

Where Computer Science, Linguistics, and Biology Meet:





Using lexical chaining to analyze biomedical text

Chrysanne DiMarco




**David R. Cheriton School of Computer Science
University of Waterloo**


NATURAL LANGUAGE RESEARCH GROUP

The problem

-  Scientific articles contain much implicit knowledge about biomolecular interactions, e.g., protein-protein interactions.
-  These interactions can help researchers to determine the biological functions of the proteins involved.
-  Currently, protein-interaction databases exist but not all data is biologically valid, i.e., interactions occur under lab conditions, but not in the living cell..
-  Too many papers on protein interactions exist even for a single protein for biologists to read and validate manually.

Our approach

-  We know that proteins involved in interactions tend to have similar biological functions.
-  Biologically similar terms in protein-interaction contexts in scientific articles may therefore provide evidence that the interaction occurs in the living cell.
-  We are using a discourse-analysis method—*lexical chaining*—to find such strings of semantically related terms.

Current research in biomedical information extraction

- 📄 Natural language methods in biomedical information extraction rely on partial syntactic analysis, limited semantic understanding.
- 📄 The basic approach:
 - Tokenization.
 - Part-of-speech tagging.
 - Shallow parsing.
- 📄 Simple grammatical templates to match protein interactions:
ProteinA interacts with/ binds to/ associates with proteinB.
- 📄 But template-only analysis misses a great deal of semantic info!

What is 'lexical chaining'?

- 📄 First introduced by Morris and Hirst (1991).
- 📄 Derives from the concept of *textual cohesion*:
A text is not just a random set of sentences, but 'sticks together' by various means to form a unified whole.
- 📄 Forms of cohesion include:
 - Grammatical cohesion (reference, substitution, ellipsis, conjunction)
 - Lexical cohesion (semantic relationships between words)
- 📄 Formally, a *lexical chain* is a sequence of semantically related words within a topical unit of a text.
- 📄 A document may contain many lexical chains.

An example of a lexical chain

- 1) John has a *Jaguar*.
- 2) He loves the *car*.
- 3) John works in the *garage* taking care of his *Jaguar*.

Lexical chain = {*Jaguar*, *car*, *garage*, *Jaguar*}



Lexical semantic relationships:

{*Jaguar*, *car*}: hyponymy

{*car*, *garage*}: collocation

{*garage*, *Jaguar*}: collocation

A simple lexical-chaining algorithm

-  Lexical chains can be computed by grouping sets of words (usually noun instances) that are semantically related by repetition, synonymy, hypernymy/hyponymy, etc:
1. Select a set of candidate words (all noun instances).
 2. For each candidate word, find an appropriate chain using some 'relatedness criterion' among members of the chain.
 3. If such a chain can be found for current word, insert in chain. Otherwise, start a new chain.
-  But this process is difficult and computationally costly:
- Each word must be assigned to only one chain.
 - Chains must be optimized to be longest/strongest possible.

A linear-time algorithm (Silber and McCoy 2002)

Step 1: For each noun instance

For each sense of the noun instance

Compute all scored ‘metachains’

- All possible chains that can be made up of word senses in the text having identity, synonymy, or hypernymy relationship.

e.g., for noun sense *person*, {*John*, *machine*} is a metachain because both *John* and *machine* have the sense *person*.

Step 2: For each noun instance

For each metachain to which the noun belongs

Keep noun instance in metachain to which it ‘contributes most’

Delete noun instance from all of its other metachains

Update scores of other metachains

At conclusion: Highest-scoring chains will be left.

An improved algorithm

(Enss 2006)




- 📄 Enss showed that ‘perfect’ word sense disambiguation before lexical chaining significantly reduced number of incorrect chains.
- 📄 However, his results also showed that in some cases errors in word sense disambiguation allowed correct lexical chains that were not possible with perfect disambiguation.

