RELAZIONE
DI
WEB SEMANTICO

AMLO
Advanced Machine Learning Ontology

Andrea Ingargiola - 1026589
andrea.ingargiola@studio.unibo.it
Riccardo Omiccioli - 1054475
riccardo.omiccioli@studio.unibo.it
Cecilia Teodorani - 1032918
cecilia.teodorani@studio.unibo.it

Novembre 2023

Indice

1 Introduzione **3**

2 Analisi generale del dominio **4**
2.1 Ontologie integrate 4

3 Ontologia **6**
3.1 In dettaglio 8
3.1.1 Persona 8
3.1.2 Ruolo 9
3.1.3 Gruppo 10
3.1.4 Ente 11
3.1.5 Progetto 11
3.1.6 Tecnologia 12
3.1.7 Competizione 13
3.1.8 Edizione 14
3.1.9 Categoria 15
3.1.10 Tipo di Machine Learning 15
3.1.11 Strumento 16
3.1.12 Strato 17
3.2 Regole SWRL 18
3.2.1 isPersonDifferent 18
3.2.2 isCoauthor 19
3.2.3 isAuthorWinner 19
3.2.4 isTeamOf 19

4 Dati e Interrogazioni **20**
4.1 Inserimento dati 20
4.2 Interrogazioni 20
4.2.1 Top 10 persone che hanno fatto più progetti 20
4.2.2 Persona che ha vinto più competizioni 21
4.2.3 Organizzazioni che hanno finanziato più progetti 21
4.2.4 Numero di tecnologie in ogni tipo di Machine Learning 22
4.2.5 Media delle persone che lavorano in un gruppo 22
4.2.6 Algoritmi che hanno numero di layer $15 \leq x \leq 25$ 22
4.2.7 Numero di layer degli algoritmi 23
4.2.8 Riepilogo di una tecnologia 23

5 Conclusioni **25**

Capitolo 1

Introduzione

Nell'ultimo decennio, è diventato sempre più comune sentir parlare di Machine Learning e Intelligenza Artificiale nella vita quotidiana, poiché queste tecnologie vengono sempre più utilizzate nei sistemi software che la popolazione utilizza quotidianamente. Un esempio tangibile di questa diffusione è l'uso comune del riconoscimento facciale nei nostri smartphone per sbloccarli. Parallelamente, c'è stato un aumento significativo nell'interesse e nello studio di questi campi nelle università e nelle aziende, poiché continuano a evolversi e a diventare sempre più complessi nel tempo. Anche se esiste già un'ontologia chiamata "Machine Learning Ontology"[4], essa potrebbe non essere esaustiva per soddisfare tutte le necessità di modellazione in questo vasto campo.

Pertanto, è stata presa la decisione di creare un'ontologia dedicata, chiamata **AMLO** (**Advanced Machine Learning Ontology**), con l'obiettivo di organizzare in modo completo le tecnologie di questo dominio. Questa ontologia comprenderà non solo le tecniche di Machine Learning e gli elementi matematici utilizzati, ma includerà anche informazioni sulle persone coinvolte nella loro creazione, sugli enti che le hanno finanziate e sulle competizioni in cui alcune di queste tecnologie hanno ottenuto riconoscimenti.

Capitolo 2

Analisi generale del dominio

La Advanced Machine Learning Ontology si concentra sull'intero dominio del Machine Learning, cioè modella ciò che riguarda algoritmi e modelli che consentono ai computer di apprendere e migliorare le proprie prestazioni in specifici compiti senza richiedere una programmazione esplicita. Per garantire una rappresentazione completa di modelli e algoritmi, sono state incluse informazioni sugli sviluppatori di tali tecnologie, cioè le persone coinvolte nei progetti di sviluppo finanziati da vari possibili enti. Poiché l'ontologia copre le aree di Computer Vision e Deep Learning, sono state inserite informazioni riguardanti le competizioni in cui alcune reti neurali hanno partecipato e/o vinto, comprese le diverse edizioni annuali e gli organizzatori.

In generale, l'ontologia si compone dei seguenti concetti:

- tecnologia: comprende algoritmi e modelli nell'ambito del Machine Learning;
- tipo di Machine Learning: suddivisione utilizzata attualmente nel campo, cioè Supervised Learning, Unsupervised Learning e Reinforcement Learning;
- persona: indica un essere umano che ha organizzato una competizione o che ha sviluppato una tecnologia;
- ente: una università, un'azienda o un'organizzazione che hanno finanziato progetti di ricerca, all'interno dei quali lavorano persone hanno sviluppato tecnologie, oppure dipendenti che hanno organizzato alcune edizioni delle competizioni;
- competizione: evento annuale in cui si sfidano varie tecnologie e che solo una di esse vince.

2.1 Ontologie integrate

È fondamentale allineare AMLO con altre ontologie, poiché ciò facilita notevolmente l'interoperabilità con esse e garantisce l'accuratezza dei concetti definiti. Inoltre, questa pratica semplifica l'estensione dell'ontologia per l'introduzione di nuovi concetti precedentemente non inclusi. Nello specifico, DBpedia, FOAF, MLO, AIISO e DublinCore sono le ontologie scelte da integrare in AMLO.

- **DBpedia** [1]: prefisso "dbo", è un progetto che estrae informazioni enciclopediche da Wikipedia e le organizza in dati strutturati disponibili in formato RDF (Resource Description Framework). Sono stati utilizzati anche i **DBpedia resources**, prefisso "dbr", cioè gli oggetti che rappresentano entità o concetti specifici del mondo reale a cui è associata una pagina corrispondente su Wikipedia;

- **Friend of a Friend** (FOAF) [8]: prefisso "foaf", è un'ontologia semantica utilizzata per rappresentare informazioni personali e relazioni sociali online delle persone in modo strutturato e standardizzato;

- **Machine Learning Ontology** (MLO) [4]: prefisso "mlo", è un'ontologia nel campo del Machine Learning, che comprende algoritmi, problemi, strumenti e altre caratteristiche correlate;

- **Academic Institution Internal Structure Ontology** (AIISO) [9]: prefisso "aiiso", fornisce classi e proprietà per descrivere la struttura organizzativa interna di un'istituzione accademica;

- **DublinCore** [11]: prefisso "dcterms", è uno standard internazionale utilizzato per descrivere risorse digitali. Utilizzato per gestire i metadati riguardanti autori, licenza e titolo dell'ontologia.

Nell'ontologia sviluppata, sono stati utilizzati solo gli elementi ritenuti utili e necessari delle ontologia appena citate. Dal capitolo successivo, questi elementi saranno individuabili grazie al prefisso della rispettiva ontologia.

Capitolo 3

Ontologia

Per la creazione dell'ontologia, sono state utilizzate le tecnologie standard del Web Semantico. Inizialmente, è stato usato il **Resource Description Framework** (RDF) [3], in cui le informazioni sono rappresentate in forma di triple (soggetto, predicato, oggetto), dove il soggetto identifica una risorsa, il predicato indica una relazione e l'oggetto rappresenta il valore associato. Questo formato RDF consente l'elaborazione dei dati tramite motori di ricerca semantici, strumenti di manipolazione dati e ragionatori per l'inferenza.

