TEXTRACTOR: A NEW APPROACH FOR N-GRAM, COLLOCATION AND MULTI-WORD EXPRESSION EXTRACTION

A Thesis

Presented to

the Faculty of the Department of Computer Science

University of Houston

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

By

Vasanthi Vuppuluri

May 2015

TEXTRACTOR: A NEW APPROACH FOR N-GRAM, COLLOCATION AND MULTI-WORD EXPRESSION EXTRACTION

Vasanthi Vuppuluri

APPROVED:

Dr. Rakesh M Verma, Committee Chairperson
Dept. of Computer Science

Dr. Kam-Hoi Cheng
Dept. of Computer Science

Dr. Latha Ramchand
Dept. of Finance

Dean, College of Natural Sciences and Mathematics

This material is based upon work supported by the National Science Foundation under Grant No. 1241772. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Acknowledgements

It gives me immense pleasure in expressing my regards to the people who supported me and helped me in making my thesis possible. First and foremost, I would like to thank my thesis advisor Dr. Rakesh Verma. Every part of this thesis is a result of his continued guidance and encouragement, without which it would not have been possible. I would like to thank my thesis committee members Dr. Kam-Hoi Cheng and Dr. Latha Ramchand for keeping me on the right track through out the duration of my thesis.

The thesis would have been incomplete without timely suggestions and comments from my colleagues, Avisha Das, Keith Dyer, Luis Felipe, Nirmala Rai, and Tanmay Thakur at ReMiND (Reasoning, Mining, Natural Language Processing and Defense) lab. Special thanks to Avisha Das and Nirmala Rai for reviewing my thesis documentation and providing valuable suggestions. I am thankful to Yvette Elder for assisting me in all the administrative work.

Finally, my sincere regards to my parents and the Almighty for encouraging and supporting me through each milestone of my academic journey.

TEXTRACTOR: A NEW APPROACH FOR N-GRAM, COLLOCATION AND MULTI-WORD EXPRESSION EXTRACTION

An Abstract of a Thesis

Presented to

the Faculty of the Department of Computer Science

University of Houston

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

By

Vasanthi Vuppuluri

May 2015

Abstract

There is so much knowledge available on the Internet now, which represents a great opportunity for automatic, intelligent text processing and understanding, but there are major problems in finding legitimate sources of information and overcoming rate limitations on search engine APIs. The work in this thesis describes methods that combine the knowledge of World Wide Web (WWW) and the power of Internet search with the knowledge extracted from dictionaries. This thesis presents Textractor, an un-supervised, domain independent general-purpose n-gram, collocation and multi-word expression (MWE) extraction software written in Python. It is modular and allows the user to choose from and compare different methods for identifying n-grams, collocations and MWEs including statistical, dictionary and Internet-based. This thesis shows that it is very hard to identify collocations based on statistical information from the given text document alone (although this might seem obvious some systems do use it), and that dictionary and Internet-based techniques, when combined properly, can be very effective sources of collocations and MWEs without their respective drawbacks. This method can overcome the limitations of current Natural Language Processing techniques. For example, Textractor can recognize collocations and MWEs even when the complete sentence is not present, and when the domain knowledge of the data is not known. It is currently designed to work with text in English but can easily be extended to other languages.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Automatically extracting phrases from the documents, be they structured, un-structured, or semi-structured, is always an important yet challenging task; and often involves processing of human language texts with the help of Natural Language Processing (NLP).

The overall goal is to create a more easily machine-readable text to process the sentences and extract certain kinds of phrases from them. Subtasks of phrase extraction include N-gram, collocation, and multi-word expression (MWE) extraction, these three subtasks are the goals of this thesis. These are fundamental tasks in NLP with a long history of research in this area, some of which are mentioned in Chapter 2. But, collocation and MWE extraction are very hard tasks - even humans have to memorize them, since there is no algorithm one can use to tell if "*yellow journalism*" is an idiom. One either knows this from being taught or one does not.

Moreover, the frequency information on how many times a phrase occurs in a document cannot help in deciding that this is an idiom. So, just as a human must use his memory as a base for this task, this thesis uses the WWW as a knowledge base to identify them. The software in this thesis is named 'Textractor'.

1.1 Terminology

1.1.1 N-gram

An *n-gram* is a contiguous sequence of n items in a given text. With respect to this thesis, items are words in a given sequence of text. An *n-gram* of size 1 is referred to as a 'unigram', an n-gram of size 2 is referred to as a 'bigram', an n-gram of size 3 is referred to as a 'trigram'. Larger sizes are referred to by the value of n, e.g., "four-gram", "five-gram", and so on. Some examples of n-grams are: {hello, computer science, he is at, go up the hill, once in a blue moon, etc}. N-grams are essentially n words in a sequence, and sometimes those sequences may not have a definite meaning.

1.1.2 Collocation

A *collocation* is a sequence of words or terms that co-occur more often than would be expected by chance. Although not always, they usually have some special meaning associated with them. Collocations are expressions that become established through repeated context-dependent use. 'hello world', 'ethernet cable', 'stumble upon' are

some examples of bigram collocations. There are seven main types of collocations[1]: adjective+noun, noun+noun (such as collective nouns), noun+verb, verb+noun, adverb+adjective, verbs+prepositional phrase (phrasal verbs), and verb+adverb.

The traditional method of performing collocation extraction is to find a formula based on the statistical quantities of those words to calculate a score associated to every word pair. Proposed formulas are mutual information, t-test, z test, chi-squared test, and likelihood ratio, as in [4] and [26] for example.

1.1.3 Multi-word expression

A multi-word expression (MWE) is a phrase made up of a sequence of two or more words that has properties that are not predictable from the properties of the individual words or their normal mode of combination. Multi-word expressions are a key problem for the development of large-scale, linguistically sound natural language processing technology. There are various kinds of multi-word expressions (for example, idioms are MWEs), as discussed in [31], and each of them should be analyzed in distinct ways. Some examples of MWEs are 'study guide', 'yellow journalism', 'quick fix'.

[1] https://www.englishclub.com/vocabulary/collocations-samples.htm (05/19/2014)

1.2 Need for extraction and motivation

N-grams are widely used, in NLP techniques, especially for automatic extraction of words and phrases, and for text categorization based on n-gram statistics. N-grams are also used in the detection of malicious code [1]. Automatic extraction of key phrases from abstracts with the help of term frequency and n-grams was experimented with in [13]. [6] describes an n-gram-based text categorization approach that is tolerant of textual errors. N-grams are also widely used in speech recognition software and are effective in modeling language data. N-grams are also used for evaluation of summaries as explained in [20], and can also be used in designing Word-Sense Disambiguation software. An efficient n-gram extraction technique plays an important role in several of the model NLP tasks. Although there are several n-gram extraction techniques available, there are certain flaws that need to be addressed in order for them to be near perfect. The n-gram extraction technique presented in Textractor addresses those issues without compromising on the quality of the n-grams.

Collocations are important in a number of applications. They help language learners in learning the natural language, for example, "with the help of collocations mistakes like 'powerful tea' instead of 'strong tea' can be avoided" as explained in [2]. Collocations also help in understanding the social phenomenon through language. Collocations can also be used in summarizers. Most of the existing collocation extraction rely on frequency values of n-grams and domain knowledge. Textractor satisfies the need for a domain independent collocation extraction technique that

does not depend on the frequency of the collocation with respect to the document it is present in.

Multi-word expressions play an important role in NLP. MWEs exist in almost all languages and are hard to extract, as there is no algorithm that can precisely outline the structure of an MWE. MWEs are important for natural language generation, and significantly influence machine translation and semantic tagging. MWEs are also used in document indexing, information retrieval, and text summarization. Efficiently extracting MWEs significantly improves many areas of NLP. But most of the MWE extraction techniques are biased in a way that they focus on a specific domain or make use of statistical techniques alone, which results in poor performance. Textractor makes use of knowledge from WWW in deciding if a phrase is a MWE rather than solely depending on frequency measures or following rules of a specific domain.

1.3 Contribution of this thesis

As discussed in the previous section, n-gram, collocation, and MWE extraction are important tasks which call for efficient techniques. The problem with existing n-gram extraction software is that they merely classify all the items in the text into n-grams based on their order of occurrence. For example, given a sentence "Hello world. How are you?", existing techniques can return a bigram as "world ." or a trigram as "hello world ."; instead of unigram "hello" and bigram "hello world". As a result, the frequency values are not accurate which results in false outputs. This

issue is addressed by Textractor. Punctuation and sentence endings are correctly recognized by the technique in this thesis and n-grams are extracted accordingly. As a result there is no overlapping of words from multiple sentences and punctuation does not skew the frequency count.

