Data Integration: Arlecchino System

Matteo Wissel, Paolo Di Simone, Pietro Baroni
Dipartimento di Ingegneria Informatica Roma 3
Roma 22/02/23

Introduzione

Data Integration

La qualità delle decisioni prese in un contesto sempre più Data-Driven è direttamente influenzato delle informazioni possedute e dalla facilità di acquisizione di esse. Tuttavia, molto spesso, tali informazioni sono memorizzate in sorgenti distinte.

A tal proposito negli anni sono stati presentati numerose soluzioni di Data-Integration. Tali sistemi hanno l'obiettivo di integrare informazioni provenienti da sorgenti distinte in base alla relazione semantica tra le diverse colonne delle tabelle di partenza.

Use Case

L'obiettivo principale del progetto è stata la creazione di uno schema unico contenente informazioni derivanti dai dataset sviluppati da 11 team distinti durante l'Homework 5 del corso di Ingegneria dei Dati mediante tecniche di Web-Scraping. Tali dataset contengono informazioni eterogenee relative ad aziende memorizzate in formati distinti.

A tale scopo è stato implementato un sistema *ad-hoc*, Arlecchino System, capace di eseguire tutti i passaggi di Data Integration (Schema Matching e Record Linkage) indipendentemente dalle tabelle in input.

Dataset

Il Dataset di partenza è costituito da 44 tabelle (4 per ogni team) provenienti da 14 pagine web distinte.

Clustering del Dataset

In quanto numerose tabelle del Dataset sono state generate estraendo dati da una medesima sorgente web, per l'implementazione della pipe-line di Data Integration, abbiamo ritenuto opportuno raggruppare le singole tabelle rispetto ai siti web sorgenti (*Figura 1*).

	Sorgente	Numero file
1	ambitiobox	1
2	ariregister	1
3	cbinsights	2
4	companiesmarketcap (cmc)	8
5	disfold	9
6	forbes	3
7	ft	4
8	globaldata	1
9	govuk	2
10	hithorizons	1
11	infoclipper	1
12	teamblind	1
13	valuetoday	6
14	wikipedia	3

Figura 1-Tabelle per cluster

Analisi dei sorgenti

Poiché l'obiettivo principale del progetto è individuare le relazioni semantiche tra le diverse tabelle, per avere una visione più dettagliata del Dataset si è ritenuto opportuno classificare i clusters rispetto al contenuto informativo delle loro tabelle.

Le labels scelte sono quattro:

1. Dati finanziari;
2. Dati geografici;
3. Dati relativi al personale;
4. Dati giuridici.

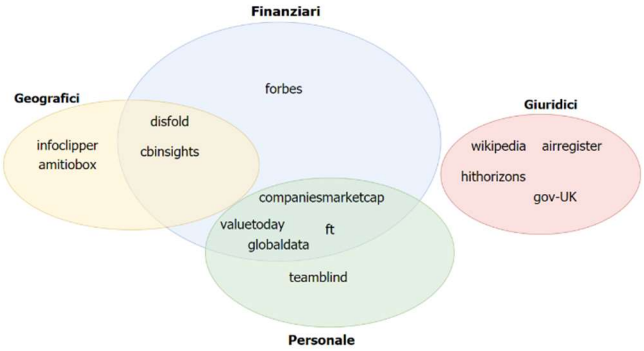

Figura 2-Classificazione clusters

Come si può vedere dalla *Figura 2*, non esiste un cluster che contenga tutti e quattro i tipi di informazione, questo implica che, per avere una visione unificata delle quattro tipologie di informazioni ricercate, è necessario intervenire integrando le tabelle del Dataset.

Schema Mediato Target

Per le motivazioni elencate nel capitolo precedente lo schema mediato finale proposto (*Figura 3*) è tale da racchiudere informazioni derivanti da ognuna delle quattro aree di interesse e permettere quindi di avere una visione unificata dei dati interessati.

