A STUDY ON THE FUNDAMENTALS OF SEMANTIC ROLE LABELING

by

Tin Wing (Winona) Wu
B.Sc., Simon Fraser University, 2007

PROJECT SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

In the
School of Computing Science

SIMON FRASER UNIVERSITY

Spring 2010

APPROVAL

Name: **Tin Wing (Winona) Wu**

Degree: **Master of Science**

Title of Project: **A STUDY ON THE FUNDAMENTALS OF SEMANTIC ROLE LABELING**

Examining Committee:

Chair: **Name**
Dr. Jim Delgrande
Professor, Computing Science

_____

Name
Dr. Fred Popowich
Professor, Computing Science

_____

Name
Dr. Veronica Dahl
Professor, Computing Science

_____

Name
Dr. Maite Taboada
Associate Professor, Linguistics

Date Defended/Approved: December 22, 2009

Declaration of
Partial Copyright Licence

ABSTRACT

The natural language processing (NLP) community has recently experienced a growing interest in semantic role labeling (SRL) – the process of assigning a WHO did WHAT to WHOM, WHEN, WHERE, WHY and HOW structure to text. The increased availability of annotated resources enables the development of statistical approaches specifically for SRL. This holds potential impact in NLP applications.

In this project, we describe the linguistic background of the SRL problem, major resources that are used and an overview of general approaches in computational systems. We reproduce the approaches to SRL based on Pradhan's ASSERT system extending the work of Gildea and Jurafsky. We examine the system and its individual components, including its annotated resources, parser, classification system, and the features used. We then examine the results obtained by the system and its components. We also assess the challenges in SRL and identify the opportunities for useful further research in SRL.

Keywords: semantic role labeling, support vector machine, classification, parsing

ACKNOWLEDGEMENTS

I would like to thank my supervisor Dr. Fred Popowich for his valuable suggestions, remarks, comments, discussion and encouragement.

I would also like to thank Dr. Veronica Dahl, Yudong Liu and all the people in SFU Natural Language Lab for assisting my project report and their valuable comments.

And a special thanks to my lovely family for their encouragement and support.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1: NATURAL LANGUAGE PROCESSING

We have all been fascinated by the prediction about the future of technology in science fiction stories and movies. A few of the remarkable fictional machines include WALL-E, a garbage collecting robot who has been left to clean up the mess on Earth, C-3PO a droid in Star Wars that can understand and translate six million forms of communication, and HAL 9000 in 2001 A Space Odyssey who is capable of not only carrying intelligent conversation with humans, but also interpreting emotions and reasoning. However, our current state of technology is not as advanced as our fantasy. Our current computers need knowledge about human language and algorithms to be able to process natural (human) language. We enter this knowledge using specific formats, so that computers can extract the necessary information. We can then develop applications based on this knowledge and associated processes.

Computers are applied to a wide range of tasks, and many of these tasks are relatively easy for programmers to design and implement the necessary software. However, there are many tasks that are impossible or difficult. Recent advances are bringing machine learning techniques into the mainstream. Machine Learning is the study of methods for programming computers to learn. This capacity to learn from experience, analytical observation, and other means, results in a system that can continuously self-improve and thereby offer increased efficiency and effectiveness (AAAI, 2008).

Scientists have conducted a great deal of research in intelligent systems that perceive their environment and take action that maximizes their chances of success. Scientific advancements are making it possible for people to talk to smart computers. Research in speech recognition, artificial intelligence, powerful chips and virtual environments is likely to produce this intelligent interface. Natural Language Processing (NLP) is a field in Artificial Intelligence concerned with the interaction between computers and human (natural) language using techniques in computer science and linguistics. One goal of NLP is to design and develop efficient algorithms to analyze, understand, and generate languages that humans use naturally.

1.1 Challenges of NLP

To understand language requires defining concepts such as *word* and *phrase* and figuring out how to link these concepts together in a meaningful way for language processing tasks. Natural language is a medium of communication that is easiest for a human to learn and use. It is very difficult for a computer to master, because of the highly ambiguous nature of natural language.

I made her duck. (1.1)

Jurafsky and Martin (2000) show an example sentence that demonstrates a number of ambiguities cause different meanings of the sentence in (1.1). Figure 1-1 shows five different meanings that (1.1) could have, each of which exemplifies an ambiguity at some level.

a. I cooked waterfowl for her.

b. I cooked waterfowl belonging to her.

c. I created the (plaster?) duck she owns

d. I caused her to quickly lower her head or body.

Figure 1-1 Different meanings for sentence (1.1)

First, the words *duck* and *her* are syntactically and morphologically ambiguous. *Duck* can be verb or a noun, while *her* can be a dative pronoun or a possessive pronoun. Second, the word *make* is semantically ambiguous; it can mean create or cook. It is impossible to tell which without knowing the properties of the sentence.

In order to have better understanding of this highly ambiguous natural language, NLP systems often begin with word-level understanding to interpret the meaning of individual words. Following word-level understanding, NLP systems may perform syntactic analysis that determines the structure of the input text. This structure consists of a hierarchy of *phrases*, the smallest of which are the *basic symbols* (or *words)* and the largest of which is the *sentence*. On the other hand, NLP systems also apply techniques to obtain semantic interpretation of the input text. Semantic interpretation is the process of mapping a syntactically analyzed text of natural language to a representation of its meaning.

1.2 Application of NLP

The goal of NLP is to “accomplish human-like language processing”; therefore, Liddy (2001) suggested that a full NLP System would be able to:

1. Paraphrase an input text

2. Translate the text into another language

3. Answer questions about the contents of the text

4. Draw inferences from the text

Liddy (2001) also stated that even though NLP systems have not been able to draw inferences from text by themselves, NLP has made serious inroads into accomplishing goals 1 to 3. It provides both theories and implementations for a range of applications. The most frequent applications utilizing NLP include the following:

- Information Retrieval (IR) / Extraction (IE) – IR provides a list of potentially relevant documents in response to a user's query, while IE turns large collections of text into structured representation to capture useful information
- Question Answering – responds to questions that are posed in natural language. A QA system parses incoming questions, matches the queries against its knowledge base and presents the appropriate information to the user (Katz, Borchardt, & Felshin, 2002).
- Dialogue and Conversational Agents – use computational linguistics techniques to interpret and respond to statements made by the user in ordinary natural language (Lester, Branting, & Mott, 2004).
- Machine Translation – the use of computers to automate some or all of the process of translating from one language to another (Jurafsky & Martin, 2000). It ranges from the 'word-based' approach to applications that include higher levels of analysis.

1.3 Sub-tasks of NLP

Many NLP applications deal with both generation and understanding of languages. Researchers focus on a wide range of sub-tasks resulting in state-of-the-art technology for robust, broad-coverage natural language processing in many languages. These subtasks cover areas such as:

- Parsing – to determine the grammatical structure of a text with respect to a given formal grammar. Chunking is also used to identify short phrases in text.
- Part of Speech (PoS) tagging – marks up words in a text as corresponding to particular part of speech labels (e.g. noun, verb).
- Word sense disambiguation – to select the meaning that makes the most sense in a context where the word has more than one meaning.
- Text segmentation – to identify word boundaries. This becomes a non-trivial task for some written languages like Chinese and Japanese that do not have single-word boundaries.
- Speech recognition – converts spoken words to text.

Fundamental to research on these subtasks is the notion of evaluation. For many of these subtasks there are standard evaluations techniques and corpora. Standard evaluation metrics from information retrieval include precision, recall and a combined metric called an F_1 measure (Jurafsky & Martin, 2000). Precision is a measure of how much of the information that the system returned is correct, also known as accuracy. Recall is a measure of how much relevant information the system has extracted from text, thus a measure of the coverage

of the system. The F_1 measure balances recall and precision. A corpus is often divided into three sets: training set, development set and testing set. Training set is used for training systems, whereas the development set is used to tune parameters of the learning systems and select the best model. Testing set is used for evaluation. Cross-corpora evaluation is used in some tasks, for which a fresh test set different from the training corpora is used for evaluation.

1.4 Structure of this project

In this project, we are studying a subtask of NLP called Semantic Role Labeling (SRL) that has a great deal of potential for significant impact in NLP applications. In Section 2, we learn about the linguistic background of SRL, major resources, general approaches and basic features use for developing SRL systems. We further investigate the computational approaches to SRL by experimenting with Pradhan's (2005) ASSERT system in Section 3. We present the lesson we learn from reproducing the ASSERT system in Section 4 and conclude the project in Section 5.

2: SEMANTIC ROLE LABELING

The general problem of interpreting semantics involves the determination of the semantic relations among the entities and the events they participate in (Màrquez, Carreras, Litkowski, & Stevenson, 2008). Given a sentence, one formulation to interpret semantic consists of detecting basic event structures such as "who" did "what" to "whom", "when" and "where". Early examples of NLP systems, like the chatterbot ELIZA (Weizenbaum, 1966), use a collection of decomposition rules triggered by keywords to simulate a natural language conversation with a human as psychotherapist. ELIZA has inspired modern NLP applications like the chatterbot ALICE (Wallace, 1995) by applying pattern-matching rules to create a simple illusion of understanding. Current information extraction and dialogue understanding systems are often based on domain-specific frame and slot templates. A new set of slots is required for natural language understanding tasks for each new application domain. Semantic Role Labeling (SRL) (Carreras & Màrquez 2005) is a task in NLP to analyze propositions expressed by some target verbs in a given sentence, and fill all the constituents in the sentence with less domain-specific semantic roles for each target verb.

As with many areas in computational linguistics and NLP, developments in SRL are built on research on manually created semantic grammars and other resources for supporting text interpretation. Graeme Hirst (1987) presented a

theoretically motivated foundation for semantic interpretation using a framework that facilitates the resolution of both lexical and syntactic ambiguities. Recently, medium-to-large corpora have been manually annotated with semantic roles in FrameNet (Fillmore, Ruppenhofer, & Baker, 2004), PropBank (Palmer, Gildea, & Kingsbury, 2005), and NomBank (Meyers, et al., 2004), enabling the development of statistical approaches specifically for SRL. This holds potential for significant impact in many NLP applications, such as Information Extraction, Question Answering, Summarization, and Machine Translation; as well, any NLP tasks that require some kind of semantic interpretation.