The collocation extraction technique proposed in this thesis is domain independent and does not make use of frequency count of the candidate collocation phrase in the given input thus eliminating any chances of error that may be a result of frequencies. The existing domain dependent collocation extraction software requires patterns to be assigned by the user before collocations can be extracted. These patterns are usually parts–of–speech (POS) based as mentioned in [29]. Given a POS pattern, their software extracts the collocations based on the possible combinations and their respective frequency information. Unfortunately, these techniques result in a large number of false positives. Textractor proposes collocation extraction techniques that determines if a candidate is a collocation or not based on the knowledge from WWW, and statistical and probability techniques thus eliminating the issues caused with existing collocation extraction techniques.

This thesis proposes a MWE extraction technique. The major problem with existing software is that they are designed to identify MWEs based on their structure and language information. Textractor provides a single solution that can handle both the cases. This technique makes use of knowledge from WWW similar to the collocation extraction technique in extracting MWEs. The technique in Textractor is domain independent and can easily be extended to languages other than English.

The most important contribution of this thesis is that Textractor is the first of

its kind to extract all n-grams, collocations, and MWEs from a given text document, that is both unsupervised and domain independent.

1.4 Organization of this thesis

Chapter 2 does a survey and literature study of existing solutions, approaches, and techniques for extracting n-grams, collocations, and MWEs from a given text document. Chapter 4 gives the details of the datasets used in the experiments. Chapters 3 is the crux of the work. It describes the extraction models. Chapter 5 discusses the performance of Textractor. Chapter 6 concludes with the challenges faced and planned future work.

Chapter 2

Existing Phrase Extraction Techniques

Recently, a large number of tools and techniques have been proposed to aid the extraction of n-grams, collocations, and multi-word expressions from text corpora. The existing techniques however, are not able to deal with collocations and MWEs in an efficient way, with most of them using the term MWE for a collocation to begin with. This section is a survey on the prior work directly related to the extracting n-grams, collocations, and MWEs. The survey is categorized into sections based on the extraction category.

2.1 N-gram extraction

Identification and analysis of n-grams and collocations is supported by [4], using a software named Text-NSP. Text-NSP allow extraction of n-grams from both contiguous and non-contiguous sequences with the help of Perl regular expressions. Text-NSP allows counting n-gram frequencies and applies two n-gram filtering techniques. The first technique filters n-grams that are made entirely of stop words and the second technique filters the n-grams based on their frequency values. Text-NSP also supports association measures for n-grams like, log-likelihood ratio ("a statistical test used to compare the goodness of fit of two models"[1]), Fisher's exact test ("is used to examine the significance of association between the two kinds of classification"[2]), and Pearson's Chi-squared test ("to assess goodness of fit and tests of independence"[3]). The major issues with this software are crossovers between sentences when identifying n-grams and punctuation are considered as a term in the n-gram. For example, (Hello, World!) will be a 4-gram (Hello , world !) rather than being a bi-gram (Hello world). This causes inconsistencies in the frequencies of generated n-grams and also requires post processing to be done before using the generated n-grams. We believe that because of these inconsistencies there are many false positives when this information is used to extract collocations. N-grams in a topical model are used to index topics and topical phrases in [42].

[1] `http://en.wikipedia.org/wiki/Likelihood-ratio_test`
[2] `http://en.wikipedia.org/wiki/Fisher\%27s_exact_test`
[3] `http://en.wikipedia.org/wiki/Pearson\%27s_chi-squared_test\#Definition`

2.2 Collocation extraction

Xtract is a general purpose software proposed by [33] for collocation extraction from text using a combination of n-grams and mutual information measure, with around 80% precision when tested on a dataset of 4000 collocations randomly selected from a 10 million-word corpus of stock market news reports. Xtract defines collocation as an arbitrary and recurrent word combination. It is a lexicographic tool implemented based on statistical methods for retrieving and identifying collocations from large text corpora. The paper [33] states that the results are dependent on type and size of the corpus.

Researchers in [24] focused on general collocations. They attempt to compare a range of bigram collocation extraction techniques as well as they implement a new technique based on tests for lexical substitutability (words that can be substituted for each other in the context of class). This technique uses bigram co-occurrence frequencies obtained from 5.3 million words of the British National Corpus (BNC). Bigram co-occurrence frequencies were obtained from 5.3 million word British National Corpus (BNC) corpus and for each pair of words in this frequency data a score is assigned, where the higher the score, the better the collocation.

Researchers in [3] and [30] focused on phrasal verbs. A range of techniques for extracting English verb-particle constructions from raw text corpora based on the

output of a POS tagger, chunker, chunk grammar, and dependency parser were proposed in [3]. This resulted in an accuracy of 74.9% for intransitive verbs and 89.7% for transitive verbs when all four components are combined together into a single classifier. Methods for automatic acquisition of verb-particle constructions $VPCs$ taking into account their statistical properties and some regular patterns in combinations of verbs and particles were proposed by [30]. This technique can detect idiomatic VPCs form compositional. This technique has an overall result of 61.9% when detecting idiomatic VPCs.

Compound nouns are the focus of [14]. Their technique shows that the web can be employed to obtain frequencies for bigrams that are unseen in a given corpus. Their evaluation technique works on proving that web frequencies can reliably predict human plausibility judgements. The UCS toolkit by [9] is a collection of libraries and scripts for the statistical analysis of co-occurrence data. Datasets are annotated with association scores from a wide range of built-in measures, ranked, and sorted with UCSPerl subsystem.

A combination of lexical association measures and empirical results of several methods for automatic collocation extraction were presented in [25]. Some of the included methods are *point-wise mutual information, Pearson's Chi-squared test. z-score, odds ratio, and squared log-likelihood ratio.* Their best performing method has an accuracy of 66.49%. Centrality-based methods for identifying keywords and key phrases in collocation networks are proposed by [17].

2.3 MWE extraction

MWEs were investigated in [44] and [41]. MWEToolkit by [29] is a tool for automatic extraction of MWEs from a monolingual corpora of general purpose English. However, these techniques require linguistic information to extract MWEs from the corpora and the accuracy can further be improved. For example, MWEToolkit requires patterns of the MWEs to be detected and declared prior to the execution along with appropriate .dtd files to assist the tagger, i.e., in-order to extract Noun-Noun phrases we need to specify "<pat><w pos="NN*"/ ><w pos="NN*"/ ></pat >" in the patterns.xml file. Ramisch wrote a book on multi-word expression acquisition [28].

Multiple sources of linguistic information in the task of identifying multiword expressions in natural language texts were used by [36]. A semi-supervised approach to extracting multiword aspects of user-written reviews that belong to a given category was discussed by [38]. Various strategies to predict both syntactic dependency parsing and contiguous MWE recognition were investigated by [5]. Multiple Sequence Alignment to recognize MWEs were used by [19] and is motivated from gene recognition.

A supervised and automatic idiom and literal classification algorithm treating idioms as semantic outliers and identifying semantic shift as outlier detection was described by [27] and [10]. The latest work on MWE identification is by [32]. This

is a supervised approach for identifying MWEs, trained and tested on English web corpus with an accuracy of around 60%; however, from our perspective MWEs in [32] are essentially collocations. An experimental evaluation of the significance of phrasal verb treatment for obtaining better quality statistical machine translation results is explained by [15]. Two structured prediction models for joint parsing and multiword expression identification were worked on by [12].

Resolving ambiguity between idiomatic and non-idiomatic expressions using a connectionist model and continuation class model was explained by [37]. Work on Italian and English idioms of maritime type and the integration of idiomatic expressions in the terminological database *Mariterm* was presented in [21]. Their dataset consists of 200 Italian idioms. Whether a text instance of potentially idiomatic expression is actually used idiomatically in a given context or not is investigated by [11]. Context-based idiom detection using the IDIX corpus was explained by [34]. American National Corpus datasets for idioms was tagged by [35]. Possible architecture for a multilingual database for idioms was described by [40]. Work on automatically identifying verb-particle constructs as idioms was done by [30]. An approach for distinguishing literal and non-literal use of idiomatic expressions using a combination of supervised and unsupervised approaches was proposed by [18].

Chapter 3

Models

This chapter describes different models that make up Textractor. Before coming up with the models for Textractor, an experiment was conducted to prove that frequency information and association measures alone cannot be used in determining if an n-gram is a collocation. For this experiment a set of 3000 English sentences were extracted from the novel 'Hard Times' authored by Charles Dickens.

Steps involved are as follows:

Step-1: Any non-ascii characters are eliminated from the dataset of 3000 sentences

Step-2: N-grams are extracted from the input dataset

Step-3: Frequencies of both n-grams and individual words are obtained, and normalized to calculate the mean and standard deviation

Step-4: N-grams are partitioned into 8 ranges based on their frequencies. For example, range-1 if $frequency < mean - 3 * standarddeviation$, range-2 if $mean - 3 * standarddeviation < frequency < mean - 2 * standarddeviation$, ... , range-8 if

$frequency > mean + 3 * standarddeviation$

Step-5: Four association measures (Dice, PMI, MLE, T-score [29]) are calculated for each n-gram. Once all the scores are obtained, mean and standard deviation for each measure are calculated and n-grams are partitioned to 8 ranges as in Step-4

Step-6: Aonaware dictionary API[1] is used to obtain the gold standard

Step-7: F-score is computed as a harmonic mean of Recall and Precision. Recall, Precision, and F-score are discussed in detail in Chapter 5.