General information-extraction algorithm

1. Tokenize input text.
2. Tag each word with its part-of-speech.
3. Group sequences of words into phrases using cascaded finite state machines.
4. Identify noun entities and actions of interest.
5. Identify and link coreferring noun entities.
6. Instantiate grammatical ‘templates’ with entities and activities.
This is a form of ‘shallow parsing’.

Sample template: *NounPhrase1 Verb Preposition NounPhrase2*

Combining the algorithms...

-  Need to determine each 'protein interaction context':
 - A single sentence.
 - A whole paragraph.
 - Part of a paragraph. Difficult!!
-  Need to recognize protein entities, other biological terms:
 - Acronyms, abbreviations.
 - New protein names.
 - Developing biological ontologies.
-  Need to define set of protein interaction templates:
 - Defining a '*stylistic grammar*' for biomedical writing?

A sample PPI lexical chain

...Cdc13p interacts with Pol1p, Sir4p, Zds2p and Imp4p. Moreover, CDC13-deleted yeast cells... with point **mutations** in this region caused **defects** in progressive *cell growth* and in *cell cycle control*. These cells also have **defects** in *telomere length regulation*. Thus, we conclude that the N-terminal region of Cdc13p is involved in *telomere maintenance*, *teleomere length regulation* and *cell growth control* through its interaction with its binding proteins.

Scoring lexical chains

- 📄 Hirst and St.-Onge (1997) adapted Morris and Hirst's semantic distance algorithm to propose formula for scoring strength of a lexical chain.
- 📄 They view semantic relationships between words as a *graph*.
- 📄 Relationships between words in the WordNet 'graph' have:
 - Direction (\uparrow , \downarrow , \rightarrow , \leftarrow).
 - Varying weights:
 - Extra-strong: repetition
 - Strong: standard WordNet relation between adjacent concepts
 - Medium-strong: allowable 'path' of relations between concepts
- 📄 Score of lexical chain = sum of weights of word pairs in chain.

What a lexical chainer needs: A linguistic ontology

- 📄 Silber & McCoy and Hirst & St.-Onge use WordNet, an online lexical database as knowledge source for the lexical semantic relations used in constructing lexical chains.
- 📄 Words (nouns, verbs, adjectives, adverbs) are organized into ‘synonym sets’, known as *synsets*.
- 📄 A synset represents the concept underlying a group of words that are (near-) synonymic.
- 📄 Synsets can be related by various lexical semantic relations:
 - Synonymy, antonymy, hyponymy (*IsA*), meronymy (*part-whole*).
- 📄 Sample synset: {*car, auto, automobile, machine, motorcar*}

What we needed: A domain-specific ontology

The Gene Ontology:

- Commonly used biological ontology.
- Hierarchical vocabulary to describe gene and gene products in any organism.