2.1 Semantic Roles

A semantic role in language is a type of relationship that a syntactic constituent has with a predicate. This predicate is often the verb of a sentence and typical semantic arguments include Agent, Patient, Instrument, etc. and adjunctive arguments indicating Locative, Temporal, Manner, Cause, etc (Carreras & Màrquez 2005) .The bracketing of the sentence in Figure 2-1 is broken down into arguments which are labelled with semantic roles:

[The girl on the swing $_{\text{AGENT}}$] [**whispered** $_{\text{PRED}}$] to [the boy beside her $_{\text{RECIPIENT}}$]

Figure 2-1 Semantic role consisting of a predicate with Agent and Recipient arguments identified

Although there is substantial agreement on major semantic roles, such as Agent and Theme, there is no consensus on a definitive list of semantic roles, or even whether such a list exists (Màrquez, Carreras, Litkowski, & Stevenson,

2008). At the specific end of the spectrum are domain-specific roles such as FROM_AIRPORT, TO_AIRPORT, or DEPART_TIME, or verb-specific roles such as EATER and EATEN for the verb *eat.* The opposite end of the spectrum consists of theories with only two core roles, Proto-Agent and Proto-Theme. In between, theories proposed approximately ten general semantic roles called thematic roles. Gildea and Jurafsky (2002) gave a number of examples showing how the thematic roles are assigned in Table 2-1:

Role	Example Sentence
AGENT	**Henry** *pushed* the door open and went in.
CAUSE	Jeez, **that** *amazes* me as well as riles me.
DEGREE	I **rather** *deplore* the recent manifestation of Pop; it doesn't seem to me to have the intellectual force of the art of the Sixties.
EXPERIENCER	It may even have been that **John** *anticipating* his imminent doom ratified some such arrangement perhaps in the ceremony at the Jordan.
FORCE	If this is the case can it be *substantiated* **by evidence from the history of developed societies**?
GOAL	Distant across the river the towers of the castle rose against the sky straddling the only land *approach* **into Shrewbury**.
INSTRUMENT	In the children with colonic contractions **fasting motility** did not *differentiate* children with and without constipation.
LOCATION	These fleshy appendages are used to detect and *taste* food **amongst the weed and debris on the bottom of a river.**
MANNER	His brow *arched* **delicately.**
NULL	Yet while she had no intention of surrendering her home, **it** would be *foolish* to let the atmosphere between them become too acrimonious.
PATH	The dung-collector *ambled* slowly **over**, one eye on Sir John.
PATIENT	As soon as a character lays a hand on this item, the skeletal Cleric *grips* **it** more tightly.
PERCEPT	What is *apparent* is **that this manual is aimed at the non-specialist technician, possibly an embalmer who has good knowledge of some medical procedures.**
PROPOSITION	It says that rotation of partners does not *demonstrate* **independence**.
RESULT	All the arrangements for stay-behind agents in northwest Europe collapsed, but Dansey was able to *charm* most of the governments in exile in London **into recruiting spies.**
SOUCE	He heard the sound of liquid slurping in metal container as Farrell *approached* him **from behind.**
STATE	Rex *spied* out Sam Maggott **hollering at all and sundry and making good use of his over-sized red gingham handkerchief.**
TOPIC	He said, "We would urge people to be aware and be *alert* **with fireworks** because your fun might be someone else's tragedy."

Table 2-1 Abstract semantic roles with representative examples from FrameNet corpus

The FrameNet project (Baker, Fillmore, & Lowe, 1998) proposed semantic roles that are neither as general as the abstract thematic roles, nor as specific as the thousands of potential verb-specific roles. FrameNet roles are defined for each semantic frame. Frames are schematic representations of the conceptual

structures and patterns of beliefs, practices, institutions, images, etc. that provide a foundation for meaningful interaction in a given speech community (Fillmore, Ruppenhofer, & Baker, 2004).

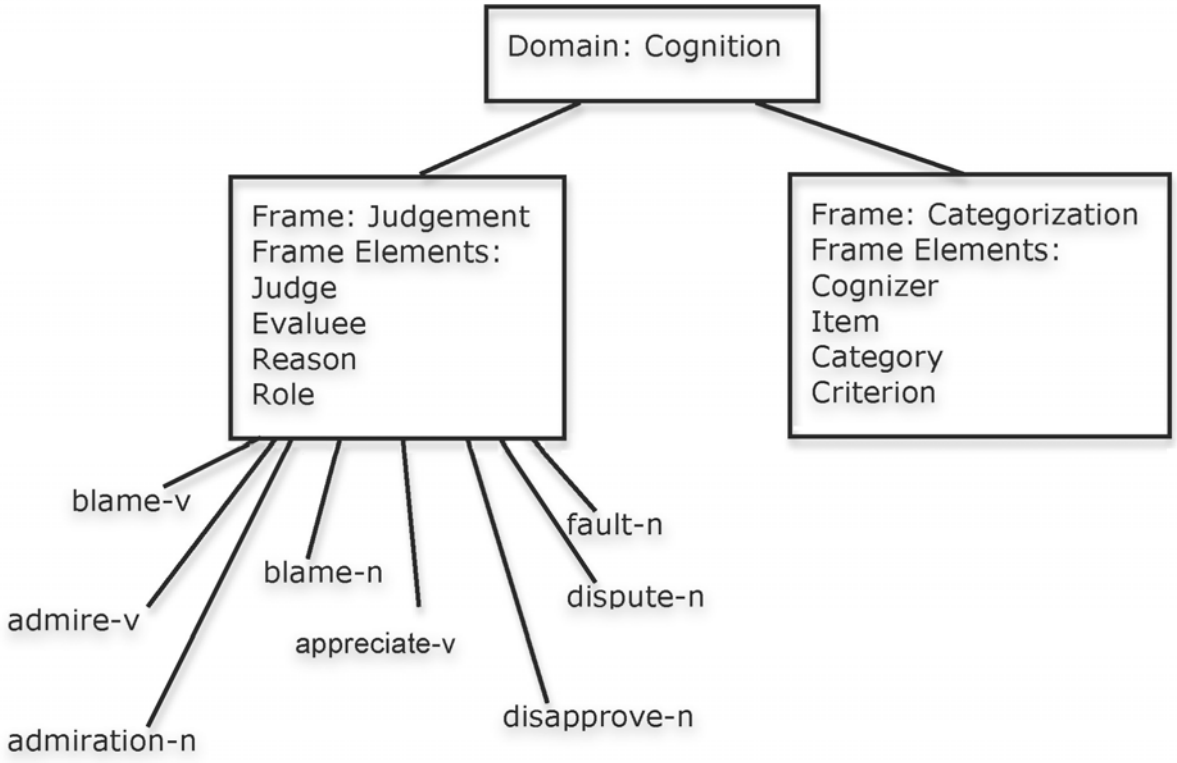

Figure 2-2 Example of semantic frames and their relationships to other frames

From Figure 2-2, the frame JUDGEMENT frame shown on the left of the figure has the roles JUDGE, EVALUEE, and REASON, is semantically related to verbs such as *blame, admire,* and *praise,* and nouns such as *fault* and admiration. We refer the roles for a given frame as frame elements. A hand-annotated example from the JUDGEMENT frame is shown in Figure 2-3:

[She JUDGE] **blames** [the Government EVALUEE] [for failing to do enough to help REASON]

Figure 2-3 Example sentence annotated with the JUDGEMENT semantic frame elements

Defining semantic roles at this intermediate frame level helps avoid some of the well-known difficulties of defining a unique small set of universal, abstract thematic roles, while also allowing some generalization across the roles of different verbs, nouns, and adjectives, each of which adds additional semantics to the general frame, or highlights a particular aspect of the frame (Gildea & Jurafsky, 2002). The difference between thematic roles and semantic frames is that thematic roles tend to be arguments mainly of verbs, frame elements can be arguments of any predicate, and the FrameNet database thus includes nouns and adjectives as well as verbs.

2.2 Data Set

A major focus of work in the computational linguistics community is on the mapping between the predicate argument structure that determines the roles, and the syntatic realization of the recipients of those roles. These linguistic approaches to semantic roles have greatly influenced current work on SRL, leading to the creation of significant computational lexicons capturing the foundational properties of predicate-argument relations.

In the FrameNet project (Fillmore, Ruppenhofer, & Baker, 2004), lexicographers define a frame to capture some semantic situation (e.g., Arrest), identify lexical items as belonging to the frame (e.g., apprehend and bust), and devise appropriate roles for the frame (e.g., Suspect, Authorities, Offense). They

then select and annotate example sentences from the British National Corpus and other sources to illustrate the range of possible assignments of roles to sentence constituents for each lexical item (at present, over 141,000 sentences have been annotated) (Màrquez, Carreras, Litkowski, & Stevenson, 2008) . Fillmore, Ruppenhofer, & Baker (2004) shows marked up example sentences for the REVENGE frame in Figure 2-4.

1. [They $_{\text{AVENGER}}$] took **revenge** [for the deaths of two loyalists prisoners $_{\text{INJURY}}$].
2. Why hadn't [he $_{\text{AVENGER}}$] sought to **avenge** [his child $_{\text{INJURED PARTY}}$] ?
3. The Old Bailey was told [he $_{\text{AVENGER}}$] was desperately in love and wanted to **get back** [at the women $_{\text{OFFENDER}}$] ["for ending their relationship" $_{\text{INJURY}}$]

Figure 2-4 Marked up example sentences for REVENGE frame from FrameNet corpus

The existence of the FrameNet corpus enabled Gildea and Jurafsky (2002) to develop the first statistical machine learning approach to SRL, using seven lexical and syntactic features including the phrase type of each constituent, its grammatical function, and position in the sentence. Gildea and Jurafsky has often been used as the baseline study of SRL. The Senseval-3 task (Litkowski, 2004) called for the development of systems to meet the same objectives as the Gildea and Jurafsky study. The data for this task would be a sample of the FrameNet hand-annotated data.

Although this research has encouraged refinements and extensions on Gildea and Jurafsky's approach, the FrameNet data has not been used extensively. One issue is that the corpus is not a representative sample of the language, but rather consists of sentences chosen manually to illustrate the

possible role assignments for a given lexical item (Màrquez, Carreras, Litkowski, & Stevenson, 2008).

