ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 10, October 2016

Analysis of Probabilistic Parsing in NLP

Krishna Karoo, Dr.Girish Katkar

Research Scholar, Department of Electronics & Computer Science, R.T.M. Nagpur University, Nagpur, India

Head of Department, Department of Computer Science, Taywade College, Koradi, Maharashtra, India

ABSTRACT: The most explanatory method for presenting what actually happens within a Natural Language processing system is by means of the 'levels of language' approach. This is also referred to as the synchronic model of language and is distinguished from the earlier sequential model, which hypothesizes that the levels of human language processing follow one another in a strictly sequential manner. Psycholinguistic research suggests that language processing is much more dynamic, as the levels can interact in a variety of orders. Introspection reveals that we frequently use information we gain from what is typically thought of as a higher level of processing to assist in a lower level of analysis. For example, the pragmatic knowledge that the document you are reading is about biology will be used when a particular word that has several possible senses (or meanings) is encountered, and the word will be interpreted as having the biology sense of necessity, the following description of levels will be presented sequentially. The key point here is that meaning is conveyed by each and every level of language and that since humans have been shown to use all levels of language to gain understanding, the more capable an NLP system is, the more levels of language it will utilize.

I. INTRODUCTION

LEVELS OF NATURAL LANGUAGE PROCESSING

A. PHONOLOGY

This level deals with the interpretation of speech sounds within and across words. There are, in fact, three types of rules used in phonological analysis: 1) phonetic rules – for sounds within words; 2) phonemic rules – for variations of pronunciation when words are spoken together, and; 3) prosodic rules – for fluctuation in stress and intonation across a sentence. In an NLP system that accepts spoken input, the sound waves are analyzed and encoded into a digitized signal for interpretation by various rules or by comparison to the particular language model being utilized.

B. MORPHOLOGY

This level deals with the componential nature of words, which are composed of morphemes – the smallest units of meaning. For example, the word preregistration can be morphologically analyzed into three separate morphemes: the prefix pre, the root registra, and the suffixtion. Since the meaning of each morpheme remains the same across words, humans can break down an unknown word into its constituent morphemes in order to understand its meaning. Similarly, an NLP system can recognize the meaning conveyed by each morpheme in order to gain and represent meaning. For example, adding the suffix –ed to a verb, conveys that the action of the verb took place in the past. This is a key piece of meaning, and in fact, is frequently only evidenced in a text by the use of the -ed morpheme.

C. LEXICAL

At this level, humans, as well as NLP systems, interpret the meaning of individual words. Several types of processing contribute to word-level understanding – the first of these being assignment of a single part-of-speech tag to each word. In this processing, words that can function as more than one part-of-speech are assigned the most probable part-of speech tag based on the context in which they occur. Additionally at the lexical level, those words that have only one possible sense or meaning can be replaced by a semantic representation of that meaning. The nature of the representation varies according to the semantic theory utilized in the NLP system. The following representation of the meaning of the word launch is in the form of logical predicates. As can be observed, a single lexical unit is decomposed into its more basic properties. Given that there is a set of semantic primitives used across all words; these simplified lexical representations make it possible to unify meaning across words and to produce complex interpretations, much the same as humans do.

launch (a large boat used for carrying people on rivers, lakes harbors, etc.)
((CLASS BOAT) (PROPERTIES (LARGE)
(PURPOSE (PREDICATION (CLASS CARRY) (OBJECT PEOPLE))))

The lexical level may require a lexicon, and the particular approach taken by an NLP system will determine whether a lexicon will be utilized, as well as the nature and extent of information that is encoded in the lexicon. Lexicons may be quite simple, with only the words and their part(s)-of-speech, or may be increasingly complex and contain information on the semantic class of the word, what arguments it takes, and the semantic limitations on these arguments, definitions of the sense(s) in the semantic representation utilized in the particular system, and even the semantic field in which each sense of a polysemous word is used.

D. SYNTACTIC

This level focuses on analyzing the words in a sentence so as to uncover the grammatical structure of the sentence. This requires both a grammar and a parser. The output of this level of processing is a (possibly delinearized) representation of the sentence that reveals the structural dependency relationships between the words. There are various grammars that can be utilized, and which will, in turn, impact the choice of a parser. Not all NLP applications require a full parse of sentences, therefore the remaining challenges in parsing of prepositional phrase attachment and conjunction scoping no longer stymie those applications for which phrasal and clausal dependencies are sufficient. Syntax conveys meaning in most languages because order and dependency contribute to meaning. For example the two sentences: 'The dog chased the cat.' and 'The cat chased the dog.' differ only in terms of syntax, yet convey quite different meanings.