F-score for each of the approaches is as follows:

Table 3.1: F-score comparison of frequency-based and association measures-based methods

	Frequencies	Dice	PMI	MLE	T-score
Range-1					
Range-2					
Range-3					
Range-4	0.021		0.036	0.021	0.006
Range-5	0.052		0.028	0.052	0.029
Range-6	0.085		0.026	0.026	
Range-7	0.030		0.030	0.030	
Range-8	0.025	0.03	0.025	0.030	

F-score values in the Table 3.1 are observed to be very low, which demonstrates that

[1] `http://www.programmableweb.com/api/aonaware-dictionary` 05/04/2014

frequency information and association measures are not helpful in determining if an n-gram is a collocation.

Models implemented by Textractor are as follows:

Section 3.1: N-gram extraction model

Section 3.2: Collocation extraction model

Section 3.3: MWE extraction model

3.1 N-gram extraction model

This model extracts n-grams given an input text file. Two heuristics are proposed based on whether or not the POS of the input text is to be considered. Both heuristics were tested for up to a n value of 50, and we believe that the model works for the larger values of n as well. Here we overcome the problems we faced when trying to extract n-grams using the existing software[2]. N-grams are generated on a sentence basis with no overlap and punctuation is not treated as an n-gram literal, with the n-gram containing exactly n 'words'. Steps involved are as follows:

[2] `http://www.d.umn.edu/~tpederse/nsp.html` (04/17/2013)

3.1.1 Stopwords removal

Stopwords can be removed from the input if necessary. This is an option and the user can decide whether they want the stopword removed. The word list from the Natural Language Tool Kit (NTLK) corpus is used.

3.1.2 Non-ASCII character removal

Sometimes, while extracting text from different file formats, non-ascii characters might slip into the input read and cause issues by being unreadable. To eliminate this problem, all the text read form the input file is cleaned by the non-ascii characters before progressing further.

3.1.3 N-grams without POS tags

This model is designed to help the users who want to extract n-grams from an input text file for a given value of n but are not interested in the POS of the words involved. This model was not used in the experiments conducted for this paper. This can be used when a user wants to obtain n-grams alone and has no further plans of extracting collocations and MWEs.

3.1.4 N-grams with POS and word position

This model allows the user to extract n-grams from the text along with the POS of the word and the position of the word in the input file. This is useful for keeping

track of the words that might have different POS associated with them at different locations in the input file.

Algorithm 1 Algorithm: N-gram extraction

```
1: procedure N-GRAM EXTRACTION
2:     POS tag the input text
3:     for sentence i ∈ InputTextFile do
4:         if n ≤ length(sentence) then
5:             extract n–grams from sentence
6:             position(n) = position of n–gram identified by sentence number and
   word number
7:             POS(n) = parts–of–speech of n–gram
```

N-gram extraction steps can be understood with an example as follows:

Example: Hello world. How are you?

Requirement: Bigrams

Expected bigrams: {Hello world}, {How are}, {are you}

Output from N-gram Statistic Package[3]: {Hello world}, {world .}, {. How}, {How are}, {are you}, {you .}.

As we can see from the output, NSP returns a lot of junk and there is an overlap between the words from different sentences. This issue is addressed by Textractor.

[3] `http://www.d.umn.edu/~tpederse/nsp.html`

3.2 Collocation extraction model

This model extracts collocations from the input file.

Input. n-grams identified in section 3.1

There are two modules the user can choose for extracting the collocations. These two modules help us observe how the POS affects a phrase being identified as a collocation, as in most scenarios collocations are expected to be of a certain defined structure.

Collocations WITH-OUT POS restrictions. This module checks if an n-gram is a collocation irrespective of the POS of the words in the n-gram.

Collocations WITH POS restrictions. There are two approaches a user can choose here.

–An n-gram is a collocation if and only if POS of at least half of the words in the n-gram are from the set {Noun, Adjective, Verb, Adverb} and at least half the words in the n-gram are not auxiliary verbs. Observing that this model was a bit too strict on n-grams, we opted the model below in our experiments.

–An n-gram is a collocation if at least one of the words in the n-gram is from the set {Noun, Adjective, Verb, Adverb}.

Although the user can choose between WITH and WITH-OUT POS restrictions, the steps involved in the process are essentially the same. They are as follows:

3.2.1 Dictionary search

We start with the straightforward method of determining if an n-gram is a collocation, that is to check for the n-gram in a phrasal dictionary. If an n-gram is present in a phrasal dictionary, then by the definition of collocation, that n-gram is not a random co-occurrence of words, but rather is a collocation. In this paper, we used WordNet dictionary from NLTK corpus. We also implemented search using Glosbe and Aonaware dictionaries in some of our modules, but these can be used only when working with a small text corpus, as they do have limitations on the number of search requests, 3000 per month. But even when using API, these dictionaries tend to block the MAC address if repeated search requests are sent by the program. This is one of the major issues when using online dictionaries, with the other issue being that there are not many phrasal dictionaries available online that contain a good number of valid English phrases.

Algorithm 2 Algorithm: Collocation Extraction using Dictionary search

```
1: procedure COLLOCATION EXTRACTION - DICTIONARY SEARCH
2:     for n–gram in N − grams extracted do
3:         if n–gram ∈ WordNet dictionary then
4:             n–gram is a collocation
```

Example:

n-gram = forty winks

Definition of the n-gram from WordNet dictionary = sleeping for a short period of time (usually not in bed)

Since WordNet has a definition for the n-gram, we can say that it is not a random

co-occurrence of words and that 'forty winks' is a collocation.

3.2.2 Internet search

After searching the phrasal dictionaries at our disposal, we then explore the largest source of data in determining if an n-gram is a collocation, the 'Internet'. For this purpose we used Bing search API[4]. This module searches for a phrase in the web using Bing search API and retrieves the top 10 search results of the search. From each of the results retrieved, the title and url are extracted. Now, the module checks if any keyword that is synonymous to the word 'dictionary' or any dictionary is present in either the title or url. If the keywords are present the module then checks if the exact match of the n-gram being searched for is present in the url or if the stemmed keyword is present in the stemmed title (stemming is a process where words are reduced to their root form,[5] e.g., 'running' is reduced to 'run'). This is to avoid missing any keywords because it might be represented in a different form. Snowball stemmer is used to stem the words. Once a match is found, that n-gram is declared as a collocation. The two steps involved in this module ensure that the n-gram is not a random co-occurrence of words, implying that the n-gram is a collocation.

The Bing search API used in the experiments is not consistent in providing the search results as the results keep fluctuating from time to time. Access to a stable web search API will also improve the performance of Textractor.

[4] `https://datamarket.azure.com/dataset/bing/search`
[5] `http://en.wikipedia.org/wiki/Stemming`

Algorithm 3 Algorithm: Collocation Extraction using Internet search

```
1: procedure COLLOCATION EXTRACTION - INTERNET SEARCH
2:     for n–gram in N − grams extracted do
3:         Titles = Top 10 search titles which have words synonymous to 'dictionary'
4:         URLs = Top 10 search URLs whose titles meet the requirement in line3
5:         if n–gram ∈ Titles then
6:             n–gram is a collocation
7:         if n–gram ∈ URLs then
8:             n–gram is a collocation
```

Example:

n-gram = zip it

Number of times there was a match for the n-gram in the titles = 1

Number of times there was a match for the n-gram in the urls = 4

Since there was a match for the n-gram, we consider it as a collocation.

3.2.3 Internet search and statistical value

Although the Internet search module is efficient in most cases, in certain situations the top 10 results might not be sufficient to meet all the requirements of the module. Hence, another technique is employed in this module to determine if an n-gram is a collocation. This module uses Bing Search API to obtain the total number of search results returned (S_n), when searched for the n-gram. Then each of the words in the n-gram is replaced by 5 random words that are of the same parts-of-speech as the word being replaced. This is done for words whose POS is from {Noun, Adjective, Verb, Adverb} if we take POS into consideration. After each replacement, the n-gram with one of its words replaced is searched for in the web using Bing search API

and the total number of search results returned are obtained. Once all the replacements are done and search results, $\{S_{11}, S_{12}, S_{13}, S_{14}, S_{15}, S_{21}, ...S_{n5}\}$ are obtained, an average is computed as, S_{avg}. Now a statistical value, *Stat* is calculated as follows, $Total\_number\_of\_search\_results$ = Number of replacements made = Total number of search results in the search results set (can be n*5 when all the words in an n-gram are replaced)

$$S_{avg} = \frac{\sum_{i=1}^{5}\sum_{j=1}^{n} S_{ij}}{Total\_number\_of\_searches\_with\_non\_zero\_results}$$

$$Stat = 1 - \frac{S_{avg}}{S_n}$$

Here, 'Stat' can be a negative value but has an upper bound of 1.0. If 'stat' is a user-defined value between 0.5 and 1.0, we consider the n-gram to be a collocation.