Other works focused on the extraction of predicate argument structures has resulted in the Proposition Bank (PropBank). The PropBank (Palmer, Gildea, & Kingsbury, 2005) takes a practical approach to semantic representation, adding a layer of predicate-argument information (semantic roles) to the syntactic structures of the Penn Treebank. The development of the PropBank was inspired by the research on VerbNet (Kipper, Dang, & Palmer, 2000). VerbNet regularizes and extends the original Levin classes that categorize verbs according to shared meaning and behaviour. Additionally, for each verb and each sense it defines the set of possible roles for that verb usage, called the roleset. PropBank contains annotated semantic roles for all the verbs in the Penn Treebank corpus (the Wall Street Journal [WSJ] news corpus) using the definition of the verb senses from VerbNet. This provides a representative sample of text with role-annotations, in contrast to FrameNet's reliance on manually selected, illustrative sentences. In addition, PropBank's composition allows for consideration of the statistical patterns across natural text. Although there is some concern about the limited genre of its newspaper text, this aspect has the advantage of allowing SRL systems to benefit from state-of-the-art syntactic parsers like Charniak (2000) and other resources developed with the WSJ TreeBank data such as name entity recognition system IdentiFinder™ (Bikel, Schwartz, & Weischedel, 1999). Moreover, current work is extending the PropBank annotation to balanced corpora such as the Brown corpus.

PropBank (Palmer, Gildea, & Kingsbury, 2005) has emerged as a primary resource for research in SRL. The verbs in PropBank have been tagged with coarse-grained senses and with inflectional information. Each verb has a frameset; the frameset lists the allowed role labels in which the arguments are designated by number (starting from zero like ARG0). Each numbered argument is provided with an English language description specific to that verb. Verbs with different senses have different framesets. While designations of ARG0 and ARG1 are intended to indicate the general roles of Agent and Theme/Patient across verbs, other argument numbers do not consistently correspond to general (non-verb-specific) semantic roles. For example, the verb *decline* has two framesets. **Decline.01** in Figure 2-5 has a set of arguments describing components related to going down that is different from the set of arguments of **decline.02** in Figure 2-6 that indicate the roles of rejection.

Frameset: **decline.01** “go down incrementally”
ARG1: entity going down
ARG2: amount gone down by, EXT
ARG3: start point
ARG4: end point
Ex: … [it net income $_{ARG1}$] **declining** [42% $_{ARG2\text{-}EXT}$] [to $121 million $_{ARG4}$] [in the first 9 months of 1989 $_{ARGM\text{-}TMP}$].

Figure 2-5 decline frameset: "go down incrementally"

Frameset: **decline.02** “demure, reject”
ARG0: agent
ARG1: rejected thing
Ex: … [A spokesman $_{ARG0}$] **declined** [*trace* to elaborate $_{ARG1}$]

Figure 2-6 decline frameset: "demure, reject"

The CoNLL-2004 Shared Task used the annotations provided from the PropBank to come up with machine learning strategies addressing the SRL problem on the basis of only partial syntactic information, avoiding the use of full

parsers and external lexico-semantic knowledge bases. Other levels of processing treated in the previous editions of the CoNLL shared task such as: part-of-speech (PoS) tags, chunks, and name entities, were provided for the development of the system. The preprocessors corresponded to the state-of-the-art system for each level of annotation. The best system, presented by the most experienced group on the task (Hacioglu, Pradhan, Ward, Martin, & Jurafsky, 2004), achieved a moderate performance of 69.49 for the F_1 measure. It is based on a Support Vector Machine (SVM, refer to Section 3.2.4 for more information) tagging system, performing IOB decisions on the chunks of the sentence (I is used to mark a word inside the phrase, O marks for outside the phrase, and B marks for beginning of the phrase), and exploited a wide variety of features based on partial syntax. CoNLL 2004 (Carreras & Màrquez 2004) summarized that most of the systems advance the state-of-the-art on SRL on the basis of partial syntax. However, state-of-the-art systems working with full syntax still perform substantially better.

Compared to the shared task of CoNLL-2004, the CoNLL-2005 shared task aimed at evaluating the contribution of full parsing in SRL using complete syntactic trees from two alternative parsers. A substantially enlarged training corpus, PropBank, was used to input information for the task for testing the scalability of learning-based SRL systems to big datasets and to compute learning curves to see how much data is necessary for training. Preprocessing from the previous editions of the CoNLL shared task, i.e., words, PoS tags, base chunks, clauses, and name entities, and annotation of predicate-argument

structure of the PropBank corpus were also available. However, a cross corpora evaluation was performed using a fresh test set from the Brown corpus to test the robustness of the presented systems. CoNLL-2005 (Carreras & Màrque 2005) reported that the best system presented by (Punyakanok, Roth, & Wen-Tau 2005) achieves an F_1 at 79.44 on the WSJ test. Furthermore, the performance of such an SRL module in a real application would be about ten points lower, as demonstrated in the evaluation on the sentences from the Brown corpus.

The CoNLL 2008 shared task (Surdeanu, Johansson, Meyer, Màrquez, & Nivre, 2008) took a different approach by proposing a unified dependency-based formalism, which modelled both syntactic dependencies and semantic roles. Using this formalism, this shared task merged both the task of syntactic dependency parsing and the task of identifying semantic arguments and labeling them with semantic roles. In this task, the SRL problem addressed not only propositions centered around verbal (PropBank) predicates but also around nouns (NomBank).

The NomBank (Meyers, et al., 2004) is an annotation project related to the PropBank project. It provides argument structure for common nouns in the Penn Treebank corpus, and it uses essentially the same framework as PropBank to annotate arguments of nouns. Differences between PropBank and NomBank stem from differences between noun and verb argument structure. In NomBank, the various arguments and adjuncts of the head nouns are labelled with *roles* (sets of argument labels for each sense of each noun). (2.1) shows an example of noun predicate *gift* in NomBank.

[Her $_{ARG0}$] **gift** of [a book $_{ARG1}$] [to John $_{ARG2}$]. (2.1)

The CoNLL 2009 Shared Task (Hajič 2009) built on the CoNLL 2008 task and extended it to multiple languages. The core of the task was to predict syntactic and semantic dependencies and their labeling. Data was provided for both statistical training and evaluation, which extract these labelled dependencies from manually annotated Treebanks such as the Penn Treebank for English, the Prague Dependency Treebank for Czech and similar Treebanks for Catalan, Chinese, German, Japanese and Spanish languages, enriched with semantic relations (such as those captured in the Prop/NomBank and similar resources). Great effort has been devoted to provide the participants with a common and relatively simple data representation for all the languages, similar to the 2008 English data.

Role-annotated data makes it available for many research opportunities in SRL including a broad spectrum of probabilistic and machine learning approaches. We have introduced datasets associated with SRL; we are now prepared to discuss the main approaches to automatic SRL.

2.3 Approaches to Automatic SRL

Given a sentence and a designated verb, the SRL task consists of identifying the boundaries of the arguments of the verb predicate (argument identification) and labeling them with semantic roles (argument classification). The most common architecture for automatic SRL consists of the following steps to achieve these subtasks.

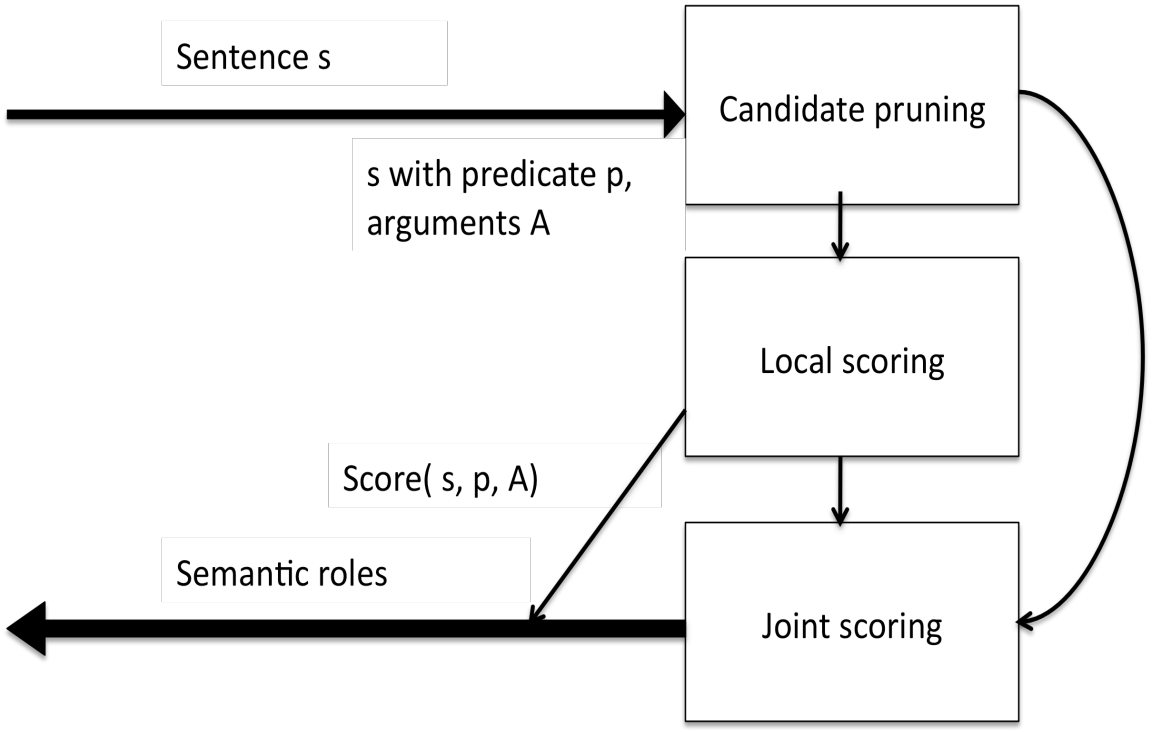

Figure 2-7 The common architecture for automatic SRL

The first step in SRL typically consists of identifying potential predicates and filtering (or pruning) the set of argument candidates for a given predicate. Arguments may be a continuous or discontinuous sequence of words; any subsequence of words in the sentence is an argument candidate. Xue and Palmer (2004) developed simple heuristic rules to filter out constituents that are clearly not semantic arguments to the target predicate. Their approach first designated the predicate as the current node of the syntactic tree, and collected its sister nodes (constituents attached at the same level as the predicate) and the sisters' immediate children. It then reset the current node to its parent node and collected its sisters and sisters' children nodes until it reached the top-level node. These simple rules greatly reduced the set of candidate arguments, while maintaining a very high recall. Figure 2-8 illustrates an example to identify related

semantic arguments using these rules. The circled nodes in Figure 2-8 are the constituents that the rules keep.