E. SEMANTIC

This is the level at which most people think meaning is determined, however, as we can see in the above defining of the levels, it is all the levels that contribute to meaning. Semantic processing determines the possible meanings of a sentence by focusing on the interactions among word-level meanings in the sentence. This level of processing can include the semantic disambiguation of words with multiple senses; in an analogous way to how syntactic disambiguation of words that can function as multiple parts-of-speech is accomplished at the syntactic level. Semantic disambiguation permits one and only one sense of polysemous words to be selected and included in the semantic representation of the sentence. For example, amongst other meanings, 'file' as a noun can mean either a folder for storing papers, or a tool to shape one's fingernails, or a line of individuals in a queue. If information from the rest of the sentence were required for the disambiguation, the semantic, not the lexical level, would do the disambiguation. A wide range of methods can be implemented to accomplish the disambiguation, some which require information as to the frequency with which each sense occurs in a particular corpus of interest, or in general usage, some which require consideration of the local context, and others which utilize pragmatic knowledge of the domain of the document.

F. DISCOURSE

While syntax and semantics work with sentence-length units, the discourse level of NLP works with units of text longer than a sentence. That is, it does not interpret multi sentence texts as just concatenated sentences, each of which can be interpreted singly.

Rather, discourse focuses on the properties of the text as a whole that convey meaning by making connections between component sentences. Several types of discourse processing can occur at this level, two of the most common being anaphora resolution and discourse/text structure recognition. Anaphora resolution is the replacing of words such as pronouns, which are semantically vacant, with the appropriate entity to which they refer. Discourse/text structure recognition determines the functions of sentences in the text, which, in turn, adds to the meaningful representation of the text. For example, newspaper articles can be deconstructed into discourse components such as: Lead, Main Story, Previous Events, Evaluation, Attributed Quotes, and Expectation.

G. PRAGMATIC

This level is concerned with the purposeful use of language in situations and utilizes context over and above the contents of the text for understanding the goal is to explain how extra meaning is read into texts without actually being encoded in them. This requires much world knowledge, including the understanding of intentions, plans, and goals. Some NLP applications may utilize knowledge bases and inferencing modules. For example, the following two sentences require resolution of the anaphoric term 'they', but this resolution requires pragmatic or world knowledge.

The city councilors refused the demonstrators a permit because they feared violence. The city councilors refused the demonstrators a permit because they advocated revolution.

Summary of Levels

Current NLP systems tend to implement modules to accomplish mainly the lower levels of processing. This is for several reasons. First, the application may not require interpretation at the higher levels. Secondly, the lower levels have been more thoroughly researched and implemented. Thirdly, the lower levels deal with smaller units of analysie, e.g. morphemes, words, and sentences, which are rule-governed, versus the higher levels of language processing which deal with texts and world knowledge, and which are only regularity-governed. As will be seen in the following section on Approaches, the statistical approaches have, to date, been validated on the lower levels of analysis, while the symbolic approaches have dealt with all levels, although there are still few working systems which incorporate the higher levels.

II. APPROACHES TO NATURAL LANGUAGE PROCESSING

Natural language processing approaches fall roughly into four categories:
Symbolic, Statistical, Connectionist, and Hybrid.

Symbolic and statistical approaches have coexisted since the early days of this field. Connectionist NLP work first appeared in the 1960's.For a long time, symbolic approaches dominated the field. In the 1980's, statistical approaches regained popularity as a result of the availability of critical computational resources and the need to deal with broad, real-world contexts. Connectionist approaches also recovered from earlier criticism by demonstrating the utility of neural networks in NLP. This section examines each of these approaches in terms of their foundations, typical techniques, differences in processing and system aspects, and their robustness, flexibility, and suitability for various tasks.

A. SYMBOLIC APPROACH

Symbolic approaches perform deep analysis of linguistic phenomena and are based on explicit representation of facts about language through well-understood knowledge representation schemes and associated algorithms. In fact, the description of the levels of language analysis in the preceding section is given from a symbolic perspective.

The primary source of evidence in symbolic systems comes from human-developed rules and lexicons.

A good example of symbolic approaches is seen in logic or rule-based systems. In logic based systems, the symbolic structure is usually in the form of logic propositions. Manipulations of such structures are defined by inference procedures that are generally truth preserving. Rule-based systems usually consist of a set of rules, an inference engine, and a workspace or working memory. Knowledge is represented as facts or rules in the rule-base. The inference engine repeatedly selects a rule whose condition is satisfied and executes the rule .Another example of symbolic approaches is semantic networks. First proposed by Quillian to model associative memory in psychology, semantic networks represent knowledge through a set of nodes that represent objects or concepts and the labeled links that represent relations between nodes. The pattern of connectivity reflects semantic organization, that is; highly associated concepts are directly linked whereas moderately or weakly related concepts are linked through intervening concepts. Semantic networks are widely used to represent structured knowledge and have the most connectionist flavor of the symbolic models.