Figura 3-Schema Mediato

Architettura

Il sistema implementato per la creazione della tabella integrata (Schema mediato) è riportato in *Figura 4.*

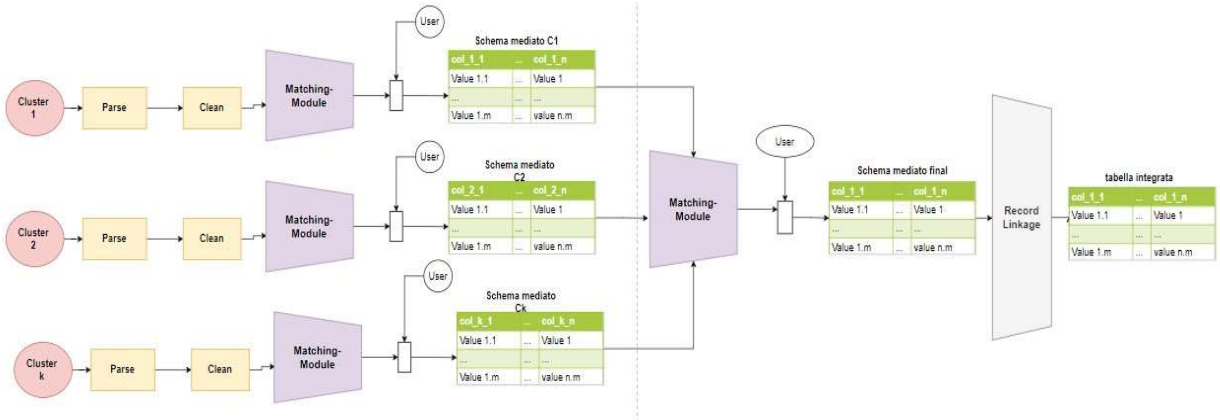

Figura 4-Architettura Arlecchino

L'architettura di Arlecchino può essere suddivisa nei seguenti tre moduli:

1. **Parsing & Cleaning Module**: modulo specializzato nella pulizia e standardizzazione dei dati presenti nelle tabelle di partenza;
2. **Matching Module**: modulo custom specializzato nell'operazione di Schema Matching;
3. **Record Linkage Module**: modulo specializzato nelle operazioni di Record Linkage (Blocking, Linkage e Join).

Funzionamento

Arlecchino System persegue il task di Data Integration applicando in modo iterativo il Matching Module. Quest'ultimo, infatti, viene prima applicato su ogni cluster per generare uno schema mediato di cluster (visione unificata dei dati interessanti presenti all'interno di un singolo cluster), lo stesso modulo viene poi utilizzato per integrare i singoli schemi mediati di cluster ed ottenere lo schema mediato finale. Le istanze di Matching Module che operano sui cluster inoltre, inferiscono informazioni sul problema all'istanza finale, in quanto alimentano un dizionario unificato dei sinonimi individuati nei singoli cluster (*dizionario sinonimi pregressi*)

Una volta generato lo schema mediato finale, quest' ultimo viene dato in input al Record Linkage Module il quale genera lo schema Integrato finale.

In questa relazione approfondiamo il ***Matching Module*** ed il ***Record Linkage Module***.

Schema matching – Matching Module

Formulazione problema

Il problema di Schema Matching è stato formulato nel seguente modo:

$$\forall\, column\ c \in\ C \rightarrow S = f(c)\ \ dove\colon\ S\ set\ of\ sinonimi\ di\ c, \qquad C\ set\ of\ columns, \qquad |C| = N$$

Analizzando la funzione f(c) si nota come domino e codominio (search space) hanno entrambi cardinalità O(N) ed il numero totale di confronti potenziali è $O(N^2)$.

Architettura Matching Module

L'operazione di schema matching è stata svolta interamente dal Matching Module, un modulo custom responsabile nell'individuazione dei sinonimi delle colonne in input mediante analisi *data-wise e column-wise.* La Figura 5 riassume l'architettura del modulo.

