Noun Compound Interpretation

Using a Data Driven Approach

By

Chinmay Vadgama (14CEUOS074)

A project submitted
in
partial fulfillment of the requirements
for the degree of

BACHELOR OF TECHNOLOGY
in
Computer Engineering

Internal Guide
Dr. C. K. Bhensdadia
Head of Department
Dept. of Comp. Engg.

External Guide
Prof. Pushpak Bhattacharyya
FNAE, Director, IIT Patna.
Professor, Dept. of CSE, IIT Bombay

Faculty of Technology
Department of Computer Engineering
Dharmsinh Desai University

April 2018

CERTIFICATE

This is to certify that the project work titled

Noun Compound Interpretation:
Using a Data Driven Approach

is the bonafide work of

Chinmay Vadgama (14CEUOS074)

carried out in the partial fulfillment of the degree of
Bachelor of Technology in **Computer Engineering**
at **Dharmsinh Desai University** in the academic session
December 2017 to April 2018

Dr. C. K. Bhensdadia
Guide & Head
Dept. of Computer Engg.

Faculty of Technology
Department of Computer Engineering
Dharmsinh Desai University
April 2018

Acknowledgements

I would like to express my sincere gratitude to **Prof. Pushpak Bhattacharyya** for giving me a valuable opportunity. I sincerely enjoyed his session on different concepts of NLP. I find myself lucky to have a chance to get the benefit of his immense knowledge, support and guidance.

I would also like to express my deepest gratitude and special thanks to **Dr. C. K. Bhensdadia** - my internal guide and Head, Computer Engineering Department, DDU, for allowing me this valuable internship at prestigious institute. I thank him for his moral support and for giving me personal attention in difficult situations.

Furthermore, I would like to express special thanks to **Mr. Girishkumar Ponkiya**, Ph.D Scholar for his crucial role during my internship. I thank him for his continuous support, motivation and sharing his knowledge. I benefited a lot from fruitful discussions with him throughout my internship period. I would like to thank **Mr. Kevin Patel**, Ph.D. Scholar for his valuable suggestions, insightful comments and technical assistance during project implementation.

I would also like to show my gratitude to all the members of **CFILT** (Center for Indian Language Technology), IIT Bombay, for sharing their pearls of wisdom and being an integral part of my internship here. I would also like to thank my friends - other interns at IIT Bombay - for their constant help and assistance.

I am also thankful to all my family members for their unconditional support and motivation throughout my internship.

Abstract

Noun compounds are sequences of more than one nouns acting as a single noun. Noun compounds are very productive as most of them appear only once in a large corpus. The interpretation of noun compound is a challenging task in the field of natural language processing. Noun compound interpretation refers to the process of extracting the missing information about the relation between the individual nouns of the noun compound. Here, the problem is to identify noun compounds from a text, parse it if required, and extract the semantic relation between the components of a noun compound. This task of extracting the dropped information, or paraphrasing the noun compound using verb and prepositions, is known as noun compound interpretation.

In this report, I have tried to look at the problem of noun compound interpretation. From literature, it is evident that automatic interpretation requires more than just semantics of individual components. So, I have improved an existing system based on a data-driven approach to understand how components of a noun compound can be related in real sentences. Apart from that, researchers have reported that there are about millions of noun compounds in large corpora. But, for our experimentation, I have a relatively small annotated dataset. So, I propose a semi-supervised approach to utilize such unlabeled data, along with the labeled data.

I have followed optimal ways to gather required data for the system. I needed to extract millions of sentences for a noun compound for raw data. Then, meaningful parsing data was extracted from state-of-the-art tools using API and function calls. I have carefully crafted different features to represent a noun compound. Each feature-set was represented by vector representation. For learning some of the features' representation, deep learning based approach was used. To cluster all noun-compounds available I have used different classifiers. I feed the clustered data to different classification algorithms to classify each noun compound for its semantic relation.

In later part of the report, I discuss an analysis of the current system and conclude with takeaways from this system implementation.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

1.1 Natural Language Processing

Natural Language Processing (NLP) is the field of computer science and linguistics concerned with the interaction between computers and human (natural) languages. Understanding natural language requires extensive knowledge about outside world and ability to manipulate it. Thus it is considered as a subset of artificial intelligence. NLP has significant overlap with computational linguistics. NLP researchers aim to gather knowledge on how human being understand and manipulate natural languages to perform the desired task.

1.1.1 Goals of NLP

There are two major goals of NLP.

Scientific Goal: In this goal, the focus is on the understanding the way languages operate. Consider the following example:

Monica went to the *bank* to withdraw money. The word *bank* has two senses: 'river bank' and 'financial bank'. Which of these two senses was meant in the example? To understand that, we look at the context(neighbours) of the word *bank*.

So, here, our major goal is to figure out how humans understand language. By looking at the context, they decide the meaning. But for machines, it is somewhat more difficult. For instance, in the above example, it is difficult for a machine to decide which among the context words {*Monica, went, withdraw, money*} should be used to figure out the meaning.

Engineering Goal: In this goal, the focus is on building NLP systems: Question Answering systems, Dialogue systems, *etc.* The engineering goal does not care whether the systems work with or without understanding how humans understand language.

The most striking feature that distinguishes a natural language from an artificial language (such as programming languages, *etc.*) is the presence of ambiguity in different constructs. For example, words ending with 's' are not always plural (*virus, news, species, etc.*). Proper resolution of such ambiguities is a crucial requirement to getting machines understand language. To tackle this, researchers study NLP at different stages which we discuss in the next subsection.

1.1.2 Stages of NLP

As discussed earlier researchers study NLP at different stages of complexity, each with its own set of ambiguities. These stages are subfields of NLP, and are as follows:

Phonetics and Phonology

This stage deals with the processing of speech. There are several types of ambiguities possible. Some are discussed below:

- Homophones: This ambiguity is related to words which sound similar but, having a different meaning.

  Example: In Gujarati, *LOCHO* has two meanings: (1) A famous dish from Surat, (2) Problem.

- Near Homophones: It is related with the intensity of particular sound is different in a word having a similar spelling.

  Example: In Gujarati, *kalam* (Pen) vs *Kalam* (Dr. APJ abdul kalam).

- Word Boundary: It is concerned with how to split rapid speech into words having meaningful context.

  Example: In Gujarati, *Chaapani* can be split as *chaa pani* (refers to Tea) or *cha aapni* (refers to Give me tea).

- Phrase Boundary: It is concerning with splitting phrase in a meaningful context while delivered rapidly.

  Example: In Gujarati, *"Deevanthidarbarma, chheandharughor"* can be splitted as *"Deeva nathi darbarma, chhe andharu ghor"* (Due to absence of a lamp in room, there is darkness) and *"Deevanthi darbarma, chhe andharu ghor"* (Due to presence of deewan, there is dark mood in room).

- Disfluency: These are the words which have no meaning at all, but at the time of speaking, human use to organize the thoughts.

  Example: Words like *oh*, *aah*, *uh*, *etc* are used during speach but those words have no meaning at all.

Morphology

Morphology deals with word formation rules from root words. Morphology is identification, analysis, and description of the structure of a word.

Example:

- Noun: *Cars* comes from the root word *Car*. (Structure: Plural, Gender making)
- Verb: *Created* is a past tense of *To create*.

Some facts related to morphology:

- Languages that are reach in morphology: Dravidian language(Tamil, Telugu, Malayalam)
- Languages that are poor in morphology: English, Chinese
- Languages having rich morphology have advantage of easier processing at higher stages of processing
- By using Finite state machine one can produce morphology analysis.

Lexical Analysis

The 'Lexicon' of a language is its vocabulary, that includes its words and expressions. It is about dictionary making of a word and accessing properties of a word.

Example: Dictionary entry of *Dog*,

- noun (lexical property)
- take 's' as plural (morph property)
- animal, four-legged, carnivore (semantic property)

This type of dictionary information can be used for Question Answering, Information Retrieval, *etc*.

Challenges in Lexical analysis:

- Part of Speech Disambiguation:
  - Dog as noun: in the sense of animal

  - Dog as verb: in the sense of 'To run after'

- Sense Disambiguation:
  - Dog (as animal)
  - Dog (as very detestable person)

- Needs word relationship in context:
  Example: *Ground breaking ceremony*: here *break* is not about fracturing something

So, the conclusion is lexical analysis can not be performed in isolation of morphological and syntactic analysis. Higher level of knowledge is needed for disambiguation.

Syntactic Analysis

Syntactic analysis is useful for checking the grammatical structure of the sentence. In this stage, words are transformed into structures that show how the words are related to each other. Some word sequence may be rejected if they violate the language grammatical rules. Following are the ambiguities occur in syntactic analysis:

- Scope ambiguity:

  No smoking areas will allow hookas inside.

  Depending on the scope of *No*, the sentence can have two meanings:
  - *No smoking areas* will allow hookas inside.
  - Not even a single smoking area will allow hookas inside.

- Preposition phrase attachment:

  I saw the boy with a telescope.

  This sentence has two meanings based on the correspondence of a proposition. If *with* corresponds to *I*, then it's meaning will be *I saw a boy with the help of a telescope.* But if *with* corresponds to *boy* then its meaning will be *I saw a boy who had a telescope.*

Semantic Analysis

It derives an absolute (dictionary definition) possible meaning from context. The sentence is represented in one of the unambiguous forms like predicate calculus, semantic net, frame, conceptual dependency, conceptual structure, etc.

Examples:

- *People laugh heartily*. (subject verb adverb).
  Here, laugh and heartily makes sense of verb adverb pair. So this is semantically correct.
- *Sleep furiously* (verb adverb).
  Here, sleep is peaceful activity and furious is intense vigorous state and this two words are mutually incompatible. So this is semantically incorrect.

The ambiguity at this stage arises from a confusion of semantic roles and representation. For example, in the sentence *Visiting Aunts can be boring* so, here aunts can have two roles. For example aunts as a visitors (agent role) or aunts who are being visited (object role)?

Pragmatics

It is a very challenging task of NLP. It derives knowledge from external commonsense information. It means understanding a purposeful use of language in situations, particularly those aspects of language which require world knowledge, user intention, sentiment, belief world, etc. All of these are highly complex tasks.

The following example illustrates how complex this task may be:

> *Tourist (checking out of the hotel): "Waiter, go upstairs to my room and see if my sandals are there; do not be late; I have to catch the train in 15 minutes."*
>
> *Waiter (running upstairs and coming back, panting): "Yes! Sir, they are there."*

It is clear that the waiter didn't understand the pragmatics(actual sense) of the situation and fell short of the expectation of the tourist. Larger context, history, intent, sentiment, tone, *etc*. come into play by making the task enormously difficult.