Symbolic approaches have been used for a few decades in a variety of research areas and applications such as information extraction, text categorization, ambiguity resolution, and lexical acquisition. Typical techniques include: explanation-based learning, rule-based learning, inductive logic programming, decision trees, conceptual clustering, and K nearest neighbor algorithms.

B. STATISTICAL APPROACH

Statistical approaches employ various mathematical techniques and often use large text corpora to develop approximate generalized models of linguistic phenomena based on actual examples of these phenomena provided by the text corpora without adding significant linguistic or world knowledge. In contrast to symbolic approaches, statistical approaches use observable data as the primary source of evidence. A frequently used statistical model is the Hidden Markov Model (HMM) inherited from the speech community. HMM is a finite state automaton that has a set of

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 10, October 2016

states with probabilities attached to transitions between states. Although outputs are visible, states themselves are not directly observable, thus "hidden" from external observations.

Each state produces one of the observable outputs with a certain probability. Statistical approaches have typically been used in tasks such as speech recognition, lexical acquisition, parsing, part-of-speech tagging, collocations, and statistical machine translation, statistical grammar learning, and so on.

C. CONNECTIONIST APPROACH

Similar to the statistical approaches, connectionist approaches also develop generalized models from examples of linguistic phenomena. What separates connectionism from other statistical methods is that connectionist models combine statistical learning with various theories of representation - thus the connectionist representations allow transformation, inference, and manipulation of logic formulae. In addition, in connectionist systems, linguistic models are harder to observe due to the fact that connectionist architectures are less constrained than statistical ones.

Generally speaking, a connectionist model is a network of interconnected simple processing units with knowledge stored in the weights of the connections between units . Local interactions among units can result in dynamic global behavior, which, in turn, leads to computation. Some connectionist models are called localist models, assuming that each unit represents a particular concept. For example, one unit might represent the concept "mammal" while another unit might represent the concept "whale". Relations between concepts are encoded by the weights of connections between those concepts. Knowledge in such models is spread across the network, and the connectivity between units reflects their structural relationship. Localist models are quite similar to semantic networks, but the links between units are not usually labelled as they are in semantic nets. They perform well at tasks such as word-sense disambiguation, language generation, and limited inference. Other connectionist models are called distributed models. Unlike that in localist models, a concept in distributed models is represented as a function of simultaneous activation of multiple units. An individual unit only participates in a concept representation. These models are well suited for natural language processing tasks such as syntactic parsing, limited domain translation tasks, and associative retrieval. Comparison among Approaches we have seen that similarities and differences exist between approaches in terms of their assumptions, philosophical foundations, and source of evidence. In addition to that, the similarities and differences can also be reflected in the processes each approach follows, as well as in system aspects, robustness, flexibility, and suitable tasks.

Process: Research using these different approaches follows a general set of steps, namely, data collection, data analysis/model building, rule/data construction and application of rules/data in system. The data collection stage is critical to all three approaches although statistical and connectionist approaches typically require much more data than symbolic approaches. In the data analysis/model building stage, symbolic approaches rely on human analysis of the data in order to form a theory while statistical approaches manually define a statistical model that is an approximate generalization of the collected data. Connectionist approaches build a connectionist model from the data. In the rule / data construction stage, manual efforts are typical for symbolic approaches and the theory formed in the previous step may evolve when new cases are encountered. In contrast, statistical and connectionist approaches use the statistical or connectionist model as guidance and build rules or data items automatically, usually in relatively large quantity. After building rules or data items, all approaches then automatically apply them to specific tasks in the system. For instance, connectionist approaches may apply the rules to train the weights of links between units.

System aspects: By system aspects, we mean source of data, theory or model formed from data analysis, rules, and basis for evaluation.

Data: As mentioned earlier, symbolic approaches use human introspective data which are usually not directly observable. Statistical and connectionist approaches are built on the basis of machine observable facets of data, usually from text corpora.

Theory or model based on data analysis: As the outcome of data analysis, a theory is formed for symbolic approaches whereas a parametric model is formed for statistical approaches and a connectionist model is formed for connectionist approaches.

Rules: For symbolic approaches, the rule construction stage usually results in rules with detailed criteria of rule application. For statistical approaches, the criteria of rule application are usually at the surface level or under-specified. For connectionist approaches, individual rules typically cannot be recognized.