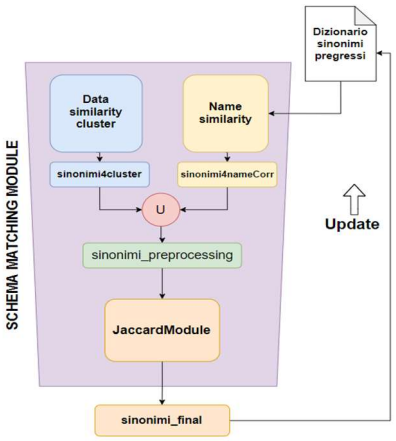

Figura 5-Matchin Module

Il modulo è caratterizzato da due componenti: Preprocessing e JaccardModule. Ognuno dei moduli ha l'intento di individuare per ogni colonna la lista di sinonimi considerando similarità di nome della colonna e similarità a livello di dato.

Preprocessing

La componente Preprocessing ha il compito di generare un dizionario parziale di sinonimi tale da ridurre il *search space*. L'obbiettivo principale è quello di ridurre il numero di confronti operati da JaccardModule (collo di bottiglia della pipeline) massimizzando la *Recall* e definendo per ogni colonna una lista di possibili sinonimi computati unendo i sinonimi identificati dai blocchi *NameSimilarity* e *DataCorrelation.*

Questi due ricercano relazioni semantiche tra colonne operando in domini distinti:

1. La ***NameSimilarity*** opera a livello di schema (*Figura 6*)

Algorithm 1 NameSimilarity

Require: T_{name}
  for all $(c, c') \in (C \times C)$ **do**
    $score_1(c, c') \leftarrow avg(3gram(c, c'), 4gram(c, c'))$
    $score_2(c, c') \leftarrow Jaccard(sin_{pre}(c), sin_{pre}(c'))$
    $A \leftarrow (sin_{pre}(c) - sin_{pre}(c'))$
    $B \leftarrow (sin_{pre}(c') - sin_{pre}(c))$
    $score_3(c, c') \leftarrow avg(\sum_{c_a \in A, c_b \in B} score_1(c_a, c_b))$
    $Score(c, c') \leftarrow avg(score_1(c, c'), score_2(c, c'), score_3(c, c'))$
    if $Score(c, c') \geq T_{name}$ **then**
      $match \leftarrow (c, c')$
    end if
  end for

Figura 6-Name Similarity module

   In questa fase si fa inoltre uso del *dizionario dei sinonimi pregressi,* un dizionario contenente tutti i sinonimi individuati e validati dal sistema fino a quel punto.
2. La ***DataCorrelation*** opera a livello di dati. Trasforma ogni colonna in un vettore di 17 features ed individua i sinonimi mediante utilizzo di un modello ML non supervisionato (*K-Means*)

E' stato empiricamente testato che il numero di confronti operato dal JaccardModule in presenza di *Preprocessing* si riduce di un fattore medio di 8.

JaccardModule

I sinonimi computati nello step di Preprocessing vengono dati in input al JaccardModule, quest'ultimo genera un file .csv per ogni colonna c presente nel dizionario dei sinonimi in input contenente tutte le colonne reputate sinonimi. Per ogni file, il modulo individua i sinonimi veri operando anche in questo caso sia schema-wise che data-wise. Nello specifico, la logica può essere riassunta come in *Figura 7*.

Algorithm 2 JaccardModule

Require: T_{edit}, $T_{jaccard}$
  for all $A \in directory$ **do**
    for all $(i, j) \in \mathcal{A}$ **do**
      $score_{edit} \leftarrow Levenshtein(col(i), col(j))$
      if $score_{edit} \leq T_{edit}$ **then**
        $match \leftarrow (i, j)$
      else
        $score_{jaccard} \leftarrow Jaccard(i, j)$
        if $score_{jaccard} \geq T_{jaccard}$ **then**
          $match \leftarrow (i, j)$
        end if
      end if
    end for
  end for

Figura 7-Logica JaccardModule

Output

L'output del MatchingModule è un dizionario di sinonimi in cui, per ogni colonna definita nello schema mediato, sono presenti una lista di possibili colonne semanticamente uguali (sinonimi). L'utente seleziona i match eliminando eventuali errori del sistema.