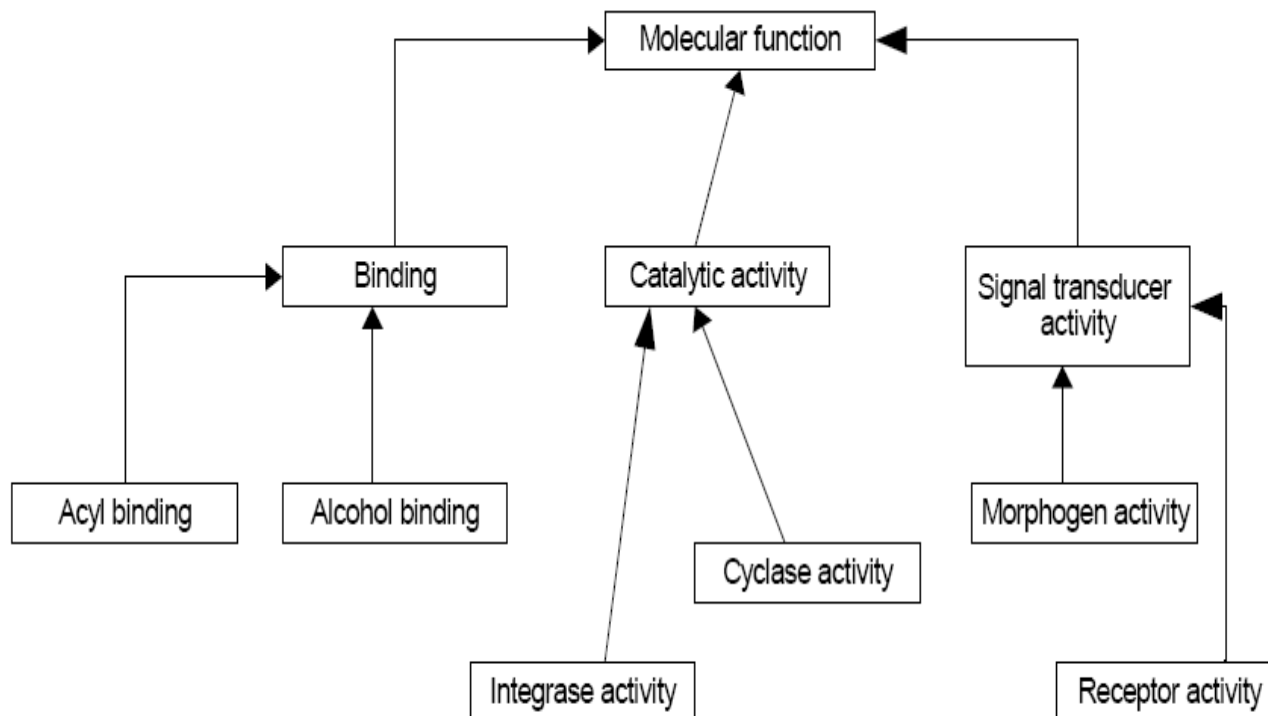
Structure of the ontology:

- Biological processes.
- Molecular functions.
- Cellular components.





Limitation:

- Protein-protein interaction domain is not covered!

From Gene Ontology to “PPIWordNet”



A protocol for constructing “PPIWordNet” (He 2007)

-  Semi-automatic domain concept determination (Hassan).
-  Manual concept relationship determination (Musso, Yu).
-  Integration of protein-specific concepts with the Gene Ontology.
-  Evaluation of “PPIWordNet” in lexical-chaining application.

Process 1:

PPI ontology determination

Goal:

- To determine key concepts and relationships in the PPI domain.

'Middle-out' strategy:

- Identify a core of basic domain concepts.
- Specify and generalize as necessary.

Corpus-based methodology:

- Deciding how discriminating a term is for PPI domain compared to other subject domains.
- Representing how important/relevant a term is to application goals.

Process 1:

PPI ontology determination (cont'd)

Select corpora:

- BioCreAtIvE PPI task corpus and Wikipedia.

Extract discriminating terms:

- Automatically extract discriminating terms using *tf-idf*.
- Manually select terms relevant to PPI research.
- Output < 200 terms.

