Continual Learning over temporal data

January 29, 2021

1 Caso D'Uso

All'interno di un contesto aziendale [1], le tracce deviant possono essere potenzialmente di questi tipi:

- Le situazioni previste dalle regole aziendali che non vanno a buon fine (es. reclami di prodotti guasti o consegne in ritardo).
- Vincoli aziendali che non vengono soddisfatti (es., non viene gestita la risposta al cliente come atteso dalle regole).
- Si violano vincoli di integrità referenziale presenti nei dati.

Questi sono tutti esprimibili tramite vincoli di integrità, che sono rilevanti nel constesto di ragionamento abduttivo, in quanto impongono delle restrizioni negli stati consentiti all'interno di un database. Tuttavia, ci chiediamo se questi vincoli possano essere totalmente espressi utilizzando solamente Declare o Declare Data Aware. Uno dei principali problemi di questo linguaggio è che non consente di esprimere vincoli tra termini appartenenti ad eventi differenti, quando i termini non sono valori temporali:

- ad esempio, Declare riesce a gestire il caso errato in cui l'ordine viene spedito prima che arrivi il pagamento del cliente,
- ma non riesce a gestire il caso in cui venga spedito un prodotto con il nome differente rispetto a quello richiesto dal cliente contenuto...

Osserva inoltre che questo caso d'uso non può essere nemmeno caratterizzato dai descrittori quali `SHAP` o `LIME`, o anche un albero binario o RIPPERk, in quanto effettuano spiegazioni che analizzano singoli valori attribuiti alle variabili. Quindi, nei constesti reali, è necessario utilizzare un linguaggio più espressivo.

D'altro canto, sia Declare sia Declare Data Aware non riesce ad esprimere regole di chain precedence e chain consequence tra tre eventi, ma le dipendenze tra eventi sono solamente tra due a due. D'altro canto, sebbene Frequent Rule Mining riesce ad esprimere correlazioni tra più di due eventi, queste correlazioni non sono stabilite tramite un nesso temporale, in quanto si trattano di correlazioni di coesistenza, indipendentemente dal fattore temporale.

Quindi, è necessario utilizzare una descrizione più potente. Inoltre, l'approccio di explainability proposto precedentemente in [2] non consente di differenziare variabili presenti in differenti eventi, e non considera la possibilità che un evento

possa accadere più volte all'interno dello stesso processo. Sebbene questo possa essere facilmente rappresentabile associando ad ogni evento il suo numero di istanza, estendendo così la dimensione della rappresentazione vettoriale arbitrariamente, questa soluzione non è sufficientemente generale, e non scala per dati arbitrari, in quanto è sempre necessario stabilire a priori il numero delle istanze.

2 Learning vs Mining

La rappresentazione della conoscienza si basa sulla rappresentazione formale di pattern, funzioni, ipotesi e dati. Si assume quindi che esista un linguaggio dei dati o esempi $\mathcal{L}_e$, che descrive le istanze dei dati osservabili, mentre le descrizioni di questi dati sono caratterizzabili da un linguaggio delle ipotesi , che forniscono una *spiegazione* dei dati. Nei problemi di apprendimento come in **machine learning**, il linguaggio delle ipotesi è in particolare l'insieme delle funzioni $\mathcal{L}_h = \mathcal{Y}^{\mathcal{L}_e}$ che mappano i dati su di un determinato dominio $\mathcal{Y}$. Queste ipotesi cercheranno quindi di apprendere una funzione incognita $f \colon \mathcal{L}_e \to \mathcal{Y}$ dei dati da un insieme di esempi $E \subseteq \mathcal{L}_e \times \mathcal{Y}$ che minimizzi la funzione di perdita *loss*, ovvero:

$$\hat{f} = \arg\min_{h \in \mathcal{L}_h} loss(h, E)$$

A questo punto, vogliamo valutare la tesi in base la quale una classificazione dei dati reali tramite una funzione fornisce una descrizione riduttiva della realtà, che può essere rappresentata tramite una rappresentazione più completa e meno approssimata.

A questo punto, riduciamoci ad un problema binario di classificazione come il problema di soddisfacibilità, per cui $\mathcal{Y} = \{ \top, \bot \}$. Supponiamo a questo punto che $\mathcal{Y}$ fornisca l'etichettamento dei dati $\mathcal{L}_e$ per i quali i dati aziendali sono stati etichettati manualmente da un utente (quindi, con errore) distinguendo quali tracce possono essere deviant da quali no. In aggiunta, un business analyst fornisce un insieme di vincoli di dominio $\theta \in \Theta$, dove si forniscono le caratteristiche che devono essere possedute da tutte le tracce ritenute deviant. Nell'assunzione preliminare che Θ caratterizzi completamente le tracce $t \in \mathcal{L}_e$ per cui $f(t) = \top$, Θ è necessario per controllare che non vengano generate regole $H \subseteq \mathcal{L}_h$ da E che validano le tracce marcate come deviant. In particolare, il principio abduttivo (es. Castor) mi garantice (con qualche approssimazione) che:

- $\Theta, H \vDash \{ (\mathbf{x}, y) \in E \mid y = 1 \}$
- $\Theta, H \nvDash \bot$

In particolare, possiamo stimare numericamente il primo predicato tramite la metrica di precision e caratterizzare la seconda tramite metriche di inconsistenza di una teoria logica tramite MIS.