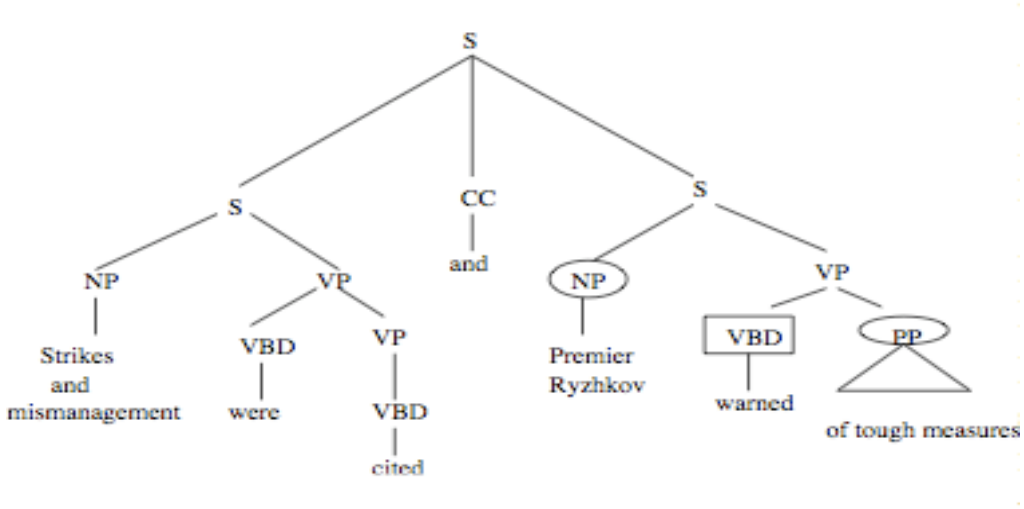

Figure 2-8 An example for identify related semantic argument using rules from Xue and Palmer (2004)

The second step consists of a local scoring of argument candidates by calculating the probabilities of a candidate argument to be labelled by each of the possible role labels, plus an extra “no-argument” label meaning that the candidate should not be considered an argument in the solution. A crucial aspect in local scoring is the representation of candidates with features, rather than the particular choice of classification algorithm. Argument identification and classification may be treated jointly or separately in the local scoring step. In the latter case, a pipeline of two sub-processes is typically applied, first scoring between “argument” and “no-argument” labels, and then scoring the particular argument labels.

The third step in SRL is to apply a joint scoring (or global scoring) in order to combine the predictions of local scorers to produce a good structure of

labelled arguments for the predicate. Some models may apply re-ranking to select the best among a set of candidate complete solutions produced by a base SRL system. (Johansson & Nugues, 2008) had the highest ranked system using a re-ranking strategy in the closed challenge of CoNLL 2008 shared task. Their thorough system addressed all facets of the task with state-of-the-art methods. It used a second-order parsing model for argument identification/classification models separately tuned for PropBank and NomBank. The second-order parsing model uses feature function not only of head-dependent links, but also of sibings and children of the dependent. The system used global learning model with global constraint features to correct bias problems introduced by the previous architecture, and finally integrated syntactic and semantic analysis in a reranking step, which maximize the joint syntactic-semantic score in the top k solutions. This novel task is attractive both from a research perspective and an application-oriented perspective. It is believed that the proposed dependency-base representation is a better fit for many applications. It hopes to expand this effort with evaluations on multiple languages in CoNLL 2009 shared task, and on larger out-of-domain corpora.

There are other variations in the three-step architecture. Systems may bypass one of the steps, by doing only local scoring, or skipping directly to joint scoring. An important consideration within this general SRL architecture is the combination of systems and input annotations. Most SRL systems include some kind of combination to increase robustness, gain coverage, and reduce effects of parse errors. The combination can be as simple as selecting the best among the

set of complete candidate solutions, but usually consists of combining fragments of alternative solutions to construct the final output. The gain in performance from the combination step is consistently between two and three F_1 points. However, a combination approach increases system complexity and penalizes efficiency.

2.4 Features Engineering

As previously noted, devising the features with which to encode candidate arguments is crucial for obtaining good results in the SRL task. Given a verb and a candidate argument (a syntactic phrase) to be classified in the local scoring step, three types of features are typically used:

1. Features that characterize the candidate argument and its context;
2. Features that characterize the verb predicate and its context;
3. Features that capture the relation (either syntactic or semantic) between the candidate and the predicate

Gildea and Jurafsky (2002) presented a compact set of features, which has served as the core of most of the subsequent SRL work:

Feature	Description
Predicate	The predicate itself.
Path	The minimal path from the constituent being classified to the predicate.
Phrase Type	The syntactic category (NP, PP, etc.) of the constituent being classified.
Position	The relative position of the constituent being classified with regard to the predicate (before or after)
Voice	Whether the predicate is active or passive
Head Word	The syntactic head of the phrase
Sub-categorization	The phrase structure rule expanding the parent of the predicate.

Table 2-2 Basic features

Extensions to these features have been proposed in various directions. Exploiting the ability of some machine learning algorithms to work with very large feature spaces, features have largely extended using the representation of the constituent and its context, including among others: first and last words (and part-of-speech) in the constituent, bag-of-words, n-grams of part of speech, and sequence of top syntactic elements in the constituent. For instance, Surdeanu et al. (2003) generalized the concept of headword with the content word feature. Xue and Palmer (2004) presented the syntactic frame features, which capture the overall sentence structure using the verb predicate and the constituent as pivots.

2.5 Evaluation

The standard experiment in automatic SRL can be defined as follows: Given a sentence and a target predicate appearing in it, find the arguments of the predicate and label them with semantic roles. A system is evaluated with respect to precision, recall and F_1 measure. Precision (p) is the proportion of arguments

predicted by a system that are correct. Recall (r) is the proportion of correct arguments that are predicted by a system. F_1 measure computes the harmonic mean of precision and recall, and is the final measure to compare the performance of system. It is formulated as

$$F_1 = \frac{2pr}{(p+r)} \qquad (2.2)$$

Performance can be divided into two components: 1) precision, recall and F_1 of unlabeled arguments, meaning the segmentation accuracy of the system 2) the classification accuracy of assigning semantic roles to the arguments that have been correctly identified. An argument is considered correct when both its boundaries and the semantic role label match a gold standard. Credits may be given for partial matching; if a system assigns the incorrect predicate sense, it still receives some points for the arguments correctly assigned.

The Gildea & Jurafsky (2002) study assembled a set of suitable metrics on accuracy, precision and recall to evaluate the performance of an automatic SRL system. The CoNLL 2004 shared task is evaluated with respect to precision, recall and the F_1 measure. For an argument to be correctly recognized, the words spanning the argument as well as its semantic role have to be correct. On the other hand, the CoNLL 2008 evaluation measures consist of three different scores: syntactic dependencies are scored using the labelled attachment score (LAS), the semantic dependencies are evaluated using the labelled F_1 score. The overall task is scored with a harmonic mean of macro precision and recall scores calculated by averaging the two previous scores.

3: SRL SYSTEMS

3.1 Gildea & Jurafsky

The Gildea & Jurafsky (2002) system is based on statistical classifiers trained on roughly 50,000 sentences that were hand-annotated with semantic roles by the FrameNet project. It then parsed each training sentence into a syntactic tree and extracted the lexical and syntactic features listed in Section 2.4. It used lexical clustering algorithms to generalize across possible fillers of roles.

Test sentences were parsed, were annotated with these features, and were then passed through the classifiers. The system achieved up to 82% accuracy in identifying the semantic role of pre-segmented constituents correctly. At the more difficult task of simultaneously segmenting constituents and identifying their semantic role, the system achieved 65% precision and 61% recall.

3.2 ASSERT

Pradhan & Jurafsky (2004) proposed a machine-learning algorithm for shallow semantic parsing, extending the work of Gildea and Jurafsky. The ASSERT system (Pradhan et al., 2004) first replaced the statistical classification algorithm with one that uses Support Vector Machines. It evaluated a series of modifications and a number of new features to improve its performance. Adding

features that are generalizations of the more specific features was helpful; these features were named entities, headword part of speech and verb clusters. The system reformulated the task as a combined chunking and classification problem, allowing its algorithm to be applied to new languages or genres of text for which statistical syntactic parsers may not be available. It used the PropBank 2002 corpus and was evaluated using both hand corrected TreeBank syntactic parses, and actual parses from the Charniak parser. On the task of assigning semantic labels to the PropBank corpus, the ASSERT system has a precision of 84% and a recall of 75%. In this section, we study the main components of ASSERT: PropBank, syntactic parser, classification, and understand how they are used in ASSERT.

3.2.1 PropBank

As we mentioned in Section 2.2, PropBank is a corpus in which predicate argument relations are marked for almost all occurrences of verbs in the Wall Street Journal (WSJ) part of the Penn TreeBank. In this section, we will look into PropBank in greater detail and examine how it is used in ASSERT.

Recall that PropBank labels the arguments of a verb sequentially from ARG0 to ARG5, where ARG0 is the AGENT (usually the subject of a transitive verb) ARG1 is the PATIENT (usually its direct object). Note that A0 and A1 frequently correspond to the AGENT and PATIENT of the proposition, other argument numbers do not consistently correspond to general semantic roles. Figure 3-1 provides examples for labeling numbered arguments of a verb:

[John $_{ARG0}$] **broke** [the window $_{ARG1}$]

[The window $_{ARG1}$] **broke**

Figure 3-1 Example sentences annotated with numbered arguments

In addition to the numbered arguments, the PropBank annotation also involves assigning functional tags to all modifiers of the verb, such as manner (MNR), locative (LOC), temporal (TMP) and others as shown in Figure 3-2.