Algorithm 4 Algorithm: Collocation Extraction using Internet search and Statistical value

```
1: procedure COLLOCATION EXTRACTION - INTERNET SEARCH AND STATISTICAL VALUE
2:     for n–gram in N − grams extracted do
3:         S_n = Total search results of the original phrase
4:         New phrases = replace each word with 5 words of same POS
5:         Search results = list(Total search results returned for each new phrase)
6:         S_avg = Average of the search results of new phrases
7:         Stat = 1 − S_avg / S_n
8:         if Stat > 0.5 then
9:             n–gram is a collocation
```

Example:

n-gram: the previous government. Total number of search results for the n-gram = 446000

Word 1: ‘the’. We do not choose any words for the word ‘the’ as we are taking the POS of the words into consideration here. Total number of search results keeping the word ‘the’ as it is = 446000

Word 2: ‘previous‘. Words chosen for ‘previous’ as ‘adjective’ = [‘assistant’, ‘intermediate’, ‘industrywide’, ‘ample’, ‘weird’]. Total number of search results of the phrase formed by replacing ‘previous’ with each of the chosen words = [4310, 2500, 2, 84, 28100]

Word 3: ‘government’. Words chosen for ‘government’ as ‘noun’ = [‘bellwether’, ‘debut’, ‘Uptick’, ‘buyer’, ‘bomb’]. Total number of search results of the phrase formed by replacing ‘government’ with each of the chosen words = [21, 1970, 802, 41800, 2140]

$$S_{avg} = 47975.3636$$

$$S_n = 446000$$

$$\text{Stat} = 0.892432$$

Since Stat >0.5, we conclude that the n-gram ‘the previous government’ is a collocation.

3.2.4 Internet search and probability

The latest approach we implemented in determining if an n-gram is a collocation or not. In comparison to the technique in Section 3.2.3, this technique does not use as many internet search queries and has a better recall. There are two variations of the

technique that differ in Step-6.

Method-1:

The steps involved are as follows:

Step-1: The letter 'a' is searched for using Bing search API. Since 'a' is both an letter and a word in English, the total number of search results returned will act as a baseline, we call it 'Universe of English web pages' on internet, denoted by 'U_a'.

Step-2: Each of the words in the input file are searched for using Bing Search API and the total number of search results returned are obtained, denoted by T(w_1), T(w_2), T(w_3), ... , T(w_n),

where T(w_1) = Total number of search results of the word, w_1.

Step-3: Each of the n-gram is searched for using Bing Search API as a phrase, and the total number of search results are saved, denoted by T(N)

Step-4: Probability of the n–gram, P(N) = $\frac{T(N)}{U_a}$

Step-5: Probability of the word w_1 = $\frac{T(w_1)}{U_a}$

Given an n–gram, probability of each of the words in the n–gram {P(w_1), P(w_2), P(w_3) ... P(w_n)}, and n–gram, P(N) as a whole are calculated

Step-6: If the probability of the n–gram is greater than the product of the probabilities of individual words in the n–gram, then this technique declares that n–gram as a collocation if:

$$P(N) > P(w_1) * P(w_2) * P(w_3) * ... * P(w_n)$$

Example: Method-1:

n-gram: rags to riches

Algorithm 5 Algorithm: Collocation Extraction using Internet search and Probability - Method I

```
1: procedure COLLOCATION EXTRACTION - INTERNET SEARCH AND PROBABILITY
2:     for n–gram in N − grams extracted do
3:         T(N) = Total search results of the original phrase
4:         U_a = Universe of English web pages available on the Internet
5:         P(N) = T(N)/U_a
6:         for word in n-gram do
7:             T(w_i) = Total search results returned for the word 'i'
8:             P(w_i) = Probability of the word 'i'
9:         if P(N) > P(w_1) * P(w_2) * P(w_3) * ... * P(w_n) then
10:            n–gram is a collocation
```

Universe of English web pages = U_a = 3170000000

Total search results of the n–gram = T(N) = 2430000

Total search results of the word, rags = T(w_1) = 3690000

Total search results of the word, to = T(w_2) = 4294967295

Total search results of the word, riches = T(w_3) = 16700000

Probability of the n–gram = P(N) = $\frac{T(N)}{U_a}$ = 0.000766561514196

Product of individual word probabilities = P(w_1) * P(w_2) * P(w_2) = 8.3085437642e-06

Here, since $P(N) > P(w_1)*P(w_2)*P(w_3)*...*P(w_n)$, 'rags to riches' is a collocation.

Method-2:

In this method the uniqueness of the words in the n-gram is also taken into consideration. The steps involved are same except for step-6.

Step-6: When all the words in the n-gram are unique, if the probability of the n–gram,

P(N) is greater than product of individual probabilities of the words in the n–gram divided by the factorial of 'n', then the n-gram is a collocation.

When the words in the n-gram are not unique, the product of individual word probabilities is divided by factorial of 'n' and multiplied by the product of $n_i!$, where, n_i = number of occurrences of i^{th} unique word, 'i' ranges from 1 to k, and k = total number of unique words in the n-gram. This can be represented as follows:

When all words in the n-gram are unique: n-gram is a collocation if

$$P(N) > \frac{1}{n!} \text{ P}(w_1) * \text{P}(w_2) * \text{P}(w_3) * ... * \text{P}(w_n)$$

When the words in the n-gram are not unique: n-gram is a collocation if

$$P(N) > \frac{n_1!n_2!n_3!...n_k!}{n!} \text{ P}(w_1) \text{ P}(w_2) \text{ P}(w_3) ... \text{ P}(w_n)$$

where, k = total number of unique words in the n–gram

n_k = number of occurrences of the k^{th} unique word in the n–gram

Example: Method - II

n-gram: so far so good

Universe of English web pages = U_a = 4294967295

Total search results of the n-gram as a phrase = T(N) = 49400

Probability of the n-gram = P(N) = $\frac{T(N)}{U_a}$ = 0.000787

Total search results of the word, so = T(w_1) = 657000000

Total search results of the word, far = T(w_2) = 191000000

Total search results of the word, good = T(w_3) = 787000000

Algorithm 6 Algorithm: Collocation Extraction using Internet search and Probability - Method II

```
1: procedure COLLOCATION EXTRACTION - INTERNET SEARCH AND PROBABILITY
2:     for n–gram in N − grams extracted do
3:         T(N) = Total search results of the original phrase
4:         U_a = Universe of English web pages available on the Internet
5:         P(N) = T(N) / U_a
6:         for word in n-gram do
7:             T(w_i) = Total search results returned for the word 'i'
8:             P(w_i) = Probability of the word 'i'
9:         if All the words in the n-gram are unique then
10:            if P(N) > 1/n! P(w_1) * P(w_2) * P(w_3) * ... * P(w_n) then
11:                n–gram is a collocation
12:        else
13:            if P(N) > (n_1!n_2!n_3!...n_k!)/n! P(w_1) P(w_2) P(w_3) ... P(w_n) then
14:                n–gram is a collocation
```

Now, RHS = $\frac{n_1!n_2!n_3!...n_k!}{n!}$ P(w_1) P(w_2) P(w_3) ... P(w_n) = 7.94489482619e-06

Since, $P(N) > RHS$, 'so far so good' is a collocation.

3.2.5 Internet search and mean, standard deviation

All the n-grams that reach section Internet search and Probability also pass through this module. This module is similar to the module above until the part of extracting the search results for the n-gram as whole S_n and each of the replacement phrase, S = $\{S_{11}, S_{12}, S_{13}, S_{14}, S_{15}, S_{21}, ...S_{n5}\}$.

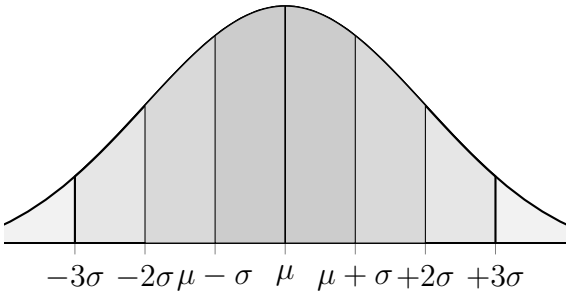

Figure 3.1: Bell-curve representing 8 ranges based on calculated mean and standard deviation

Now, mean (μ) and standard deviation (σ) are calculated for the values in S. The area under the bell-curve is categorized into 8 ranges based on μ and σ as in Figure 3.2.5. Each of the n-gram is placed into one of the 8 ranges beneath the bell-curve in Figure 3.2.5 depending on the following criteria:

$$\text{Left most range : } S_n < \mu - 3\sigma$$

$$-3\sigma \text{ to } -2\sigma : \mu - 3\sigma < S_n < \mu - 2\sigma$$

$$-2\sigma \text{ to } \mu - \sigma : \mu - 2\sigma < S_n < \mu - \sigma$$

$$\mu - \sigma \text{ to } \mu : \mu - \sigma < S_n < \mu$$

$$\mu \text{ to } \mu + \sigma : \mu < S_n < \mu + \sigma$$

$$\mu \text{ to } \mu + \sigma : \mu < S_n < \mu + \sigma$$

$$\mu + \sigma \text{ to } +2\sigma : \mu + \sigma < S_n < \mu + 2\sigma$$

$$+2\sigma \text{ to } +3\sigma : \mu + 2\sigma < S_n < \mu + 3\sigma$$

$$\text{Right most range: } \mu + 3\sigma < S_n$$

We used this module as an independent approach to compare the results obtained here to those from section 3.2.4. From our experiments we observed that the n-grams

that fall into the ranges to the right of $\mu + \sigma$ are usually collocations. An interesting observation is, 'P' value of the n-gram is always greater than or equal to 0.9 when that n-gram is in the 'right most range' of the bell-curve.