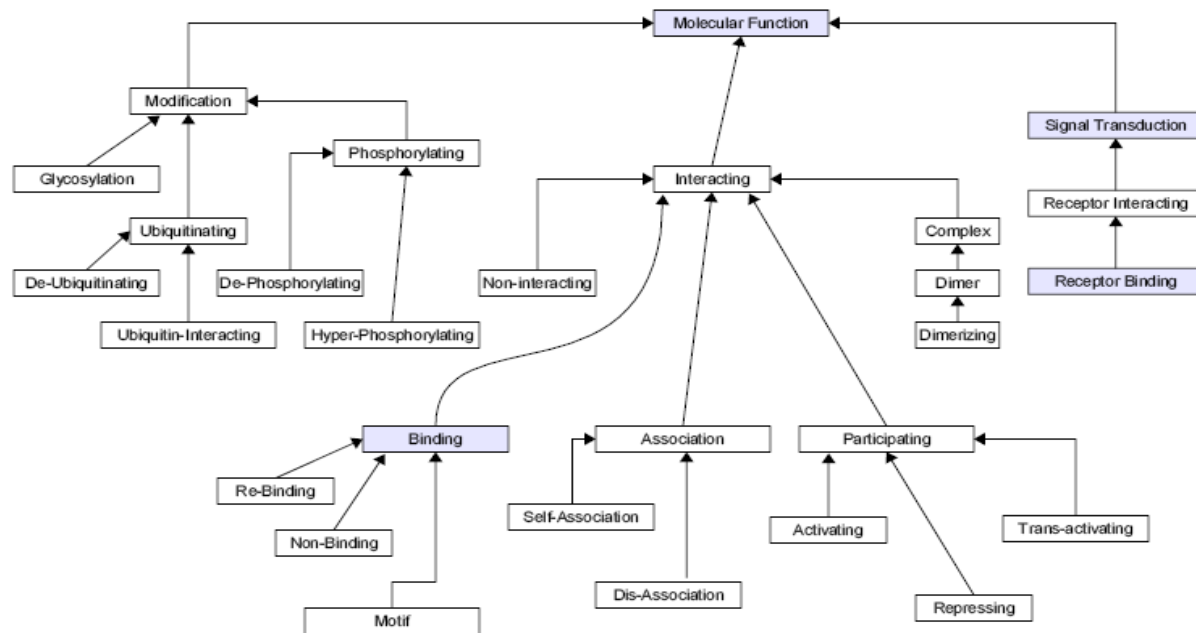
Construct glossary:

- Organize terms into synsets.
- Categorize according to Gene Ontology semantic types:
 - Biological process, molecular function, interaction property, also methods.
- Identified 54 'seed' (starting-point) terms.

Process 2: PPI ontology construction

- 📄 A process of linking/merging subparts of the ontology.
- 📄 For each category:
 - For each seed term:
 - Construct the local hierarchy of directly related terms (using *IsA*, *part-of* relations).
 - Domain experts added parents, children, siblings for each seed term.
 - Expand the local hierarchy recursively.
 - Link local hierarchies together using common terms.

Sample of initial ontology

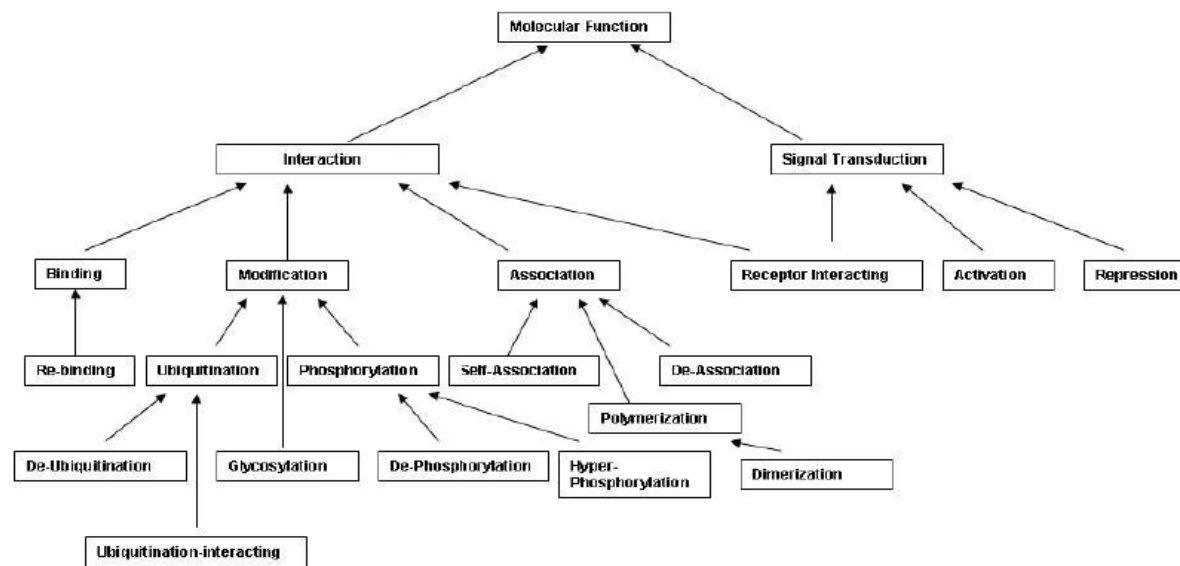


Process 2: PPI ontology construction (cont'd)