Discourse

The meaning of an individual sentence may depend on the sentences that precede it and may influence the meaning of the sentences that follow it.

Example: *Santa came home. He distributed gifts to children.*

Here, *He* is referred to *Santa*. How to infer this? This can be done by discourse integration.

1.1.3 Applications of NLP

NLP has found its way to many applications, some of which are as follows:

Sentiment Analysis

Sentiment Analysis deals with analyzing the sentiment expressed by different people in their opinion. For instance, a review written by a customer on an online shopping site. There are different problems in sentiment analysis, ranging from simple positive/negative sentiment classification to assigning a sentiment score, or finding the sentiment of entire review vs. finding the sentiment for individual entities in a review (*e.g.* camera quality: good, battery life: bad).

Information Retrieval

Information Retrieval deals with fetching information requested by a user from the Internet. The vast amount of data present on the web makes it a daunting task to identify relevant pages, and return them in a ranked manner such that they are useful to the user. Imagine the scenario where the page you were looking for ends up being the 100th search result instead of being among the top 10 in Google. Would you still use it?

There are many NLP problems that may arise in Information Retrieval too. For instance, if a user searches for *apple* - what is (s)he looking for? The fruit or the billion-dollar company?

Information Extraction

Information Extraction deals with finding structured information from unstructured text. Many NLP challenges manifest themselves in different forms. For instance, Noun Compound Interpretation deals with uncovering the hidden verb structure between two nouns (orange juice - juice MADE OF orange). Co-reference resolution deals with finding which noun does a pronoun refer to (Ram gave a book to Shyam. He was happy. - Who is he?) Such tasks arise in a different domain-specific task. For instance, medical information extraction deals with identifying medical diagnosis, treatments, etc. from unstructured texts created naturally by medical staff at various medical facilities.

Question Answering

Question Answering systems are one of the most complicated applications of NLP. The task is to automatically generate a valid answer for a given question. It is a complex task and needs help from almost all stages of NLP. Recently, QA capabilities of

machines were debated in the news when IBM Watson won the Jeopardy - a reality question answering competition.

Machine Translation

Machine Translation is considered as the ultimate goal of NLP. In fact, it is also one of the major reasons for the birth of computation - the British wanted to translate German military messages during the World War II, which led to the development of the first computing machine - ENIGMA, and the rest is history. We have come a long way from those days. Currently, Google Translate is one of the most advanced machine translation system in the world. Yet, it miserably fails while trying to translate अपनी आज़ादी को हम हर गीज मिटा सकते नहीं (returns translation as *We cannot erase every geys to our independence*)

1.2 Introduction to Noun Compound

Typically, a *noun compound* is a continuous sequence of two or more nouns acting as a single noun. Individual nouns in a noun compound are called the components of noun compound. The semantics of such expressions are not confined to simply combining semantics of individual components, but it requires explicit information in addition to the semantics of the individual words in the expression. For example, *"apple juice"* means *"juice made from apple"* or *"juice extracted from apples"*. Information related to how *apple* and *juice* are related is missing in *"apple juice"*, and we used words like *"made from"* and *"extracted from"* to convey this relation. Such information is intuitive for human, but very difficult for a machine to understand.

Noun Compound Interpretation means extracting (or unrevealing) the hidden relationship between the component nouns. This can be done in different ways like paraphrasing the noun compound, assigning labels to a given noun compound and many more. We mainly consider the technique of assigning labels from the available repositories of semantic labels.

In the above example, the meaning of a compound can be conveyed by combining the meaning of its components. But this is not true for all the noun compounds. For example, a noun compound like *lady finger* has nothing to do with a *lady* or *finger*. If the meaning of a compound can be derived by combining the meaning of its components, then such compounds are called *compositional compounds; non-compositional*, otherwise. In this work, we are interested in an interpretation of only compositional noun compounds.

1.2.1 Aim

Noun Compounds are highly productive and its usage is increasing day to day language. Being productive in nature, most of the noun compounds appear only once in a large corpus. Moreover, the information about the relation between the components of noun compound is dropped out. Such relations must be intuitive for human, but not for a machine. Our goal is to build a system that can extract such hidden relation for a given noun+noun compound.

1.2.2 Motivation

The information related to the relation between the components of a noun compound is crucial for most NLP tasks. Any sophisticated NLP task needs to extract this information. Not extracting this information would affect the performance of the system to a large extent. Once the information is available, it could be further processed for other NLP tasks like machine translation, information retrieval and many more. Next, we will have a look at various tasks of NLP where the noun compound interpretation helps.

1.2.3 Noun Compounds in various NLP tasks

In this section, we will discuss the importance of noun compound interpretation for various NLP tasks.

Machine Translation

Machine Translation is the task of automatic translating a sentence from one language to another language using a computer. The quality of machine translation system is measured based on adequacy (measuring the meaning transfer) and fluency (quality of generating a sentence).

ENG: Honey Singh became the latest victim of a celebrity death hoax.
हिंन्दी: *हनी सिंह पृसिद्ध व्यक्ति मौत अफवाह के ताजा सिकार बने।
हिंन्दी: हनी सिंह पृसिद्ध व्यक्ति की मौत के बारे में अफवाह के ताजा सिकार बने।

Noun compound in one language need not be a noun compound in another language. For example, as shown above, the "*celebrity death hoax*" cannot be translated literally in Hindi. We need to add few words (like "के बारे में") to make the sentence fluent.

Textual Entailment

Textual entailment in NLP is a directional relation between text fragments. The relation holds whenever the truth of one text fragment follows from another text.

For example (Nakov, 2013):
Text: *"Geneva headquarters of the WTO."*
Hypothesis: *"WTO headquarters are located in Geneva."*

To check whether *"Geneva headquarters of the WTO"* could entail *"WTO headquarters are located in Geneva"* can be performed by understanding the semantics of the noun compound *"WTO Geneva headquarters"*.

Information Retrieval

Information Retrieval is the task of obtaining documents from large collection relevant to the required information. Generally, humans use very few words in a query and they tend to create compound words. If the system cannot understand the correct or intended meaning of compound in the query, then it might fail to retrieve relevant documents.

"Easton's research does conclusively show that there are economic benefits in the legalization of marijuana."
Query: drug legalization benefits the economy: True/False?

In the above example, the system must be able to understand that legalization of a drug is one of the possible paraphrases of drug legalization, and marijuana is a type of drug. Without such information, the system might treat such documents, like shown in the example, as irrelevant.

Text Summarization

Automatic text summarization is the process of reducing a text document with a computer program in order to create a summary that retains the most important points of the original document. To reduce the size of the document, one can identify the phrases which can be written as compounds.

E.g., *"Joey hides the keys in his dog house."*

Here, *dog* and *house* forms a kind of summary of the information: "*a place or house where the dog lives*".

1.3 Outline of the Report

In this chapter, we discussed the basics of NLP and importance of extracting semantic relations of noun compound. In chapter 2 we discuss the semantics and theories of noun compound. chapter 3 discusses the technology and tools used for the project. In chapter 4, the 3 step process for noun compound interpretation and different approaches for the automatic interpretation of noun compound.

We propose the idea of semi-supervised approach for noun compound interpretation in chapter 5. We also discuss the results of our experiments in this chapter. In chapter 6, we summarize with conclusion and future work.

Chapter 2

Semantics of Noun Compounds

Natural language creates new words by means of compounding. Any sophisticated NLP system supposed to understand the meaning of such compounds. Noun Compounds cannot be ignored by Natural Language Processing (NLP) applications as they are abundant in writing text. Understanding syntax and semantics of a noun compound are potentially important for almost any NLP task. Here, we are interested in building a system which can understand the meaning of noun compounds automatically.

In this chapter, we will talk about the semantics of noun compound. We begin by talking about different criteria, for deciding if a compound is a noun compound. We then examine different definitions for noun compound. We will also talk about different ways of representing the semantic relations between the component nouns. (Nakov, 2013) summarized various criteria for compounds. In the next section, we give an overview of the compounds, and criteria for qualification as a compound.

2.1 Compounds

An important characteristic of the English language is a process of creating new words by means of compounding. The Dictionary of Grammatical Terms in Linguistics defines the process of compounding as follows (Trask, 1993):

> "The process of forming a word by combining two or more existing words: *newspaper*, *chocolate milk*, *babysit*, *video game*."

Since the process of compounding constructs new words, these words can in turn combine with other words to form longer compounds, and this process can be repeated indefinitely, e.g., *fault, fault analysis, fault analysis system,* etc. But the compounds of length more than two are less frequent in general. Following are some criteria which help us decide if a sequence of words can be categorized as a compound or not.

2.1.1 Orthographic Criteria

Many compounds are at least partially lexicalized and are written as a single word or hyphenated. Concatenated orthography is a reliable indicator of compounds, but hyphenated spelling is less so. Say for an example *silkworm* and *snowball* are the compounds, but *US-China* is partially lexicalized compound. Again, according to this criteria, *"US-China relations"* is not a compound.

Overall, orthography is not a good criterion for English as concatenation is rare, except small lexicalized words like Sunday, textbook. Also, variations in spelling found in same noun compound make it less effective. Like, it would be inconsistent to consider *healthcare* and *health-care* as noun compounds but not *health care*. These criteria do not

consider the compounds with word separated by a space. So, *"put on"*, and *"US-China relations"* are not compounds as per this criteria, despite they are.

2.1.2 Phonological Criteria

Chomsky and Halle (1968) gave a phonological definition for noun compounds in English: "the words preceding a noun will form a compound with it if they receive the primary stress." So, *blackboard* is a compound (black receives primary stress), but *black board* is not (equal stress).

This criterion is problematic for following reasons:

- Stress could differ across dialects and even across speakers of same dialects.
- Stress criteria fail while dealing with written text as no information regarding stress is available.

2.1.3 Morphological Criteria

A compound should inflect as a whole only, while compound-internal inflections should not be allowed. For example, *apple cake* can generate inflected form like *apple cakes*, but not *apples cakes* or *apples cake* cannot.

There are few languages like Russian for which the inflection is unstable. On the other hand, it works very well for languages like Turkish.

2.1.4 Syntactic Criteria

This criteria check whether a compound is treated as a single unit in syntax, i.e., whether syntax can "see" the individual words that form it.

