the datasets before and after satisfying the data requirements and Figure 9 shows the predictive performance of models trained by different datasets based on different data requirements. In experiment (a), the dataset satisfies data requirements of $(tree, above, sign)$, $(building, above, sign)$ and $(car, above, sign)$ and is

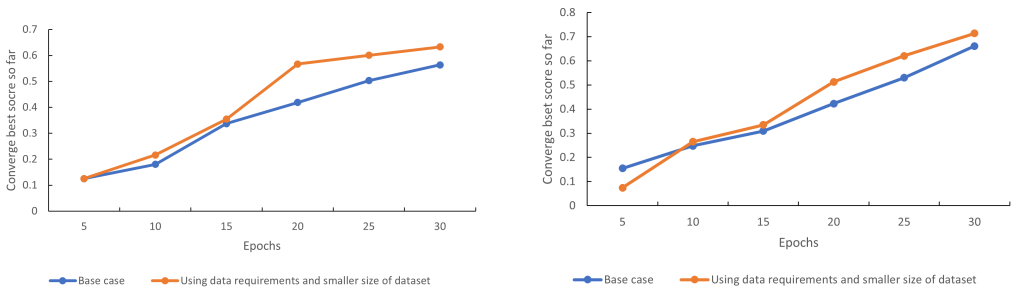
smaller than the baseline dataset. In experiment (b), the dataset satisfies data requirements of $car, behind, sign)$ and $(car, under, sign)$ and is smaller than the benchmark dataset. The results indicate that despite the size of the dataset satisfied with the data requirements being smaller than the baseline dataset, the training performance is better than the initial status.

Table 6. YOLO performance report for detecting label *Sign* before and after constructing dataset using data requirements.

	Exp.	Datasets Satisfy the Data Requirements	Data Size	mAP%50
Constructing Datasets Randomly (Base Case)	1a	-	321	0.7748
	2a	-	1316	0.8717
	3a	-	1768	0.8695
	4a	-	3260	0.7754
Constructing Datasets Using Data Requirements	1b	*req*5	321	0.8018
	2b	*req*15	1316	0.9015
	3b	*req*16 *req*17 *req*20	1768	0.8712
	4b	*req*21 *req*22	3260	0.7839

Table 7. The size of dataset before and after satisfying the data requirements.

Exp.	Base Case	Satisfying Data Requirements
Experiment 3.	1768	1300
Experiment 4.	3260	2500



(**a**) (**b**)

Figure 9. The mAP_0.5:0.95 scores of model trained by candidate dataset and smaller size dataset constructed by data requirements. (**a**) comparison of dataset satisfies the data requirements mentioned in Experiment 3 and candidate dataset. (**b**) comparison of dataset satisfies the data requirements mentioned in Experiment 4 and candidate dataset.

The previous experiment demonstrated how the recognition capability of the system changes when a dataset contains data that fully satisfies the data requirements and data that fail to satisfy the requirements. Next, we would like to quantitatively compare how the proportion of data satisfying different requirements in the dataset affects the results.

We use coverage rate to divide the candidate dataset based on whether satisfy the criteria, given a dataset D and a data requirement req satisfy the criteria, the coverage rate of req is the number of items in D that matches req. Mathematically, the coverage rate is defined as follows:

$$cov(L, D) = |\{t \in D \mid M(t, req) = 1\}| \quad (9)$$

where $M(t, req) = 1$ means an item matches the L. When D is known, we can simplify $cov(req, D)$ with $cov(req)$, we would like each data in the D to match the criterion or we just want the rate to be higher than a threshold ξ.

We denote $S(D)$ as the state model obtained by the ML system after training with a specific dataset, and $S(D')$ as the state model obtained by the ML system after training with a different dataset filtered by the value of attribute nodes or the combination of entity ontology nodes and relation ontology nodes based on *Attribute Coverage Criterion* and *Relation Coverage Criterion*.

The candidate datasets are divided into different subsets based on the coverage criteria and their corresponding distribution rates. Each subset represents a subsystem with specific prediction abilities. Table 8 illustrates the performance of subsystems that satisfy the different coverage criteria distributions. For example, when the percentage of subsets satisfied with coverage combinations $req1$ increases from 20% to 40%, and the percentage of subsets satisfied with coverage combinations $req2$ increases from 30% to 40%, while the percentage of subsets satisfied with coverage combinations $req3$ decreases from 50% to 20%, the subsystem evolves from $S1$ to $S2$, which exhibits different predictive behaviors. In this case, the accuracy of $S2$ in predicting red and round traffic signs slightly improves compared to $S1$. Similarly, when the subsets change from being satisfied with the combination of $req1 + req5$ to being satisfied with the combination of $req1 + req6$, the subsystem evolves from $S4$ to $S5$. This change occurs because the relation coverage combination based on the relation coverage criterion transforms from $(tree, behind, sign)$ to $(building, behind, sign)$. As a result, the accuracy of predicting red, round, and small traffic signs slightly decreases.