A questo punto, dobbiamo tuttavia supporre che Θ non caratterizzi completamente le tracce non-deviant. Possiamo osservare che esistono necessariamente:

- tracce che sono deviant sia per i dati sia per Θ (true negatives)

- tracce che non sono deviant sia per i dati sia per Θ (true positives)
- tracce che sono solo deviant nei dati ma non per i vincoli in Θ
- tracce che sono non deviant nei dati ma deviant per i vincoli

Osserva che non possiamo ascrivere direttamente questi due casi d'uso ad errori di classificazione (falsi positivi e falsi negativi), in quanto può essere che Θ non mi fornisca una caratterizzazione completa dei dati di mio interesse.

Esempio 1. *Supponiamo che la caratterizzazione Θ contenga solamente un predicato θ che distingue le tracce che hanno un ticket di assistenza problemi dalle altre. Tuttavia, non tutte le tracce θ caratterizzano correttamente le tracce problematiche: infatti, un utente può etichettare manualmente quelle tracce come problematiche se è stato richiesta assistenza ma questa non è stata effettuata, o se non sono stati segnalati problemi ma è stato consegnato un prodotto prima di ricevere il saldo dell'acquisto del cliente.*

Quindi, è necessario estendere Θ almeno come due modelli Θ_1 e Θ_2, dove la riparazione di Θ tramite gli esempi positivi positivi è descritta da $\Theta_1 \vee \Theta_2$. L'obiettivo dell'apprendimento è quindi quello di estendere Θ, e non quello di ottenere solamente gli elementi che si conformano ad esso. Questa è quindi una *prima critica al procedimento puramente abduttivo*, che **non considera possibile aggiornare a tempo d'esecuzione i vincoli** Θ per ottenere un miglioramento della teoria, senza però cambiare il marcamento dei dati. Nel contempo, questa fornisce anche un'evidenza della *mancanza di adeguata espressività dei modelli di machine learning*, in quanto non sono in grado di esprimere molteplici ipotesi per una stessa caratterizzazione dei dati. D'altro canto, gli algoritmi di **data mining** non generano un singolo modello $\hat{f}$ che descrive i dati, ma più possibili descrizioni o ipotesi h che soddisfano un criterio di qualità sui dati $\mathcal{Q}$, quali la precisione/accuratezza dei risultati. Gli algoritmi di data mining sono quindi genericamente descrivibili dalla seguente funzione di enumerazione delle soluzioni:

$$Th(\mathcal{Q}, (P, N)) = \{\, h \in \mathcal{L}_h \mid \mathcal{Q}(h, P, N) \,\}$$

Assumiamo a questo punto di poter aggiornare Θ fino a convergenza desiderata, ovvero finché le regole estratte dai dati marcati positivi coincidono esattamente con una possibile teoria estesa Θ. Il processo di abduzione così come descritto in Castor non considera la possibilità che, dato comunque un insieme di vincoli Θ, siano comunque presenti delle tracce che descrivono modelli inconsistenti, in quanto Castor non tratta la generazione di regole contemplanti la negazione. Questa richiesta è tuttavia presente in Declare, dove esistono dei template di negazione, o comunque predicati sui dati che sono contradditori. È quindi necessario generare generalizzazioni tra i dati (*mgg*) allo scopo di ottenere qual è il minimal consistent subset che caratterizzi entrambe le tracce.

References

[1] André Petermann, Martin Junghanns, Robert Müller, and Erhard Rahm. Foodbroker - generating synthetic datasets for graph-based business analytics. In *Big Data Benchmarking - 5th International Workshop, WBDB 2014,*

Potsdam, Germany, August 5-6, 2014, Revised Selected Papers, volume 8991 of *Lecture Notes in Computer Science*, pages 145–155. Springer, 2014.

[2] Williams Rizzi, Chiara Di Francescomarino, and Fabrizio Maria Maggi. Explainability in predictive process monitoring: When understanding helps improving. In Dirk Fahland, Chiara Ghidini, Jörg Becker, and Marlon Dumas, editors, *Business Process Management Forum*, pages 141–158, Cham, 2020. Springer International Publishing.