In seguito, è stato adottato RDFS (**Resource Description Framework Schema**) [10], un'estensione di RDF che permette di organizzare e definire le informazioni in modo più strutturato e significativo. RDFS fornisce infatti costrutti per definire classi, sottoclassi, proprietà di collegamento tra risorse, sottoproprietà, relazioni tra classi e inferenze di base.

Infine, è stato inserito anche OWL2 (**Web Ontology Language 2**) [5], un linguaggio che offre una modellazione semantica avanzata rispetto alle tecnologie precedenti. OWL2 include una vasta gamma di costrutti per la rappresentazione di concetti complessi e restrizioni su classi e proprietà. Inoltre, esso supporta il ragionamento automatico, permettendo di derivare nuove informazioni dai dati esistenti. OWL2 supporta RDF e RDFS, in quanto costituisce una loro estensione.

Nella Tabella 3.1 sono presenti le metriche principali dell'ontologia AMLO.

Axioms	1351
Logical axiom count	843
Declaration axioms count	182
Class count	28
Object property count	34
Data property count	10
Individual count	111
Annotation property count	8
Class assertion	74
Object property assertion	173
Data property assertion	435
Annotation assertion	326

Tabella 3.1: Metriche dell'ontologia

La Figura 3.1 illustra lo schema dell'ontologia AMLO, il quale verrà successivamente approfondito. Le data properties sono evidenziate in verde, mentre gli elementi in giallo denotano possibili istanze di ciascuna classe. Per rappresentare le relazioni di sottoclasse, sono state utilizzate linee tratteggiate.

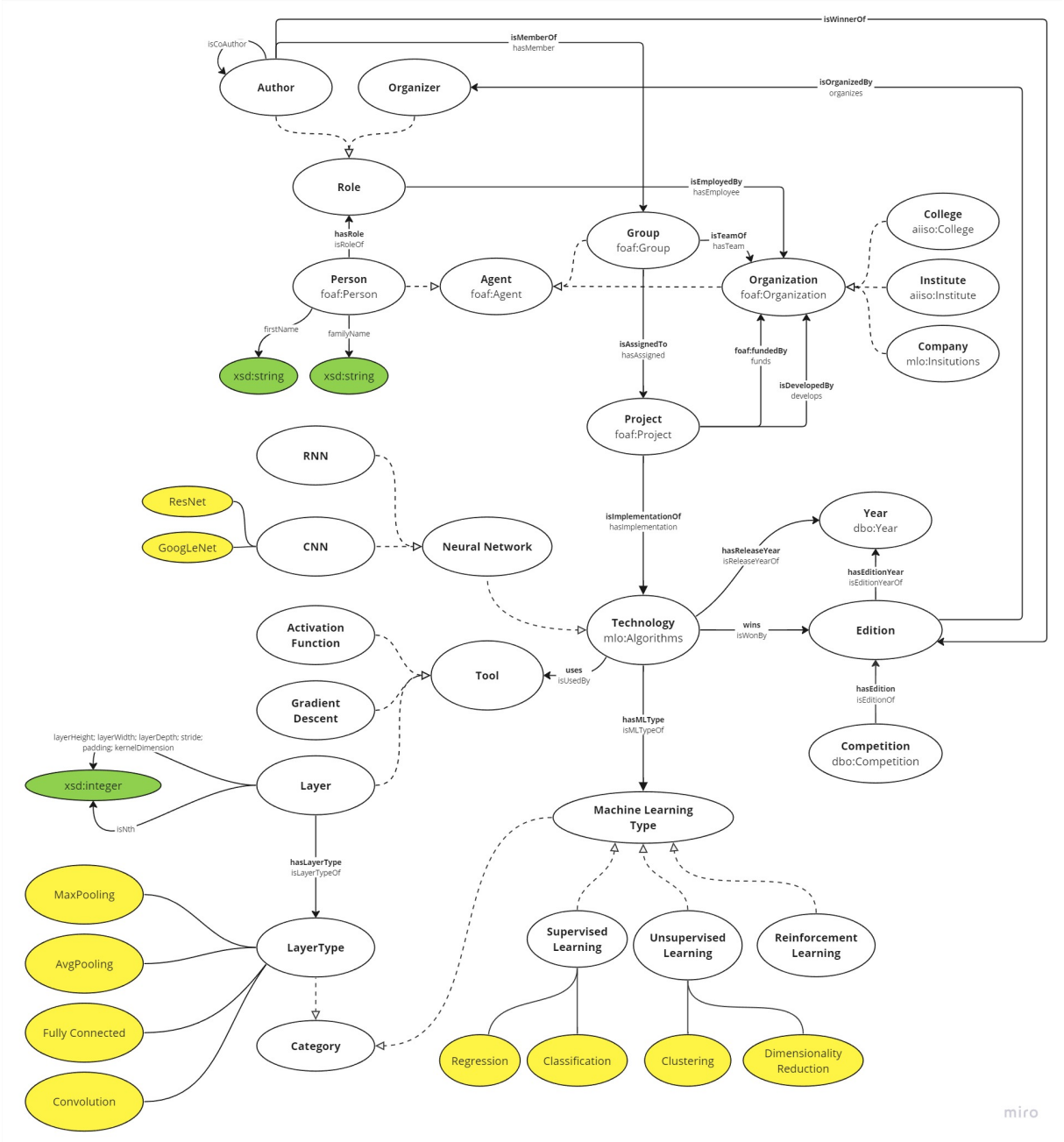

Figura 3.1: Schema AMLO

3.1 In dettaglio

Di seguito verranno riportate le classi implementate nell'ontologia che sono ritenute più importanti. Per ciascuna di esse verranno anche indicate eventuali object e data properties, caratteristiche e relazioni con altre classi.

3.1.1 Persona

Il termine *Persona* rappresenta un individuo generico, inteso come un soggetto fisico che esiste nel mondo reale. Questo concetto è importato direttamente da *foaf:Person*, ma nell'ambito di AMLO, sono state apportate alcune estensioni, principalmente per quanto riguarda le proprietà di dati *familyName* e *firstName*. Queste proprietà consentono l'assegnazione di un nome e un cognome a una persona per scopi di identificazione. È importante notare che nell'ontologia sviluppata, non è stato affrontato il concetto di omonimia. Di conseguenza, due persone sono considerate diverse se hanno nomi e/o cognomi diversi.

Inoltre, viene importata la sovraclasse *foaf:Agent*, che rappresenta un soggetto più generale capace di compiere azioni. Ciascuna persona nell'ontologia può essere associata a uno o più ruoli specifici.