For an example, *"It is a green ribbon, not a red one"*, where one refers to the *ribbon*, not to the *green ribbon*, which shows that *green ribbon* is transparent to syntax, and thus it should not be considered a compound, according to this criterion. This is not a bullet-proof criterion as sometimes some "impossible" sentences are generated.

2.1.5 Semantic Criteria

This criterion mainly focuses on the meaning of the noun compound as a whole. The words forming compounds must be in a permanent or habitual relationship (permeanace). The compounds must be at least partially compositional. Compositional is in turn a matter of degree. Some noun compounds are completely non-compositional.

The example is *honeymoon* has nothing to do with *honey* or the *moon*. Therefore, there is an independence between the components of the nouns (*"honey"* and *"moon"*) and noun compound (*"honeymoon"*). On the other hand, metaphorical *birdbrain* is highly compositional where it means a *"stupid person"*. Often it refers to *"a person with a short attention span"*, i.e., some or the other way, implicitly it implies *"one having a small brain which resembles to the size that of bird."*

In this section, we discuss the criteria for compound qualification. There are no full-proof criteria. But, these criteria might help in detecting/finding compounds from text. From the next section, we shall discuss noun compounds, a special category of compound.

2.2 Noun Compound

Typically, a sequence of two or more nouns acting as a single noun is called noun compound. But, there are no definite rules or algorithmic steps to identify a noun compound. Not every sequence of nouns is a noun compound. Many definitions of noun compounds (or similar terms) have been proposed by various scholars. In this section, we shall briefly explain various similar terms, and how they are different from the noun + noun construction.

2.2.1 Similar Terms

Different scholars have used different terms to refer to the structures we are discussing about. The main reason for using different terms seems the restrictions on what should be considered noun compounds. Following are the some popular terms:

Noun Premodifiers : (Quirk and Widdowson, 1985) had adopted a very open definition. Their grammar permits virtually any constituent to appear before a noun to form a noun premodifier. Their definition thus includes out-in-the-wilds cottage and similar constructions. The difficulty with this definition lies in distinguishing compounding from adjectival modification.

Compounds : Chomsky and Halle (1991) had taken a phonological definition in which words preceding noun form a compound if they receive the primary stress. Therefore, blackboard is a compound, black board is not. The problem with this track is that pronunciations vary amongst speakers, that what is a compound for one person may not be for another.

Complex Nominals : According to (Levi, 1978), she chooses to include certain adjec-

tives along with nouns as possible compounding elements and so calls her noun compounds as complex nominals. The adjectives included are non-predicating adjectives (ones that supposedly cannot be ascribed via a copula construction).

An *"electrical engineer"* is an example of non-predicative adjective modifier, as electrical cannot be a predicate for engineer, i.e., *"that engineer is (a/an) electrical"* is ungrammatical sentence. Since adjectives are difficult to divide into predicating and non-predicating, this ambiguity causes computational difficulties during identification.

Under complex nominals, this theory clearly separates the compounds with a normalized form of a predicate as head with the rest.

Noun+Noun Compounds: (Downing, 1977) defines noun compounds as any sequence of nouns that itself functions as a noun. While this is more restrictive than any of the previous three, it is relatively unambiguous. Leonard (1984) also takes this path, using the term noun sequences.

Though most scholars consider (Downing, 1977) definition during noun compound identification, most annotated dataset has also considered the non-predicative adjective.

After having a look at the criteria for Noun Compounds, let us understand types of noun compounds by (Nakov, 2013). It gives us the brief idea of how these noun compounds are further categorized on what different ways.

2.3 Types of Noun Compounds

The two most important types of noun compounds are endocentric and exocentric. The Lexicon of Linguistics defines them as follows:

Endocentric compound : A type of compound in which one member functions as the *head* and the other as its *modifier*, attributing a property to the head. The relation between the members of an endocentric compound can be schematized as "AB is (a) B". Like the compound *"armchair"*. Here, both the nouns retain their semantic contexts. Also, the syntactic feature of the chair is preserved.

Exocentric compound : A term used to refer to a particular type of compound, viz., compounds that lack a head. Often these compounds refer to pejorative properties of human beings. A compound *"wise nose"* (in normal usage) does not refer to a *nose* that is *wise*. In fact, it does not even refer to a *nose*, but to a human being with a particular

property. A similar term used for such compounds in Sanskrit is *bahuvrihi*.

There is a third category of compounds, known as **copulative or coordinative** (*dvandva* in Sanskrit). They form an entity that is the sum of the two nouns that form the compound and is also distinct from either of them.

Copulative compounds are often confused with a fourth category of compounds known as **appositional**, where each of the nouns denotes independently a different aspect of the entity that the compound represents. E.g., *coach-player*, *coach-player* is somebody who is both a *coach* and a *player*. Other examples are *learner-driver*, *sofa-bed*.

Compounds can be also formed by reduplication. There are various kinds of reduplication in English such as exact (e.g., *bye-bye*), ablaut (e.g., *chit-chat*) rhyming (e.g., *walkie-talkie*, *hokey-pokey*), contrastive (e.g., *"I'll make the tuna salad, and you make the salad-salad."*), etc. Despite this variety, reduplication is not very common nor is it very productive in English, except probably for the last two categories. However, it is very common in some other languages.

So, in a nutshell, we will mainly deal with binary noun compounds, i.e., a noun compound with two components. Here, each noun compound would have a *"head"* and a *"modifier"*. A *"head"* of noun compound is a component which has the primacy in the meaning, and *"modifier"* is providing additional information or is supporting the head in some or the other manner. Say for an example, *"student protest"*, where *"protest"* is a head (as *"student protest is a protest"*), and *"student"* is a modifier as shown in Figure 2 . These will eventually help us to fetch out the relation between those two nouns, and later on we can use this relation to predict the unknown forthcoming noun compound.

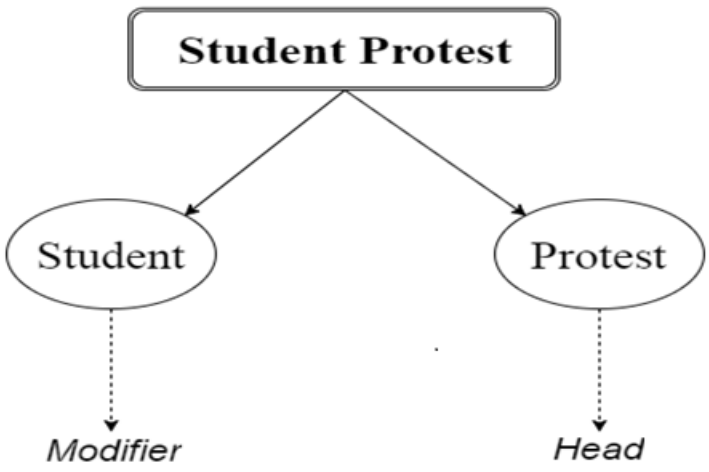

Fig. 2.1 "student protest" as a Noun Compound

Therefore, even after learning about noun compounds and its types, noun compound Interpretation is potentially difficult because of the following reasons:

- Noun compounds are very productive. This high productivity means that these compounds cannot be listed in a dictionary. A study on introducing new words in English, over fifty years, shows that compounding is the most frequent word formation process, covering 68% of the new words; 90% of these new compounds are noun compounds (Algeo (1991). (Baldwin, Bannard, Tanaka, & Widdows, 2003) have shown that static English dictionary has only 27% coverage of these compounds.
- Noun compounds are abundant in English language. For supervised approach of noun compound interpretation, annotated dataset is required.
- Noun Compound interpretation needs to consider pragmatic information. These are the most difficult tasks of NLP as it needs a large knowledge base and proper representation of the available knowledge. This knowledge representation would help us infer the relation between the components of noun compound.

The central idea of our work is to extract the implicit information from a noun compound using pragmatic information, i.e., how components of a compound can be "linked" in sentences.

A typical way of processing noun compound can be seen as a pipeline of three steps:

- Firstly, we need to identify the noun compounds from a given text. The previous part of the chapter describes this.
- After noun compound identification, we parse the noun compound, if necessary.
- A crucial part of the task is finding the relation between the component nouns. This can be done, by paraphrasing the noun compound, or assigning a semantic relation to the noun compound.

In this work, we are interested in the development of a system which can extract semantic relation between the components of a compound. For relation representations, we used a set of predefined abstract relations. In the following section, we discuss various inventories of semantic relation for noun compound.

2.4 Semantic Representations

The motivation behind solving this problem of interpreting the noun compounds come from the problem faced by other NLP applications. So, after processing a noun

compound, we need to present a semantic information in such a way that is can be used by other NLP applications. In this section, we discuss about various semantic representations, proposed in literature. We can categorize all theories into two types: set of abstract semantic relations, and paraphrasing the noun compound.

2.4.1 Levi's Theory

The (Levi, 1978) study is built on base of language study. The primary objective behind this study was to understand how human create nominal compounds. When the relation between two entities becomes intuitive for human, they simply drop the semantic relation related detail by deleting the predicates and prepositions. Levi argued that such relations can be recovered in the form of the abstract predicates. She proposed set of abstract relation for this type of noun compounds, and she called such predicates as Recoverably Deletable Predicates (RDPs).

E.g., *"pie made of apples"* becomes *"apple pie."*

2.4.2 Warren's Theory

(Warren, 1978) also proposed a set of relations for noun compounds. Initially, she considered only noun+noun compounds. But, later the same was revised and it includes a special type of adjective. This theory also considers the orthographically connected words, like *gunman*. In contrast to Levi, Warren's theory was based on corpus study. Warren studied characteristics of noun compounds in BNC corpus and, based on that study, she proposed hierarchical relations of four levels. She called top-level relations as *major semantic classes*, second level relations as *semantic classes*, third level relations as *main groups*, and fourth level relations as *subgroups.*

2.4.3 Barker and Szpakowicz

(Barker and Szpakowicz, 1998) had created an inventory of 20 relations. For each of the relations, definition was given by a paraphrase. In the same paper, they said that the inventory might evolve as experimental evidence suggests changes. But, later no one has tried to change it.

Later, (Kim and Baldwin, 2005) argued that this inventory can help the NLP tasks, and they prepared an annotated dataset of this.

2.4.4 Tratz and Hovy

All of the previously proposed inventories are either deep semantics or they have a very poor inter-annotator agreement. In addition, most theories don't cover most of the noun compounds. To solve this problem, Tratz and Hovy (2010) proposed new inventory. They compared and mapped relations from existing theories to the relations in their inventory of 43 relations. They also annotated 17509[1] noun compounds extracted from the WSJ section of the Penn Treebank. As per our knowledge, this is the biggest annotated dataset available as of now. The second largest dataset has 2169 samples, and it was prepared by (Kim and Baldwin, 2005) using the inventory of (Barker and Szpakowicz, 1998).