Basis for Evaluation: Evaluation of symbolic systems is typically based on intuitive judgments of unaffiliated subjects and may use system-internal measures of growth such as the number of new rules. In contrast, the basis for evaluation of statistical and connectionist systems are usually in the form of scores computed from some evaluation function.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 10, October 2016

However, if all approaches are utilized for the same task, then the results of the task can be evaluated both quantitatively and qualitatively and compared.
Robustness: Symbolic systems may be fragile when presented with unusual or noisy input. To deal with anomalies, they can anticipate them by making the grammar more general to accommodate them. Compared to symbolic systems, statistical systems may be more robust in the face of unexpected input provided that training data is sufficient, which may be difficult to be assured of. Connectionist systems may also be robust and fault tolerant because knowledge in such systems is stored across the network. When presented with noisy input, they degrade gradually.
Flexibility: Since symbolic models are built by human analysis of well-formulated examples, symbolic systems may lack the flexibility to adapt dynamically to experience.
In contrast, statistical systems allow broad coverage, and may be better able to deal with unrestricted text for more effective handling of the task at hand. Connectionist systems exhibit flexibility by dynamically acquiring appropriate behaviour based on the given input. For example, the weights of a connectionist network can be adapted in real time to improve performance. However, such systems may have difficulty with the representation of structures needed to handle complex conceptual relationships, thus limiting their abilities to handle high-level NLP. Suitable tasks: Symbolic approaches seem to be suited for phenomena that exhibit identifiable linguistic behaviour. They can be used to model phenomena at all the various linguistic levels described in earlier sections. Statistical approaches have proven to be effective in modelling language phenomena based on frequent use of language as reflected in text corpora. Linguistic phenomena that are not well understood or do not exhibit clear regularity are candidates for statistical approaches. Similar to statistical approaches, connectionist approaches can also deal with linguistic phenomena that are not well understood. They are useful for low-level NLP tasks that are usually subtasks in a larger problem. To summarize, symbolic, statistical, and connectionist approaches have exhibited different characteristics, thus some problems may be better tackled with one approach while other problems by another. In some cases, for some specific tasks, one approach may prove adequate, while in other cases, the tasks can get so complex that it might not be possible to choose a single best approach. In addition, as Klavans and Resnik pointed out, there is no such thing as a "purely statistical" method. Every use of statistics is based upon a symbolic model and statistics alone is not adequate for NLP. Toward this end, statistical approaches are not at odds with symbolic approaches. In fact, they are rather complementary. As a result, researchers have begun developing hybrid techniques that utilize the strengths of each approach in an attempt to address NLP problems more effectively and in a more flexible manner.

III. NATURAL LANGUAGE PROCESSING APPLICATIONS

Natural language processing provides both theory and implementations for a range of applications. In fact, any application that utilizes text is a candidate for NLP. The most frequent applications utilizing NLP include the following:
• Information Retrieval – given the significant presence of text in this application, it is surprising that so few implementations utilize NLP. Recently, statistical approaches for accomplishing NLP have seen more utilization, but few systems other than those by Liddy and Strzalkowski have developed significant systems based on NLP
• Information Extraction (IE) – a more recent application area, IE focuses o the recognition, tagging, and extraction into a structured representation, certain key elements of information, e.g. persons, companies, locations, organizations, from large collections of text. These extractions can then be utilized for a range of applications including question-answering, visualization, and data mining.
• Question-Answering – in contrast to Information Retrieval, which provides a list of potentially relevant documents in response to a user's query, question-answering provides the user with either just the text of the answer itself or answer-providing passages.
• Summarization – the higher levels of NLP, particularly the discourse level, can empower an implementation that reduces a larger text into a shorter, yet richly constituted abbreviated narrative representation of the original document.
• Machine Translation – perhaps the oldest of all NLP applications, various levels of NLP have been utilized in MT systems, ranging from the 'word-based' approach to applications that include higher levels of analysis.
• Dialogue Systems – perhaps the omnipresent application of the future, in the systems envisioned by large providers of end-user applications. Dialogue systems, which usually focus on a narrowly defined application, (e.g. your refrigerator or home sound system), currently utilize the phonetic and lexical levels of language. It is believed that utilization of all the levels of language processing explained above offer the potential for truly habitable dialogue systems.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 10, October 2016

While NLP is a relatively recent area of research and application, as compared to other information technology approaches, there have been sufficient successes to date that suggest that NLP-based information access technologies will continue to be a major area of research and development in information systems now and far into the future. Acknowledgement Grateful appreciation to Xiaoyong Liu who contributed to this entry while she was a Ph.D. student and a Research Assistant in the Center for Natural Language Processing in the School of Information Studies at Syracuse University.