3.3 MWE extraction model

This model extracts MWEs.

Input. Collocations obtained from section 3.2.

Here we focus on the definition of MWEs, i.e., "*properties of individual words in a phrase differ from the properties of the phrase in itself*". Hence, in simple terms, we look at what individual words in a collocation mean and what the collocation means as a whole. If the meaning of collocation is different from what the individual words in the collocation try to convey, then by definition of MWEs, that collocation is a MWE. The beauty of this approach is that his is similar to how a human processes a phrase and differentiates it as being a collocation or MWE. The problem is, resources available to us do not contain all of the MWE phrases. We hope that with the advancement in technology we would be able to do a much better job of obtaining the phrase definitions in the near future.

Steps involved in the process of MWE extraction are as follows:

3.3.1 Definition extraction

This step is the most important step in determining if a collocation is a MWE. The definitions of the collocation phrase (D_p) and individual words (as per the POS if possible, sometimes a dictionary may not have definitions for a word for given POS, in which case definition of the word is obtained without taking POS into consideration) in the collocation are obtained, $\{D_{W1}, D_{W2}, ...D_{Wj}\}$. For this we use WordNet, WordNik dictionary API, and Bing search API. Here,

$D_p = \{D_1, D_2, D_3, ...D_k\}$

$D_{W1} = \{D_{11}, D_{12}, D_{13}, ...D_{1n}\}$

$D_{W2} = \{D_{21}, D_{22}, D_{23}, ...D_{2m}\}$, and so on.

3.3.2 Recreating definitions

Once we have the definitions of each words and those of the collocation phrase, each of the definition is POS tagged using the NLTK POS tagger and only the words whose POS tag is from {noun, adverb, adjective, verb} are considered (our observations of several MWEs proved that MWEs in general have at least one of the mentioned POS tags in-order for the phrase to have a meaning) and the definitions are recreated after stemming the words using the Snowball Stemmer[6] as, RD_p and $\{RD_{W1}, RD_{W2}, ...RD_{Wn}\}$ with only those words present. Here,

$RD_p = \{RD_1, RD_2, RD_3, ...RD_k\}$

$RD_{W1} = \{RD_{11}, RD_{12}, RD_{13}, ...RD_{1n}\}$

[6]http://snowball.tartarus.org/download.php

$RD_{W2} = \{RD_{21}, RD_{22}, RD_{23}, ...RD_{2m}\}$, and so on.

Now, each of the word in the original collocation is replaced with its definitions which results in a set of new phrases {P} as follows:

$P = \{RD_{11}RD_{21}...RD_{j1}, RD_{12}RD_{21}...RD_{j1}$

$, RD_{1n}RD_{2m}...RD_{jl}\}$

To avoid any confusion regarding how the procedure is implemented an example is provided below.

3.3.3 Subtraction

Each of the phrase present in {P} is subtracted from each of the recreated definition in $\{RD_p\}$.

3.3.4 MWE Result

There are two options the user can choose in deciding if a collocation is a MWE. They are:

–By Union

–By Intersection

By Union: This is a lenient way of deciding if a collocation is a MWE. Here, if at least one word survives the subtraction step above, then that collocation is declared to be a MWE.

By Intersection: This is a more stricter way of deciding if a collocation is a MWE. Here, a collocation is a MWE if and only if at least one word survives all the subtractions.

Algorithm 7 Algorithm: MWE Extraction

```
1: procedure MWE EXTRACTION
2:     for collocation in Collocations extracted do
3:         D_p = Definition of the collocation phrase
4:         RD_p = Recreated definitions of the collocation phrase
5:         for word in collocation do
6:             D_wi = Definition of the word 'i'
7:             RD_wi = Recreated definiions of the the word 'i'
8:         Recreating definition phrases, P
9:         P = {RD_11 RD_21 ... RD_j1, RD_12 RD_21 ... RD_j1, RD_1n RD_2m ... RD_jl}
10:        Subtraction. S = {RD_p} − {P}
11:        MWE result - by Union.
12:        if S is non-empty then
13:            collocation phrase is a MWE
14:        MWE result - by Intersection
15:        if At least one word survives all subtractions then
16:            collocation phrase is a MWE
```

The MWE steps can easily be understood with an example as follows:

Example - Definition extraction.

D_p = Definition of 'forty winks' = {sleeping for a short period of time (usually not in bed)}

D_{W1} = Definitions of 'forty' as a 'Noun' = {the cardinal number that is the product

of ten and four}

D_{W2} = Definitions of 'winks' as a 'Noun' = {a very short time (as the time it takes the eye to blink or the heart to beat), closing one eye quickly as a signal, a reflex that closes and opens the eyes rapidly}

Example - Recreating definitions.

RD_p = {sleep period time bed}

RD_{W1} = {number product ten}

RD_{W2} = {time time eye blink heart beat, eye signal, reflex}

P = {number product ten time time eye blink heart beat, number product ten eye signal, number product ten reflex}

Example - Subtraction.

$RD_p - P$ = {sleep period time bed} - {number product ten time time eye blink heart beat, number product ten eye signal, number product ten reflex}

= {sleep period bed, sleep period time bed, sleep period time bed}

Count of each word that after subtraction = {sleep: 3, period: 3, time: 2, bed: 3}

Example - MWE Result.

By Union: Since $RD_p - P$ is a non-empty set, the phrase 'forty winks' is a MWE

By Intersection: At least one word in $RD_p - P$ is present as many times as those of recreated definitions. Hence 'forty winks' is a MWE.

Chapter 4

Datasets

For the experiments in this thesis, we used different datasets extracted from vocabulary.com, englishclub.com, and Oxford Dictionary of Idioms. The datasets and their extraction process is explained here.

4.1 Collocation dataset

A collocation dataset is obtained from various sources as mentioned below. A set of 400 collocations were extracted from each of the sources based on their POS structure. This dataset consists of 100 Adjective+Noun collocations (obtained from[1] [2]),

[1] http://esl.about.com/od/grammarstructures/a/g_intadj_2.htm
[2] http://www.myenglishteacher.eu/blog/

100 Noun+Noun collocations (obtained from[3]), 100 Verb+Noun collocations (obtained from[4] [5] [6] [7] [8]), and 100 Verb+Preposition collocations (obtained from english-club.com[9] [10] [11]). Each of these collocations are given to Textractor to verify the performance of the Textractor when a complete sentence is not present, and also to compare the performance of Textractor with MWEToolkit which requires us to declare the POS patterns of collocations to be extracted. Performance results are presented in section– 5.2.2.3.

4.2 Idiom example sentences dataset

An idiom dataset is obtained from englishclub.com[12]. From the website, 198 idioms are randomly chosen and 198 example sentences that exemplify those 198 idioms are used. These 198 example sentences that are manually extracted serve as our dataset. Since all idioms are essentially MWEs and all MWEs are collocations, and collocations are n–grams to begin with, idioms would be the perfect choice for testing out the software in this thesis. N-grams are first extracted, followed by extracting collocations from those n-grams; and once collocations are obtained they are further

[3] http://www.vocabulary.com/lists/201117\#view=definitions\&word=cross\%20hair

[4] http://www.stgeorges.co.uk/blog/verbnouncollocationsenglishbutlersareback/

[5] http://www.bbc.co.uk/worldservice/learningenglish/grammar/learnit/learnitv351.shtml

[6] http://www.scielo.cl/pdf/signos/v45n78/a03.pdf

[7] http://www2.elc.polyu.edu.hk/CILL/eap/2004/u5/verbs\&nounspart2.htm

[8] http://www.johnsesl.com/templates/grammar/collocations.php

[9] http://www.englishclub.com/vocabulary/phrasalverbslist.htm

[10] http://www.learnenglishtoday.com/phrasalverbs/phrasalverbs_A.html

[11] http://grammar.ccc.commnet.edu/grammar/phrasals.htm (07/23/2014)

[12] https://www.englishclub.com/ref/Idioms/ (02/23/2015)

processed to extract MWEs. This dataset facilitates the evaluation of false positive rate of Textractor.