Category Name	% (Frequency)	Example
Casual Group		
COMMUNICATOR OF COMMUNICATION	0.77	court order
PERFORMER OF ACT/ACTIVITY	2.07	police abuse
CREATOR/PROVIDER/CAUSE OF	2.55	ad revenue
Purpose/Activity Group		
PERFORMENGAGE_IN	13.24	cooking pot
CREATEPROVIDE/SELL	8.94	nicotine patch
OBTAINACCESS/SEEK	1.5	shrimp boat
MODIFYPROCESS/CHANGE	1.5	eye surgery
MITIGATEOPPOSE/DESTROY	2.34	flak jacket
ORGANIZESUPERVISE/AUTHORITY	4.82	ethics board
PROPEL	0.16	water gun
PROTECTCONSERVE	0.25	screen saver
TRANSPORTTRANSFER/TRADE	1.92	freight train
TRAVERSEVISIT	0.11	tree traversal
Ownership, Experience, Employment, and Use		
POSSESSOR + OWNEDPOSSESSED	2.11	family estate
EXPERIENCER + COGINITIONMENTAL	0.45	voter concern
EMPLOYER + EMPLOYEEVOLUNTEER	2.72	team doctor
CONSUMER + CONSUMED	0.09	cat food
USERRECIPIENT + USEDRECEIVED	1.02	voter guide
OWNEDPOSSESSED + POSSESSION	1.2	store owner
EXPERIENCE + EXPERIENCER	0.27	fire victim
THING CONSUMED + CONSUMER	0.41	fruit fly
THINGMEANS USED + USER	1.96	faith healer
Temporal Group		
TIME [SPAN] + X	2.35	night work
X + TIME [SPAN]	0.5	birth date

[1]This dataset was revised later and it contains 19K+ annotated examples of Noun Compounds.

Location and Whole+Part/Member of		
LOCATIONGEOGRAPHIC SCOPE OF X	4.99	hillside
WHOLE + PART/MEMBER OF	1.75	robot arm
Composition and Containment Group		
SUBSTANCEMATERIALINGREDIENT + WHOLE	2.42	plastic bag
PART/MEMBER + COLLECTION/CONFIG/SERIES	1.78	truck convoy
X + SPATIAL CONTAINER/LOCATION/BOUNDS	1.39	shoe box
Topic Group		
TOPIC OF COMMUNICATIONIMAGERYINFO	8.37	travel story
TOPIC OF PLANDEALARRANGEMENTRULES	4.11	loan terms
TOPIC OF OBSERVATIONSTUDYEVALUATION	1.71	job survey
TOPIC OF COGNITIONEMOTION	0.58	jazz fan
TOPIC OF EXPERT	0.57	policy wonk
TOPIC OF SITUATION	1.64	oil glut
TOPIC OF EVENT/PROCESS	1.09	lava flow
Attribute Group		
TOPIC/THING + ATTRIB	4.13	street name
TOPIC/THING + ATTRIB VALUE CHARAC OF	0.31	earth tone
Attributive and Coreferential		
COREFERENTIAL	4.51	fighter plane
PARTIAL ATTRIBUTE TRANSFER	0.69	skeleton crew
MEASURE + WHOLE	4.37	hour meeting
Other		
HIGHLY LEXICALIZED FIXED PAIR	0.65	pig iron
OTHER	1.67	contact lens

Table 2.1 Semantic relations of Largest Dataset by (Tratz and Hovy, 2010)

2.4.5 Paraphrasing

The meaning of a noun compound can be expressed using a paraphrase. For example, *"student protest"* and *"student price"* can be explained using paraphrases *"protest performed by students"* and *"special price for the benefits of students"*, respectively. In this type of semantic representation, the semantic relations need not come from a small set of predefined patterns or rules. Again, this type of paraphrasing might be subjective to the interpreter. For instance, *"chocolate bar"* can be paraphrased in many ways, and substituting synonym can give rise to even more paraphrases.

E.g., Chocolate Bar

a. Bar made of Chocolate

b. Bar consist of Chocolate

c. Bar tasted like Chocolate

d. Bar, melted into Chocolate

In this section, we had a look at various inventories of semantic relations for noun compounds. For our experiments, we have used an inventory proposed by (Tratz & Hovy, 2010).

2.5 Summary

In this chapter, we discussed linguistics aspects of compounds and noun compounds. We discussed various criteria for compounds, noun compound and related terms, and various representations for the semantics of noun compounds.

In the next chapter, we will be discussing about our work for automatic noun compound interpretation. Based on the analysis of this work, we shall explain how our idea is novel and different from the existing work.

Chapter 3

Resources, Technology & NLP Tools

Having acquired a theoretical background of the project, we are now moving towards implementation of our system. Before we learn the system in depth, we present the description of tools and libraries that we are going to use for building the system. In this chapter, we shall give brief overview of lexical resources and their implementation, scientific library, and machine learning libraries. We shall restrict ourselves to only those tools which has been used for our experiments.

3.1 Resources

For computation and inferencing purpose, a computer need knowledge, and such knowledge should be represented in computation friendly format. WordNet is one such resources which gives information about how words in a language can be related, and how one can make meaning from tokens – primary units. In this section, we give brief overview of the WordNet, and its pythonic implementation.

WordNet is a large lexical database of English. Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept. Synsets are interlinked by means of conceptual-semantic and lexical relations. The result is a network of meaningfully related words and concepts. WordNet superficially resembles a thesaurus, in that it groups words together based on their meanings.

NLTK defines an infrastructure that can be used to build NLP programs in Python. It provides basic classes for representing data relevant to natural language processing; standard interfaces for performing tasks such as part-of-speech tagging, syntactic parsing, and text classification; and standard implementations for each task that can be combined to solve complex problems. It provides easy to use interface to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries.

Google-Word2Vec is a pretrained word-embedding. it includes word vectors for a vocabulary of 3 million words and phrases that google trained on roughly 100 billion words from a Google News dataset. The vector length is 300 features.

3.2 Technology

For our task related to NLP, we use **Python** as a programming language. Python is a simple yet powerful programming language that efficiently works with linguistic data processing. Being an interpretation language, it supports interactive exploration. Python is object oriented which permits code re-usability and encapsulation. It comes with

extensive standard libraries, is easy to use and is very suitable for processing linguistic data. It is the most popular programming language in the NLP community. So we use python language for our project.

3.2.1 Scientific Libraries

As mentioned earlier, Python comes with extensive libraries for different tasks like numerical processing, graphical programming, etc. We hereby describe the scientific libraries we use for numerical processing.

Numpy (v.1.14.1) : NumPy, one of the libraries in python, is very efficient for numerical processing. It has a powerful N-dimensional array object. It supports operations over large multidimensional arrays and matrices which brings a manifold increase in the computation speed of the system. It also includes linear algebra and fourier transforms.

Pandas-(v.0.22.0) : pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with structured (tabular, multidimensional, potentially heterogeneous) and time series data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real world data analysis in Python.The two primary data structures of pandas, Series (1-dimensional) and DataFrame (2-dimensional), handle the vast majority of typical use cases in finance, statistics, social science, and many areas of engineering. pandas is built on top of NumPy and is intended to integrate well within a scientific computing environment with many other 3rd party libraries.

scikit-learn [sklearn-(v.0.19.1)] : This package includes implementation of most-of the basic machine learning algorithms, including but not limited to clustering, classification, evaluation, data processing, etc.

3.2.2 Optimization Libraries

Python contains numerous packages that are well known for optimizing execution time and space.We will discuss one which we have extensively used throughout the implementation of system.

multiprocessing-(v.0.70a1) : The multiprocessing package offers both local and remote concurrency, effectively side-stepping the Global Interpreter Lock by using subprocesses instead of threads. A prime example of this is the Pool object which offers a convenient means of parallelizing the execution of a function across multiple input

values, distributing the input data across processes (data parallelism).
We have successfully implemented many blocks of time consuming code with multiprocessing.Pool with multiple processes.

Sample code-snippet to use multiprocessing.Pool is described as below.

```
from multiprocessing import Pool

def f(x):
    return x*x

if __name__ == '__main__':
    p = Pool(5)
    print( p.map(f, [1, 2, 3]) )

Output : [1, 4, 9]
```

3.2.3 Advanced Machine Learning Libraries

Keras : Keras is a high-level neural networks API, written in Python and capable of running on top of TensorFlow, CNTK, or Theano.
We have used keras to create nueral network for learning numerical representation for complex features like dependencies and pos-tags.

3.3 NLP Tools

To extract important information from web-extracted sentences, we used various NLP tools. Here, we give brief information about NLP tools we used. The approach for extracting the information will be discussed while discussing feature extraction in Chapter 5.

3.3.1 Stanford CoreNLP

In order to perform parsing on the sentences, we use the tool Stanford CoreNLP. Stanford CoreNLP integrates many of Stanford's NLP tools, including the part-of-speech (POS) tagger, the named entity recognizer (NER), the parser, the coreference resolution system, sentiment analysis, bootstrapped pattern learning, and the open information extraction tools.

We have extracted features for our system after parsing data through Stanford CoreNLP. We have used extended-dependencies and pos tagging mechanisms to parse data from given sentences containing constituent nouns of noun compound.

A dependency parser analyzes the grammatical structure of a sentence, establishing relationships between ”head” words and words which modify those heads.
a sample dependency parse graph :

Sentence : ***”Thousands of students participated in protest at JNU . ”***

Dependency Graph :

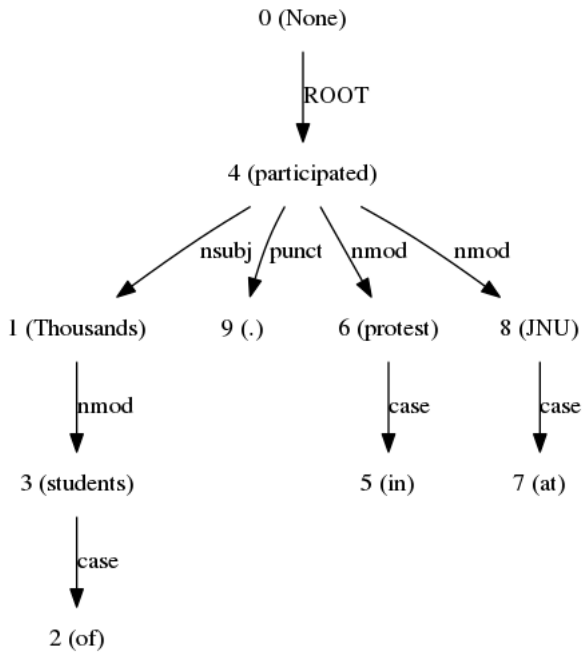

Fig. 3.1 Sample Dependency Parse Graph for sample sentence.