Based on the above experiments, we have identified elements that can significantly impact the accuracy of the image recognition system. For instance, increasing the number of images containing small signs in the scene can improve the system's accuracy in recognizing traffic signs. Additionally, due to the obstruction caused by trees, the shape of traffic signs can change. Therefore, increasing the proportion of image data containing trees in a dataset can often enhance the performance of the image recognition system compared to increasing the proportion of images containing buildings behind the traffic signs. Environmental features such as lighting and weather can also cause changes in the color of traffic signs. These features are all part of the domain model we proposed, which indicates that the data with such features have higher quality. Increasing the proportion of data containing the features present in the domain model can help the ML system better understand the special situation to have the ability to handle different scenarios. Therefore, correctly establishing the domain model can help avoid collecting irrelevant or redundant data, ensuring the completeness and accuracy of data collection.

Table 8. Recognition accuracy based on different dataset distributuon.

	Dataset Distribution	**mAP_0.5**
$S1$	$cov(req1, D1) = 20\%$,$cov(req2, D1) = 30\%$, $cov(req3, D1) = 50\%$	0.7615
$S2$	$cov(req1, D2) = 40\%$,$cov(req2, D2) = 40\%$, $cov(req3, D2) = 20\%$	0.7832
$S3$	$cov(req4, D3) = 20\%$, $cov(req2, D3) = 30\%$, $cov(req3, D3) = 50\%$	0.7963
$S4$	$cov(req1 + req5, D4) = 40\%$, $cov(req2 + req5, D4) = 40\%$, $cov(req3 + req5, D4) = 20\%$	0.7398
$S5$	$cov(req1 + req6, D5) = 40\%$, $cov(req2 + req6, D5) = 40\%$, $cov(req3 + req6, D5) = 20\%$	0.7216

5.3. Discussion

Taking an object detection system as an example, this paper elucidates the process of constructing data requirements based on target entities or their relationships. The initial step involves constructing the domain model for the given problem. While building this

model, whether the problem can abstract semantic entities should be determined. Because it makes users understand the problem more easily and generates data requirements through semantic information. Meanwhile, it is imperative to consider potential relationships among entities and identify those entities directly related to the target entity. This careful consideration is essential as entities closely associated with the target entity exert a significant impact, ultimately influencing the detection performance aimed at the said target entity.

For example, in a tumor recognition system, the identification of benign and malignant tumors is mainly influenced by factors such as the shape, texture, density, and size of the tumor. These features are typically used to establish domain models to achieve accurate identification and classification of tumors. Additionally, the likelihood of having a tumor is also influenced by independent factors such as the patient's age and gender, which are directly related to the tumor in the domain model. Therefore, it is necessary to focus more on the above-mentioned factors through the process of collecting, while factors such as color features, surrounding environmental features, and motion features, which do not have a significant impact on the identification results, do not need to be given too much attention.

When handling ML systems that do not involve detection functionality, such as prediction and classification systems, the process of constructing requirements for all entity objects within the domain model can be executed. However, the effectiveness of this approach is heavily contingent on the computational performance of the machine learning system. Therefore, our subsequent research endeavors are directed towards minimizing the complexity associated with generating data requirements as much as possible. Moreover, in the case of sentiment analysis systems, because the sentiment analysis task primarily involves extraction of three elements—entity, aspect and opinion—from the text to analyze the positive or negative evaluations directed towards a specific object, we can construct the domain model based on these three elements [24]. The approach can even be considered for improving the quality of datasets for multi-modal systems [25–27]. By providing structured and standardized representations of domain knowledge, it is possible to combine different types of data such as text, images, and audio to unified entities and relationships in ontology to achieve the correlation and integration of different modal data.

In fact, irrespective of the system type, as long as the problem domain allows for the extraction of multiple semantic entities, where entities have inherent aspects, and there are strong or weak relationships among these entities, the method proposed in this paper can be considered to enhance the quality of datasets used for training machine learning systems.