4.3 Oxford Dictionary of Idioms dataset

This dataset is a collection of idioms obtained from the Oxford dictionary of idioms. The text file consisting of 1629 idioms is our input. N-grams are the first to be extracted, then collocations from those n-grams are ext racted, and once collocations are obtained they are further processed to extract MWEs similar to the dataset mentioned in Section 4.2.

Preprocessing and Sanitization:
PDFMiner[13] was used to extract text as XML from the PDF version of Oxford Dictionary of Idioms and then a python script was used to extract idioms from the .xml file into a text file. Also, any non-ASCII characters are ignored while writing the idioms to the text file.

[13] `http://www.unixuser.org/euske/python/pdfminer/` (11/28/2014)

Chapter 5

Results

In this chapter we discuss the performance of Textractor. Comparing the performance of Textractor with existing n-gram, collocation, and MWE extraction software resulted in some interesting and encouraging observations.

Datasets used:

Dataset-1: A set of 400 collocation phrases discussed in Section- 4.1

Dataset-2: A set of 198 idiom example sentences from englishclub.com discussed in Section- 4.2

Dataset-3: A set of 1629 idioms obtained from Oxford dictionary of idioms mentioned in Section- 4.3

F-score is calculated as follows:

$$\text{F-score} = \frac{2*Recall*Precision}{Recall+Precision}\%$$

Where,

Recall = Proportion of relevant phrases(n-grams, collocations, MWEs) retrieved

Precision = Proportion of retrieved phrases that are actually relevant

Recall and Precision can be calculated as follows:

$$\text{Recall} = \frac{Relevant\_phrases\_retrieved}{Total\_\#\_of\_relevant\_phrases}$$

$$\text{Precision} = \frac{Relevant\_phrases\_retrieved}{Total\_phrases\_retrieved}$$

5.1 Performance

Textractor's collocation extraction module has a F-score of 85.4% with POS restrictions and 92.90% with-out POS restrictions when tested on Dataset-3. MWE extraction module has a F-score of 51.77% by intersection approach and 77.61% by union approach on Dataset-3. The results are encouraging and we believe that the F-score of MWE extraction module can be improved a lot if we have access to better phrasal dictionaries.

Advantages of Textractor's techniques. Textractor's unsupervised approaches will improve as more definitions are available on the World Wide Web. A second advantage of our approaches is that they can be easily generalized to languages beyond English as resources for them become available on the Internet. A third advantage is domain-independence.

5.2 Result comparison

In this section, results from Textractor are compared with the results from existing software, Text-NSP, MWEToolkit, and AMALGr.

5.2.1 N-gram extraction

5.2.1.1 Textractor vs. Text-NSP

Textractor and Text-NSP from [26] are used to obtain n-grams from the input datasets.

All the collocations in Dataset-1 range from bi-grams to 5-grams, idioms in Dataset-2 range from bi-grams to 8-grams and idioms in Dataset-3 ranges from bi-grams to 11-grams. Idioms are first separated into text files based on their n-gram count. For example, all bi-grams are placed in one file, all tri-grams in another and so on. Each of these files is given to both the software for extracting n-grams and F-score is calculated.

Performance of Textractor and Text-NSP on extracting n-grams from Dataset-1 are documented in Table 5.1.

From the results documented in Table 5.2, we observe that on Dataset-3 Textractor performed 39.39% better than NSP. The reason being, NSP reads the entire input as one large sequence rather than individual sentences resulting in an overlap

Table 5.1: N-gram extraction: Textractor vs. Text-NSP - on Dataset-1

	Textractor			Text-NSP		
	Recall	Precision	F-Score	Recall	Precision	F-Score
Adj+Noun	100	100	100	100	61.53	76.19
Noun+Noun	100	100	100	100	25.44	40.56
Verb+Noun	100	100	100	98.42	22.91	37.17
Verb+Prep	100	100	100	100	22.08	36.17

between idioms when extracting N-grams. For example, we are looking at two different idioms 'yellow journalism' and 'forty winks' to extract bi-grams, NSP extracts {(yellow journalism), (journalism forty), (forty winks)} resulting in an entirely new bigram (journalism forty) that was never intended to be in the first place. As a result, although NSP demonstrates good recall, precision is considerably low resulting in low F-score even after helping NSP by providing it with bigram idioms as test data for extracting bi-grams, trigram idioms as test data for extracting tri-grams and so on.

5.2.2 Collocation extraction

The results here are from implementing the following techniques to extract collocations:

Technique-1: This technique is a combination of dictionary search, internet search,

Table 5.2: N-gram extraction: Textractor vs. Text-NSP - on Dataset-3

n-gram value	n-gram count	Textractor F-score	Text-NSP F-score
2	138	88.99%	79.49%
3	584	100%	48.27%
4	407	96.65%	34.48%
5	245	96.27%	31.77%
6	152	96.02%	27.44%
7	72	90.77%	22.56%
8	26	90.19%	20.35%
9	3	100%	27.35%
10	1	100%	100%
11	1	100%	100%
Total	1629	95.99%	56.50%

Internet search with statistical value, and internet search and mean, standard deviation techniques discussed in the sections 3.2.1, 3.2.2, 3.2.3 and 3.2.5

Technique-2: This technique is a combination of dictionary search, Internet search, Internet search with probability, Internet search and mean, standard deviation techniques discussed in the sections 3.2.1, 3.2.2, 3.2.4, and 3.2.5

5.2.2.1 Textractor Collocation Extraction Performance

Textractor's collocation extraction module has an F-score of 85% with POS restrictions and 92.9% without POS restrictions when Technique-1 is implemented

on Dataset-3. Results are documented in the Table 5.3. Textractor's Technique-2 when implemented on Dataset-3 has a F-score of 35.32% with Model-1 and a F-score of 62.12% with Method-2. Results are documented in the Table 5.4. The probable reason for the recall of Technique-2 on Dataset-3 being low is, the probabilities of individual words are high compared to the probability of the idiom. As a result, the product of probabilities of individual words is likely to be high in most scenarios, thus declaring the idiom is not a collocation. If we observe the F-score of Method-1 and Method-2 from Technique-2, it can be observed that F-score of Method-2 is higher. The probable reason for this being, the product of individual word probabilities is multiplied by inverse of 'n!', thus reducing the value as a whole.

Textractor's collocation extraction module has a recall of 80.3% with POS restrictions when Technique-1 is implemented on Dataset-2, and recalls 82.6% and 32.68% when Method-1 and Method-2 of Technique-2 are implemented on Dataset-2. Results are documented in the Table 5.5. (Note: Although the recall is high, precision is low on this dataset, as the gold standard does not contain several collocations that were identified by Textractor. The gold standard was put together by two people who are not linguists. The gold standard is being improved by obtaining collocations from Oxford Collocation Dictionary.)

5.2.2.2 Textractor vs. NLTK

NLTK has association measures to extract bigram and trigram collocations. These association measures depend on frequency information to decide if a n-gram is a collocation. When NLTK bi-gram measures were run on 138 bi-gram idioms from

Table 5.3: Textractor: Extracting collocations from idioms. Each column represents Recall(R), Precision(P) and F-score(F) values. Total = Cumulative R, P and F of all n-grams - on Dataset-3 using Technique-1

n-grams	with POS			without POS		
	R	P	F	R	P	F
bi-grams	64.49	84.76	73.24	86.23	88.80	87.49
tri-grams	80.09	92.02	85.65	90.06	96.16	93.01
4-grams	76.41	86.14	80.98	91.40	97.78	94.48
5-grams	84.48	91.59	87.89	91.83	97.40	95.51
6-grams	92.10	96.55	94.27	90.78	96.50	93.55
7-grams	83.33	92.30	87.58	86.11	92.53	89.20
8-grams	84.61	91.66	87.99	84.61	91.66	87.99
9-grams	100	100	100	100	100	100
10-grams	100	100	100	100	100	100
11-grams	100	100	100	100	100	100
Total	80.23	90.44	85.04	90.17	95.82	92.90

Table 5.4: Textractor: Extracting collocations from idioms. Each column represents Recall(R), Precision(P) and F-score(F) values. Total = Cumulative R, P and F of all n-grams - on Dataset-3 using Technique-2

n-grams	Method-1			Method-2		
	R	P	F	R	P	F
bi-grams	18.11	92.59	30.30	23.18	91.42	36.99
tri-grams	21.06	97.61	34.64	36.81	97.28	53.41
4-grams	15.97	97.01	27.42	43.24	96.17	59.66
5-grams	23.67	100	38.38	55.91	95.80	70.61
6-grams	28.28	95.55	43.65	77.63	96.72	86.131
7-grams	33.33	92.30	48.97	70.83	91.07	79.68
8-grams	38.46	90.90	54.05	53.84	87.5	66.66
9-grams	66.66	100	80.0	100	100	100
10-grams	100	100	100	100	100	100
11-grams	100	100	100	100	100	100
Total	21.60	96.70	35.32	45.91	96.02	62.12

Table 5.5: Textractor: Extracting collocations from Dataset-2 using both Technique-1 and Technique-2

n-grams	Technique-1	Technique-2	
		Method-1	Method-2
Recall	80.3	10.03	32.68
Precision	1.6	2.45	2.93
F-score	3.13	3.94	5.39

Dataset-3, NLTK returned a total of 279 bigram collocation candidates. NLTK generated n-grams by sequentially combining the words from all 138 bigram collocations given as input. When the frequency threshold was set to 'at least two' only 4 of those candidates were declared as collocations. Whereas in reality, all 138 are collocations. Similarly, for 584 trigram idioms, 1748 collocation candidates were returned, of which 9 had a frequency greater than one.