Mr Bush met him privately, in the White House, on Thursday.

REL: met

ARG0: Mr. Bush

ARG1: him

ARGM-MNR: privately

ARGM-LOC: in the White House

ARGM-TMP: on Thursday

Figure 3-2 Example annotated with functional tags

Finally, PropBank annotation involves finding antecedents for 'empty' arguments of the verbs.

I made a decision [*] to leave.

REL: leave

ARG0: [*] -> I

Figure 3-3 Example with an empty category

The subject of the verb '*leave*' in this example of Figure 3-3 is represented as an empty category [*] in Treebank. In PropBank, all empty categories are linked with their associated NPs within the same sentence.

3.2.1.1 Frame Files

The argument labels for each verb are specified in the frame files. A set of argument labels and their definitions is called a frameset. Each frame file provides a unique identifier for the verb sense, a meaning for that verb sense, and the set of expected arguments with the ARG-numbers and a description for that ARG (Kingsbury & Palmer, 2003). Example sentences demonstrating various syntactic realizations for that frameset are included following the definitions. Figure 3-4 provides an example of a frame file for the verb '*expect*':

Roleset ID: expect.01

Roles:

ARG0: expected

ARG1: thing expected

Figure 3-4 Frame file for verb '*expect*'

Given the sentence in Figure 3-5, we show the roles in the frame in Figure 3-4 filled with the constituents from the sentence.

Portfolio managers expect further declines in interest rate.

ARG0: Portfolio managers

REL: expect

ARG1: further declines in interest rates

Figure 3-5 An example sentence associated with frame from Figure 3-4

Because the meaning of each argument number is depending on the verb, the verb usage in a sentence, or verb sense, it is impossible to provide one set of semantic roles for all senses of the verb. For example, the two senses of the verb '*leave*' in the examples below take different arguments:

Mary left the room (3.1)

Mary left her daughter-in-law her pearls in her will (3.2)

In such cases, frame files distinguish two or more verb senses and define argument labels specific to each Frameset. The frameset for (3.1) and (3.2) are shown in Figure 3-6 and 3-7 respectively.

Frameset leave.01 "move away from":

ARG0: entity leaving

ARG1: place left

Figure 3-6 Frameset for leave.01

Frameset leave.01 "give":

ARG0: giver

ARG1: thing given

ARG2: beneficiary

Figure 3-7 Frameset for leave.02

Despite this generality, ARG0 is often assigned as an 'AGENT'-type role, while ARG1 consistently has a PATIENT or THEME role as well. During annotating, we first select the frameset and then assign the argument labels as specified for this frameset. Some frame files have multiple framesets; it is absolutely necessary to check the frame file to see if the verb has more than one frameset. In some cases, frame files define not only several framesets for each verb, but also several predicates. If a verb has a particle (marked as PRT in TreeBank), then it is being considered as a different predicate, and has a different set of semantic roles. For example, the frame file for the verb 'keep' defines three predicates: predicate 'keep' (which has 3 framesets), and

predicates '*keep_up'* and '*keep_on'*.

3.2.1.2 Adjunct tags

Adjunct tags are general arguments that any verb may take optionally. There are 13 types of adjunct as reflected in Table 3-1. The following definition is extracted from the 2005 Annotation guidelines for PropBank; please refer to it for a detailed explanation (Babko-Malaya, 2005).

Adjuncts	**Description**
DIR	Directional modifiers show motion along some path
LOC	Locative modifiers indicate where some action takes place
MNR	Manner adverbs specify how an action is performed
EXT	ARGM-EXT indicates the amount of change occurring from an action, and is used mostly for numerical adjuncts, quantifiers, and comparatives.
REC	These include reflexives and reciprocals such as *himself, itself, themselves, together, each other, jointly, both.*
PRD	These are used to show that an adjunct of a predicate is in itself capable of carrying some predicate structure.
PNC	Purpose clauses are used to show the motivation for some action.
CAU	Similar to "Purpose clauses", these indicate the reason for an action.
DIS	These are markers that connect a sentence to a preceding sentence
ADV	These are used for syntactic elements which clearly modify the event structure of the verb in question, but which do not fall under any of the headings above. As opposed to ARGM-MNR, which modifies the verb, ARGM-ADVs usually modifies the entire sentence.
MOD	Modals are usually: *will, may, can, must, shall, might, should, could,* and *would*.
NEG	Negation is elements such as "not", "n't", "never", "no longer" and other markers of negative sentences.

Table 3-1 List of Adjunct tags in PropBank

In addition to the semantic roles described in the rolesets in Section 3.1, verbs can take any of a set of adjunct-like arguments (ARGMs), distinguished by one of the function tags shown in Table 3. While the PropBank provides meaning

using these adjunct tags, it does not distinguish the different roles played by a verb's grammatical subject or object. The same verb used with the same syntactic sub-categorization can assign different semantic roles.

3.2.1.3 Annotation of null elements

Null elements used in the Penn Treebank are annotated as shown in Table 3-2 (Babko-Malaya, 2005). They are often used to connect with components in passive sentences, fronted and dislocated arguments, questions and wh-phrases and relative clauses.

[*T*]	(trace of A-movement, including parasitic gaps)
[(NP *)]	(arbitrary PRO, controlled PRO, and trace of A-movement)
[0]	(null complementizer, including null wh-operator)
[*U*]	(unit)
[*?*]	(placeholder for ellipsed material)
[*NOT*]	(anti-placeholder in template gapping)
[*RNR*]	(pseudo-attach: right node raising)
[*ICH*]	(pseudo-attach: interpret constituent here)
[*EXP*]	(pseudo-attach: expletive)
[*PPA*]	(pseudo-attach: permanent predictable ambiguity)

Table 3-2 Different types of null elements

Arguments with null elements represent arguments realized in other parts of the sentences. The role of the reference is the same as the role of the referenced argument, an annotation of R- tag prefixed to the label of the referent, e.g. R-A1, is used in the CoNLL 2004 corpus (Carreras & Màrquez 2004). However, null elements are not produced by a syntactic parser, the developers of the ASSERT system decided not to consider them in the experiment.

3.2.2 Syntactic Processing

Semantic roles are closely related to syntax, and, therefore, automatic SRL heavily relies on the syntactic structure of the sentence. Syntactic structure of the sentence is often used for extracting useful features. Thus, it has become a common practice to use full parse trees to define argument boundaries and extract relevant information for training classifiers to disambiguate between role labels. Punyakanok, Roth and Yih (2008) and Surdeanu et al.(2007) have shown that a system working with partial parsing can do almost as well as a system working with full parses, with differences in F score of only 3 points. Punyakanok, Roth and Yih (2008) and Surdeanu et al.(2007) also reported that incorrect syntactic constituents caused many errors in SRL. By using many parses, the recognition of semantic roles is more robust to parsing errors.

Other promising approaches draw on dependency parsing rather than traditional phrase structure parsing (Johansson & Nugues, 2007). Dependency parsing (Covington, 2000) is to use dependency grammar to draw links connecting individual words, this concept occurs naturally for ones who want to explain agreement or case assignment. Dependency-parsed tree should make more sense semantically than those produced by constituent approaches. The new format gave a 23% error reduction for semantic role labeling classification.

Gildea and Jurafsky (2002) used the parser of Collins (2003) to generate parses from its data to extract features. Pradhan chose the Charniak parser over Collins parser for the ASSERT system for two reasons. First, at the time the source code only for the Charniak parser (2000) was available, and it could be modified to accept data from standard input required for the interactive parsing

application; and second, preliminary experiments of the ASSERT system indicated that the Charniak parser was faster than the Collins' parser. In these generated parses, about 6% of the arguments have boundaries that did not align exactly with any of the hand-generated phrase boundaries.

Charniak statistical parser (2000) is based on a probabilistic generative model. It returns the parse π that maximizes the probability p (π |s) for any s. The model assigns a probability to a parse by top-down processing that considers each constituent c in π and for each c first guessing the pre-terminal of c, t(c) (t for "tag"), then the lexical head of c, h (c), and then the expansion of c into further constituents e (c). Thus, Equation 1 gives the probability of a parse:

$$\begin{aligned} p(\pi) \quad = \quad & \prod_{c \in \pi} p(t(c) \mid l(c), H(c)) \\ & \cdot p(h(c) \mid t(c), l(c), H(c)) \\ & \cdot p(e(c) \mid l(c), t(c), h(c), H(c)) \end{aligned}$$

Equation 1 Probability of a parse

Where l (c) is the label of c (e.g., noun phrase, verb phrase) and H (c) is the relevant history of c. At the time, maximum entropy approach had been strongly recommended to probabilistic model builders for its flexibility. The use of a maximum entropy inspired model for conditioning and smoothing allows many different conditioning events to be combined and evaluated. Modifying the set of features used can easily change the probability.

The Charniak parser achieved 90.1% average precision/ recall for sentences of length 40 and less, and 89.5% for sentences of length 100 and less when trained and tested on the "standard" sections of the Wall Street Journal Treebank.

3.2.3 New Features

Pradhan (2004) experimented with several features on top of the basic features proposed by Gildea and Jurafsky (2002), to find out their effect on the argument classification and argument identification tasks. Two of these new features were obtained from the Surdeanu et al. (2003) literature, that reported performance gains by adding named entities in constituents and the headword part of speech.

3.2.3.1 Name entities in constituents

Following Surdeanu et al. (2003), some of these name entities such as location and time are expected to be particularly important for the adjunctive arguments ARGM-LOC and ARGM-TMP. Seven named entities (PERSON, ORGANIZATION, LOCATION, PERCENT, MONEY, TIME, DATE) were tagged using IdentiFinder^{TM} (Bikel, Schwartz, & Weischedel, 1999) and were added as 7 binary features.

3.2.3.2 Headword part of speech

Surdeanu et al. (2003) showed that using the part of speech (POS) of the headword gave a significant performance boost to their system. Therefore, this feature is added for the ASSERT system.