6. Conclusions

The construction of ML systems involves four main stages: data collection, data preprocessing, model training and model deployment. This paper specifically focused on the first stage and proposed a requirement analysis method tailored to ML systems. Unlike obtaining training datasets solely through non-functional requirements during the data collection stage, this method incorporates ontology and utilizes some of its concepts to semantically model the dataset in the domain model, thereby enhancing the functional requirements. This approach provides users with more specific evaluation metrics for datasets, in addition to abstract measures such as fairness and non-discrimination. Additionally, it offers formal expressions combined with the specific scenarios targeted by the system to automatically generate data requirements. The performance of the YOLOV5-based image recognition system was evaluated on the candidate CCTSDB2021 dataset and the dealed-CCTSDB2021 dataset filtered through our data requirements method. The model trained by the processed dataset exhibited superior performance over the baseline model.

Firstly we construct a domain model, our study introduces the concepts of entity, relation, attribute, and value based on the concepts of Class and ObjectProperty in ontology, which form a semantic topological graph. Formal expressions are provided to distinguish these four concepts, enabling the model file to be parsed into metadata that need to be referenced for data requirement construction. Next, we propose coverage criteria that

cover elements in the domain model. These criteria will derive data requirements through the algorithm we provided, automatically generating corresponding data requirements according to the metadata. Additionally, a supporting tool is developed based on this theory, consisting of three components: the ontology analysis module, the data requirements specification module, and the dataset construction generation module. This tool will assist users in generating data requirement specifications more efficiently and effectively, requiring little additional manual effort.

In the future, we intend to incorporate automated reasoning mechanisms in ontology to improve the development of domain models and explore more comprehensive coverage criteria. In our example, we solely utilized image data to demonstrate our approach. The inclusion of other data types, such as audio, text, and other multi-modal data, in our approach, will be a key focus of our future research.

Author Contributions: Conceptualization, L.J. and X.W.; methodology, L.J.; software, L.J.; validation, L.J. and X.W.; formal analysis, L.J. and X.W.; data curation, L.J.; writing—original draft preparation, L.J.; writing—review and editing, L.J. and X.W.; visualization, L.J.; supervision, X.W. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the school-level project of Shanghai University "Software Requriements Analysis Method for Underground Railway System".

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding authors.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Strickland, E. Andrew Ng, AI Minimalist: The Machine-Learning Pioneer Says Small is the New Big. *IEEE Spectr.* **2022**, *59*, 22–50. [CrossRef]
2. Habibullah, K.M.; Gay, G.; Horkoff, J. Non-functional requirements for machine learning: Understanding current use and challenges among practitioners. *Requir. Eng.* **2023**, *28*, 283–316. [CrossRef]
3. Ahmad, K.; Bano, M.; Abdelrazek, M.; Arora, C.; Grundy, J. What's up with requirements engineering for artificial intelligence systems? In Proceedings of the 2021 IEEE 29th International Requirements Engineering Conference (RE), Notre Dame, IN, USA, 20–24 September 2021; pp. 1–12.
4. Zhang, J.; Zou, X.; Kuang, L.D.; Wang, J.; Sherratt, R.S.; Yu, X. CCTSDB 2021: A more comprehensive traffic sign detection benchmark. *Hum.-Centric Comput. Inf. Sci.* **2022**, *12*, 1–21.
5. Yang, Z.; Qi, P.; Zhang, S.; Bengio, Y.; Cohen, W.W.; Salakhutdinov, R.; Manning, C.D. HotpotQA: A dataset for diverse, explainable multi-hop question answering. *arXiv* **2018**, arXiv:1809.09600.
6. Yu, F.; Chen, H.; Wang, X.; Xian, W.; Chen, Y.; Liu, F.; Madhavan, V.; Darrell, T. Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 2636–2645.
7. Ghiasi, G.; Cui, Y.; Srinivas, A.; Qian, R.; Lin, T.Y.; Cubuk, E.D.; Le, Q.V.; Zoph, B. Simple copy-paste is a strong data augmentation method for instance segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 2918–2928.
8. Gupta, N.; Patel, H.; Afzal, S.; Panwar, N.; Mittal, R.S.; Guttula, S.; Jain, A.; Nagalapatti, L.; Mehta, S.; Hans, S.; et al. Data Quality Toolkit: Automatic assessment of data quality and remediation for machine learning datasets. *arXiv* **2021**, arXiv:2108.05935.
9. Pan, H.; Xi, Y.; Wang, L.; Nan, Y.; Su, Z.; Cao, R. Dataset construction method of cross-lingual summarization based on filtering and text augmentation. *PeerJ Comput. Sci.* **2023**, *9*, e1299. [CrossRef] [PubMed]
10. Yang, S.; Xiao, W.; Zhang, M.; Guo, S.; Zhao, J.; Shen, F. Image data augmentation for deep learning: A survey. *arXiv* **2022**, arXiv:2204.08610.
11. Cubuk, E.D.; Zoph, B.; Shlens, J.; Le, Q.V. Randaugment: Practical automated data augmentation with a reduced search space. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, Seattle, WA, USA, 14–19 June 2020; pp. 702–703.
12. Wightman, R.; Touvron, H.; Jégou, H. Resnet strikes back: An improved training procedure in timm. *arXiv* **2021**, arXiv:2110.00476.
13. Yao, Y.; Zhang, J.; Shen, F.; Liu, L.; Zhu, F.; Zhang, D.; Shen, H.T. Towards automatic construction of diverse, high-quality image datasets. *IEEE Trans. Knowl. Data Eng.* **2019**, *32*, 1199–1211. [CrossRef]