IV. STATISTICAL PARSING

Statistical parser, like statistical tagging (chapter 3), requires a corpus of hand-parsed text. There are such corpora available, the most notably being the Penn tree-bank (Marcus et al. 1993). The Penn tree-bank is a large corpus of articles from the ***Wall Street Journal*** that have been tagged with Penn tree-bank tags and then parsed according to a simple set of phrase structure rules conforming to Chomsky's government and binding syntax. The parsed sentences are represented in the form of properly bracketed trees. A statistical parser works by assigning probabilities to possible parse of a sentence and returning the most likely parse as a set of possible parse trees of s which we denote by $\tau(s)$, a probabilistic parser finds the most likely parse φ of s as follows :

$$\varphi = \text{argmax}_{\varphi \in \tau(s)} P(\varphi \mid s)$$
$$= \text{argmax}_{\varphi \in \tau(s)} P(\varphi, s)$$
$$= \text{argmax}_{\varphi \in \tau(s)} P(\varphi)$$

In order to construct a statistical parser, we have to first find all possible parses of a sentence, then assign probabilities to them, and finally return the most probable parse. We discuss an implementation of statistical Parsing based upon probabilistic context-free grammars (PCFGs). But Before proceeding ahead to this discussion, let us have a look at why we need probabilistic parsing at all. What advantages do these parsers offer?

The first benefit that a probabilistic parser offers is removal of ambiguity from parsing. We have seen earlier, that sentences can have multiple parse trees. The parsing algorithm discussed so far in this chapter has no means to decide which parse is the correct or most appropriate one. Probabilistic parsers assign probabilities to parses. These probabilities are then used to decide the most likely parse tree structure of an input sentence.

Another benefit this parser offers is related to efficiency. The search space of possible tree structures is usually very large. With no information on which sub-trees are more likely to be a part of the final parse tree, the search can be quite time consuming. Using probabilities to guide the process the search becomes more efficient.

A probabilistic context-free grammar (PCFG) is a CFG in which every rule is assigned a probability (Charniak 1993). It extends the CFG by augmenting each rule A $\rightarrow \alpha$ in set of productions *P*, with a conditional probability ρ:

$$A \rightarrow \alpha\ (\rho)$$

Where ρ gives the probability of expanding a constituent using the rule.

$$A \rightarrow \alpha$$

A. NATURAL LANGUAGE PROCESSING AND INFORMATION RETRIEVAL

Let us now define PCFG. A PCFG is defined by the pair (G, f), where
G is a CFG and f is a positive function defined over the set of rules such
That, the sum of the probabilities associated with the rules expanding a
particular non-terminal is I (Infante-Lopez and Maarten de Rijke 2006).

$\sum f(A \rightarrow a) = 1$

An example of PCFG is shown in Figure. We can verify that for each non-terminal, the sum of probabilities is l.

f(S → NP VP) + f(S → VP) = 1
f(NP → Det Noun) + f(NP → Noun)+ f(NP → Pronoun)
+ f(NP → Det Noun PP) = 1.0
f(VP → Verb NP) + f(NP → Verb)+ f(VP → VP PP) = 1.0
f(Det → this) + f(Det → that) + f(Det → a) + f(Det → the) = 1.0
f(Noun → paint) + f(Noun → door) + f(Noun → bird)
+ f(Noun → hole) = 1.0

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 10, October 2016

S → NPVP S → VP NP → Det Noun NP → Noun NP → Pronoun NP → Det Noun PP VP → Verb NP VP → Verb VP → VP PP PP → Preposition NP Det → this Det → that Det → a Det → the Noun → paint	0.8 Noun → door 0.2 Noun → bird 0.4 Noun → hole 0.2 Verb → sleeps 0.2 Verb → sings 0.2 Verb → open 0.5 Verb → saw 0.3 Verb → paint 0.2 Preposition → from 1.0 Preposition → with 0.2 Preposition → on 0.2 Preposition → to 0.25 Pronoun → she 0.35 Pronoun → he 0.25 Pronoun → they	0.2 5 0.2 5 0.2 5 0.2 0.2 0.2 0.2 0.2 0.3 0.2 5 0.2 0.2 5 0.3 5 0.3 5 0.2 5

A probabilistic contex-tree grammar (PCFG)
Similarly, the condition is satisfied for the other non-terminals.

V. ESTIMATING RULE PROBABILITIES

The next question is how are probabilities assigned to rules? one way to estimate probabilities for a PCFG is to manually construct a corpus of a parse tree for a set of sentences, and then estimate the probabilities of each rule being used by counting them over the corpus. The MLE estimate for a rule A → a is given by the expression

$$P_{MLE}\ (A \rightarrow a) = \frac{Count\ (A \rightarrow a)}{\sum a\ Count\ (A \rightarrow a)}$$

If our training corpus consists of two parse trees, we will get the estimates as shown in Table for the rules.