Stanford CoreNLP uses **Universal Dependencies**[1] for dependency parsing. Stanford CoreNLP API gives 3 types of different dependencies as following :

basicDependencies : The basic dependency representation forms a tree, where exactly one word is the head of the sentence, dependent on a notional ROOT and all other words are dependent on another word in the sentence.

[1]Find Description of Universal Dependencies at http://universaldependencies.org/u/dep/all.html

enhancedDependencies: As described in (Schuster and Manning, 2016) ,in addition to the basic dependency representation, which is obligatory for all UD treebanks, it is possible to give an enhanced dependency representation, which adds (and in a few cases changes) relations in order to give a more complete basis for semantic interpretation. The enhanced representation is in general *not a tree* but a general *graph* structure

enhancedPlusPlusDependencies : As described (Schuster and Manning, 2016) addition to enhancedDependencies , enhanced++ dependencies takes into account of Partitives and light noun constructions, Multi-word prepositions, Conjoined prepositions and prepositional phrases & Relative pronouns.

Except from Dependencies we have used POS tagging mechanism of Stanford-CoreNLP, which uses *part of speech*(POS) tags from *PennTreebank*[2]

3.3.2 SENNA

To get semantic role labels for noun compound with respect to verb with lowest dependency for both constituent nouns in dependency parse graph/tree we use the tool SENNA. We use the python wrapper of this tool called **practNLPTools** (Das, 2014). For semantic role labeling, the tool uses the relations available from **PropBank**.

3.4 Summary

In this chapter, we discussed various building blocks for our experiments. We gave overview of lexical and computational resources. We also explained Theano, a library we used for performance tuning. From the next chapter, we shall discuss our experiments, outcomes, and analysis of the system.

[2]Penn-Treebank POS tags at https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html

Chapter 4

Related Work And Experiments

The previous chapters gave us an idea related to the noun compound semantics and the tools and resources required for the same. We also know that the task of extracting the semantic relation is challenging. After having the knowledge about the importance of extracting semantic relation between the noun compounds and the difficulties that arise, we now discuss in this chapter various approaches already proposed for the task

4.1 Noun Compound Processing

Processing of noun compounds can be considered as a process of three phases: (i) identification of a noun compound, (ii) Parsing of the noun compound, and (iii) extracting the semantic relation between the components of the noun compound.

4.1.1 Noun Compound Identification

Before we go for identification of noun compound, it should be clear which compounds are considered as noun compounds. Different research scholars have given different criteria for a compound to be called as a noun compound. In the computational field, most consider the (Downing, 1977) definition of a noun compound:

> "Noun compounds are any sequence of two or more nouns acting as a single noun."

In our work, we consider only **compositional** noun compounds. Compositional noun compounds are those noun compounds whose meaning can be derived by combining the meaning of individual nouns.

4.1.2 Parsing a Noun Compound

The second step in noun compound processing is parsing the noun compound. If there are more than two nouns in a noun compound, it parsing is required. For example,

(a). Liver cell antibody
(b). Liver cell line

Here, in (a) liver cell is related to antibody. In example (b) cell line has some relation with liver While parsing a noun compound, we can have only two noun compounds at any node. So we can represent long noun compounds using brackets in groups of two nouns. This can be represented as:

[[liver cell] antibody]

[liver [cell line]]

Thus as the brackets change, the meaning of the noun compound changes..

4.1.3 Noun Compound Interpretation

The identification of noun compounds and parsing are not very difficult. A trained model performs well. The main challenge is assigning a label from the available semantic relations repositories. This is because these relations are highly semantic in nature. Lower inter- annotator agreement for all available datasets is the indication of this fact.

We will focus here on the interpretation part. In the next section, we discuss automatic interpretation of noun compound and different approaches proposed in the literature for the same.

4.2 Automatic Interpretation of Noun Compound

The main aim of our proposed work is that when a noun compound is given to the system, it should be able to extract the semantic information. The system, thus, is able to assign a semantic label to a noun compound automatically.

The figure below gives an overview of the systems proposed in the literature for automatic interpretation of noun compounds.

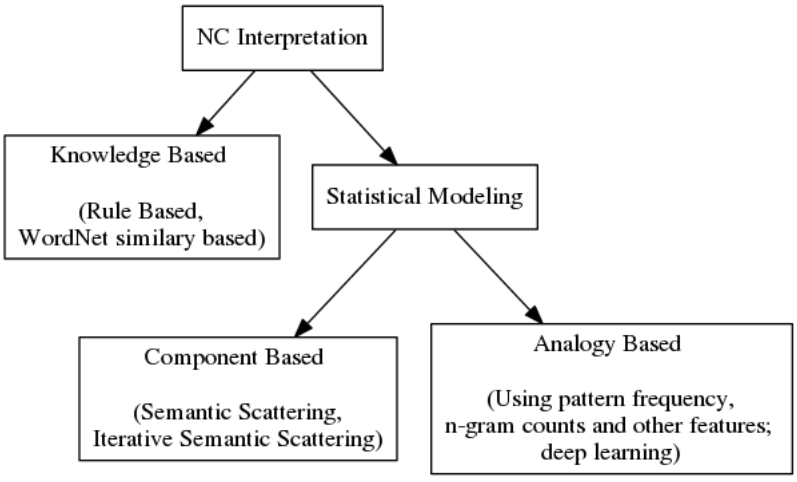

Fig. 4.1 Methods for Automatic Interpretation of noun compounds

4.2.1 Knowledge Based Approach

Some researchers have developed systems for automatic (or semi-automatic) interpretation of noun compounds using predefined rules (Vanderwende, 1994) and (Rosario et al., 2002).

(Vanderwende, 1994) manually created a set of weighted rules to identify the semantic relations between the components. The rules assigned a semantic relation based on the type of the components. Later these approaches accounted him an advantage that no weights were associated with the rules. On the same note, it bagged a drawback that the degree to which those rules were satisfied were not expressible and so in many of the cases the most plausible interpretation of the noun compound will not be produced.

Furthermore, as per as (Kim and Baldwin, 2005), the other alternative approach is to understand various methods of calculating the similarity between the component nouns which includes the entire path based and information based similarities. To know that how those components of nouns are related to each other by specifying them into a set of class labels (relations, they had proposed 20 class labels as shown in 4.2 . As stated by him, *WUP similarity*is a path based similarity of the synset of noun in NLTK where it denotes how similar two word senses are, based on the depth of the two senses in the taxonomy and that of their Least Common Subsumer (most specific ancestor node).

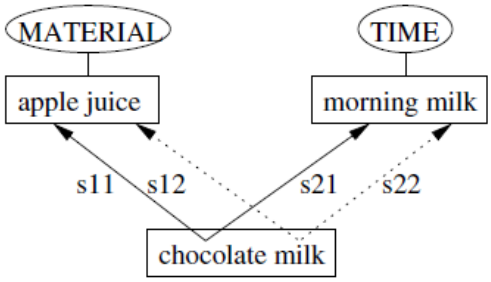

Fig. 4.2 Approach from Kim and Baldwin based on WUP Wordnet Similarity

Therefore, as per Kim and Baldwin (2005), Fig. 4.2 shows the correspondences between two training noun compounds, *apple juice* and *morning milk*, and a test noun compound, *chocolate milk*;Table 4.1 lists the noun pairings and noun–noun similarities based on WordNet. Each training noun is a component noun from the training data, each test noun is a component noun in the input, and Sij provides a measure of the noun–noun similarity in training and test, where t1 is the modifier and t2 is the head noun in the noun compound in question. The similarities in Table 4.1 were computed by the WUP method as implemented in WordNet Similarity.

	% (Training Noun)	Test Noun	S_{ij}
t1	Apple	chocolate	0.71
t2	juice	milk	0.83
t1	morning	chocolate	0.27
t2	milk	Milk	1

Table 4.1 WordNet-based similarities for component nouns in the training and test data

The simple product of the individual similarities (of each modifier and head noun, respectively) gives the similarity of the noun compound pairing. For example, the similarity between *chocolate milk* and *apple juice* is 0.60, while that between *chocolate milk* and *morning milk* is 0.27. Note that although milk in the input noun compound also occurs in a training example, the semantic relations for the individual noun compounds differ. That is, while *apple juice* is a juice made from apples (MATERIAL), morning milk is milk served in the morning (TIME). By comparing the similarity of both elements of the input NC, we are able to arrive at the conclusion that *chocolate milk* is more closely related to *chocolate milk*, which provides the correct semantic relation of MATERIAL (i.e. *milk* made from/flavored with *chocolate*). Unlike word sense disambiguation systems, our method does not need to determine the particular sense in which each noun is used.

So, in our experiments for the implementation of paper, a set of test samples and train samples were tested to predict the label (relation of 4.2) of test sample and were later on compared to the actual labels of those test samples. Therefore, WUP similarity was calculated for each combination of head-head and modifier-modifier of the noun compound pairs. The similarity measure was combined and computed with new value and the one which is best match was selected to predict the label for particular test sample.

The best match was calculated by:

$$s_A(((N_{i,1}), N_{i,2}), (B_{j,1}, B_{j,2})) = \frac{((\alpha * S_1) + S_1) * ((1 - \alpha) * S_2 + S_2)}{2}$$

Where, S_1 and S_2 are similarities of head and modifier combinations and α is the weighted value which is suitably assumed to be 0.5.

Similar rules based approach was proposed by (Rosario et al., 2002) for the medical domain. They had created a taxonomy of nouns in the medical domain, and created rule based on types of noun compounds.