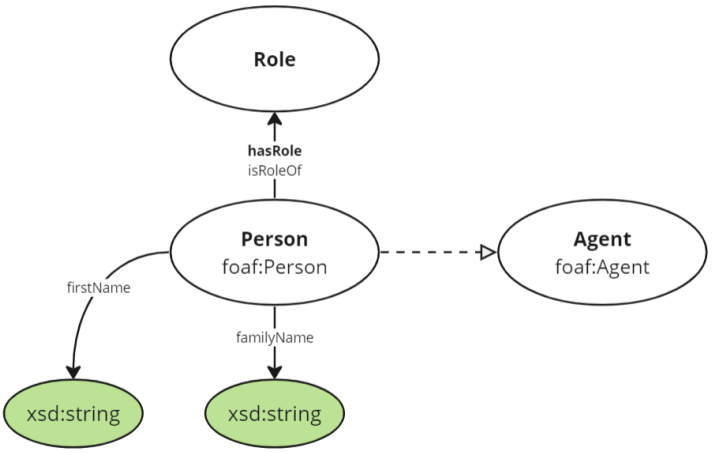

Figura 3.2: Schema Persona

object property	domain	range	inverse of
hasRole	Person	Role	isRoleOf

Tabella 3.2: Object Properties di Persona

Persona ha solo una object property, *hasRole*, all'interno del suo contesto. Questa proprietà è asimmetrica e irriflessiva ed è utilizzata per associare una *Persona* ad un *Ruolo* specifico che essa ricopre, come *Autore* o *Organizzatore*.

3.1.2 Ruolo

In questa ontologia, viene definito un *Ruolo* (:Role) come il lavoro svolto da una persona, come ad esempio essere un autore o un organizzatore. Ogni persona è rappresentata all'interno dell'ontologia in modo univoco, ma ogni volta che partecipa a un gruppo di autori o è coinvolta nell'organizzazione di una competizione, assume un ruolo diverso. Di conseguenza, una persona può avere più ruoli, ciascuno associato a un contesto specifico.

La classe *Ruolo* si specializza in due sottoclassi: *Autore* (:Author) e *Organizzatore* (:Organizer). Un *Autore* è una persona che fa parte di un gruppo di lavoro che contribuisce a un progetto, mentre un *Organizzatore* si occupa dell'organizzazione delle diverse edizioni delle competizioni in cui alcune tecnologie partecipano. Un *Ruolo* può essere associato a un *Ente* per il quale lavora.

```
###  https://github.com/RiccardoOmiccioli/AMLO/Role
:Role rdf:type owl:Class ;
        owl:equivalentClass [ rdf:type owl:Class ;
                owl:unionOf ( :Author
                        :Organizer
                )
        ] ;
        rdfs:label "Role"@en ,
                "Ruolo"@it .
```

Nella classe *Ruolo* è stato inserito un vincolo per indicare che per un dato *Ruolo* esso può essere solamente *Autore* o *Organizzatore*.

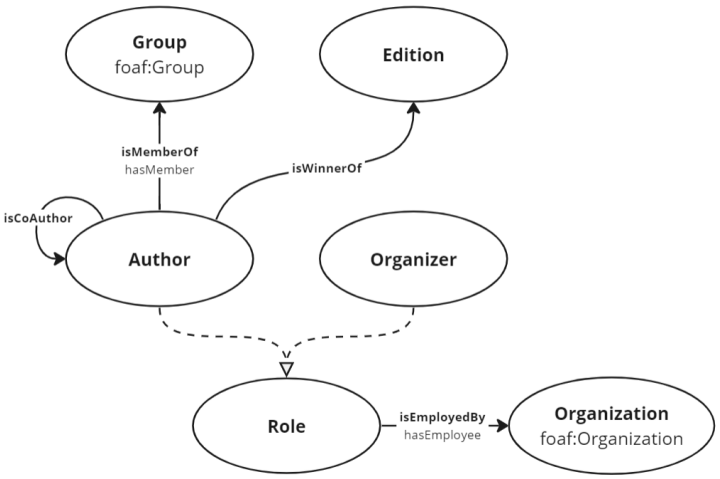

Figura 3.3: Schema Ruolo

object property	domain	range	inverse of
isCoAuthor	Author	Author	
isWinnerOf	Author	Edition	
isMemberOf	Author	Group	hasMember
isEmployedBy	Role	Organization	hasEmployee

Tabella 3.3: Object Properties di Ruolo

La classe *Ruolo* presenta diverse proprietà oggetto, alcune delle quali sono predefinite, mentre altre vengono inferite tramite l'aggiunta di regole SWRL. La proprietà oggetto *isCoAuthor* è utilizzata per associare un *Autore* a un altro *Autore* che ha collaborato nello stesso gruppo di lavoro. Questa proprietà è simmetrica e viene inferita automaticamente tramite SWRL.

La seconda proprietà inferita è *isWinnerOf*, che collega un *Autore* a una specifica *Edizione* vinta da una tecnologia di Machine Learning a cui ha contribuito. Essendo una proprietà unidirezionale, questa è classificata come asimmetrica e irriflessiva. Anche questa viene inferita tramite SWRL.

Le ultime due proprietà relative a un *Ruolo* sono *isMemberOf* e *isEmployedBy*, utilizzate rispettivamente per indicare l'appartenenza di un *Autore* a un *Gruppo* e di un *Ruolo* a un *Ente*. Poiché queste proprietà indicano l'appartenenza di un *Ruolo* specifico a un solo *Gruppo* o *Ente*, vengono considerate funzionali, asimmetriche e irriflessive.

3.1.3 Gruppo

La classe *Gruppo* è anch'essa una sottoclasse di *foaf:Agent* ed è importata da *foaf:Group*. Il concetto di gruppo è piuttosto generale, in linea con l'interpretazione fornita nell'ontologia FOAF. Tuttavia, all'interno di AMLO, viene utilizzato il concetto di *Gruppo* per rappresentare un'aggregazione di *Autori* che collaborano per lavorare su uno o più progetti legati alle tecnologie di Machine Learning.

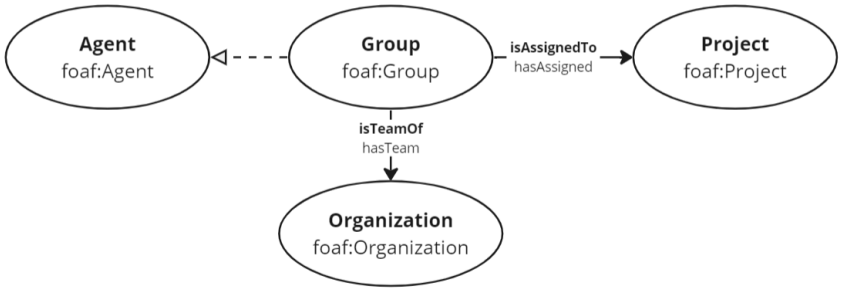

Figura 3.4: Schema Gruppo

object property	domain	range	inverse of
isAssignedTo	Group	Project	hasAssigned
isTeamOf	Group	Organization	hasTeam

Tabella 3.4: Object Properties di Gruppo

La classe *Gruppo* è associata a una proprietà oggetto chiamata *isAssignedTo*, che indica l'assegnamento di un *Gruppo* a un *Progetto*. Poiché è possibile che un *Gruppo* possa essere associato a più di un *Progetto*, questa proprietà non è funzionale ma è asimmetrica e irriflessiva.