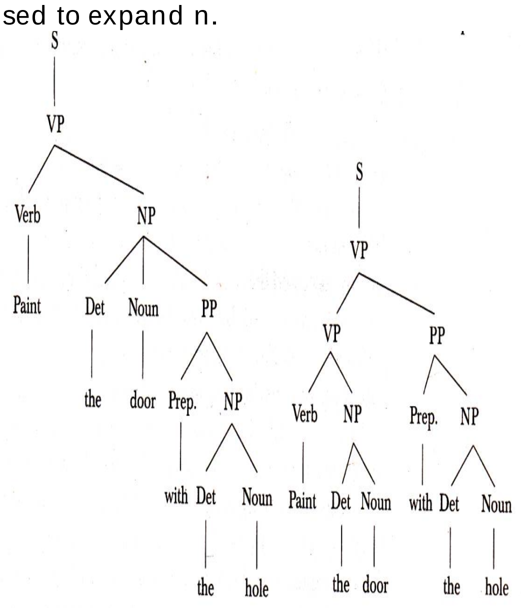
The MLE for the grammar rules used in trees of Figure

Rule	Count (A → a)	Count A	MLE estimates
S → VP	2	2	1
NP → Det Noun PP	1	4	0.25
NP → Det Noun	3	4	0.75
[illegible]	[illegible]	3	0.66
[illegible] PP	[illegible]	3	0.33
Det → the	2	2	1
Noun → hole	2	4	0.5
Noun → door	2	4	0.5
Prep → with	1	1	1
Verb → Paint	1	1	1

We now turn to another important question-what do we do with these probabilities? We assign a probability to each parse tree φ of a sentence *s*. The probability of a complete parse is calculated by multiplying the probabilities for each of the rules used in generating the parse tree:

$$P(\varphi, s) = \frac{\pi\, p\{r(n)\}}{n \in \emptyset}$$

where n is a node in the parse tree φ and r is the rule used to expand n.



Two possible parse trees

A. NATURAL LANGUAGE PROCESSING, AND INFORMATION RETRIEVAL

For a sentence, the parse trees generated by a PCFG are the same as those generated by a corresponding CFG. However, the PCFG assigns a probability to each parse. The probability of the two parse trees of the sentence *Paint the door with the hole* (shown in Figure) can be computed as follows:

P(t1) = 0.2 * 0.5 * 0.2 * 0.2 * 0.35
 * 0.25 * 1.0 * 0-25 * 0.4 * 0'35 * 0.25
 = 0.0000030625

P(t2) = 0.2 * 0.2* 0.5 * 0.2* 0.4* 0.35
 * 0.25 * 1 * 0.25 * 0.4 * 0.35 * 0.25
 = 0.000001225

In this example, the first tree has a higher probability leading to correct interpretation.

We can calculate probability to a sentence *s* by summing up probabilities of all possible parses associated with it

$$= \sum_{t \in \tau(s)} P(t)$$

Thus, the sentence will have the probability

$$P(t1) + P(t2)$$
$$= 0.0000030625 + 0.000001225$$
$$= 0.0000042875$$

B. PARSING PCFGS

Given a PCFG, a probabilistic parsing algorithm assigns the most likely parse φ to a sentence s.

$$\varphi = \text{argmax } T_{\in\tau(S)} P(T \mid S)$$

where τ(S) is the set of all possible parse trees of s.

We have already discussed a number of parsing algorithms for CFG. We will now discuss the probabilistic CYK parsing algorithm. We have already discussed the basic CYK parsing algorithm. Its probabilistic version was first introduced by Ney (1991) and is generally preferred over probabilistic Earley parsing algorithm due to its simplicity. We begin with notations that we use in the algorithm.

As in the basic CYK parsing algorithm, $w = w_l\ w_2\ w_3\ w_i \ldots w_j \ldots w_n$ represents a sentence consisting of n words. Let φ [I,j,A] represent the maximum probability parse for a constituent with non-terminal A spanning words i,i+1, up to i+j-1. This means it is a sub-tree rooted at A that derives sequence of j words beginning at position i and has a probability greater
than all other possible sub-trees deriving the same word sequence. An array named BP is used to store back pointers. These pointers allow us to recover the best parse. The output for a successful parse is the maximum probability parse φ[1,n,S], which corresponds to a parse tree rooted at S and spanning the sequence of words $w_1\ w_2\ w_3 \ldots w_{10}$., i.e. the whole sentence.

