**Exposé zur Diplomarbeit**

# Relationsextraktion durch Frequent Patterns in Dependency Graphen

Stefan Pietschmann

17. Februar 2009

Betreuer: Prof. Dr. Ulf Leser

## 1 Hintergrund

Die Anzahl der Veröffentlichungen im biomedizinischen Sektor wächst exponentiell. Dies macht Verfahren notwendig, in der großen Menge von Publikationen (man spricht dabei von einem Korpus) relevante Informationen automatisch zu finden. Eine wichtige Aufgabe ist die Extraktion von Relationen zwischen biomedizinischen Entitäten. Typische Entitäten sind Proteine, Gene und Krankheiten und wichtige Relationen sind Protein-Protein-Interaktionen, Protein-Funktion-Zusammenhänge und Gen-Krankheit-Beziehungen.

Der einfachste Ansatz zur Relationsextraktion ist die Sätze einer Publikation auf das gemeinsame Vorkommen von Entitäten zu überprüfen [1]. Tritt dieser Fall ein, wird dies als Relation zwischen den Entitäten gewertet. Dieses Vorgehen findet trivialerweise alle vorhandenen Relationen in einem Satz, was einen Recall von 100% bedeutet. Allerdings ist die Precision sehr niedrig, da viele der gefundenen Relationen gar keine sind.

Ein verfeinerter Ansatz ist die Einbeziehung der Umgebung der Entitäten im Satz, d. h., dass ein gemeinsames Vorkommen von Entitäten nur dann als Relation gewertet wird, wenn diese und ihre umliegenden Wörter einem bestimmten Muster (engl.: Pattern) entsprechen [2]. Dadurch wird eine hohe Precision erreicht, da idealerweise alle Sätze ohne Relationen nicht den Patterns entsprechen und herausgefiltert werden. Allerdings sind Patterns schnell zu fein, so dass viele korrekte Relationen ebenfalls nicht

gefunden werden, was einen niedrigen Recall zur Folge hat. Den Abgleich des Satzes mit Patterns bezeichnet man als Pattern Matching.

Da die Publikations-Datenbanken in natürlich-sprachlichem Text vorliegen, ist es zudem möglich, Verfahren aus dem Natural Language Processing (NLP) in den Extraktionsprozess einzubinden, man spricht in dem Zusammenhang von biomedizinischem NLP (BioNLP). So werden bei vielen Pattern Matching-Verfahren nicht nur die Wörter an sich untersucht, sondern auch ihre linguistischen Eigenschaften, insbesondere ihre Wortarten (engl.: Part Of Speech (POS)) [3] und/oder ihre syntaktischen Beziehungen zueinander [4]. Durch diese Zusatzinformationen kann die Qualität der Patterns, und damit auch Precision und Recall erhöht werden. Die benötigten Patterns werden dabei entweder per Hand erstellt [5] oder automatisch gelernt [6].

Die dritte große Gruppe der Relationsextraktoren sind statistische Klassifizierer, wie z. B. Support Vector Machines [7], welche anhand von Trainingsdaten gelernt werden und auf Grundlage des gelernten Modells bei Eingabe eines Satzes entscheiden, ob dieser eine Relation enthält oder nicht. Auch diese Verfahren benutzen oft syntaktische Zusatzinformationen.

# 2 Ziel

Ziel der Arbeit ist die Untersuchung eines neuen Ansatzes zur Relationsextraktion auf Basis von Pattern Matching mit Syntaxinformationen. Dabei sollen die Patterns durch Frequent Subgraph Mining (FSM) und geeignete Vor- und/oder Nachverarbeitungsschritte in einem bereits mit Relationen annotierten Korpus gefunden werden. Eine der im BioNLP am häufigsten diskutierten Relationsextraktionsarten ist die von Protein-Protein-Interaktionen (PPI). Um das zu entwickelnde System gut mit anderen vergleichbar zu machen, soll deshalb auch hier ein mit PPIs annotiertes Korpus benutzt werden. Prinzipiell soll das Vorgehen aber auch auf andere Relationen anwendbar sein.

FSM-Verfahren geben bei Eingabe einer Menge von Graphen alle Subgraphen aus, die häufig vorkommen. Im BioNLP werden hauptsächlich zwei große Syntaxgrapharten unterschieden: Constituent Trees und Dependency Graphen, welche jeweils eine linguistische Repräsentation eines Satzes oder einer Phrase darstellen. Constituent Trees sind geordnete Bäume, deren Blätter den Wörtern im Satz entsprechen, deren restliche Knoten linguistischen Constituents entsprechen und deren Kanten die rekursive Struktur dieser Constituents wiederspiegeln. Dependency Graphen sind Graphen, deren Knoten den Wörtern im Satz und deren Kanten linguistischen Beziehungen zwischen den Wörtern

entsprechen. In dieser Arbeit soll FSM auf Dependency Graphen angewandt werden, da deren Struktur potentiell mehr semantische Informationen liefert [8, S. 430].