Inoltre, tramite una regola SWRL, la proprietà *isTeamOf* viene dedotta automaticamente, collegando un *Gruppo* all'*Ente* responsabile dello sviluppo del *Progetto* su cui il

Gruppo sta lavorando. Dato che un *Gruppo* può contribuire a vari progetti e questi possono essere gestiti da differenti enti, un *Gruppo* potrebbe fungere da team per più enti. Di conseguenza, questa object property non è funzionale, ma è asimmetrica e irriflessiva.

3.1.4 Ente

La classe *Ente* (*foaf:Organization*) è una sottoclasse della classe più generale *foaf:Agent* ma, a differenza di *foaf:Group*, è utilizzata per rappresentare concetti più specifici. Inoltre, *Ente* si specializza ulteriormente in sottoclassi che rientrano in categorie specifiche come *Azienda* (*mlo:Institutions*), *Istituto* (*aiiso:Institute*) o *Università* (*aiiso:College*). Viene utilizzata la classe *Ente* per esprimere concetti legati al finanziamento o allo sviluppo di tecnologie di Machine Learning da parte di entità riconosciute.

```
###  http://xmlns.com/foaf/0.1/Organization
foaf:Organization rdf:type owl:Class ;
        rdfs:subClassOf foaf:Agent ;
        owl:equivalentClass [ rdf:type owl:Class ;
                owl:unionOf ( mlo:Institutions
                        aiiso:Institute
                        aiiso:College
                )
        ] ;
        rdfs:isDefinedBy foaf: ;
        rdfs:label "Organization"@en ,
                "Ente"@it .
```

Nella classe *Ente* è stato inserito un vincolo per indicare che per un dato *Ente* esso può essere solamente *Università*, *Istituto* o *Azienda*.

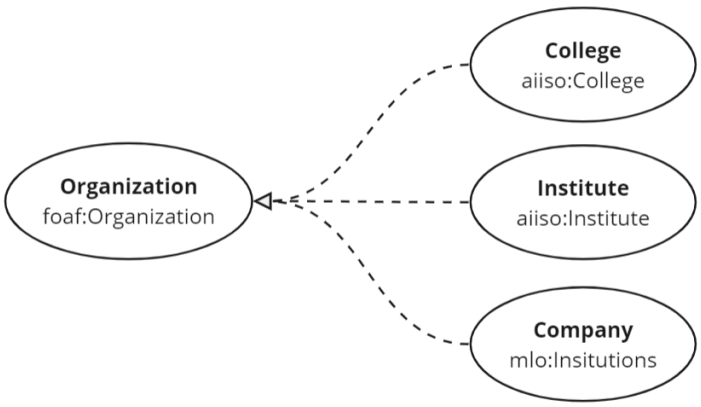

Figura 3.5: Schema Ente

3.1.5 Progetto

Progetto è definito tramite *foaf:Project* e, in conformità con quanto specificato in FOAF, rappresenta un lavoro collettivo finalizzato al raggiungimento di un obiettivo, che in questo contesto è lo sviluppo di una tecnologia di Machine Learning. Nell'ontologia, *foaf:Project*

costituisce un punto di connessione tra i concetti di gruppo che sviluppa una tecnologia, ente che finanzia un progetto e la tecnologia implementata a cui ci si riferisce.

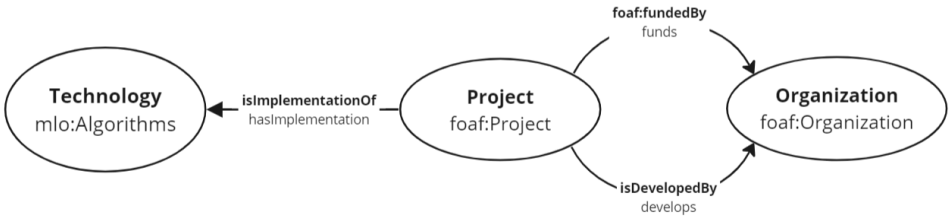

Figura 3.6: Schema Progetto

object property	domain	range	inverse of
foaf:fundedBy	Project	Organization	funds
isDevelopedBy	Project	Organization	develops
isImplementationOf	Project	Technology	hasImplementation

Tabella 3.5: Object Properties di Progetto

La classe *Progetto* importa da FOAF la proprietà *foaf:fundedBy* per indicare l'ente che finanzia il *Progetto*. La proprietà *isDevelopedBy* associa anch'essa un *Ente* a un *Progetto*, ma in questo caso indica l'*Ente* responsabile dello sviluppo del *Progetto*. Poiché è stato stabilito che un solo ente si occupa di ciascun *Progetto*, la proprietà *isDevelopedBy* è indicata come funzionale, asimmetrica e irriflessiva. D'altra parte, *foaf:fundedBy* non è considerata funzionale poiché nel dominio modellato potrebbe esserci più di un *Ente* che finanzia lo stesso *Progetto*.

L'ultima proprietà relativa a un Progetto è *isImplementationOf*, che indica a quale *Tecnologia* il *Progetto* fa riferimento. Anche in questo caso, un *Progetto* può fare riferimento a una sola *Tecnologia*, poiché si presume che per tecnologie o versioni diverse ci siano progetti di sviluppo separati. Di conseguenza, questa proprietà presenta nuovamente caratteristiche funzionali, asimmetriche ed irriflessive.

3.1.6 Tecnologia

La classe *Tecnologia* costituisce il nucleo centrale della Advanced Machine Learning Ontology, in quanto rappresenta un'implementazione specifica di un algoritmo, modello o tecnologia nel campo del Machine Learning. Questo concetto di *Tecnologia* è importato da *mlo:Algorithms* e rappresenta la materializzazione di un concetto teorico nel dominio del Machine Learning. Per ogni tecnologia implementata, è prevista la correlazione con al massimo un progetto di implementazione. Inoltre, versioni diverse relative allo stesso algoritmo vengono considerate come tecnologie implementate distinte.

È importante notare che ogni tecnologia implementata può partecipare a una sola edizione di una competizione specifica, poiché per edizioni diverse è necessario sviluppare implementazioni diverse, ciascuna costituendo una variante separata di uno specifico algoritmo di Machine Learning.