The algorithm is given in above Figure. Like the basic CYK algorithm, the first step is to generate items of length 1. However, in this case, we have to initialize the maximum probable parse trees deriving a string of length 1, with the probabilities of the terminal derivation rules used to derive them.

$$\varphi\,[i,1,A] = P(A \rightarrow w_i)$$

Similarly, recursive step involves breaking a string into all possible ways and identifying the maximum probable Parse.

$$\varphi\,[i,j,A] = \max \varphi\,[i,k,B] \times \varphi(k,j,C] \times P(A \rightarrow BC)$$

The rest of the step follow those of basic CYK parsing algorithm.

C. PROBLEMS WITH PCFG

The PCFG is not without disadvantages. Its first problem lies in the independence assumption. We calculate the probability of a parse tree assuming that the rules are independent of each other. However, this is not true. How a node expands depends on its location in the parse tree. For example, Francis et al. (1999) showed that pronouns occur more frequently as subjects rather than objects. These dependencies are not captured by a PCFG, as the probability of, say, expanding an NP as a pronoun versus a lexical NP, is independent of whether the NP appears as a subject or an

```
Initialization:
        for i := 1 to n do
                for all rules $A \rightarrow w_i$ do
                        $\varphi[i,1,A] = P(A \rightarrow w_i)$
Recursive Step:
        for j = 2 to n do
                for i = 1 to n-j +l do
                begin
                $\varphi[i,1,A] = \emptyset$
                for k=1 to j=1 do
$\varphi[i,j,A] = \max \varphi[i,k,B] \times \varphi[k,j,C] \times P(A \rightarrow BC)$,

such that,
$A \rightarrow BC$ is a production rule in grammar
                BP[I,j,A] = { k, A, B}
        end
```

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 10, October 2016

object.

Another problem associated with a PCFG is its lack of sensitivity to lexical information. Lexical information plays a major role in determining correct parse in case of PP attachment ambiguities and coordination ambiguities (Collins 1999). Two structurally different parses that use the same rules will have the same probability under a PCFG, making it difficult to identify the correct or most probable parse. The words appearing in a parse may make certain parses unnatural. This however, requires a model which captures lexical dependency statistics for different words. Such a model is presented next.

D. LEXICALIZATION

In PCFG, the chance of a non-terminal expanding using a particular rule is independent of the actual words involved. However, this independence assumption does not seem reasonable. Words do affect the choice of the rule. Investigations made on tree bank data suggest that the probabilities of various common sub-categorizalton frames differ depending on the verb that heads the verb phrase (Manning and Schutze 1999). This suggests the need for lexicalization, i.e., involvement of actual words in the sentences, to decide the structure of the parse tree. Lexicalization is also helpful in choosing phrasal attachment positions. This model of lexicalization is based on the idea that there are strong lexical dependencies between heads and their dependents, for example, between a head a head noun and its modifiers, or between a verb and a noun phrase object where the noun phrase object in turn can be approximated by its head noun.

One way to achieve lexicalization is to mark each phrasal node in a parse tree by its head word. Figure 4.16 is an example of such a tree. One way to implement this model is to use a lexicalized grammar. A lexicalized grammar is a grammar in which every finite structure is associated with one or more lexical heads. A context free grammar is not lexicalized as no lexical item is associated with its rule, such as, S → NP VP. Nor is the PCFG lexicalized. In order to convert a PCFG into lexicalized grammar, each of its rules must identify a head daughter, which is one of the constituents appearing on its right hand side, for example head daughter of S → NP VP rule is VP. The head word of a node in the parse tree is set to the head word of its head daughter. For example, the head of a verb phrase is the main verb. Hence, the head of the node VP in the Figure is jumped. Similarly, the head of a constituent expanded using the rule S → NP VP is the head of VP.

The probability in a lexicalized PCFG is calculated for each rule or head of a phrase and head of the sub-phrase or head of the phrase (i.e., head/mother head) combination. For –example, we need to collect the following rule/head Probability :

VP (jumped) → verb (jumped) PP (over)
VP (jumped) → verb (jumped) PP (into)
VP (jumped) → verb (jumped) PP (to)

An example of the head or mother head combination is the following type of probability:

“What is the probability that an NP whose mother’s head is over has the head fence?"

With a lexicalized PCFC, the probability of the parse tree is computed as the product of the probability of the head of the constituent c given the probability of the mother m(c) and the probability of the rule used to expand constituent c given the head of constituent (Charniak 1997):