 Refine the ontology:

- Conceptualization.
 - Given a set of terms, over a given domain, discover/invent the abstract concepts that can be explicitly/implicitly represented by the terms.
 - Two types:
 - A concept represented by multiple terms.
 - Multiple concepts represented by a single term.
- Formalization.
 - Discover relations between concepts and formalize them.

Sample of refined ontology



Process 3: Integration with Gene Ontology

- 📄 Ontology of PPI domain concepts and Gene Ontology combined to form “PPIWordNet”.
- 📄 Integration cases:
 - Unique concept used *as is*.
 - Same concept having different subclasses *expanded* to include all available subclasses.
 - Same concept having different superclasses *specialized* to use more-detailed relation.

Evaluating PPIWordNet

Goals:

- Test PPIWordNet in lexical-chain analysis of PPI articles.
- Investigate characteristic discourse structure of PPI genre.

Hypotheses:

- PPIWordNet ontology can enable deeper discourse-analysis using lexical chaining than current biomedical IE methods.
- Lexical chains occur throughout PPI texts, can be evaluated by various metrics related to their quantity and quality.

Experiment:

Lexical chaining using PPIWordNet

 Test set:

- 100 full-text articles from BioCreative PPI task corpus.

 Steps:

- Step 1: Use only Gene Ontology (GO) on test set.
- Step 2: GO + Method terms.
- Step 3: GO + Method terms + Interaction Property terms.
- Step 4: GO + Method terms + Interaction Property terms + Molecular Function terms.

Evaluation metrics

 Quantity of lexical chains in an article.

 Quality of lexical chains:

- ***Strength:** Indicating importance of a topic.*
 - Chain length.
 - Chain density.
- ***Richness:** Providing more information, so stronger ‘evidence’ for interaction validity.*
 - Number of unique concepts (‘lemmas’) in a chain.

Results of experiment (He 2007 MMath)

Measurement	GO	GO + Method	GO + Method + IP	GO + Method + IP + MF
# of terms	60020	60038	60070	60089
# of chains	4536	5030	5652	5898
# of chains per doc	45.36	50.30	56.52	58.98
# of chains per para	1.84	2.04	2.30	2.40
average length	5.23	5.16	5.03	5.10
average span	7.44	7.46	7.26	7.46
average lemmas	1.19	1.20	1.29	1.40
average density	0.62	0.61	0.61	0.60


Summary

- ❏ PPIWordNet produced significant improvements in quantity of lexical chains, mild improvements in chain quality:
- ❏ Size of ontology increased each step by average of 2%.
Resulting increase in number of chains was 30% — very significant compared to trivial increase in ontology.
- ❏ Average length of a chain decreased by average of 0.76% between each step.
- ❏ Lemmas (unique concepts) increased each step by average of 5.9% — significant compared to changes in ontology size.

On-going work

- 📄 Improvements to lexical-chaining algorithm:
 - Semantic relatedness function.
 - Use of non-classical (i.e., non-WordNet relations).
- 📄 Expand and refine PPIWordNet.
- 📄 Better metrics for lexical chains.
- 📄 Long-term goal:
 - Judging the validity of protein-protein interactions using *tractable* computational discourse analysis.

Acknowledgements

 The “BioWebDoc Project” is collaborative work with:

- Xiaofen He
- Shady Shehata Hassan
- Gabriel Musso (Toronto)
- Zhou Yu (Toronto)