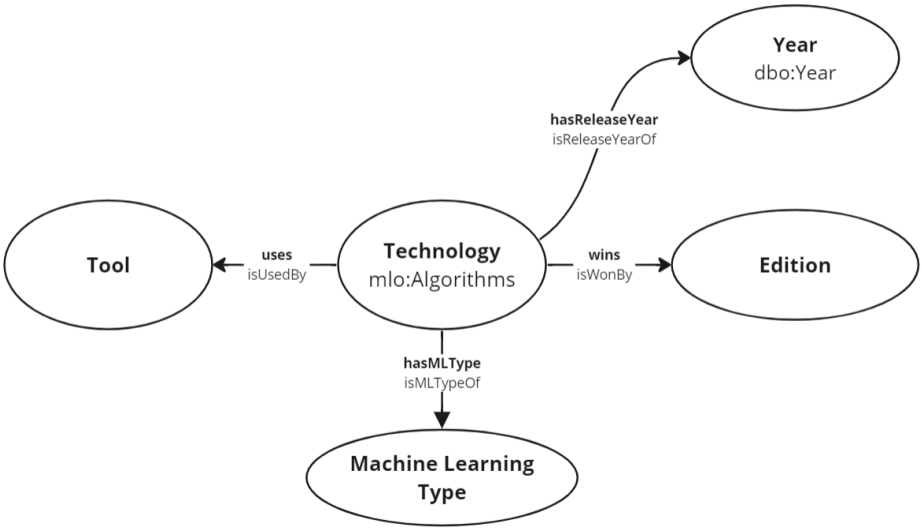

Figura 3.7: Schema Tecnologia

object property	domain	range	inverse of
hasReleaseYear	Technology	Year	isReleaseYearOf
wins	Technology	Edition	isWonBy
hasMLType	Technology	MachineLearningType	isMLTypeOf
uses	Technology	Tool	isUsedBy

Tabella 3.6: Object Properties di Tecnologia

La classe più complessa all'interno del dominio considerato, *Tecnologia*, presenta diverse proprietà. Una delle più semplici è *hasMLType*, che associa una *Tecnologia* a un *Tipo di Machine Learning*. Poiché ciascuna *Tecnologia* può avere un solo *Tipo di Machine Learning*, questa proprietà è stata definita come funzionale, asimmetrica ed irriflessiva.

Per indicare l'anno di rilascio di una *Tecnologia*, è stata utilizzata la proprietà *hasReleaseYear*. Poiché versioni diverse di una *Tecnologia* sono considerate distinte, anche questa proprietà è stata dichiarata come funzionale, asimmetrica ed irriflessiva.

La vittoria di una *Tecnologia* in una *Edizione* di una *Competizione* è modellata tramite la proprietà *wins*. Dal momento che ogni *Edizione* presenta variazioni uniche rispetto alle altre, una *Tecnologia* può vincere al massimo una *Edizione*, poiché per vincere edizioni diverse è necessario sviluppare implementazioni diverse per rispondere ai requisiti specifici dell'edizione stessa. Pertanto, anche questa proprietà è funzionale, asimmetrica ed irriflessiva.

Infine, per indicare un generico *Strumento* utilizzato per l'implementazione di una *Tecnologia*, è stata utilizzata la proprietà *uses*. Questa proprietà è stata mantenuta più generica e meno restrittiva rispetto alle altre, indicandola come asimmetrica ed irriflessiva.

3.1.7 Competizione

Competizione è importata da *dbo:Competition* e rappresenta un evento all'interno del quale diverse tecnologie si sfidano con lo scopo di risolvere uno specifico obiettivo. Diverse

competizioni possono avere scopi diversi, come ad esempio il riconoscimento facciale di esseri umani o il riconoscimento di pattern in dati scientifici. Tuttavia, nelle diverse edizioni della stessa competizione, il tema centrale rimane lo stesso, con alcune possibili variazioni nei dettagli.

Figura 3.8: Schema Competizione

object property	domain	range	inverse of
hasEdition	Competition	Edition	isEditionOf

Tabella 3.7: Object Properties di Competizione

La classe *Competizione* presenta una sola object property, *hasEdition*, che collega una *Competizione* a varie istanze di una *Edizione*. Poiché la *Competizione* è un concetto astratto in questo dominio, può essere associata a diverse edizioni che concretizzano nel mondo reale la sfida definita dalla *Competizione* corrispondente. Questa proprietà è relativamente semplice e presenta solo caratteristiche di asimmetria e irriflessività.

3.1.8 Edizione

Edizione rappresenta l'effettiva realizzazione di una competizione negli anni, coinvolgendo diverse tecnologie di Machine Learning nel tentativo di raggiungere un obiettivo specifico. In ciascuna *Edizione*, una delle tecnologie viene identificata come la migliore, vincitrice della competizione stessa. Inoltre, ogni edizione può essere associata agli *Organizzatori* responsabili della pianificazione e dello svolgimento dell'evento. Per indicare l'anno di svolgimento di una specifica edizione di una competizione, viene utilizzata la classe *dbo:Year*.

Figura 3.9: Schema Edizione

object property	domain	range	inverse of
hasEditionYear	Edition	Year	isEditionYearOf
isOrganizedBy	Edition	Organizer	organizes

Tabella 3.8: Object Properties di Edizione

Riferendosi l'*Edizione* ad un evento reale che si tiene in un determinato *Anno*, questo concetto viene modellato attraverso l'object property *hasEditionYear*. Poiché ogni *Anno* è associato a una *Edizione* differente di una specifica *Competizione*, questa proprietà è stata definita come funzionale, asimmetrica ed irriflessiva.

La pianificazione e l'organizzazione di una *Edizione* specifica di una *Competizione* sono gestite da uno o più organizzatori. Pertanto, è stata inserita la object property *isOrganizedBy*, che è asimmetrica e irriflessiva.

3.1.9 Categoria

La classe *Categoria* (:Category) è concepita per rappresentare aspetti del dominio che possono essere organizzati in famiglie o categorie. Nell'ambito dell'ontologia sviluppata, le sottoclassi di *Categoria* includono *Tipo di Strato* (:LayerType) e *Tipo di Machine Learning* (:MachineLearningType), ciascuna delle quali può contenere ulteriori sottoclassi o istanze specifiche per classificare aspetti rilevanti al proprio dominio.

```
###  https://github.com/RiccardoOmiccioli/AMLO/Category
:Category rdf:type owl:Class ;
        owl:equivalentClass [ rdf:type owl:Class ;
                owl:unionOf ( :MachineLearningType
                        :LayerType
                )
        ] ;
        rdfs:label "Category"@en ,
                "Categoria"@it .
```

Nella classe *Categoria* è stato inserito un vincolo per indicare che per una data *Categoria* esso può essere solamente *Tipo di Strato* o *Tipo di Machine Learning*.