3.2.3.3 Verb clustering

Since the training data is relatively limited, any real world test set will contain predicates that have not been seen in training. Using predicate cluster as a feature can incorporate some information about the predicate. The distance

function used for clustering is based on the intuition that verbs with similar semantics will tend to have similar direct objects. For example, verbs such as “eat”, “devour”, “savor”, will tend to all occur with direct objects describing food. The verbs were clustered into 64 classes using the probabilistic co-occurrence model of Hofmann and Puzicha (1998). The clustering algorithm used a database of verb-direct-object relations extracted by Lin (1998). The verb class of the current predicate is then used as a feature.

3.2.3.4 Partial Path

Path is one of the most salient features for the argument identification task. However, it is also the most data sparse feature. To overcome this problem, ASSERT tried generalizing the path by adding a new feature that contains only the part of the path from the constituent to the lowest common ancestor of the predicate and the constituent.

3.2.3.5 Verb sense information

The arguments that a predicate can take depend on the word sense of the predicate. Each predicate tagged in the PropBank corpus is assigned a separate set of arguments depending on the sense in which it is used. Table 3-3 illustrates the argument set for the predicate ‘*talk’*. Depending on the sense of the predicate *talk,* either ARG1 or ARG2 can identify the ‘*hearer’.* Absence of this information can be potentially confusing to the learning mechanism.

	Sense 1: speak		Sense 2: persuade/ dissuade	
Talk	**Tag**	**Description**	**Tag**	**Description**
	ARG0	Talker	ARG0	Talker
	ARG1	Subject	ARG1	Talked to
	ARG2	Hearer	ARG2	Secondary action

Table 3-3 The argument set for the predicate *talk*

3.2.3.6 Head word of prepositional phrases

Many adjunctive arguments, such as temporal and locatives, occur as prepositional phrases in a sentence, and it is often the case that the head words of those phrases, which are always prepositions, are not very discriminative, e.g. “in the city”, “in a few minutes”, both share the same head word “in” and neither contain a name entity.

3.2.3.7 First and last word/ POS in constituent

Some arguments tend to contain discriminative first and last words; therefore, first and last word are used as new features along with their part of speech.

3.2.3.8 Ordinal constituent position

In order to avoid false positives of the type where constituents far away from the predicate are spuriously identified as arguments, this feature is added to concatenate the constituent type and its ordinal position from the predicate, e.g.: first NP to the right of the predicate, second PP from the predicate.

3.2.3.9 Constituent tree distance

This is a finer way of specifying the present position feature, by defining the

tree distance of the phrase from the predicate.

3.2.3.10 Constituent relative features

These are nine features representing the phrase type, headword and headword part of speech of the parent and left and right siblings of the constituent in focus. These were added on the intuition that encoding the tree context this way might add robustness and improve generalization.

3.2.3.11 Temporal cue words

There are several temporal cue words that were not captured by the named entity taggers and were considered for addition as binary features indicating their presence.

3.2.3.12 Dynamic class context

In the task of argument classification, these are dynamic features that represent the hypotheses of at most two previous nodes belonging to the same tree as the node being classified.

3.2.4 Classification

Support Vector Machines (SVMs) have been shown to perform well on text classification tasks, where data is represented in a high dimensional space using sparse feature vectors. An SVM constructs a hyperplane that separates the training data into two binary classes. The optimal hyperplane is to find a good separation that maximizes the distance between hyperplane and the nearest training data points. This distance is also called a margin. In general, the larger

the margin means a lower generalization error of the classifier.

ASSERT formulated the parsing problem as a multi-class classification problem and uses an SVM classifier. However, SVMs are binary classifiers. There are two common approaches for extending SVMs to multi-class classification problems. The first is known as a PAIRWISE approach, where a separate binary classifier is trained for each of the class pairs and their outputs are combined to predict the classes. This approach requires the training of $n \times \frac{(n-1)}{2}$ binary classifiers. The second, known as the ONE VS ALL (OVA) approach, involves training *n* classifiers for an *n*-class problem. The classifiers are trained to discriminate between examples of each class, and those belonging to all other classes combined.

ASSERT is built using TinySVM along with YamCha as SVM training and test software. YamCha is a generic, customizable, and open source text chunker oriented toward a lot of NLP tasks, such as POS tagging, named entity recognition, base NP chunking, and text chunking. YamCha (Kudo & Matsumoto, 2000) has outstanding performance in chunking for the CoNLL 2000 Shared Task. It is also used to extract name entities in the molecular biology domain (Takeuchi & Collier, 2002) and in Japanese (Asahara & Matsumoto, 2003).

4: PUTTING THE COMPONENTS TOGETHER

Now that we have seen the various components of an SRL system, let us consider issues related to combining them and evaluating their behaviour.

Recall that we start with the PropBank files, and we need to extract two types of information from the PropBank frame files as shown in Figure 4-1. Since PropBank is stored in XML files, we can use an XML parser and regular expressions in Python NLTK (Bird, Klein, & Loper, 2009) to extract example sentences and their semantic roles based on XML tags. We store them into two text files: annotated sentences in an appropriate format for the Charniak Parser and the semantic role labels.

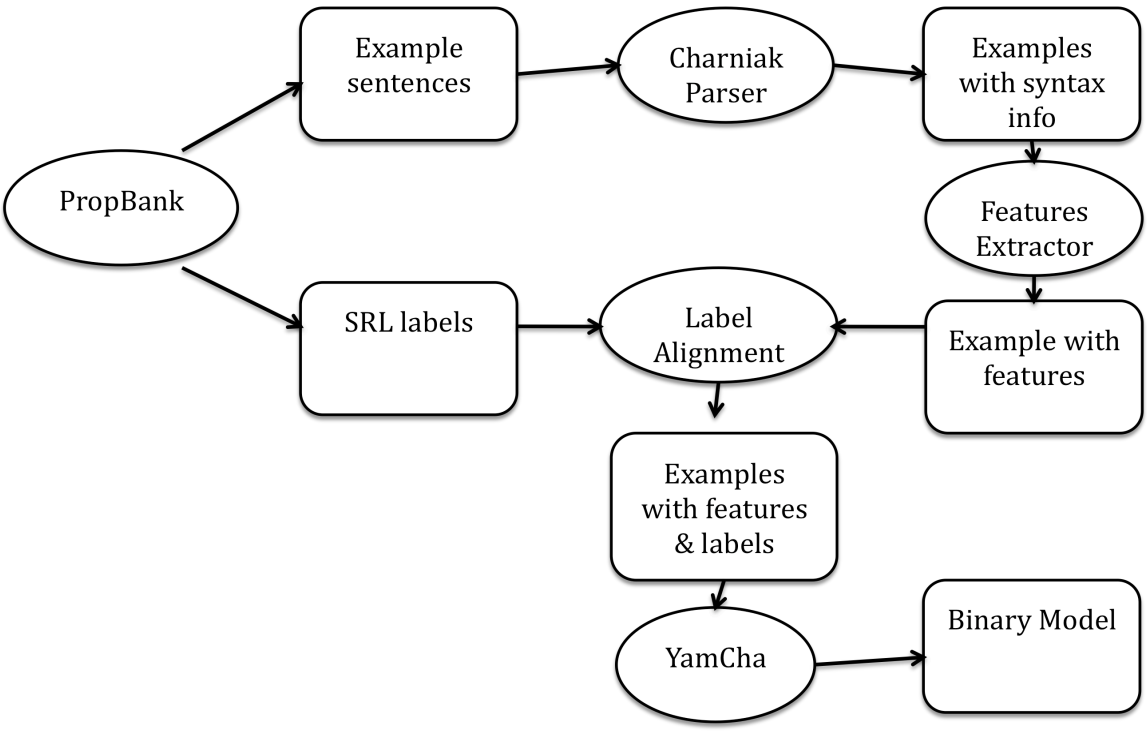

Figure 4-1 Data flow diagram for all the components

Figure 4-2 shows the format of the example sentences for the Charniak Parser in which example sentences are stored one sentence per line. The semantic role labels are stored one constituent per line as shown in Figure 4-3 for subsequent use by a label alignment process. The text file containing all the sentences from the PropBank Corpus in the format shown in Figure 4-2 is then sent to the Charniak Parser to obtain syntactic structure information. The parser returns a text file that contains part of speech tags corresponding to the constituents in the sentences.

```
And they believe the Big Board under Mr Phelan has abandoned their interest
John abandoned his pursuit of an Olympic gold medal as a waste of time
One Colombian drug boss upon hearing in 1987 that Gen Noriega was negotiating with the US to abandon
his command for a comfortable exile sent him a hand-sized mahogany coffin engraved with his name
Once he had abandoned himself to the very worst once he had quieted all the dragons of worry and
suspense there would n't be very much for Mae to do
```

Figure 4-2 Example sentences stored in the format for Charniak Parser

```
And 0
they 0
believe 0
the ARG0
Big ARG0
Board ARG0
under ARGM
Mr ARGM
Phelan ARGM
has 0
their ARG1
interest ARG1

John ARG0
his ARG1
pursuit ARG1
of ARG1
an ARG1
Olympic ARG1
gold ARG1
medal ARG1
as 0
a ARG2
waste ARG2
of 0
time ARG2
```

Figure 4-3 Semantic role labels corresponding to the first two example sentences

As shown in Figure 4-1, after the example sentences have been processed by the Charniak Parser, the tagged sentences are sent to a feature extractor to obtain the basic features from Section 2.4 and additional features from Section 3.2.3 to support the classifications. The feature extractor generates a text file consisting of feature information for each constituent of the sentences.

The text file with feature information for each example sentence then aligns with the associated semantic roles, which were collected from PropBank, again as illustrated in Figure 4-1 underneath PropBank. Yamcha uses these examples with feature information along with the semantic role labels to construct a binary model using an SVM machine learning algorithm.

During the implementation of the system described above, there was a range of integration issues related to each of the component discussed in Section 3. In the rest of Chapter 4, we will describe the issues and outline how they were addressed.

4.1 Inconsistency of PropBank 1.0

It turns out that there are many errors in PropBank 1.0. There were a handful of typos and misspellings in the corpora. Some of the example sentences did not have the correct predicates in the frame files, leading to errors when trying to parse the files correctly.