$P(\varphi, s) = \Pi\varphi\ P[(h(c)|m(c)\}\ .\ p[(r(c)h(c)]$

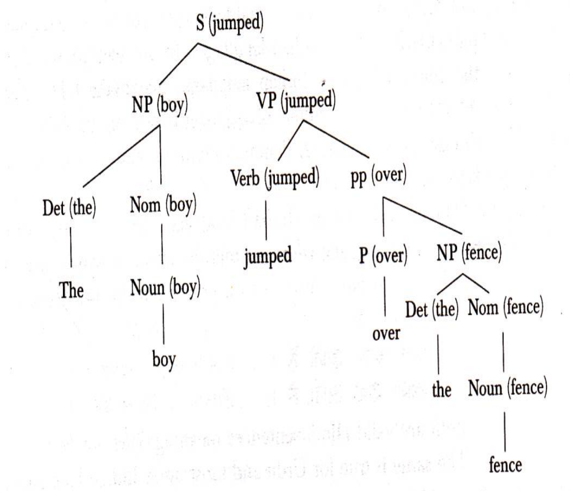
where h(c) = head of the constituent c
r(c) = rule used to expand constituent c
and m(c) = mother of the constituent c

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 10, October 2016



VI. CONCLUSION

We conclude that we can confirm our hypothesis that the accuracy differences between different machine learning algorithms using standard comparative methodology will in general be lower than the variability in accuracy resulting from interactions between algorithm parameter settings and information source selection.

REFERENCES

1. D.W. Aha, D. Kibler, and M. Albert. 1991. Instance-based learning algorithms. Machine Learning, 6:37–66.
2. Ethem Alpaydin. 1999. Combined 5_2cv F test for comparing supervised classification learning algorithms.Neural Computation, 11(8):1885–1892.
3. R.H. Baayen, R. Piepenbrock, and H. van Rijn. 1993. The CELEX lexical data base on CD-ROM. Philadelphia:Linguistic Data Consortium.
4. M. Banko and E. Brill. 2001. Scaling to very very large corpora for natural language disambiguation. In Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics, pages 26–33.Association for Computational Linguistics.
5. W.W. Cohen. 1995. Fast effective rule induction. In Proc. 12th International Conference on Machine Learning,pages 115–123.Morgan Kaufmann.
6. W. Daelemans, A. van den Bosch, and T. Weijters. 1997.Igtree: Using trees for compression and classification in lazy learning algorithms. Artificial Intelligence Review,special issue on Lazy Learning, 11:407–423.
7. Walter Daelemans, Antal van den Bosch, and Jakub Zavrel.1999. Forgetting exceptions is harmful in language learning. Machine Learning, 34:11–41.
8. W. Daelemans, J. Zavrel, K. van der Sloot, and A. van den Bosch. 2001. Timbl: Tilburg memory based learner,version 4.0, reference guide. Technical report, ILK Technical Report 01-04.
9. Thomas G. Dietterich. 1998. Approximate statistical tests for comparing supervised classification learning algorithms.Neural Computation, 10(7):1895–1923.
10. Ron Kohavi and George H. John. 1997. Wrappers for feature subset selection. Artificial Intelligence, 97(1–2):273–323.
11. Anne Kool, Walter Daelemans, and Jakub Zavrel. 2000a.Genetic algorithms for feature relevance assignment in memory-based language processing. In Claire Cardie, Walter Daelemans, Claire N´edellec, and Erik Tjong Kim Sang, editors, Proceedings of the Fourth Conference on Computational Natural Language Learning and of the Second Learning Language in Logic Workshop, Lisbon, 2000, pages 103–106. Association for Computational Linguistics, Somerset, New Jersey.
12. Anne Kool, Jakub Zavrel, and Walter Daelemans. 000b.Simultaneous feature selection and parameter optimization for memory-based natural language processing. In Ad Feelders, editor, Proceedings of the 10th BENELEARN meeting, pages 93–100. Tilburg, The Netherlands.
    Raymond J. Mooney. 1996. Comparative experiments on disambiguating word senses: An illustration of the role of bias in machine learning. In Eric Brill and Kenneth Church, editors, Proceedings of the Conference on Empirical Methods in Natural Language Processing, pages 82–91. Association for Computational Linguistics, Somerset,New Jersey.
13. Steven L. Salzberg. 1997. On comparing classifiers: Pitfalls to avoid and a recommended approach. Data Mining and Knowledge Discovery, 1(3):317–327.
14. SholomWeiss and Nitin Indurkhya. 1998. Predictive Data Mining: A Practical Guide. Morgan Kaufmann, San Francisco.
15. SholomM.Weiss and Casimir A. Kulikowski. 1991. Computer Systems That Learn: Classification and Prediction Methods from Statistics, Neural Nets, Machine Learning,and Expert Systems. organKaufmann, SanMateo,California.
16. Dietrich Wettschereck, David W. Aha, and Takao Mohri.1997. A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms.Artificial Intelligence Review, 11(1-5):273–314.
17. David H.Wolpert andWilliam G.Macready. 1995. No free lunch theorems for search. Technical Report SFI-TR-95-02-010, Santa Fe Institute, Santa Fe, NM.