Al netto della validazione dell'utente, il sistema opera un update del dizionario dei sinonimi pregressi inserendo le nuove relazioni semantiche individuate nell'esecuzione corrente.

Infine, viene creato lo schema mediato inserendo nelle colonne i dati provenienti dai relativi sinonimi. Nella *Figura 8* sono riportate le statistiche relative allo schema Mediato computato in termini di valori univoci e nulli per ogni colonna.

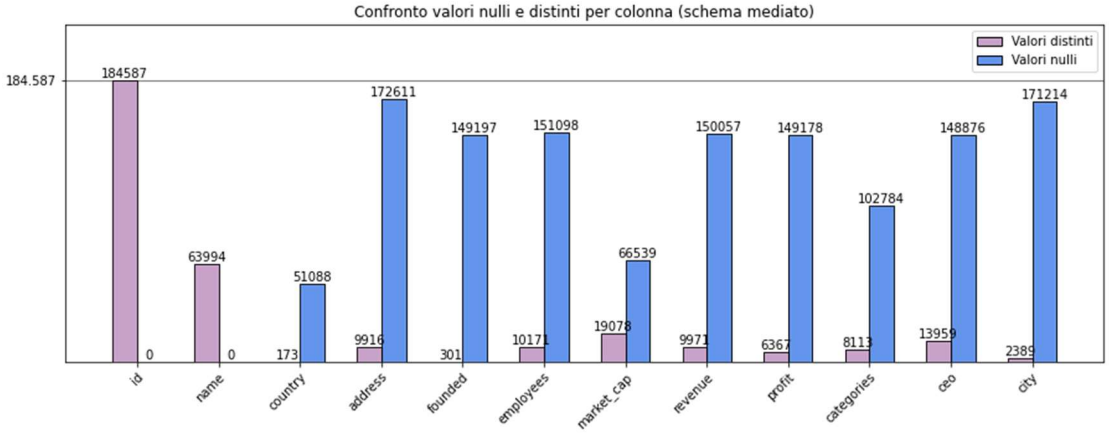

Figura 8-Statistiche Schema Mediato

Record linkage - Record Linkage Module

Proseguendo l'analisi della pipeline, l'ultimo modulo è il Record Linkage Module, la cui architettura è riassunta in *Figura 9.*

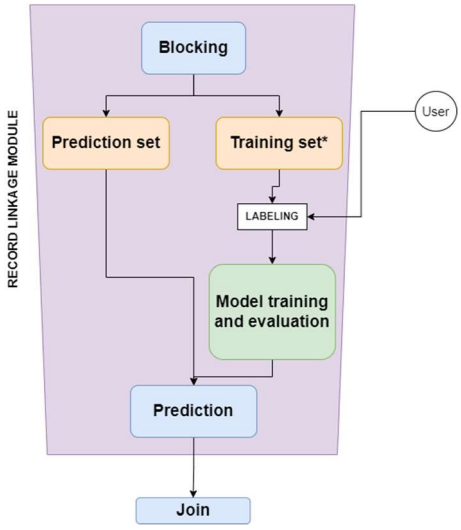

Figura 9-Architettura Record Linkage Module

Quest'ultimo riceve in input lo schema mediato computato dal MatchingModule sul quale vengono effettuate un insieme di operazioni volte ad individuare quali record fanno riferimento alla stessa entità. L'output di tali operazioni sarà utilizzato per la costruzione della tabella finale.

Il Record Linkage Module è stato implementato combinando due librerie principali:

1. py_entitymatching(Magellan);
2. Deepmatcher.