Figura 3.10: Schema Categoria

3.1.10 Tipo di Machine Learning

Come accennato in precedenza, questa classe definisce una classificazione dei tipi di Machine Learning conosciuti. *Tipo di Machine Learning* (:MachineLearningType) ha tre sottoclassi principali: *Apprendimento Supervisionato* (:SupervisedLearning), *Apprendimento non Supervisionato* (:UnsupervisedLearning) e *Apprendimento per Rinforzo* (:ReinforcementLearning). Queste sottoclassi rappresentano le tre categorie fondamentali per la classificazione degli algoritmi di Machine Learning.

```
###  https://github.com/RiccardoOmiccioli/AMLO/
   MachineLearningType
:MachineLearningType rdf:type owl:Class ;
        rdfs:subClassOf :Category ;
        owl:equivalentClass [ rdf:type owl:Class ;
```

```
                owl:unionOf ( :ReinforcementLearning
                        :SupervisedLearning
                        :UnsupervisedLearning
                )
        ] ;
        rdfs:label "Machine Learning Type"@en ,
                "Tipo Machine Learning"@it .
```

Nella classe *Tipo di Machine Learning* è stato inserito un vincolo per indicare che per un dato *Tipo di Machine Learning* esso può essere solamente *Apprendimento Supervisionato*, *Apprendimento non Supervisionato* o *Apprendimento per Rinforzo*.

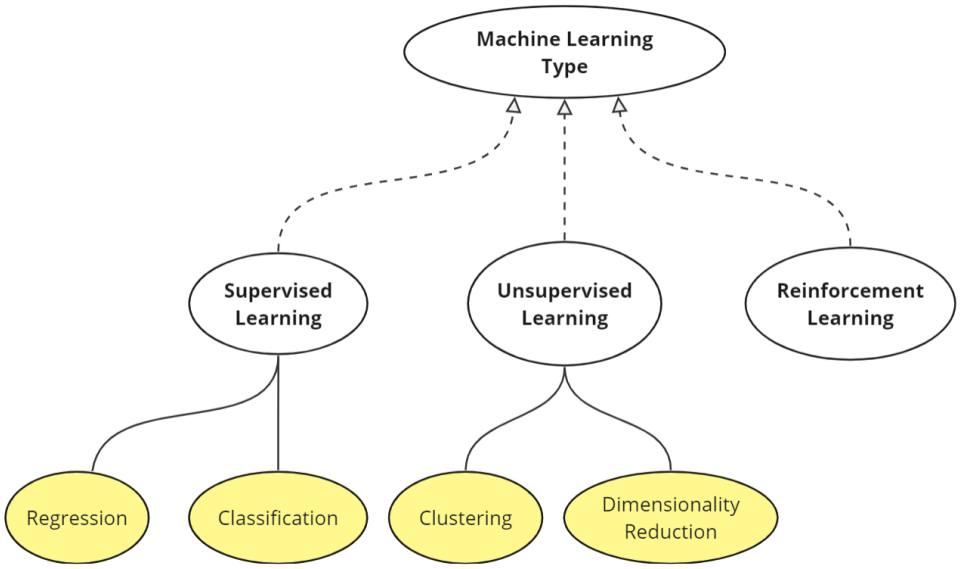

Figura 3.11: Schema Tipo di Machine Learning

3.1.11 Strumento

Strumento (:Tool) rappresenta un concetto generico che comprende elementi matematici o informatici utilizzati nelle tecnologie di Machine Learning. Nell'ambito di questa ontologia, sono state identificate sottoclassi come *Funzione di Attivazione* (:ActivationFunction), *Discesa del Gradiente* (:GradientDescent) e *Strato* (:Layer) come esempi di strumenti disponibili. Tuttavia, è importante notare che esistono numerosi altri strumenti che potrebbero essere utilizzati ma che potrebbero non essere ancora inclusi nell'ontologia.

```
###  https://github.com/RiccardoOmiccioli/AMLO/Tool
:Tool rdf:type owl:Class ;
        owl:equivalentClass [ rdf:type owl:Class ;
                owl:unionOf ( :ActivationFunction
                        :GradientDescent
                        :Layer
                )
        ] ;
        rdfs:label "Tool"@en ,
                "Strumento"@it .
```

Nella classe *Strumento* è stato inserito un vincolo per indicare che per un dato *Strumento* esso può essere solamente *Funzione di Attivazione*, *Discesa del Gradiente* o *Funzione di Attivazione*.

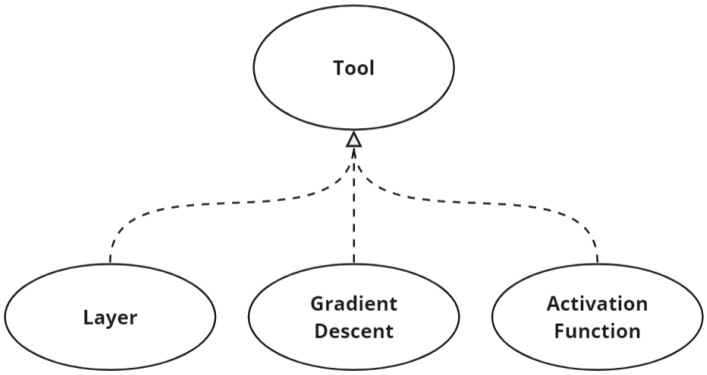

Figura 3.12: Schema Strumento

3.1.12 Strato

Strato (:Layer) è un tipo specifico di strumento utilizzato all'interno delle tecnologie di Machine Learning. Può essere caratterizzato da attributi come il numero di strato (*isNth*), le sue dimensioni altezza (*layerHeight*), larghezza (*layerWidth*) o profondità (*layerDepth*) e può essere associato a un *Tipo di Strato*, che rappresenta una delle possibili classificazioni relative al suo tipo.

```
###  https://github.com/RiccardoOmiccioli/AMLO/Layer
:Layer rdf:type owl:Class ;
        rdfs:subClassOf :Tool ;
        owl:equivalentClass [ rdf:type owl:Restriction ;
                owl:onProperty :hasLayerType ;
                owl:someValuesFrom :LayerType
        ] ;
        rdfs:label "Layer"@en ,
                "Strato"@it .
```

Nella classe *Strato* è stato inserito un vincolo per indicare che per la object property *hasLayerType* vengono accettate solamente istanze della classe *Tipo di Strato*.

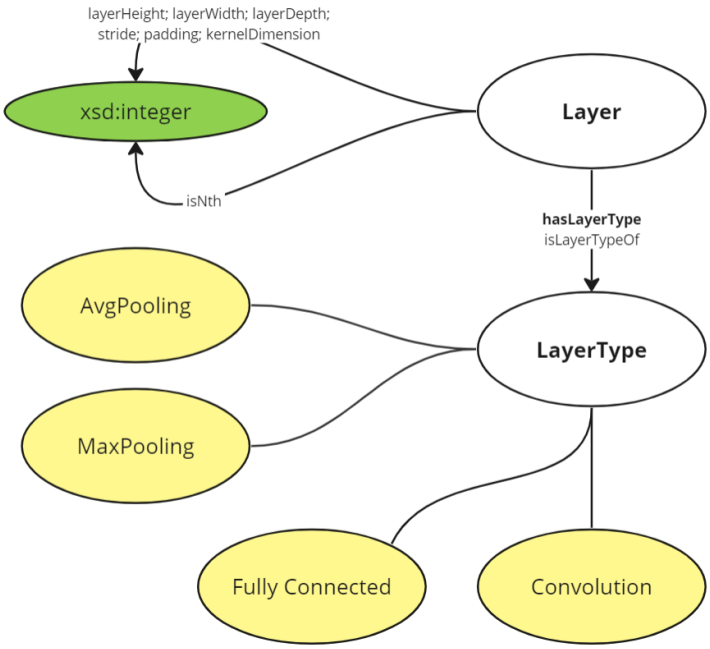

Figura 3.13: Schema Strato

object property	domain	range	inverse of
hasLayerType	Layer	LayerType	isLayerTypeOf

Tabella 3.9: Object Properties di Strato

La classe di uno *Strato* presenta una sola proprietà, *hasLayerType*, che serve per indicare il *Tipo di Strato* utilizzato. Poiché un *Strato* può essere associato a un solo *Tipo di Strato*, questa proprietà è stata dichiarata come funzionale, asimmetrica ed irriflessiva.

