Capitolo 1

**Stato dell’arte**

* 1. **Introduzione**

Il software sviluppato per questa tesi di laurea riguarda il campo del data mining, nello specifico quello testuale; cioè dato un testo, questo viene analizzato una frase per volta, ed in ogni frase viene analizzato ogni termine con l’obiettivo di trovare quelli che possono rappresentare gli eventi di una relazione causale. A tal scopo, la complessità del lavoro affrontato, ha portato a cercare soluzioni eterogenee ed applicarle sequenzialmente in tale programma. Tra le soluzioni si è partiti dal ricorrere alle varie analisi di un testo (lessicale, sintattica, semantica), e si è finiti a richiamare tecniche di machine learning, per decidere se, dati gli eventi trovati, una relazione è causale o no.

* 1. **Livello lessicale**

Per sviluppare questo parte (come per le due successive), si è ricorso all’NLP (*Natural Language Processing*), un processo di trattamento automatico delle informazioni scritte o parlate in una lingua naturale. Questo processo è reso particolarmente difficile e complesso a causa delle caratteristiche intrinseche di ambiguità del linguaggio umano. Quindi nell’analisi del testo è stata presa in considerazione una libreria (*spacy*) associata ad un noto linguaggio di programmazione (*Python*), considerata una lingua naturale specificata e, per ogni parola, individuate automaticamente alcune sue caratteristiche, o *feature*. Siccome la comprensione di un linguaggio naturale è considerata un problema IA-completo, e che a tal punto, per esempio, il linguaggio considerato legge separatamente termini formanti un’unica parola nella lingua utilizzata, si è dovuta anche creare una funzione che risolvesse quanto appena detto.

* 1. **Livello sintattico**

La parte sintattica viene sviluppata prendendo come riferimento la rappresentazione di un grafo. La frase analizzata viene rappresentata come un grafo, dove i termini che la compongono (*token*) rappresentano i nodi del grafo considerato. Ad ognuno di questi *token* vengono associate alcune delle loro *features* (in questo caso le parti del discorso (POS, *part-of-speech*)), attraverso la libreria enunciata precedentemente. In più, sono stati considerati dei collegamenti (i cosiddetti *archi* di un grafo) tra i POS di ogni token e quelli del suo successivo; questo per abilitare i *pattern* che ci servono per coinvolgere le dipendenze lessicali. I pattern utili per estrarre le relazioni causali riguardano uno specifico POS (per pattern utili sono intesi gli eventi citati nell’introduzione).

* 1. **Livello semantico**

Per i token più rappresentativi di una frase, cioè quelli tendenti a farci scoprire i pattern a noi utili, viene approfondito il loro significato. Per fare questo, si è fatto riferimento alla WSD (word sense disambiguation), un problema di computazione linguistica che tende ad esplorare e verificare il significato di una parola. Per fare questo, è stata presa in considerazione l’iperonimia dei token più rappresentativi, cioè per ognuno di essi si è risaliti all’insieme dei termini predecessori da cui deriva; stessa cosa viene fatta per i termini di quest’ultimo insieme considerato, e così via. Per capire il significato a noi utile, ogni elemento di questo insieme viene confrontato con il lemma rispettivo. Nello sviluppo del software, per l’analisi simbolica, è stata utilizzata la libreria NLTK.

* 1. **Scoperta del pattern**

Dopo aver costruito un grafo, per ogni frase, da essi vengono estratti i sotto grafi rappresentanti i pattern da utilizzare per il nostro scopo, cioè gli eventi che ci servono per l’individuazione di relazioni causali. I sotto grafi in questione comprendono, oltre agli eventi, anche il POS dell’evento abilitante il pattern e quello del token successivo ad esso (cioè, nell’ordine sequenziale della frase). La conoscenza del fatto che questo pattern matcha o no potrebbe essere un indicatore utile per verificare se l’evento è causale o no. Per fare questo, si è deciso ad applicare l’algoritmo gSpan, che estrae i sotto grafi che cerchiamo, e per farlo usa una tecnica di potatura. L’algoritmo prende le informazioni che gli servono da un file di input, e restituisce l’immagine dei grafi.

**1.6 Pattern matching**

Dopo aver ottenuto i pattern rappresentati nei sotto grafi considerati precedentemente, viene applicato il problema del *constraint satisfation*, cioè viene lanciato questo problema come problema di monomorfismo di un grafo ottenuto precedentemente sugli *enablement* riguardanti gli eventi estratti ed eseguiamo un risolutore su esso per determinare la soddisfacibilità dei pattern. Nello specifico, vengono utilizzati dei vincoli per ottenere i pattern in formato *“chiave: valore”*.

* 1. **Classificazione del grafo**

In questa sezione, sono stati utilizzati i pattern ottenuti per verificare se una relazione è causale o no. Per farlo, viene considerato il pattern *chiave* di ogni risultato ottenuto, e applicata la tecnica di *word embedding*, che accorpa le parole e ne calcola la similarità, costruendo uno spazio vettoriale contenente i vettori riguardanti le parole prese. Questa tecnica viene applicata attraverso il metodo *word2vec*, nello specifico viene utilizzata l’architettura *Skip-Gram*, che ha lo scopo di predire le parole di contesto partendo dal token corrente. Detto questo, per ogni token della frase considerata, viene creato un vettore binario con lunghezza uguale a quella della frase, che ha valore 1 nella posizione corrispondente al token considerato, 0 altrimenti. Dopodiché, vengono presi i vettori binari corrispondenti ai pattern *chiave* ed eseguito su essi il *word2vec*. Al vettore di output, che avrà la stessa lunghezza della frase considerata, verrà usato per calcolare la distribuzione di probabilità riguardante la similarità semantica del pattern considerato con gli altri token della frase; questo viene fatto con la funzione *softmax*. In seguito, viene fatta la media tra gli elementi del vettore di output, che viene utilizzata nel classificatore SVM per calcolare la misura F1 dell’accuratezza di un test; questo per individuare la relazioni causali; per maggiore sicurezza, per ogni pattern viene utilizzato anche il test di Fisher. Infine, nel problema viene applicato anche l’algoritmo di copertura sequenziale positivo/negativo, per evitare i duplicati.