Blocking

Per determinare se i record dello schema mediato siano relativi ad una stessa entità in fase di record linkage si devono eseguire confronti tra tutti i record della tabella. Nel nostro caso il numero di confronti è di oltre 34 miliardi quindi, per ridurre la complessità computazionale di tale operazione, sono stati impiegati due tipologie di blockers, offerte dalla libreri py_entitymatching.

Nello specifico:

- ***overlap blocker:*** elimina dai confronti tutti record che in un determinato campo non hanno almeno un numero, scelto dall'utente, di parole in comune;
- ***rule based blocker***: utilizza regole custom specifiche per il dominio in esame.

Le operazioni di blocking che abbiamo eseguito sono state:

1. Blocking di coppie che non hanno overlap di una parola nel nome dell'azienda;
2. Blocking di coppie che non hanno overlap di una parola nello stato dell'azienda (se presente);
3. Blocking di coppie nelle quali la distanza di Levenshtein tra i nomi delle aziende è minore di 0.7.

Tali operazioni hanno permesso di ridurre notevolemente il numero di confronti. Nello specifico:

- Confronti pre-blocking = 34.072.360.569;
- Confronti post-blocking = 2.177.713.

Modello

Una volta determinate le possibili coppie di record relative ad una stessa azienda, il modulo implementa le operazioni di Entity Matching. Queste sono state implementate utilizzando la libreria deepmatcher, la quale fornisce modelli di reti neurali specializzati in task di EM.

Tutti le soluzioni DL al problema di entity matching sono divise in tre parti:

- Attribute Embedding;
- Attribute Similarity Representation;
- Classification.

Anche nel modello utilizzato, detto Matching Model, sono presenti queste tre fasi: la prima è eseguita da fastText, ovvero un modello pre-addestrato per l'apprendimento di embeddings di parole, sull'input del modello. La seconda nel modello scelto è divisa in due sezioni l'Attribute Summarizer e l'Attribute Comparator, tra le varie versioni proposte dagli autori l'attribute summarizer scelto è quello ibrido ovvero un modello implementato mediante una RNN bidirezionale. La terza, invece, è eseguita da un layer Softmax. Abbiamo scelto l'attribute summarizer ibrido perché confrontandolo con gli altri sviluppati gli autori hanno convenuto che sia quello con maggior potere rappresentazionale.

Training

Per l'addestramento del modello, sono state estratte casualmente 1300 coppie di record. Queste sono state classificate manualmente in:

- 1, se la coppia si riferisce alla stessa entità (match);
- 0, se i due record si riferiscono ad entità diverse (no-match).

Le coppie sono state divise training, test, e validation set in rapporto 3:1:1. La distribuzione dei valori della label di tali record è abbastanza uniforme, ovvero sono presenti 695 label match e 605 label no-match.

Addestramento

Il modello di EM è stato addestrato utilizzando i seguenti iperparametri;

- Epoche: 10;
- Batch-size: 16;
- Salvataggio del modello con F1 più alta calcolata sul validation set.

Dopo aver addestrato il modello lo abbiamo impiegato per la predizione delle coppie di record definite dopo il blocking, questa operazione è durata circa 14 ore e 45 minuti, ed il risultato è stato un numero di match pari a 1.480.727 su 2.177.713 predizioni.

Join

Definiti i record relativi alla stessa azienda, questi sono stati unificati in un unico record mediante una funzione custom di join la quale inserisce nel campo x del record finale il valore con più occorrenze tra i record match. Le statistiche relative alla tabella generata prima della fase di arricchimento (Data Enrichment) sono riassunte in *Figura 10.*