NLTK results. Here all the n-grams that were extracted and n-grams whose frequency was greater than two are taken into consideration when looking for collocations.

Performance of NLTK on Dataset-1 is documented in Table 5.6.

Performance of NLTK on Dataset-2 is documented in Table 5.7.

Performance of NLTK on Dataset-3 is documented in Table 5.8.

NLTK experimental results prove that *frequency and association measures alone cannot be used to efficiently extract collocations* from given input text.

Table 5.6: Collocation extraction using NLTK on Dataset-1; where, f = frequency of n-gram

	f >=1			f >2		
	Recall	Precision	F-Score	Recall	Precision	F-Score
Adj+Noun	91	20.35	33.97	0	0	-
Noun+Noun	98	24.43	39.12	0	0	-
Verb+Noun	86	18.85	30.93	1	7.69	1.76
Verb+Prep	99	24.20	38.89	2	13.33	3.47

Table 5.7: Collocation extraction using NLTK on Dataset-2; where, f = frequency of n-gram

	Bi-grams		Tri-grams	
	f >= 1	f >2	f >= 1	f >2
Recall	100	0	97.61	1.19
Precision	50.64	0	32.66	100
F-Score	67.24	-	48.95	2.35

Table 5.8: Collocation extraction using NLTK on Dataset-3; where, f = frequency of n-gram

	Bi-grams		Tri-grams	
	f >= 1	f >2	f >= 1	f >2
Recall	94.87	10.25	86.90	8.3
Precision	2.0	1.7	1.6	3.4
F-Score	4.0	2.9	3.2	4.8

5.2.2.3 Textractor vs. MWEToolkit

In this section, performance of Textractor is compared with MWEToolkit from [29]. Although [29] describes MWEToolit as a MWE extraction software, their definition of a MWE aligns with our definition of collocation, hence, it is a valid comparative technique.

MWEToolkit requires POS patterns of collocations it is required to extract. For example, in-order to extract a verb+noun phrase, it requires a pattern 'VN' to be declared prior to executing the software. So, to be lenient on MWEToolkit, Dataset-1 is used as input (100 are verb+noun phrases, 100 are noun+noun phrases, 100 adjective+noun and 100 verb+prep collocation phrases). The results are documented in Table 5.9:

Inorder for MWEToolkit to generate collocation candidates, input is to be POS tagged first using TreeTagger POS tagger, which splits the input text into one POS tagged word per line. From these tagged words, potential collocation candidates

Table 5.9: Collocation extraction: Textractor Vs. MWEToolkit - on Dataset-1, Each value in the table is F-score calculated as the harmonic mean of Recall and Precision. T2-M1 = Technique-2 Method-1, T2-M2 = Technique-2 Method-2

	-	Adj+Noun	Verb+Noun	Noun+Noun	Verb+Prep
Textractor	F-score	96.18	95.12	100	100
(Technique-1)	Recall	92.47	90.71	100	100
	Precision	100	100	100	100
Textractor	F-score	13.08	36.06	7.69	13.08
(T2-M1)	Recall	7	22	4	7
	Precision	100	100	100	100
Textractor	F-score	42.51	47.32	34.71	21.42
(T2-M2)	Recall	27	31	21	12
	Precision	100	100	100	100
MWEToolkit	F-score	19.04	13.08	20.85	20.61
	Recall	97	78	92	100
	Precision	10.56	7.14	11.76	11.49

are generated based on the patterns defined. Recall and precision are calculated comparing the results with the input text files (files with 100 phrases each) in each case. Results show that MWEToolkit has low precision. The probable reason for this is, MWEToolkit generates collocation candidates from individual words and their POS information based on the POS patterns defined. This often results in collocations that were not present in the input file to begin with, thus resulting in very low precision.

5.2.3 MWE extraction

5.2.3.1 Textractor MWE extraction performance

On Dataset-3: Textractor's MWE extraction module has an F-score of 51.77% by intersection approach and 77.61% by union approach. To assess the F-score of Textractor's MWE extraction module, we consider n-grams declared as collocations when no restrictions on POS were imposed. Previously, out of 1629 idioms, 1533 were declared as collocations with an F-score of 92.0%. We ran our experiments on these 1533 collocations. The results are as follows: F-scores of idioms that were identified using 'Union' approach = 77.61%. And F-score of idioms that were identified using 'Intersection' approach = 51.77%. For this, we used WordNet dictionary to obtain definitions of 103 idioms, Oxford dictionary of idioms to obtain definitions of 176 idioms and internet search to obtain definitions of 975 idioms and the definitions for the rest of the idioms are not found. As the quality of phrasal dictionaries available is improved, the F-score of our MWE extraction module will be improved.

On Dataset-2: Textractor's MWE extraction module has an F-score of 73.25% by the union approach and 78.69% by the intersection approach. Improving the performance of the collocation extraction module further improves the performance of MWE extraction module and the collocations extracted previously are used for extracting MWEs.

Obtaining definitions from internet:

This is how relevant information regarding an idiom using internet search is obtained. Because definitions are not always immediately available, summaries are extracted from the top 10 search results (web pages) that were returned after an idiom is searched for using Bing search API. From these summaries, if the title of the webpage has one of the words {definition, wikipedia, define} then that summary is considered to be relevant to the idiom we are searching for, otherwise the summary returned is ignored. With this technique, Textractor obtained information on 975 idioms from Dataset-3.

5.2.3.2 Textractor vs. AMALGr

We compare our MWE extraction module with AMALGr from [32] as their definition of MWE aligns with our definition of a MWE. AMALGr requires SAID[1] corpus to be purchased from Linguistic Data Consortium (LDC) to train the software along

[1] `https://catalog.ldc.upenn.edu/LDC2003T10` (02/03/2015)

Table 5.10: MWE extraction: Textractor vs. AMALGr - on Dataset-3

(%)	Textractor (Union)	Textractor (Intersection)	AMALGr
Recall	63.41	34.92	23.07
Precision	100	100	21.91
F-score	77.61	51.77	22.47

Table 5.11: MWE extraction: Textractor vs. AMALGr - on Dataset-2

(%)	Textractor (Union)	Textractor (Intersection)	AMALGr
Recall	82.3	67.17	31.5
Precision	65.9	95.50	14.82
F-score	73.25	78.69	20.16

with other training data sets.

When tested on Dataset-3, out of 1629 idioms, 350 are tagged as MWEs by AMALGr (including both strong and weak MWEs as described in [32]) with Recall = 21.48%, Precision = 100%, F-score = 35.36%; whereas the F-score of Textractor is 43.26% when intersection is considered and 73.84% when union is considered.

When tested on Dataset-2, F-score of Textractor is 50% more than the F-score of AMALGr. We believe that Textractors' performance can further be improved if efficient phrasal dictionaries were available for research purporses.

Chapter 6

Conclusion

6.1 Challenges and future work

There are several issues we encountered while designing Textractor that we would like to answer. In search for a good gold standard to test the performance of Textractor, we came across several datasets only to realize that they were lacking in one respect or another. Although we decided to use idioms extracted from the 'Oxford dictionary of idioms' we see the need for a good gold standard data to be contributed for research purposes. Hence, in the future, we would like to come up with a large gold standard corpus of n-grams, collocations, and MWEs.

When it comes to the performance of the collocation extraction module, false positive rate is high when the dataset has a lot of non-trivial text which does not fall

into the category of either collocation or MWE. Hence, we are working on improving the collocation extraction module. Since MWE candidates are collocations initially extracted by Textractor, improving the performance of collocation modules improves MWE extraction module as well.

6.2 Conclusion

We have presented new unsupervised, domain-independent approaches for collocation and MWE extraction and their implementation in the Textractor suite of algorithms. Textractor is modular software that can extract n-grams, collocations, and MWEs from a given text document even when a full sentence is not present. The main contribution of our work is that, it utilizes resources that are at our disposal without any hard to understand logic. Our approaches are built on the very same logic humans use in understanding the language. The feasibility of our approaches proves that understanding the basic definition of what the terms n-grams, collocations and MWEs mean can help a program extract them efficiently. Textractor can be used to identify collocations, and MWEs that are other than idioms. But to demonstrate the efficiency of the software, we considered idioms, the toughest form of MWEs to identify, as our test dataset.