After modifying PropBank 1.0 to correct these problems, we realized that an SRL system could work better with the corpora provided by the CoNLL 2004 or 2005 shared task (Carreras and Màrquez 2004) instead of the original PropBank. The use of CoNLL has not been previously noted in the SRL literature. The CoNLL data uses the PropBank annotation to describe argument structure. During the creation of the CoNLL data, procedures were applied to check the consistency of propositions, looking for overlapping arguments and incorrect semantic role labels. Carreras and Màrquez (2004) reported that a total number of 68 propositions were not compliant with its procedures and were filtered out from the CoNLL dataset.

4.2 Use of the Charniak Parser

The PropBank data contains many null elements as explained in Section 3.2.1.3. These null elements are used to connect with components in passive

sentences, fronted and dislocated arguments, questions and wh-phrases and relative clauses. However, the Charniak parser is unable to interpret the purpose of these null elements contained in the training data. Figure 4-4 shows an example sentence that uses the target verb *announce*.

```
<example>
        <text>
                Kent cigarettes were sold, the company announced *Trace*
        </text>

        <arg n="0"> the company </arg>
        <rel> announced </rel>
        <arg n="1"> *Trace* </arg>
        <note> (*Trace → Kent cigarettes were sold)</note>
<example>
```

Figure 4-4 Example sentence that uses null element

The Charniak Parser is unable to identify *Trace* and ignores the meaning that *Trace* is a pointer that refers to "Kent cigarettes were sold". Therefore, we need to remove the null elements from the annotated sentences and label the dislocated arguments with their associated semantic roles. By doing so, we lose the actual location of the argument referenced by the null element.

Additionally, the Charniak parser is unable to identify some target predicate verbs from the PropBank. We removed 155 verbs from PropBank to resolve this issue; which left us with 3102 verbs for use in our system.

4.3 YamCha

After parsing the sentences with the Charniak parser, we use the feature extractor provided in the ASSERT system to retrieve feature information as

explained in Section 2.4 and Section 3.2.3. The features along with the semantic role labels are then sent to YamCha to compile a binary model. Recall that YamCha is an SVM based chunker that provides SVM classifications. The format of training data file in YamCha is easier to understand than the ones in TinySVM as we shall see below.

An example of the training data file format used in TinySVM for classifying class +1 and -1 is shown on Figure 4-5. The numbers 201, 3148 are features for the class and 1.2, 1.8 are the associated values for the features. The format of each line starts with the class followed by pairs of features and values that are separated by colons.

(BNF-like representation)
<class> .=. +1 | -1
<feature> .=. integer (>=1)
<value> .=. real
<line> .=. <class> <feature>:<value><feature>:<value> ... <feature>:<value>

```
Example (SVM)
+1 201:1.2 3148:1.8 3983:1 4882:1
-1 874:0.3 3652:1.1 3963:1 6179:1
+1 1168:1.2 3318:1.2 3938:1.8 4481:1
+1 350:1 3082:1.5 3965:1 6122:0.2
-1 99:1 3057:1 3957:1 5838:0.3
```

Figure 4-5 Format of training data in TinySVM

Figure 4-6 shows an example of training data files used in YamCha for classifying IOB tags (I marks for inside the phrase, O marks for outside the phrase, and B marks for beginning of the phrase). Each line contains information for an individual constituent, its features and its classification label. Therefore, the first line “He” is the word itself, PRP is the part of speech for “He” and B-NP is the classification label for the beginning (B) of a noun phrase (NP).

There are 3 columns for each token.

- The word itself (e.g. reckons);
- part-of-speech associated with the word (e.g. VBZ);
- Chunk(Answer) tag represented in IOB2 format;

```
He          PRP   B-NP
reckons     VBZ   B-VP
the         DT    B-NP
current     JJ    I-NP
account     NN    I-NP
deficit     NN    I-NP
will        MD    B-VP
narrow      VB    I-VP
to          TO    B-PP
only        RB    B-NP
#           #     I-NP
1.8         CD    I-NP
billion     CD    I-NP
in          IN    B-PP
September   NNP   B-NP
.           .     O

He          PRP   B-NP
reckons     VBZ   B-VP
..
```

Figure 4-6 Format of training data in YamCha

Pradhan (2005) used Yamcha to examine how a part of speech tagger followed by a syntactic phrase chunker could replace a full syntactic parser. The POS tagger and syntactic chunker were both implemented using YamCha. Pradhan observed a significant drop in F_1 score from 65.2 to 60.0 when features are derived from a flat-chunked parse. The major difference is in the derivation of the path feature. Pradhan further illustrated the effect by running both systems without using the path feature. Similar performance was obtained for the two systems, but it is believed that an important step towards bridging this gap would be to adopt a two pass approach in the word-by-word paradigm, analogous to the constituent-by-constituent paradigm.

4.4 System Performance

A large size corpus like PropBank leads to many challenges in memory allocation and file handling; therefore, we divided the corpus into 8 subparts to evaluate system performance. We extracted example sentences from the PropBank range in alphabetical order of the target verb. Table 4-1 shows the range of verbs that are contained in the 8 subparts.

Subpart	Verbs
1	abandon – certify
2	chafe - die
3	differ - dissatisfy
4	dissect - line
5	linger - power
6	practise - safeguard
7	sag - star
8	stare - zoom

Table 4-1 Ranges of verbs in all eight subparts

Each subpart has approximately 930 example sentences. These sentences along with their feature information and semantic labels were submitted to YamCha to build binary models as described in Figure 4-1. Each model takes approximately four hours to train on an AMD Athlon 64 X2 Dual Core machine with a 4200+ processor and 2GB RAM. We were able to successfully generate models for seven of the eight subparts.

We took sentences from the CoNLL2005 Brown test corpus for evaluation; 15 sentences are tailored to the verbs of each binary model for the SRL task. We randomly selected three sets of sentences and submitted them to both our

project system and the ASSERT system. Table 7 shows the precision and recall results compared to the gold standard labeling of the sentences.

	Our system		ASSERT	
	Precision	**Recall**	**Precision**	**Recall**
Subset 1	34%	42%	84%	81%
Subset 2	42%	51%	72%	60%
Subset 3	42%	57%	70%	68%
Average:	39%	50%	75%	70%

Table 4-2 Precision and Recall for our project system and ASSERT system

As discussed in Section 2.5, precision (p) is the proportion of arguments predicted by a system that are correct and recall (r) is the proportion of correct arguments that are predicted by a system. We believe that the poor performance in identifying argument boundaries have caused our project system to score only 39% in precision. Our system basically assumes every constituent has a label and tries to assign a semantic role onto it while many of these constituents should not be considered to have semantic roles assigned to them. Figure 4-7 shows the results for one sentence where the part of the sentence "and did not enter the argument" should not be consider as part of the ARGM-MNR. Figure 4-8 shows the results for the same sentence as processed by the ASSERT system, which also happens to be the gold standard for this sentence.

[Scotty $_{ARG0}$] [**accepted** $_{TARGET}$] [the decision $_{ARG1}$] [with indifference and did not enter the arguments $_{ARGM-MNR}$]

Figure 4-7 A result sentence labelled by our project system

[Scotty $_{ARG0}$] [**accepted** $_{TARGET}$] [the decision $_{ARG1}$] [with indifference $_{ARGM-MNR}$] and did not enter the arguments

Figure 4-8 Same sentence from Figure 4-7 labelled by the ASSERT system

On the other hand, constituents that are indeed part of arguments are labelled with a reasonable accuracy by our system, giving us a recall score of 50%. Figure 4-9 shows resulting sentences obtained by this project, while Figure 4-10 shows the results from the original ASSERT system. Note that "*you*", "*not*" and "*enough"* are correctly segmented as arguments and are indeed correctly labelled; the ASSERT system is unable to label "*enough"* correctly.

[You $_{ARG0}$] [do $_{ARG0}$] [not $_{ARGM-NEG}$] [**eat** $_{TARGET}$] [enough $_{ARG1}$] [honey $_{ARG1}$]

Figure 4-9 A result sentence labelled by our project system

[You $_{ARG0}$] do [not $_{ARGM-NEG}$] [**eat** $_{TARGET}$] [enough $_{ARGM-EXT}$] honey

Figure 4-10 Same sentence from Figure 4-9 labelled by the ASSERT system

[You $_{ARG0}$] do [not $_{ARGM-NEG}$] [**eat** $_{TARGET}$] [enough $_{ARG1}$] honey

Figure 4-11 Gold standard label for sentence in Figure 4-9 and Figure 4-10

In this section, we have described a system that uses the same approach as ASSERT to assign semantic roles. Given that ASSERT is effectively a "black box", we have developed a system as outlined in Figure 4-1 that attempts to get the same behaviour. We are unsure whether the black box model in ASSERT performs argument identification and argument classification at the same time. We assumed that Yamcha performed multi-class classification to perform argument identification and classification jointly. Since it compares one class with all other classes, it should assign an unlabeled class to arguments that are unlikely to be labeled. However, our experiment shows that our system fails to perform the two tasks together. We strongly encourage others to perform these tasks separately to eliminate argument candidates that are unnecessary for

argument classification. This will significantly reduce the amount of data to learn for the classification model.

5: CONCLUSION

5.1 Summary

In this project, we introduced a NLP task related to semantic interpretation called Semantic Role Labeling (SRL). We first described the linguistic background of SRL and focused on major resources such as FrameNet, PropBank and NomBank developed in the computational linguistic community like CoNLL shared tasks, which can be applied to the SRL task. We then looked at the major steps and features used in SRL systems.

We then investigated a computational implementation of SRL by experimenting with an SRL system based on Pradhan's (2005) ASSERT system, extending the work of Gildea and Jurafsky (2002). We examined its individual components, including its annotated resources, parser, classification system, and the features used. We overcame the problems of cleaning the large inconsistent PropBank data corpus and the challenges of working with large amounts of data in the Yamcha SVM-based classifier. Through our experiment, we saw the significant impact of distinguishing between the argument identification and classification tasks to label semantic roles correctly.

5.2 Future Work

SRL is no exception, as with many NLP tasks, for having challenges in applying a system to a new domain different than the domain used to develop

and train the system. Predicates in a new domain may differ from the dictionary of frames at training time. In the CoNLL -2005 task (Carreras and Màrquez 2005), WSJ-trained systems were tested on three sections of the Brown corpus annotated by the PropBank team. The performance of all systems dropped dramatically: The best system had an F_1 score below 70%, as opposed to scores in the area of 80% when tested on WSJ data. Pradhan (2008) further investigated the robustness across text genres when applying a system from WSJ to Brown corpora and discovered that the loss in accuracy takes place in assigning the semantic roles, rather than in the identification of argument boundaries.