Die Idee hinter dem Anwenden von FSM für das Finden von PPI-Sätzen ist demnach, dass diese Sätze auf Dependency-Graph-Ebene Gemeinsamkeiten haben, anhand derer man sie unterscheiden kann von Sätzen, die keine PPIs enthalten. Ob diese These gehalten werden kann, gilt es herauszufinden. Kern der Arbeit soll dabei sein, zu ergründen, welche weiteren Verarbeitungsschritte der Frequent Dependency Subgraphen (FDS) zur Qualität der Patterns beitragen können. Dabei sollen Performanzfragen bezüglich Geschwindigkeit zunächst in den Hintergrund treten.

Um die in den verschiedenen möglichen Schritten entstandenen Patterns zu evaluieren, ist ein Abgleich mit einem Gold-Standard-Korpus (siehe Abschnitt 3.1) nötig. Die Ergebnisse sollen dann schließlich mit anderen PPI-Extraktoren verglichen werden.

# 3 Vorgehensweise

Es wird voraussichtlich vier große Hauptschritte in dieser Arbeit geben, welche sich jeweils in mehrere Teilschritte gliedern. Sie können sich durchaus überschneiden und sind deshalb nicht notwendigerweise sequentiell. Die Schritte lauten:

- Dependency Graphen erstellen
- Patterns lernen
- Patterns anwenden
- Evaluation

## 3.1 Dependency Graphen erstellen

Die zum Patterns lernen benötigten Dependency Graphen müssen aus einem Korpus erstellt werden, welches als Gold-Standard fungiert. D.h., das Korpus besteht nur aus Sätzen, in denen bereits alle Proteine und ihre Interaktionen annotiert sind und es wird davon ausgegangen, dass diese Annotationen korrekt sind. Hier bieten sich handannotierte Korpora [9] oder ein automatisch generiertes Korpus [10] an.

Gute Ergebnisse im Erstellen von Syntaxbäumen (sogenanntes Parsen) von biomedizinischen Texten [11] lieferten z. B. der Charniak-Johnson Parser [12], der Charniak-Lease Parser [13], der Bikel Parser [14] oder der Stanford Parser [15]. Bei einigen Parsern

muss zuvor ein POS-Tagging-Schritt erfolgen, wofür sich der MedPost Tagger [16] eignet. Für das jeweilige Eingabeformat der Parser/Tagger muss das Korpus entsprechend angepasst werden.

Da die generierten Syntaxgraphen Constituent Trees sind, in der Arbeit aber Dependency Graphen benutzt werden sollen, müssen Erstere in Letztere umgewandelt werden. Hierfür eignet sich ein Konverter, welcher im Stanford Parser enthalten und explizit aufrufbar ist [17].

## 3.2 Patterns lernen

Das FSM der Dependency Graphen ist der erste Schritt beim Patterns lernen. State-of-the-art Miner sind beispielsweise gSpan [18] oder Gaston [19]. Hier wird das Hauptproblem sein, einen geeigneten Miner zu finden, welcher anwendbar ist auf die Struktur der Dependency Graphen (d. h. Kanten- und Knoten-gelabelte, möglicherweise zyklische Graphen). Der Parameter, wie „frequent“ Subgraphen sein sollen, muss vor dem FSM festgelegt werden.

Eine Auswahl von möglichen testbaren Weiterverarbeitungen der FDSs zur Qualitätserhöhung sind:

**Stemming** Die Reduzierung von Wörtern auf ihren Wortstamm könnte man auf alle oder ausgewählte Wörter anwenden. Ein möglicher Stemmer ist der Porter Stemmer [20].

**Groß-/Kleinschreibung ignorieren** Alle oder ausgewählte Buchstaben der Wörter könnten in Klein- oder Großbuchstaben umgewandelt werden.

**Wörter/Dependencies zusammenfassen** Mehrere Wörter bzw. Dependencies könnten durch ein einzelnes Wort bzw. eine einzelne Dependency ersetzt werden.

**Beschränkung auf bestimmte Graphen** FDSs, die bestimmten Bedingungen entsprechen/nicht entsprechen, könnten gelöscht werden (z. B. Graphen, die keine Proteine enthalten; Graphen, die Negationen enthalten).