Relation	% (Definition)	Example
AGENT	N2 is performed by N1	student protest
BENEFICIARY	N1 benefits from N2	student price
CAUSE	N1 causes N2	printer tray, flood water
CONTAINER	N1 contains N2	exam anxiety
CONTENT	N1 is contained in N2	paper tray, eviction notice
DESTINATION	N1 is destination of N2	game bus, exit route
EQUATIVE	N1 is also head	composer arranger, player coach
INSTRUMENT	N1 is used in N2	electron microscope, diesel engine
LOCATED	N1 is located at N2	building site, home town
LOCATION	N1 is the location of N2	lab printer, desert storm
MATERIAL	N2 is made of N1	carbon deposit, gingerbread man
OBJECT	N1 is acted on by N2	engine repair, horse doctor
POSSESSOR	N1 has N2	student loan, company car
PRODUCT	N1 is a product of N2	automobile factory, light bulb
PROPERTY	N2 is N1	elephant seal
PURPOSE	N2 is meant for N1	concert hall, soup pot
RESULT	N1 is a result of N2	storm cloud, cold virus
SOURCE	N1 is the source of N2	chest pain, north wind
TIME	N1 is the time of N2	winter semester, morning class
TOPIC	N2 is concerned with N1	computer expert, safety standard

Table 4.2 The Semantic relations in Noun Compound (N1= modifier, N2= head noun) by (Kim and Baldwin, 2005)

Limitations of rule based Approach

As the interpretation of noun compounds is influenced by some pragmatic knowledge, and there is no much context during interpretation, rule based system mostly fails to handle such cases. As rule based systems perform poor, we need to introduce statistical system, which can model pragmatic or contextual knowledge and help the interpretation system.

Following are two main factors challenging the automatic interpretation system:

1. **Productivity and Heterogeneous Nature:** (millions of new noun compounds are generated by people) a system cannot set predefined fact and rules for the noun compounds to be encountered in the future.

2. **Ambiguity:** the natural language itself is ambiguous in nature, so in most of the cases it is very difficult to develop a rule based deterministic system. Thus, there is a need to introduce a statistical approach for such tasks.

4.2.2 Statistical Methods

In this approach, the system used a set of training data to learn patterns automatically. Later, such patterns can be used for prediction on test data. This eliminates the need to write the rules manually.

Considering the productivity of noun compounds, the learning from the training data can be generalized and thus the system can be made flexible to other languages as well.

Supervised Approach

In supervised learning, each example is a pair consisting of an input object (typically a vector) and a desired output value (also called the supervisory signal). A supervised learning algorithm analyzes the training data and produces an inferred function, which can be used for mapping new examples. An optimal scenario will allow the algorithm to correctly determine the class labels for unseen instances. This requires the learning algorithm to generalize from the training data to unseen situations in a ”reasonable” way.

For instance, (Tratz and Hovy, 2010) presented a large annotated set of examples. The main aim behind creating this dataset was to improve inter-annotator agreement. So, they annotated dataset iteratively to remove ambiguities among annotation guideline. They used SVM (Support Vector Machine) and MaxEnt (Maximum Entropy) classifiers on the dataset, and presented promising results.

Later (Dima and Hinrichs, 2015) used deep learning based approach on the same dataset for automatic interpretation. Unlike, (Tratz and Hovy, 2010), they used word embedding instead of feature engineering, and presented similar results.

Unsupervised Approach

In the unsupervised learning approach, the input data has no expected output. The input data, in our case, is a noun compound, without any semantic label. The learning algorithm divides the training data examples into different groups and assigns a label to each group. Since the examples given to the learner are unlabeled, there is no error or reward signal to evaluate a potential solution. This distinguishes unsupervised learning from supervised learning.

When a novel example is encountered, the algorithm checks each group, finds the group to which this example is “most similar”, and the label of that group is assigned to this example.

Unsupervised learning approach just needs data and nothing else. Thus, if we have a good data source, unsupervised method can be applied to it to obtain fruitful results.

4.3 Summary

In this chapter, we discussed the three step procedure of noun compound processing. We then discuss automatic interpretation of noun compounds and various approaches for the same. In the next chapter, we would be proposing a new idea and approach for noun compound interpretation.

Chapter 5

System & Implementation

Semantic relations play major role in identifying the relation between nouns. As, we saw in the previous chapter, they hold nouns in the compound to bring out some new sense for the noun compound. Moreover, we will go through the pragmatics of the system. Thus, we will now try to capture all such information of relations into our system to deal the noun compounds effectively.

5.1 Central Idea

As the noun compounds are more productive in nature, and most of the noun compounds appear only once in a large corpus. These characteristics of the noun compounds make them a special case, and demand special treatment. So, the problem is to find out some linking relationship exist between the existing compounds and predict the new ones depicted in For noun compounds, the semantic relation between the components of a compound depends on following :

- Semantics of components of a noun compound, i.e., individual noun in noun compound.
- How components interact (or can be linked in sentences) with each other in the real world.
- Semantics of the compound, i.e., meaning of the noun compound itself.

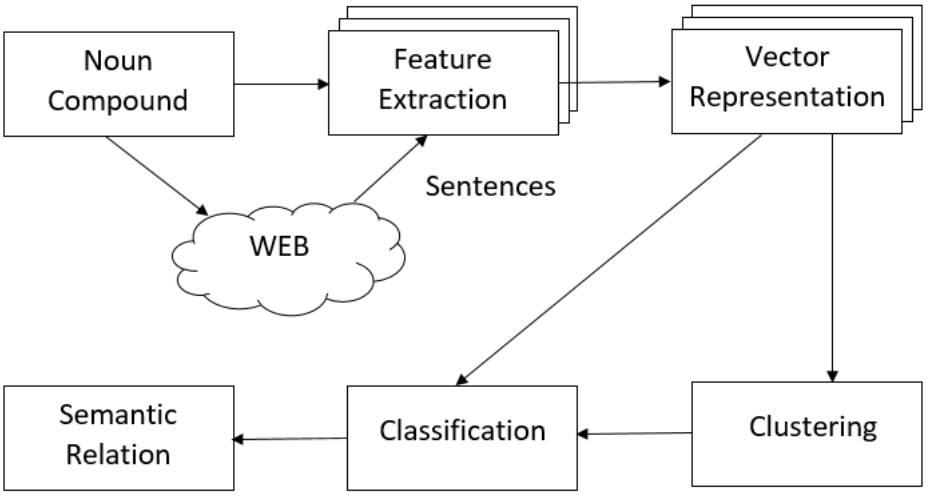

Fig. 5.1 System Architecture

For the semantics of a component, one can use lexical resources. Eventually, when extracting those relations directly, these may vary in several cases like for the noun compounds: *"customer treatment"* and *"patient treatment"*; one cannot apply the relation of

treatment interchangeably to customer and patient. As, treatment for customer would be some sort of either service or fulfilling desires or requirements of customers; whereas treatment for patients would be to provide a cure for the patient's disease.

So, for the rest two types, we can use large corpora to extract required detail. For example, given a compound *student protest*, the semantics of the *student* and *protest* can be extracted from some lexical resources. For the second information, we can extract sentences from the web containing student and protest with some words in between. Using the words between student and protest, we will try to infer the semantic relations. Our plan is to use large corpora to extract such sentences.

5.2 Feature Extraction

Once we have noun compounds, we need to process them to extract the information on how the component nouns are connected with each other. For this, we need to convert our data in a form that makes it easy to process. Feature extraction is a process of representation of the data (here, a dataset of noun compounds) into numerical form so that it can be computed for a learning algorithm.
The whole process of giving a noun compound to representing it in a numerical format is divided into the following steps:

1. Preparing Index / Word-Concordance
2. Search/Extract sentences for given Noun Compound
3. Extract different types of features of the sentences and represent it in numerical form.

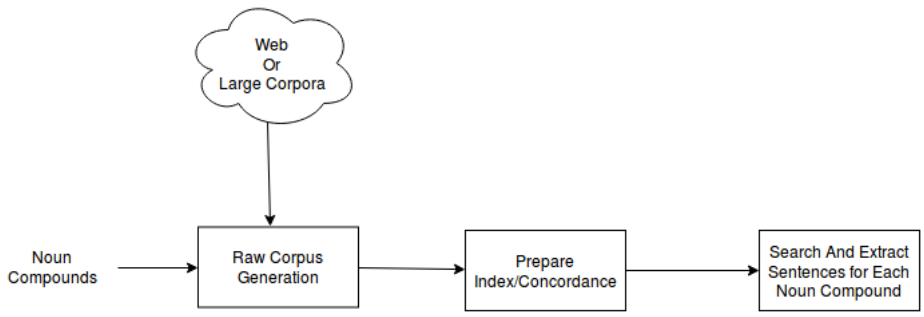

Fig. 5.2 Index Generation And Sentence Extraction

We will now discuss these steps in detail.

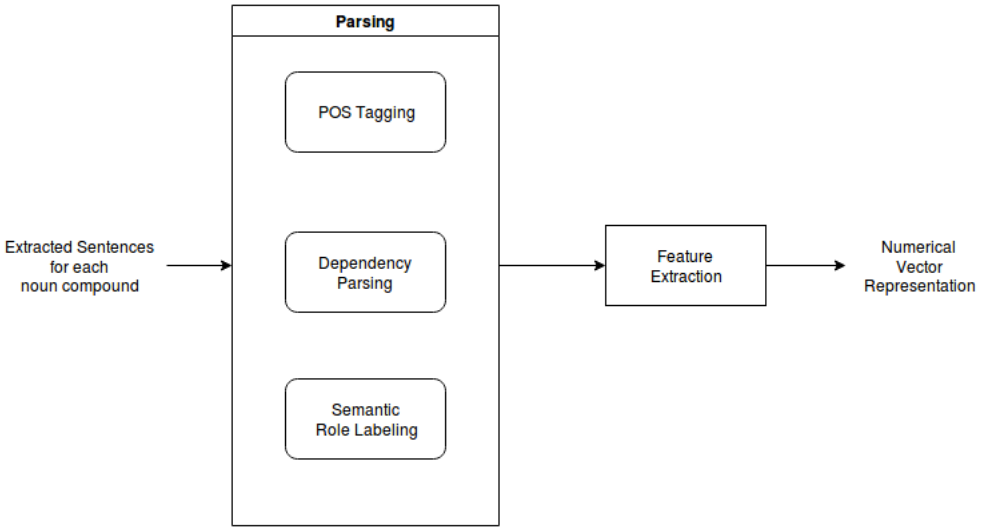

Fig. 5.3 Feature Extraction

5.2.1 Types of sentences containing Noun Compound

For My system, I have planned to extract two types of sentences for each noun compound:

Type 1: In this type of sentences, the components of a noun compound occur together. From this type of sentences, the system can extract semantics of the compound itself.

E.g., *"The country was in recession and **student protests** were frequent."*

Type 2: Here the components of a noun compound are separated by few words. The words0 (or phrases) in between the components will give hints about how the components are related in a sentence. Extracting such information from multiple sentences provides information about the relation between the components in general.