3.2 Regole SWRL

Alcune regole fondamentali per AMLO non potevano essere adeguatamente esposte attraverso i linguaggi precedentemente utilizzati. Pertanto, è stato utilizzato il **Semantic Web Rule Language** (SWRL) [6]. Estendendo OWL, SWRL consente di esprimere regole e vincoli di inferenza in modo avanzato e flessibile rispetto alle capacità di ragionamento di base fornite da OWL. Le regole SWRL consentono di definire relazioni complesse, rafforzare implicazioni e dedurre nuove informazioni nell'ontologia.

Di seguito è presente la spiegazione delle regole introdotte.

3.2.1 isPersonDifferent

Regola per indicare che ogni persona con nome e cognome differenti sono persone distinte.

```
familyName(?person1, ?familyName1) ^ familyName(?person2, ?
   familyName2) ^ firstName(?person1, ?firstName1) ^ firstName(?
   person2, ?firstName2) ^ swrlb:notEqual(?familyName1, ?
```

```
   familyName2) ^ swrlb:notEqual(?firstName1, ?firstName2) ->
   differentFrom(?person1, ?person2)
```

3.2.2 isCoauthor

Regola per indicare che due autori sono coautori se appartengono allo stesso gruppo ma sono due persone diverse.

```
hasRole(?person1, ?author1) ^ hasRole(?person2, ?author2) ^
   isMemberOf(?author1, ?group) ^ isMemberOf(?author2, ?group) ^
   differentFrom(?person1, ?person2) -> isCoAuthor(?author1, ?
   author2)
```

3.2.3 isAuthorWinner

Regola per indicare che un autore risulta vincitore di una edizione di una competizione se appartiene al gruppo che ha sviluppato la tecnologia vincitrice di quella determinata edizione.

```
Author(?author) ^ isMemberOf(?author, ?group) ^ isAssignedTo(?
   group, ?project) ^ isImplementationOf(?project, ?technology) ^
   won(?technology, ?event) -> isWinnerOf(?author, ?event)
```

3.2.4 isTeamOf

Regola per indicare che un gruppo è considerato un team dell'ente responsabile dello sviluppo del progetto su cui il gruppo sta lavorando.

```
isAssignedTo(?group, ?project) ^ isDevelopedBy(?project, ?
   organization) -> isTeamOf(?group, ?organization)
```

Capitolo 4

Dati e Interrogazioni

4.1 Inserimento dati

Essendo il programma un prototipo, sono stati inseriti solo dati sufficienti per testare la correttezza della struttura delle classi e alcune interrogazioni. I dati principali provengono dalla presentazione "Deep Learning for Image Classification" del professor Ferrara, fornita nel corso di "Visione Artificiale e Riconoscimento".

Per semplificare l'operazione, soprattutto in presenza di un elevato numero di voci ripetitive e dalla struttura simile, è stato sviluppato un piccolo programma Java. Questo permette di configurare l'architettura di una rete neurale attraverso oggetti e di stampare su un file le corrispondenti voci in sintassi Turtle (**Terse RDF Triple Language**) [7]. Ciò ha reso agevole modificare in blocco le decine di voci appartenenti alla classe Layer, evitando la necessità di riscriverle manualmente in caso di modifiche alla struttura della classe RDF corrispondente.

4.2 Interrogazioni

Per interrogare l'ontologia e le relative istanze, sono state sviluppate *query* utilizzando il linguaggio SPARQL (**Simple Protocol and RDF Query Language**) [2]. Con questo linguaggio è possibile formulare domande complesse per estrarre informazioni specifiche da dataset RDF. Di seguito, sono presentate ed analizzate le *query*, i cui risultati ci si è immaginati potessero essere interessanti.

4.2.1 Top 10 persone che hanno fatto più progetti

Avere accesso a una graduatoria di individui che hanno contribuito a diversi progetti consente di identificare gli individui più attivi nel campo del Machine Learning.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX : <https://github.com/RiccardoOmiccioli/AMLO/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>

SELECT ?person (COUNT(?project) AS ?projectCount)
WHERE {
  ?person a foaf:Person.
```

```
  ?person :hasRole/:isMemberOf/:isAssignedTo ?project.
}
GROUP BY ?person
ORDER BY DESC(?projectCount)
LIMIT 10
```

4.2.2 Persona che ha vinto più competizioni

Ottenendo la persona che ha vinto più competizioni si ottiene un nuovo tipo di informazione che non riassume semplicemente l'attività di una persona nel campo del Machine Learning, ma anche la sua bravura nel risolvere le sfide previste dalle competizioni.

Identificare la persona che ha ottenuto il maggior numero di vittorie in competizioni fornisce informazioni preziose che vanno oltre la semplice valutazione dell'attività di un individuo nel campo del Machine Learning, evidenziando l'abilità dell'individuo nel superare le sfide proposte da tali competizioni.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX : <https://github.com/RiccardoOmiccioli/AMLO/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>

SELECT ?person (COUNT(?edition) AS ?winCount)
WHERE {
  ?person a foaf:Person.
  ?person :hasRole/:isWinnerOf ?edition.
}
GROUP BY ?person
ORDER BY DESC(?winCount)
LIMIT 1
```

4.2.3 Organizzazioni che hanno finanziato più progetti

Esaminare gli enti che hanno finanziato il maggior numero di progetti può rivelarsi un elemento informativo rilevante per valutare l'entità dell'investimento e l'impegno degli enti nel settore del Machine Learning, consentendo di identificare coloro che offrono un supporto significativo alla ricerca e allo sviluppo di nuove tecnologie.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX : <https://github.com/RiccardoOmiccioli/AMLO/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>