Einige der genannten Verfahren, wie z. B. Stemming, könnten auch durchaus schon beim Dependency Graphen erstellen (siehe Abschnitt 3.1) und somit vor dem eigentlichen Finden der FDSs angewandt werden.

Weitere Möglichkeiten, die Patternqualität bereits beim Erstellen der Dependency Graphen zu beeinflussen, sind:

**Zusammenfassen von Wörtern zu Proteinen** Proteinbezeichnungen bestehen oft aus mehreren Wörtern. Da die Patterns Auskunft über die Proteine und nicht über ihre Einzelwörter geben sollen, müssen mehrwortige Proteinbezeichnungen zu einem Wort zusammengefasst werden. Wann dies geschieht, könnte die Qualität beeinflussen. Möglich wäre es beim Dependency Graphen erstellen oder nach dem FSM.

**Wahl der Dependency Graph-Art** Der Stanford-Konverter bietet die Möglichkeit vier verschiedene Varianten von Dependency Graphen zu generieren. Die Wahl könnte sowohl die Qualität der Patterns beeinflussen, als auch die Wahl des Frequent Subgraph Miners, da die mögliche Ausgabe des Konverter von Baum bis zyklischer Graph reicht.

## 3.3 Patterns anwenden

Um aus einem Korpus alle Sätze zu extrahieren, die PPIs enthalten, muss ein Abgleich des Korpus mit den gelernten Patterns erfolgen. Dazu wird der Text zunächst satzweise in die Struktur der Patterns, d. h. in Dependency Graphen, umgewandelt (siehe Abschnitt 3.1). Anschließend gleicht ein Graph-Matching-Algorithmus jeden Korpus-Dependency-Graph solange mit jedem Pattern ab, bis entweder ein Pattern matcht oder alle Patterns durchlaufen wurden. Tritt der erste Fall ein, dann wird der Satz als PPI-enthaltend klassifiziert, tritt der zweite Fall ein, als nicht-PPI-enthaltend. State-of-the-art Graph Matcher sind beispielsweise GraphFind [21] oder SAGA [22]. Hier wird das Hauptproblem sein, einen geeigneten Graph-Matching-Algorithmus zu finden, welcher anwendbar ist auf die Struktur der Dependency Graphen (d. h. Kanten- und Knotengelabelte, möglicherweise zyklische Graphen).

Potentielle Fähigkeiten des Matchers könnten eventuell die Qualität der Patterns steigern. Eine Auswahl von testbaren Möglichkeiten:

**Wildcards** Es werden Platzhalter für beliebige Knoten und/oder Kanten in den Patterns benutzt, so dass inexaktes Matching möglich ist.

**Abstandsmaße** Es wird der Abstand der Graphen zueinander berechnet und je nach Schwellwert als Matching gewertet.

## 3.4 Evaluation

Die Evaluation der gefundenen Patterns geschieht mit Hilfe eines Gold-Standard-Korpus (siehe Abschnitt 3.1), d. h. man weiß, welche Sätze in diesem Korpus PPIs enthalten

und welche nicht. Wendet man also die Patterns an, wie im vorherigen Abschnitt 3.3 beschrieben, dann braucht man nur noch die Ergebnisse mit den tatsächlichen Daten des Gold-Standards vergleichen. Zur statistischen Auswertung eignen sich die üblichen PARSEVAL-Meßgrößen Precision, Recall bzw. ihr harmonisches Mittel F-Measure. Dabei zählt als True Positive, wenn eine korrekte Relation extrahiert wird, als False Positive, wenn eine falsche Relation extrahiert wird, als False Negative, wenn eine korrekte Relation nicht extrahiert wird, und als True Negative, wenn eine falsche Relation nicht extrahiert wird.

Die berechneten Werte sollen dann mit denen anderer PPI-Extraktoren verglichen werden.

# Literatur


[1] Jenssen, T K. ; Laegreid, A ; Komorowski, J ; Hovig, E: A literature network of human genes for high-throughput analysis of gene expression. In: *Nat Genet* 28 (2001), S. 21–28

[2] Blaschke, Christian ; Andrade, Miguel A. ; Ouzounis, Christos ; Valencia, Alfonso: Automatic Extraction of Biological Information from Scientific Text: Protein-Protein Interactions. In: *Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology* Bd. 17, 1999. – ISBN 1–57735–083–9, S. 60–67

[3] Ono, Toshihide ; Hishigaki, Haretsugu ; Tanigami, Akira ; Takagi, Toshihisa: Automated extraction of information on protein-protein interactions from the biological literature. In: *Bioinformatics* 17 (2001), S. 155–161