E.g., *"Thousands of **students** participated in series of **protests** at JNU."*

5.2.2 Preparing Index / Word Concordance

I have prepared look up Index / Word Concordance for each word available in Corpus Vocabulary. This phase of system helped to reduce memory storage requirements at run time. Except storing large list of sentences to main memory , at runtime I can look up for each word to find ,on which location sentence containing given word is held at. Index format is as following:

[File for each word] contains (file Id -> list of sentence id) tuples

5.2.3 Search/Extract sentences for given Noun Compound

As Mentioned in previous subsection , Instead of searching each sentence containing a word by already available tool I have implemented Index , containing data to help search sentences for each word.
Extraction of sentences containing given Noun compound is done as following:

- Search for sentences having word Modifier noun(word1)
- Search for sentences having word Head noun(word2)
- Find sentences containing both words.

5.2.4 Extracting different features

I have extracted the following information which will be used for feature extraction:

1. List of Part of Speech (POS) Tags for verbs
2. A List of the possible enhanced dependencies assigned by the dependency parser
3. List of the Semantic Role Labels (SRL) for the sentence.
4. Word Vocabulary of available in Large Corpus from which all sentences are extracted.
5. Extended Word Vocabulary, which extends above mentioned vocabulary using derivationally related forms.
6. List of Path for each data-file available in large Corpus from which all sentences are extracted.
7. List of mapping for each word to its possible base form in wordnet, with "Noun" part of speech tag. [w1_mapping : mapping list for *modifier* nouns]

   forms(modifier) = singular(modifier) , plural(modifier)

8. List of mapping for each word to its possible base form in wordnet, with all part of speech tag.

   forms(head) = singular(head), plural(head) ∪verbal_inflection(verbal_root(w2))

We will now discuss the features that we extract, i.e. the features for each sentence.

1. **W1** : The modifier of the noun compound is fetched from vocabulary.
2. **W2** : The head of the noun compound is fetched from vocabulary.
3. **POS_tag_W1** : POS tag of the Modifier noun.
4. **POS_tag_W2** : POS tag of the Head noun.
5. **POS_Tag_main_verb** : POS tag of the verb having dependency root is the main verb of the sentence.
6. **aux** : Direct dependency named "aux" from main verb. this feature helps derivation of tense of the sentence.
7. **aux_pass** : A passive auxiliary of a clause is a non-main verb of the clause which contains the passive information. Dependency named "auxpass" in dependency graph. This feature to helps detect if sentence is in active voice or passive voice.
8. **common_ancestor** :When two nouns of the noun compound are separated by a few words, they have a common ancestor (common word) in the dependency graph of the sentence. We fetch the common ancestor in the following way: traverse the dependency graph of the sentence, make the list of the words from W2 to the ROOT say path. Now traverse the graph from W1 to ROOT. The common word that comes earliest in both path from word to ROOT is *common ancestor/common word*.
9. **common ancestor dependent towards W1 subgraph (to_w1_gl)** : Direct dependent node in dependency graph in common ancestor to W1 path.
10. **common ancestor dependent towards W1 subgraph (to_w2_gl)** : Direct dependent node in dependency graph in common ancestor to W2 path.
11. **main_verb** : The verb having dependency root is the main verb of the sentence.
12. **srl_verb** : Least level Verb that separates semantic role labels for W1 and W2.
13. **srl_W1** : semantic role label for W1 with respect to srl_verb.
14. **srl_W1** : semantic role label for W2 with respect to srl_verb.
15. **First dependency towards W1 from common ancestor (to_w1_dep)** : First direct dependency from common ancestor to W1 path.
16. **First dependency towards W2 from common ancestor (to_w2_dep)** : First direct dependency from common ancestor to W2 path.

17. **POS_aux** : POS tag for aux.
18. **POS_auxpass** : POS tag for aux_pass
19. **POS_common_ancestor** : POS tag for common_ancestor.

5.2.5 Representation of features

In the previous version of the system, each feature was presented to machine learning algorithms in form of their 1-hot representation. In this work, I try to incorporate the latest techniques for the representations.

Roughly, I categorize the features in four categories: (1) word as a feature, (2) POS-tag as a feature, (3) a dependency label as a feature, and (4) an SRL (semantic role label) as a feature. Here, I describe each of the four representations.

Word as a feature: To represent features that contain words available in vocabulary/-dictionary, I have used Google's pre-trained word embedding model word2vec[1]. Each of these feature is represented by real valued vector of size 300. We use this representation for the following features:

W1, W2, aux, aux_pass, common_ancestor, main_verb, srl_verb, to_w1_gl, to_w2_gl

POS-tag as a feature: I have created an artificial corpus of POS-tags by replacing words in a corpus by their POS-tags. Then, I trained the word2vec tool, on the POS-tag corpus. I get vector representations for each POS-tag in this way. I use these vectors as representation for following features:

w1_pos, w2_pos, main_verb_pos, aux_pos, auxpass_pos, common_ancestor_pos

Dependency-label as a feature: To represent features that contain dependency-tags, I have extracted list of enhanced dependencies from parsing data. I have trained a simple Neural Network over these enhanced-dependencies with governor and dependent words as input and the dependency label as output. I extract weight-matrix (hidden-to-output layer), and consider it as representation for the dependency tags. Fig.5.4 shows neural network layers architecture for dependency representation learning. I have considered bellow mentioned features for this category :

to_w1_dep, to_w2_dep

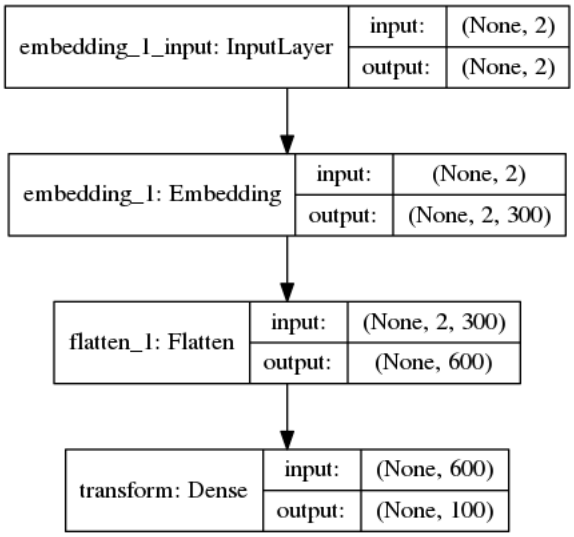

Fig. 5.4 Neural Network layers for Dependency Representation

As per Fig.5.4, Neural Network takes 2 dimensional input – governorGloss and dependentGloss – and it classifies input data into several classes of dependency. I have considered 200 dimension hidden layer *weight matrix* as representation for each dependency.

SRL (semantic role label) as a feature: To represent features that contain semantic role labels I have extracted possible semantic role labels and represented them with 1-Hot representation. I have considered bellow mentioned features for this category:

srl_w1, srl_w2

Feature Type	#	Dimension of Individual Feature	Cumulative Dimension
Word feature	9	300	2700
POS-tag feature	6	10	60
Dependency feature	2	100	200
SRL feature	2	57	114
Total			**3074**

Table 5.1 Statistics of feature dimensions.

Table 5.1 shows statistics of the feature representation. Final dimension of a feature vector for noun compound is 3074.

So , I process sentence and noun compound features as above mentioned techniques to convert them in numerical representation. Now, We move towards grouping noun compounds having similar semantic relations between them using different clustering algorithms.

[1]https://code.google.com/p/word2vec/

5.3 Clustering

As discussed earlier, noun compounds are productive. This means, people generate new noun compounds in day to day life. Scholars have shown that there are hundreds of millions noun compounds in large corpora. But, there are only a few thousands of them are annotated. Our aim was to use this unlabeled data in our system. In this situation, clustering can be of help. We can use the large unlabeled data that are available as an input to the system.

Many researchers have proposed technique to use clustering as an intermediate step before performing classification. The results have shown that clustering can be used to improve the performance of classification using the unlabeled data. Clustering can help to make the input data more intuitive. So classifier gives better performance.

Kyriakopoulou and Kalamboukis (2008) explain how clustering sets the ground for classification. Figure 8 illustrates clustering as a helping step for classification. The labeled examples of the training set are denoted with + and − signs, while the unlabeled examples of the testing set are denoted with dots. A classifier trained with given examples would find plane 'A' instead of the desired plane 'B' as shown in Fig. 5.5(a). Fig.5.5 (b) both datasets (training and testing) are clustered into two non-overlapping clusters. Fig. 5.5(c) shows the result after performing classification on this clustered data.

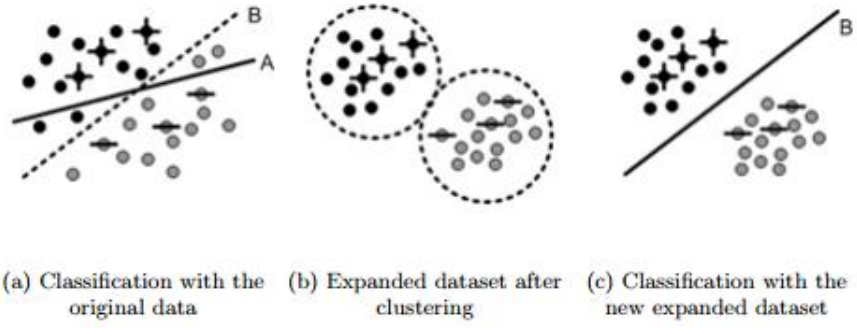

Fig. 5.5 An Insight of using clustering with classification

I come up with an idea of using the unlabeled noun compounds along with the labeled data from our system. As per our knowledge, this is the first ever attempt to perform semi-supervised interpretation of noun compounds, i.e., using a small set of labeled data and large set of unlabeled data for prediction of semantic relation in noun compound. As till now none had attempted to perform other than supervised clustering with such large corpora.

5.4 Classification

The above section helps us understand how clustering can act as a helping step for classification. We perform clustering on our training as well as test data. We expect that the prior knowledge about the nature of the testing set will help in increasing the efficiency of the classifier. The result obtained by clustering acts as an input to the classifier. This added information helps to classify examples of a more generalized level. In the next section, I describe the combining of classification and clustering.

Now, the Fig. 5.5 gives us a clear intuition, how classification can be improved if the input data is clustered beforehand. The main difference between the plane chosen in Fig. 5.5(a) and Fig. 5.5(c) is that the plane "B" is equidistant from with the clusters. This leaves a sufficient margin so that test example, can be correctly classified with more certainty. On the contrary, in case of "A", the probability that the test sample is misclassified increases as the hyperplane is not sufficiently distant from both the classes.