**Riferimenti**

1. Luca Bechini, Nicole Koenderink, Hein F.M. ten Berge, Wim Corre, Frits K. van Evert, Arianna Facchi, Olfa Gharsallah, Elena Gorriz-Mifsud, Carlo Grignani, Michael den Herder, Yolaine Hily, Eric Justes, Aurélien Lepennetier, Barbara Moretti, Paul Newell-Price, Luca Nonini, Roberto Oberti, Sonia Ramonteu, Mercedes Rois, Frank de Ruijter, Dario Sacco, Peter M. Schuler, Don Willems, Anneke Zandstra, Jan Top, *Improving access to research outcomes for innovation in agriculture and forestry: the VALERIE project, Italian Journal of Agronomy, Department of Agricultural and Environmental Sciences, University of Milano, Milano, Italy; Wageningen UR Food & Biobased Research, Wageningen, The Netherlands; Wageningen UR Plant Sciences Group, Wageningen, The Netherlands; European Forest Institute, Barcelona, Spain; Department of Agricultural Forest and Food Sciences, University of Torino, Grugliasco (TO), Italy; European Forest Institute, Joensuu, Finland; French National Institute for Agricultural Research, Paris, France; Association for the Coordination of Agricultural Technique, Paris, France; ADAS UK Ltd., Wolverhampton, United Kingdom; IT-Objects GmbH, Hagen, Germany.*
2. Bryan Rink, Cosmin Adrian Bejan, e Sanda Harabagiu, *Learning Textual Graph Patterns to Detect Causal Event Relations, Human Language Technology Research Institute, The University of Texas at Dallas Richardson, Texas 75080, USA,* 2010.
3. Yuanbin Wu, Qi Zhang, Xuanjing Huang, Lide Wu, *Phrase Dependency Parsing for Opinion Mining, Fudan University School of Computer Science,* 2009.
4. Marina Sedinkina, Folien von Desislava Zhekova, *WordNet, Language Processing and Python, CIS, LMU,* 18Dicembre, 2018.
5. Xifeng Yan, Jiawei Han, *gSpan: Graph-Based Substructure Pattern Mining, Department of Computer Science University of Illinois at Urbana-Champaign,* 2002.
6. Steven Bethard e James H. Martin, *Learning Semantic Links from a Corpus of Parallel Temporal and Causal Relations, Institute for Cognitive Science, Department of Computer Science, University of Colorado, Boulder, CO 80309, USA,* 2008.
7. Thorsten Joachims, *Text Categorization with Support Vector Machines: Learning with Many Relevant Features, Universitat Dortmund Informatik LS8, Baroper Str. 301, 44221 Dortmund, Germany,* 1998.
8. Srinivas Sankara Narayanan, *Knowledge-based Action Representations for Metaphor and Aspect (KARMA), A dissertation submitted in partial satisfaction of the requirements for the degree of Doctor of Philosophy in Engineering: Computer Science, GRADUATE DIVISION of the UNIVERSITY of CALIFORNIA at BERKELEY,* 1997.
9. Roxana Girju, *Automatic Detection of Causal Relations for Question Answering, Computer Science Department Baylor University Waco, Texas,* 2003.
10. Christopher S.G. Khoo, Syin Chan e Yun Niu, *Extracting Causal Knowledge from a Medical Database Using Graphical Patterns, Centre for Advanced Information Systems, School of Computer Engineering Blk N4, Rm2A-32, Nanyang Avenue, Nanyang Technological, University Singapore 639798,* 2000.
11. William W. Cohen, *Learning Trees and Rules with Set-valued Features, AT&T Laboratories 600 Mountain Avenue Murray Hill, NJ 07974,* 1996.
12. Edward Loper e Steven Bird, *NLTK: The Natural Language Toolkit, Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA 19104-6389, USA,* 2002.
13. Vuong M. Ngo, Tru H. Cao e Tuan M. V. Le, *WordNet-Based Information Retrieval Using Common Hypernyms and Combined Features, HCMC University of Technology and John von Neumann Institute VNU-HCM, Viet Nam,* 2018.
14. Deepak Ravichandran e Eduard Hovy, *Learning Surface Text Patterns for a Question Answering System, Information Sciences Institute, University of Southern California 4676 Admiralty Way, Marina del Rey, CA 90292-6695, USA, Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL), Philadelphia,* Luglio 2002, pp. 41-47*.*
15. Makoto Yokoo, Toru Ishid, Edmund H. Durfee e Kazuhiro Kuwabarat, *Distributed Constraint Satisfaction for Formalizing Distributed Problem Solving, NTT Communication Science Laboratories Sanpeidani Inuidani, Seika-cho Soraku-gun, Kyoto 619-02 Japan, Dept. of Electrical Engineering and Computer Science, University of Michigan Ann Arbor, MI 48109 U.S.A.,* 1992.