On the other hand, SemEval-2007 (Màrquez et al. 2007) featured the first evaluation exercise of SRL systems for languages other than English, namely for Spanish and Catalan. Xue (2008) also studied semantic role labeling for Chinese, using the Chinese PropBank and NomBank corpora. The CoNLL 2009 shared task (Hajič 2009) was dedicated to semantic role labeling using syntactic and semantic dependencies on Catalan, Chinese, Czech, English, German, Japanese and Spanish. The best system scored an average of F_1 82.64 across the seven languages. Hajič (2009) claimed that it remains unclear whether the joint learning of syntactic and semantic dependencies has a significant advantage for SRL in other languages. This shared task prepared a unified format and data for several languages for SRL and also provided three languages on out of domain data for testing purposes. There is a great opportunity in applying the techniques we have examined to other languages.

SRL systems have shown to perform reasonably well in some controlled experiments, with F_1 measures in the low 80s on standard test collections for English. Most SRL approaches require training data that is both difficult and highly expensive to produce across different genres and different languages. It is critical for the future of SRL that research broadens to include wider investigation of unsupervised and minimally supervised learning methods.

BIBLIOGRAPHY

AAAI. 2008. *AITopics / Machine Learning*. From http://www.aaai.org/AITopics/pmwiki/pmwiki.php/AITopics/MachineLearning

Asahara, M., & Matsumoto, Y. 2003. Japanese Named Entity Extraction with Redundant Morphological Analysis. *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1.* Edmonton, Canada.

Babko-Malaya, O. 2005. *PROPBANK ANNOTATION GUIDELINES* . From Proposition Bank: http://verbs.colorado.edu/~mpalmer/projects/ace/PBguidelines.pdf

Baker, C. F., Fillmore, C. J., & Lowe, J. B. 1998. The Berkeley FrameNet project. *Proceedings of the 17th international conference on Computational linguistics - Volume 1.* Montreal, Quebec, Canada.

Bikel, D. M., Schwartz, R., & Weischedel, R. M. 1999. An Algorithm that Learns What's in a Name. *Machine Learning Journal Special Issue on Natural Language Learning* , 211 - 231 .

Bird, S., Klein, E., & Loper, E. 2009. *Natural Language Processing with Python*. From Natural Language Toolkit: http://www.nltk.org/

Carreras, X., & Màrquez, L. 2004. Introduction to the CoNLL-2004 Shared Task: Semantic Role Labeling. *In Proceedings of the CoNLL-2004 Shared Task.* Boston, MA USA.

Carreras, X., & Màrquez, L. 2005. Introduction to the CoNLL-2005 Shared Task: Semantic Role Labeling. *In Proceedings of the CoNLL-2005 Shared Task.* Ann Arbor, MI USA.

Charniak, E. 2000. A Maximum Entropy Inspired Parser. *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference.* Seattle, Washington.

Collins, M. 2003. Head-Driven Statistical Models for Natural Language Parsing. *Computational Linguistics* (pp. 589 - 637). MIT Press.

Covington, M. 2000. A Fundamental ALgorithm for Dependency Parsing. *In Proceedings of the 39th Annual ACM Southeast Conference.* Athens, Georgia.

Fillmore, C., Ruppenhofer, J., & Baker, C. 2004. FrameNet and Representing the Link between Semantic and Syntactic Relations. In Huang, Chu-Ren, & W. L. Lenders, *Computational Linguistics and Beyond* (pp. 19-59). Academia Sinica.

Gildea, D., & Jurafsky, D. 2002. Automatic Labeling of Semantic Roles. *In Proceedings of the 38th Annual Conference of the Association for Computational Linguistics*, (pp. 512–520). Hong Kong.

Hacioglu, K., Pradhan, S., Ward, W., Martin, J. H., & Jurafsky, D. 2004. Semantic role labeling by tagging syntactic chunks. *In Proceedings of CoNLL-2004.* Boston, MA USA.

Hajič, J., Ciaramita, M., Johansson, R., Kawahara, D., Martí, M. A., Màrquez, L., et al. 2009. The CoNLL-2009 Shared Task: Syntactic and Semantic Dependencies in Multiple Languages. *Proceedings of the CoNLL 2009: Shared Task.* Boulder, Colorado.

Hirst, G. 1987. *Semantic Interpretation and the Resolution of Ambiguity.* Cambridge.

Hofmann, T., & Puzicha, J. 1998. *Statistical models for co-occurrence data.* Memo, Massachusetts Institute of Technology Artificial Intelligence Laboratory.

Johansson, R., & Nugues, P. 2008. Dependency based Syntactic Semantic Analysis with Propbank and NomBank. *In Proc. of CoNLL-2008 Shared Task.* Manchester, UK.

Johansson, R., & Nugues, P. 2007. Extended Constituent-to-dependency Conversion for English. *In Proceedings of NODALIDA 2007.* Tartu, ESTONIA.

Jurafsky, D., & Martin, J. H. 2000. Machine Translation. In *Speech and Language Processing.* Prentice Hall.

Katz, B., Borchardt, G., & Felshin, S. 2002. *How START works*. From The START Natural Language Question Answering System: http://start.csail.mit.edu/start-system.html

Kingsbury, P., & Palmer, M. 2003. *PropBank: the Next Level of TreeBank .* From http://w3.msi.vxu.se/~rics/TLT2003/doc/kingsbury_palmer.pdf

Kipper, K., Dang, H. T., & Palmer, M. 2000. Integrating compositional semantics into a verb lexicon. *COLING-2000 Eighteenth International Conference on Computational Linguistics.* Saarbrücken, Germany.

Kudo, T., & Matsumoto, Y. 2000. Use of Support Vector Learning for Chunk Identification. *CoNLL 2000.* Lisbon, Portugal .

Lester, J., Branting, K., & Mott, B. 2004. Conversational Agents. In *The Practical Handbook of Internet Computing.* Chapman & Hall.

Liddy, E. 2001. Natural Language Processing. In I. Marcel Decker, *In Encyclopedia of Library and Information Science* (2nd Ed. ed.). NY.

Lin, D. 1998. Automatic retrieval and clustering of similar words. *In Proceedings of the International Conference on Computational Linguistics (COLING/ACL-98).* Montreal, Quebec, Canada.

Litkowski, K. 2004. *Senseval-3 Task: Automatic Labeling of Semantic Roles*. From Semseval: http://www.senseval.org/senseval3

Màrquez, L., Carreras, X., Litkowski, K. C., & Stevenson, S. (2008. Semantic Role Labeling: An introduction to the Special Issue. *Computational Linguistics. 34*, pp. 145-159. MIT Press.

Màrquez, L., Villarejo, L., Martí, M. A., & Taulé, M. 2007. Multileve semantic annotation of Catalan and Spanish. *In Proceedings of th 4th International Workshop on Semantic Evaluations (SemEval-2007).* Prague.

Meyers, A., Reeves, R., Macleod, C., Szekely, R., Zielinska, V., Young, B., et al. 2004. The NomBank Project: An Interim Report. *In Proceedings of the HLT-NAACL 2004 Workshop: Frontiers in Corpus Annotation.* Boston.

Palmer, M., Gildea, D., & Kingsbury, P. 2005. The Proposition Bank: An Annotated Corpus of Semantic Roles. *Computational Linguistics. 31*, pp. 71-106. MIT Press.

Pradhan, S. S., Ward, W., Hacioglu, K., Martin, J. H., & Jurafsky, D. 2004. Shallow Semantic Parsing using Support Vector Machines. *in Proceedings of the Human Language Technology Conference/North American chapter of the Association for Computational Linguistics annual meeting (HLT/NAACL-2004).* Boston, USA.

Pradhan, S., Hacioglu, K., Krugler, V., Ward, W., Martin, J. H., & Jurafsky, D. 2005. Support Vector Learning for Semantic Argument Classification. *Machine Learning* , 11 - 39 .

Pradhan, S., Ward, W., & Martin, J. 2008. Towards Robust Semantic Role Labeling. *Computational Linguistics: Special Issue on Semantic Role Labeling. 34*, pp. 289-310 . MIT Press.

Punyakanok, V., Roth, D., & Wen-Tau, Y. 2005. Generalized inference with multiple semantic role labeling systems. *In Proceedings of CoNLL-2005.* Ann Arbor, MI USA.

Punyakanok, V., Roth, D., & Yih, W.-t. 2008. The Importance of Syntactic Parsing and Inference in Semantic Role Labeling. *Computational Linguistics*, *34*, pp. 257-287 .

Pustejovsky, J. 1995. *Generative Lexicon.* MIT Press.

Surdeanu, M., Harabagiu, S., Williams, J., & Aarseth, P. 2003. Using predicate-argument structures for information extraction. *In Proceedings of the 41stAnnual Meeting of the Association for Computational Linguistics.* Japan.

Surdeanu, M., Johansson, R., Meyer, A., Màrquez, L., & Nivre, J. 2008. The CoNLL-2008 Shared Task on Joint Parsing of Syntactic and Semantic Dependencies. *CoNLL 2008.* Manchester, UK.

Surdeanu, M., Màrquez, L., Carreras, X., & Comas, P. 2007. Combination strategies for semantic role labeling. *Journal of Artificial Intelligence Research(JAIR)*, *29.*

Takeuchi, K., & Collier, N. 2002. Use of support vector machine in extended name entity. *CONLL 2002.* Taipei, Taiwan.

Wallace, R. 1995. *From ELIZA to A.L.I.C.E.* From A.L.I.C.E AI Foundation: http://www.alicebot.org/articles/wallace/eliza.html

Weizenbaum, J. 1966. ELIZA--A Computer Program For the Study of Natural Language Communication Between Man and Machine. *Communication of ACM.*

Xue, N. 2008. Labeling Chinese Predicates with Semantic roles. *Computational Linguistic. 34*, pp. 225-255. MIT Press.

Xue, N., & Palmer, M. 2004. Calibrating Features for Semantic Role Labeling. *In Proceedings of EMNLP-2004.* Barcelona, Spain.