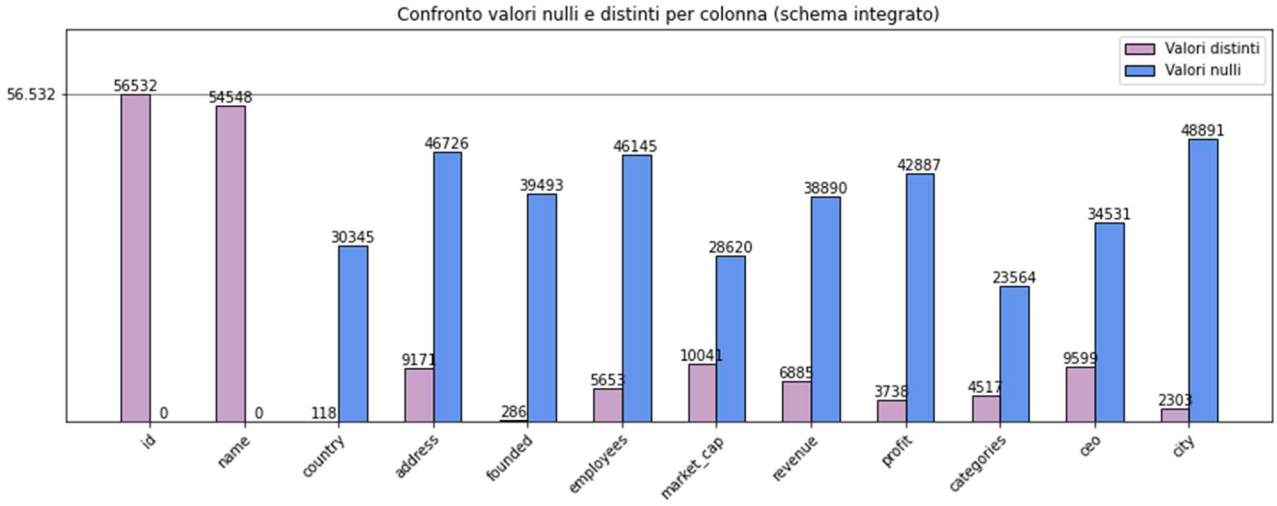

Figura 10-Statistiche schema mediato pre-arricchimento

Arricchimento

Ottenuta la tabella finale, questa è stata arricchita utilizzato l'algoritmo di merge implementato durante l'Homework 3 sull'indice ottenuto nel medesimo homework.

La ricerca nell'indice è stata svolta dando come input le colonne più significative della tabella, ovvero quelle che contengono dati che possono ricondurre più facilmente all'entità stessa ovvero Name e Ceo. Grazie a questa operazione sono stati inseriti 142 nuovi valori nelle colonne Country, Revenue, Profit, Ceo, City.

Prestazioni

Vengono riportate in questa sezione i risultati raccolti in fase di testing del sistema Arlecchino. Nello specifico, sia per il Matching Module che per il Record Linkage Model, le metriche di valutazione sono:

1. Precision: (indice del lavoro umano in fase di validazione dei sinonimi);
2. Recall;
3. F1.

Valutazione Matching Module per cluster

Cluster	Precision	Recall	F1	N.C. effettuati	N.C. senza pre-processing
ambitiobox	1	1	1	0	0
ariregister	1	1	1	0	0
cbinsights	1	1	1	13	153
cmc	0.79	1	0.86	302	1 653
disfold	0.93	0.98	0.93	321	2 415
forbes	0.58	0.9	0.61	27	171
ft	0.83	0.92	0.87	135	561
globaldata	1	1	1	0	0
govuk	1	1	1	16	136
hithorizons	1	1	1	0	0
infoclipper	1	1	1	0	0
teamblind	1	1	1	0	0
valuetoday	0.71	0.76	0.68	235	1 953
wikipedia	1	0.88	0.92	40	435

Valutazione Matching Module final

	Precision	Recall	F1	N.C. effettuati	N.C. senza pre-processing
Schema mediato	1	0.96	0.98	1 053	12 561

Valutazione Record Linkage model

	Precision	Recall	F1
Prestazioni del modello sul validation set	89.47	95.18	92.20
Prestazioni del modello sul test set	87.20	87.20	87.20