14. Li, Y.; Meng, L.; Chen, L.; Yu, L.; Wu, D.; Zhou, Y.; Xu, B. Training data debugging for the fairness of machine learning software. In Proceedings of the 44th International Conference on Software Engineering, Pittsburgh, PA, USA, 25–27 May 2022; pp. 2215–2227.
15. Giunchiglia, E.; Imrie, F.; van der Schaar, M.; Lukasiewicz, T. Machine Learning with Requirements: A Manifesto. *arXiv* **2023**, arXiv:2304.03674.
16. Zhang, R.; Albrecht, A.; Kausch, J.; Putzer, H.J.; Geipel, T.; Halady, P. DDE process: A requirements engineering approach for machine learning in automated driving. In Proceedings of the 2021 IEEE 29th International Requirements Engineering Conference (RE), Notre Dame, IN, USA, 20–24 September 2021; pp. 269–279.
17. Ries, B.; Guelfi, N.; Jahic, B. An mde method for improving deep learning dataset requirements engineering using alloy and uml. In Proceedings of the 9th International Conference on Model-Driven Engineering and Software Development, Virtual Event, 8–10 February 2021; SCITEPRESS: Setúbal, Portugal, 2021; pp. 41–52.
18. Chu, X.; Ilyas, I.F.; Krishnan, S.; Wang, J. Data cleaning: Overview and emerging challenges. In Proceedings of the 2016 International Conference on Management of Data, San Francisco, CA, USA, 26 June–1 July 2016; pp. 2201–2206.
19. Abedjan, Z.; Golab, L.; Naumann, F.; Papenbrock, T. *Data Profiling*; Springer Nature: Berlin/Heidelberg, Germany, 2022.
20. De Coste, M.; Li, Z.; Khedri, R. The prediction of mid-winter and spring breakups of ice cover on Canadian rivers using a hybrid ontology-based and machine learning model. *Environ. Model. Softw.* **2023**, *160*, 105577. [CrossRef]
21. Asudeh, A.; Jin, Z.; Jagadish, H. Assessing and remedying coverage for a given dataset. In Proceedings of the 2019 IEEE 35th International Conference on Data Engineering (ICDE), Macao, China, 8–11 April 2019; pp. 554–565.
22. Tang, K.; Niu, Y.; Huang, J.; Shi, J.; Zhang, H. Unbiased scene graph generation from biased training. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 3716–3725.
23. Barzamini, H.; Shahzad, M.; Alhoori, H.; Rahimi, M. A multi-level semantic web for hard-to-specify domain concept, Pedestrian, in ML-based software. *Requir. Eng.* **2022**, *27*, 1–22. [CrossRef]
24. Li, M.; Lu, Q.; Long, Y.; Gui, L. Inferring affective meanings of words from word embedding. *IEEE Trans. Affect. Comput.* **2017**, *8*, 443–456. [CrossRef]
25. Baltrušaitis, T.; Ahuja, C.; Morency, L.P. Multimodal machine learning: A survey and taxonomy. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *41*, 423–443. [CrossRef]
26. Khan, S.; Naseer, M.; Hayat, M.; Zamir, S.W.; Khan, F.S.; Shah, M. Transformers in vision: A survey. *Acm Comput. Surv. (CSUR)* **2022**, *54*, 1–41. [CrossRef]
27. Xu, P.; Zhu, X.; Clifton, D.A. Multimodal learning with transformers: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2023**, *45*, 2113–12132. [CrossRef] [PubMed]