SELECT ?organization (COUNT(?project) AS ?projectCount)
WHERE {
  ?organization a foaf:Organization.
  ?organization :funds ?project.
  ?project a foaf:Project.
}
GROUP BY ?organization
ORDER BY DESC(?projectCount)
LIMIT 10
```

4.2.4 Numero di tecnologie in ogni tipo di Machine Learning

Mostrare il conteggio delle tecnologie per le diverse categorie di Machine Learning fornisce un contesto che aiuta a comprendere quanto siano utilizzate le varie categorie, evidenziando eventuali tendenze in momenti specifici.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX : <https://github.com/RiccardoOmiccioli/AMLO/>

SELECT ?mlType (COUNT(?technology) AS ?technologyCount)
WHERE {
  ?mlType a :MachineLearningType.
  ?technology :hasMLType ?mlType.
}
GROUP BY ?mlType
```

4.2.5 Media delle persone che lavorano in un gruppo

Il calcolo della media delle persone coinvolte in un gruppo che sviluppa progetti di Machine Learning può essere utile per valutare il fabbisogno medio di risorse, sia dal punto di vista umano che economico, necessario per lo sviluppo di tali progetti.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX : <https://github.com/RiccardoOmiccioli/AMLO/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>

SELECT (AVG(?numMembers) AS ?averageMembersPerGroup)
WHERE {
  ?group a foaf:Group.
  {
    SELECT ?group (COUNT(?author) AS ?numMembers) WHERE {
      ?group :hasMember ?author.
    }
    GROUP BY ?group
  }
}
```

4.2.6 Algoritmi che hanno numero di layer $15 \leq x \leq 25$

Tramite questa interrogazione, si intende fornire un filtro specifico per individuare gli algoritmi che possiedono un livello di complessità strutturale che rientra in una fascia predeterminata tra i 15 e i 25 compresi. L'uso di questa query consente di concentrarsi su una categoria specifica di algoritmi, offrendo un'opportunità di analisi approfondita su quelli che presentano una specifica quantità di layer. Tale approccio permette di estrarre informazioni mirate e rilevanti, contribuendo a una migliore comprensione della distribuzione e della complessità degli algoritmi all'interno del dataset.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX mlo: <http://www.a2rd.net.br/mlo#>
PREFIX : <https://github.com/RiccardoOmiccioli/AMLO/>
```

```
SELECT DISTINCT ?technology
WHERE {
  ?technology a mlo:Algorithms.
  ?technology :uses ?layer.
  ?layer a :Layer.
  ?layer :isNth ?position.
  FILTER (?position >= 15 && ?position <= 25)
}
```

4.2.7 Numero di layer degli algoritmi

La visualizzazione del numero di layer degli algoritmi consente di presentare una classifica delle tecnologie in base al numero di layer che le compongono. Questa informazione consente di visualizzare chiaramente la gerarchia delle tecnologie in base alla complessità delle loro strutture, fornendo una panoramica dettagliata della distribuzione dei layer nelle diverse tecnologie presenti nel dataset.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX mlo: <http://www.a2rd.net.br/mlo#>
PREFIX : <https://github.com/RiccardoOmiccioli/AMLO/>

SELECT ?technology (COUNT(?layer) AS ?layerCount)
WHERE {
  ?technology a mlo:Algorithms.
  ?technology :uses ?layer.
  ?layer a :Layer.
}
GROUP BY ?technology
ORDER BY DESC(?layerCount)
LIMIT 10
```

4.2.8 Riepilogo di una tecnologia

Il riepilogo di una tecnologia serve a fornire un riassunto esauriente delle informazioni chiave relative ad essa. L'interrogazione estrae dati significativi, tra cui il progetto associato alla tecnologia, l'anno di rilascio, il gruppo di lavoro coinvolto e i relativi autori. Inoltre, la query include dettagli sulle competizioni vinte dalla tecnologia specifica. Questo riepilogo mira a fornire una visione completa del contesto e della sua realizzazione.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX : <https://github.com/RiccardoOmiccioli/AMLO/>
PREFIX mlo: <http://www.a2rd.net.br/mlo#>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX dbo: <http://dbpedia.org/ontology/>

SELECT ?technology ?project ?releaseYear ?group (GROUP_CONCAT(
   DISTINCT ?name; separator=", ") AS ?authorsList) (GROUP_CONCAT(
   DISTINCT ?edition; separator=", ") AS ?editionsWinsList)
```

```
WHERE {
  ?technology a mlo:Algorithms.
  ?technology :hasImplementation ?project.
  ?technology :hasReleaseYear ?releaseYear.
  ?project :hasAssigned ?group.
  ?group :hasMember ?member.
  ?member :isRoleOf ?person.
  ?person :firstName ?firstName.
  ?person :familyName ?familyName.
  BIND(CONCAT(?firstName, " ", ?familyName) AS ?name)
  ?technology :wins ?edition.
}
GROUP BY ?technology ?project ?releaseYear ?group
```

Capitolo 5

Conclusioni

Siamo consapevoli che, nonostante gli sforzi compiuti, l'ontologia potrebbe non coprire tutti gli aspetti salienti del dominio di riferimento. Pertanto, questa ontologia viene considerata come un prototipo per modellare gli aspetti principali.

Contrariamente all'ontologia MLO, che si focalizza sulla modellazione degli algoritmi e degli aspetti matematici generici, insieme agli autori e ai contesti di utilizzo, il nostro obiettivo è approfondire gli aspetti tecnico-matematici direttamente correlati agli algoritmi stessi. Ciò include l'inclusione di competizioni alle quali alcuni algoritmi partecipano, l'identificazione degli organizzatori delle relative edizioni e degli enti coinvolti, sia tra gli organizzatori che tra gli autori. Un'altra caratteristica distintiva di AMLO è la scelta di riutilizzare elementi da altre ontologie, essendo concetti corretti e validati, al fine di promuovere l'interoperabilità ottimale con tali ontologie.

Per lo sviluppo futuro, vengono considerati i seguenti aspetti:

- L'aggiunta di campi di applicazione specifici per i vari algoritmi;
- Il miglioramento dell'organizzazione e l'espansione delle informazioni relative a progetti, gruppi di ricerca e le relazioni con gli enti;
- L'aggiunta di informazioni riguardanti la struttura organizzativa dei dipartimenti di ricerca di università, aziende e istituti vari.

Bibliografia

[1] DBpedia. https://www.dbpedia.org/resources/ontology/.

[2] SPARQL Query Language for RDF. https://www.w3.org/TR/rdf-sparql-query/.

[3] RDF Resource Description Framework. https://www.w3.org/RDF/.

[4] Machine Learning Ontology GitHub. https://github.com/MLOntology/MLO.

[5] OWL Web Ontology Language. https://www.w3.org/TR/owl-ref/.

[6] SWRL: A Semantic Web Rule Language. https://www.w3.org/submissions/SWRL/.

[7] Turtle Terse RDF Triple Language. https://www.w3.org/TeamSubmission/turtle/.

[8] FOAF Foaf Friend of a Friend. http://xmlns.com/foaf/0.1/.

[9] AIISO Academic Institution Internal Structure Ontology. https://vocab.org/aiiso/.

[10] RDF Schema. https://www.w3.org/TR/rdf12-schema/.

[11] DublinCore Metadata Terms. https://www.dublincore.org/specifications/dublin-core/dcmi-terms/.