For my system, I use the results obtained after clustering for classification. The proposed idea is that after performing clustering of the dataset, we obtain a cluster label for each feature vector i.e. for every noun compound. I augment the feature-vector for each noun compound with 1-hot representation of the clustering information to create a new vector. I perform classification on this new dataset.

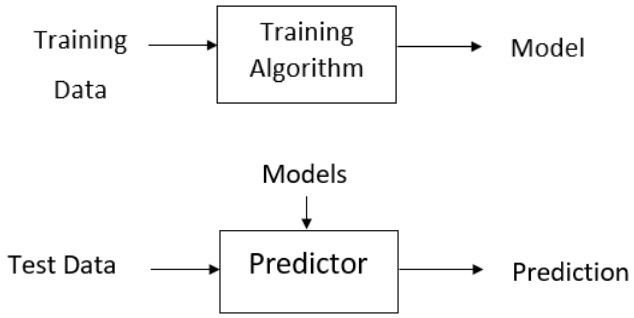

Fig. 5.6 Classification steps

I have compared the results obtained by classification with those obtained by performing classification after clustering. Here in Fig. 5.6 our vector representation and results of clustering would turn out into predicting the appropriate results for our system.

After I obtain the results of classification, I needed to measure the performance of the classifier. This can be done using the following measures:

Precision: (also called positive predictive value) it is the fraction of retrieved instances that are relevant.

Recall: (also known as sensitivity) it is the fraction of relevant instances that are retrieved.

Both precision and recall are therefore based on an understanding and measure of relevance. Suppose a computer program for recognizing dogs in scenes from a video identifies 7 dogs in a scene containing 9 dogs and some cats. If 4 of the identifications are correct, but 3 are actually cats, the program's precision is 4/7 while its recall is 4/9.

A measure that combines precision and recall is the harmonic mean of precision and recall, the traditional **F-measure** or **balanced F-score** :

$$F = \frac{2 * precision * recall}{precision + recall}$$

Thus, this section is the wind up part of my system to be in a working state, and later on, in the next section I would analyze each stage and component of the system.

5.5 Results

I have used (Tratz and Hovy, 2010) dataset for the training and testing my system. Because of time constraint, I was not able to extract a large corpora of unannotated set of noun compounds. So, I have used the same dataset without labels for clustering. I have used 5-fold validation for reporting the classification results.

Table 5.2 and 5.3 compared results performance of various classifier with additional information from K-Means and Birch algorithms, respectively.

5.6 Analysis

As it can be seen from the results, the both clustering algorithms help the SVM classifier. But ,the margin is not so significant. In case of LDA, the clustering output helps the classifier with significant margin. In case of kNN classifier, KMeans help the classifier but the Birch algorithm adds the noise degrading the performance.

It would be really interesting check performance of the system using more classifiers and clustering algorithms. Because of time constraint, I am not able to spent more time in the analysis of how clustering helps the classifier. But, insight in it will help in selection of better clustering algorithm and number of clusters.

It would be really interesting to see performance of the system on other datasets (like (Kim and Baldwin, 2005), (Girju, 2007), etc.).I am looking forward to see performance of the system on those datasets.

Classifier	**Without Clustering**			**KMeans Clusternig**			
	P	**R**	**F**	**P**	**R**	**F**	**k**
SVM (linear)	47.78	47.55	47.27	47.95	47.66	47.40	3
				48.42	48.24	**47.93**	9
				47.55	47.37	47.07	27
				47.97	47.66	47.42	43
				48.02	47.70	47.45	55
LDA	25.67	23.32	24.09	25.26	23.25	23.91	3
				25.42	23.10	23.86	9
				25.82	23.58	24.24	27
				26.45	24.08	**24.80**	43
				26.32	24.05	24.74	55
KNN	33.08	32.32	31.04	32.92	32.32	31.02	3
				32.69	32.32	30.93	9
				32.44	31.99	30.74	27
				32.74	32.39	31.04	43
				32.92	32.57	**31.28**	55

Table 5.2 Performance of various classifier with and without using additional information from **K-Means** clustering. (performance numbers are given in terms of **P**recision, **R**ecall, and **F**-score; The **k** value indicates number of clusters.)

Classifier	**Without Clustering**			**Birch Clustering**			
	P	**R**	**F**	**P**	**R**	**F**	**k**
SVM (linear)	47.78	47.55	47.27	47.63	47.33	47.10	3
				47.44	47.19	46.97	12
				47.70	47.59	47.30	15
				47.92	47.77	**47.49**	27
				47.89	47.70	47.44	30
LDA	25.67	23.32	24.09	26.37	24.05	**24.80**	3
				25.72	23.54	24.25	12
				25.35	23.50	24.04	15
				25.56	23.65	24.22	27
				25.48	23.61	24.16	30
KNN	33.08	32.32	**31.04**	32.99	32.06	30.90	3
				32.74	31.96	30.74	12
				32.84	32.10	30.82	15
				33.09	32.03	30.88	27
				33.06	31.99	30.85	30

Table 5.3 Performance of various classifier with and without using additional information from **Birch** clustering. (performance numbers are given in terms of **P**recision, **R**ecall, and **F**-score; The **k** value indicates number of clusters.)

Chapter 6

Conclusion and Future Work

6.1 Conclusion

Hereby I conclude that, After some experience in field of NLP and work with Noun Compound. I believe interpretation of semantic relation needs contextual information about placement of constituent nouns of noun compound within sentence.

During this work, I have improved sentence extraction and feature representation. I have used latest way of representing the feature using distributional semantics. I have aggregated features for several sentences for noun compound to represent one feature vector for one noun compound. I then clustered these noun compound vectors. This clustered data is fed into classification algorithm. After classification task we get predicted label for Noun Compound.

My experiment was time, space and knowledge intensive. Yet, I have successfully ran an improved version of my system. The next version will be evaluated on few months, and I expect that it will beat the state-of-the-art system. Following few points give glimpse of my system performance:

- I have successfully increased system performance in terms of classification measures as best result from previous system recorded **21.44** F-score with LDA classifier and k-means clustering(cluster size = 43) , to current system best result as described in Table 5.2 with **47.93** F-score with SVM(liner kernel) classifier and k-means clustering(cluster size = 9). Hence , I report **123.55%** of relative increase in classification measures.

- I have used Wikipedia dump for sentence extraction. There are 9.7 million sentences in the corpus. I need to clean this corpus of raw text. I need to find sentences with our custom pattern from this corpus. I have done preprocessing on this corpus to speed up the searching. I have replaced the indexing module reducing the size of the index from ~50GB on hard drive to 15.9GB, and from ~130GB in RAM to in terms of MB. Previously, loading of the index in RAM took around ~30 hours. But Now , I am doing on demand, hence, it doesn't touch the part of system which is not necessary for the searching.

- I have extracted around 2.33M sentences from the Wikipedia dump with custom filtering , and processes these sentences to get the POS-tagging, dependency parsing, constituency parsing, and semantic role labeling. In the previous version of the system, the system took around 100 hours (with 250GB RAM and 40 processor) for parsing. My new version of the system reduces by factor of ~10.

- I have used better representation of the features. This improves the performance of the system, and reduces the dimension of the vector from 34887 in previous version to 3014 in the current version.

Despite many complexities, I have managed to improve the system in terms of system performance, execution time, and memory requirement. The experiments and subjective analysis conducted after implementing my proposed system, I came up with a point that, it can perform better if all the criteria mentioned in future scope in the next section are added to the system.

6.2 Future Work

The noun compound interpretation system could be further taken ahead by overcoming several problems with proper text corpora or web source. Future work can be done on:

- More number of Noun Compound could be extracted from Web.
- Many different clustering algorithms can be explored for better pipeline of clusters to classes.
- It would be really interesting check performance of the system using more classifiers and clustering algorithms. Because of time constraint, I am not able to spent more time in the analysis of how clustering helps the classifier. But, insight in it will help in selection of better clustering algorithm and number of clusters.
- It would be really interesting to see performance of the system on other datasets (like Kim and Baldwin (2005), Girju at. el. (2007), etc.). I am looking forward to see performance of the system on those datasets.

Therefore, the system may produce far better results if these minute criteria are considered in the future for the noun compound interpretation system.

References

Barker, K. and Szpakowicz, S. (1998). Semi-automatic recognition of noun modifier relationships. In *Proceedings of the 17th international conference on Computational linguistics-Volume 1*, pages 96–102. Association for Computational Linguistics.

Dima, C. and Hinrichs, E. (2015). Automatic noun compound interpretation using deep neural networks and word embeddings. *IWCS 2015*, page 173.

Downing, P. (1977). On the creation and use of english compound nouns. *Language*, pages 810–842.

Girju, R. (2007). Improving the interpretation of noun phrases with cross-linguistic information. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, volume 45, page 568. Citeseer.

Kim, S. N. and Baldwin, T. (2005). Automatic interpretation of noun compounds using wordnet similarity. In *Natural Language Processing–IJCNLP 2005*, pages 945–956. Springer.

Levi, J. N. (1978). *The syntax and semantics of complex nominals*. Academic Press New York.

Nakov, P. (2013). On the interpretation of noun compounds: Syntax, semantics, and entailment. *Natural Language Engineering*, 19(03):291–330.

Quirk, R. and Widdowson, H. G. (1985). *English in the world: teaching and learning the language and literatures: papers of an International Conference entitled" Progress in English studies" held in London, 17-21 September 1984 to celebrate the Fiftieth Anniversary of the British Council and its contribution to the field of English studies over fifty years*. Cambridge University Press for the British Council.

Rosario, B., Hearst, M. A., and Fillmore, C. (2002). The descent of hierarchy, and selection in relational semantics. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 247–254. Association for Computational Linguistics.

Schuster, S. and Manning, C. D. (2016). Enhanced english universal dependencies: An improved representation for natural language understanding tasks. In *LREC*.

Trask, R. (1993). *A dictionary of grammatical terms in linguistics*. Linguistics/reference. Routledge.

Tratz, S. and Hovy, E. (2010). A taxonomy, dataset, and classifier for automatic noun compound interpretation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 678–687. Association for Computational Linguistics.

Vanderwende, L. (1994). Algorithm for automatic interpretation of noun sequences. In *Proceedings of the 15th conference on Computational linguistics-Volume 2*, pages 782–788. Association for Computational Linguistics.

Warren, B. (1978). Semantic patterns of noun-noun compounds. *Acta Universitatis Gothoburgensis. Gothenburg Studies in English Goteborg*, 41:1–266.