[4] Leroy, G ; Chen, H: Filling preposition-based templates to capture information from medical abstracts. In: *In Proceedings of Pacific Symposium on Biocomputing 7* Bd. 17, 2002, S. 350–361

[5] Thomas, James ; Milward, David ; Ouzounis, Christos ; Pulman, Stephen ; Carroll, Mark: Automatic extraction of protein interactions from scientific abstracts. In: *Proc. of the Pacific Symp. on Biocomputing* Bd. 23, 2002, S. 538–549

[6] Hakenberg, Jörg ; Plake, Conrad ; Leser, Ulf ; Kirsch, Harald ; Rebholz-Schuhmann, Dietrich: Genic Interaction Extraction with Alignments and Finite

State Automata. In: *Proc Learning Language in Logic Workshop (LLL05) at the 22nd Int Conf on Machine Learning*. Bonn, Germany, August 2005, S. 38–45

[7] Culotta, Aron ; Sorensen, Jeffrey: Dependency tree kernels for relation extraction. In: *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics - ACL '04* Bd. 23. Morristown, 2004, S. 423

[8] Manning, Christopher D. ; Schütze, Hinrich: *Foundations of Statistical Natural Language Processing*. Second Printing. Cambridge/London : The MIT Press, 2000

[9] Pyysalo, Sampo ; Sætre, Rune ; Tsujii, Jun'ichi ; Salakoski, Tapio: Why Biomedical Relation Extraction Results are Incomparable and What to do about it. In: *Proceedings of the Third International Symposium on Semantic Mining in Biomedicine (SMBM 2008)*. Turku, 2008, S. 149–152

[10] Hakenberg, Jörg ; Leser, Ulf ; Kirsch, Harald ; Rebholz-Schuhmann, Dietrich: Collecting a large corpus from all of Medline. In: *SMBM 2006*. Jena, Germany, April 2006, S. 89–92

[11] Clegg, Andrew ; Shepherd, Adrian: Benchmarking natural-language parsers for biological applications using dependency graphs. In: *BMC Bioinformatics* 8 (2007), S. 24

[12] McClosky, David ; Charniak, Eugene: Self-Training for Biomedical Parsing. In: *Proceedings of ACL-08: HLT, Short Papers*. Columbus, Ohio, 2008, S. 101–104

[13] Lease, Matthew ; Charniak, Eugene: Parsing biomedical literature. In: *In Proceedings of the Second International Joint Conference on Natural Language Processing (IJCNLP-05), Jeju Island, Korea*, 2005, S. 58–69

[14] Bikel, Daniel M.: Design of a multi-lingual, parallel-processing statistical parsing engine. In: *Proceedings of the second international conference on Human Language Technology Research*. San Francisco, CA, USA, 2002, S. 178–182

[15] Klein, Dan ; Manning, Christopher D.: Accurate unlexicalized parsing. In: *ACL '03: Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*. Morristown, NJ, USA, 2003, S. 423–430

[16] Smith, L ; Rindflesch, T ; Wilbur, W J.: MedPost: a part-of-speech tagger for bioMedical text. In: *Bioinformatics* 20 (2004), April, S. 2320–2321

[17] De Marneffe, Marie-Catherine ; Maccartney, Bill ; Manning, Christopher D.: Generating Typed Dependency Parses from Phrase Structure Parses. In: *LREC* Bd. 20, 2006

[18] Yan, Xifeng ; Han, Jiawei: gSpan: Graph-Based Substructure Pattern Mining. In: *ICDM '02: Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM'02)*. Washington, DC, USA, 2002, S. 721

[19] Nijssen, Siegfried ; Kok, Joost N.: A quickstart in frequent structure mining can make a difference. In: *Proceedings of the 2004 ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '04*. New York, 2004. – ISBN 1581138889, S. 647

[20] Porter, M.F. ; Rijsbergen, C.J. V. ; Robertson, S.E.: New models inprobabilistic information retrieval. In: *BritishLibrary Research and Development Report, London* 5587 (1980)

[21] Ferro, Alfredo ; Giugno, Rosalba ; Mongiovì, Misael ; Pulvirenti, Alfredo ; Skripin, Dmitry ; Shasha, Dennis: GraphFind: enhancing graph searching by low support data mining techniques. In: *BMC Bioinformatics* 9 (2008), S. S10

[22] Tian, Y. ; Mceachin, R. C. ; Santos, C. ; States, D. J. ; Patel, J. M.: Saga: a subgraph matching tool for biological graphs. In: *Bioinformatics* 23 (2007), Januar, S. 232–239. ISBN 1581135076