Bibliography

[1] Tony Abou-Assaleh, Nick Cercone, Vlado Keselj, and Ray Sweidan. N-gram-based detection of new malicious code. In *Computer Software and Applications Conference, 2004. COMPSAC 2004. Proceedings of the 28th Annual International*, volume 2, pages 41–42. IEEE, 2004.

[2] Cyntia Bailer and Andreana Marchi. Tell me who you go with and i'll tell you who you are - ideas for efl teachers about collocations. *BELT–Brazilian English Language Teaching Journal*, 4(1):30–45, 2014.

[3] Timothy Baldwin. Deep lexical acquisition of verb–particle constructions. *Computer Speech & Language*, 19(4):398–414, 2005.

[4] Satanjeev Banerjee and Ted Pedersen. The design, implementation, and use of the ngram statistics package. In *Computational Linguistics and Intelligent Text Processing*, pages 370–381. Springer, 2003.

[5] Marie Candito and Matthieu Constant. Strategies for contiguous multiword expression analysis and dependency parsing. In *ACL 14-The 52nd Annual Meeting of the Association for Computational Linguistics*. ACL, 2014.

[6] William B Cavnar, John M Trenkle, et al. N-gram-based text categorization. *Ann Arbor MI*, 48113(2):161–175, 1994.

[7] Ann Copestake, Fabre Lambeau, Aline Villavicencio, Francis Bond, Timothy Baldwin, Ivan Sag, and Dan Flickinger. Multiword expressions: linguistic precision and reusability. *LREC 2002 proceedings*, 2002.

[8] David A Evans and Chengxiang Zhai. Noun-phrase analysis in unrestricted text for information retrieval. In *Proceedings of the 34th Annual Meeting on Association for Computational Linguistics*, pages 17–24. Association for Computational Linguistics, 1996.

[9] S Evert. The ucs toolkit. *http://www.d.umn.edu/ tpederse/nsp.html*, 2002.

[10] Anna Feldman and Jing Peng. Automatic detection of idiomatic clauses. In *Proceedings of the 14th International Conference on Computational Linguistics and Intelligent Text Processing - Volume Part I*, CICLing'13, pages 435–446, Berlin, Heidelberg, 2013. Springer-Verlag.

[11] Fabienne Fritzinger, Marion Weller, and Ulrich Heid. A survey of idiomatic preposition-noun-verb triples on token level. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Bente Maegaard, Joseph Mariani, Jan Odijk, Stelios Piperidis, Mike Rosner, and Daniel Tapias, editors, *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta, may 2010. European Language Resources Association (ELRA).

[12] Spence Green, Marie-Catherine de Marneffe, and Christopher D Manning. Parsing models for identifying multiword expressions. *Computational Linguistics*, 39(1):195–227, 2013.

[13] Anette Hulth. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, pages 216–223. Association for Computational Linguistics, 2003.

[14] Frank Keller, Maria Lapata, and Olga Ourioupina. Using the web to overcome data sparseness. In *Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing-Volume 10*, pages 230–237. Association for Computational Linguistics, 2002.

[15] Valia Kordoni and Iliana Simova. Multiword expressions in machine translation. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Hrafn Loftsson, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, Reykjavik, Iceland, may 2014. European Language Resources Association (ELRA).

[16] Bruce Krulwich and Chad Burkey. The infofinder agent: learning user interests through heuristic phrase extraction. *IEEE Intelligent Systems*, 12(5):22–27, 1997.

[17] Shibamouli Lahiri, Sagnik Ray Choudhury, and Cornelia Caragea. Keyword and keyphrase extraction using centrality measures on collocation networks. *arXiv preprint arXiv:1401.6571*, 2014.

[18] Linlin Li and Caroline Sporleder. Classifier combination for contextual idiom detection without labelled data. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, pages 315–323. Association for Computational Linguistics, 2009.

[19] Ru Li, Lijun Zhong, and Jianyong Duan. Multiword expression recognition using multiple sequence alignment. In *Advanced Language Processing and Web Information Technology, 2008. ALPIT'08. International Conference on*, pages 133–138. IEEE, 2008.

[20] Chin-Yew Lin and Eduard Hovy. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 71–78. Association for Computational Linguistics, 2003.

[21] Rita Marinelli and Laura Cignoni. In the same boat and other idiomatic seafaring expressions. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, 2012.

[22] Makoto Nagao and Shinsuke Mori. A new method of n-gram statistics for large number of n and automatic extraction of words and phrases from large text data of japanese. In *Proceedings of the 15th Conference on Computational Linguistics-Volume 1*, pages 611–615. Association for Computational Linguistics, 1994.

[23] Chau Q Nguyen and Tuoi T Phan. An ontology-based approach for key phrase extraction. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 181–184. Association for Computational Linguistics, 2009.

[24] Darren Pearce. A comparative evaluation of collocation extraction techniques. In *Proceedings of the Third International Conference on Language Resources and Evaluation, LREC 2002, May 29-31, 2002, Las Palmas, Canary Islands, Spain*, 2002.

[25] Pavel Pecina and Pavel Schlesinger. Combining association measures for collocation extraction. In *Proceedings of the COLING/ACL on Main Conference Poster Sessions*, pages 651–658. Association for Computational Linguistics, 2006.

[26] Ted Pedersen, Satanjeev Banerjee, Bridget T McInnes, Saiyam Kohli, Mahesh Joshi, and Ying Liu. The ngram statistics package (text:: nsp): A flexible tool for identifying ngrams, collocations, and word associations. In *Proceedings of the Workshop on Multiword Expressions: from Parsing and Generation to the Real World*, pages 131–133. Association for Computational Linguistics, 2011.

[27] Jing Peng, Anna Feldman, and Ekaterina Vylomova. Classifying idiomatic and literal expressions using topic models and intensity of emotions. *Association for Computational Linguistics*, 2014.

[28] Carlos Ramisch. *Multiword Expressions Acquisition: A Generic and Open Framework*. Springer, 2014.

[29] Carlos Ramisch, Aline Villavicencio, and Christian Boitet. Multiword expressions in the wild?: the mwetoolkit comes in handy. In *Proceedings of the 23rd International Conference on Computational Linguistics: Demonstrations*, pages 57–60. Association for Computational Linguistics, 2010.

[30] Carlos Ramisch, Aline Villavicencio, Leonardo Moura, and Marco Idiart. Picking them up and figuring them out: Verb-particle constructions, noise and idiomaticity. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning*, pages 49–56. Association for Computational Linguistics, 2008.

[31] Ivan A Sag, Timothy Baldwin, Francis Bond, Ann Copestake, and Dan Flickinger. Multiword expressions: A pain in the neck for nlp. In *Computational Linguistics and Intelligent Text Processing*, pages 1–15. Springer, 2002.

[32] Nathan Schneider, Emily Danchik, Chris Dyer, and Noah A Smith. Discriminative lexical semantic segmentation with gaps: running the mwe gamut. *Transactions of the Association for Computational Linguistics*, 2:193–206, 2014.

[33] Frank Smadja. Retrieving collocations from text: Xtract. *Computational linguistics*, 19(1):143–177, 1993.

[34] Caroline Sporleder, Linlin Li, Philip Gorinski, and Xaver Koch. Idioms in context: The idix corpus. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Bente Maegaard, Joseph Mariani, Jan Odijk, Stelios Piperidis, Mike Rosner, and Daniel Tapias, editors, *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta, may 2010. European Language Resources Association (ELRA).

[35] Laura Street, Nathan Michalov, Rachel Silverstein, Michael Reynolds, Lurdes Ruela, Felicia Flowers, Angela Talucci, Priscilla Pereira, Gabriella Morgon, Samantha Siegel, Marci Barousse, Antequa Anderson, Tashom Carroll, and Anna Feldman. Like finding a needle in a haystack: Annotating the american national corpus for idiomatic expressions. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Bente Maegaard, Joseph Mariani, Jan Odijk, Stelios

Piperidis, Mike Rosner, and Daniel Tapias, editors, *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta, may 2010. European Language Resources Association (ELRA).

[36] Yulia Tsvetkov and Shuly Wintner. Identification of multiword expressions by combining multiple linguistic information sources. *Computational Linguistics*, 40(2):449–468, 2014.

[37] Erik-Jan van der Linden and Wessel Kraaij. Ambiguity resolution and the retrieval of idioms: two approaches. In *Proceedings of the 13th conference on Computational linguistics-Volume 2*, pages 245–250. Association for Computational Linguistics, 1990.

[38] Olga Vechtomova. A method for automatic extraction of multiword units representing business aspects from user reviews. *Journal of the Association for Information Science and Technology*, 65(7):1463–1477, 2014.

[39] Aline Villavicencio. The availability of verb–particle constructions in lexical resources: How much is enough? *Computer Speech & Language*, 19(4):415–432, 2005.

[40] Aline Villavicencio, Timothy Baldwin, and Benjamin Waldron. A multilingual database of idioms. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation, LREC 2004, May 26-28, 2004, Lisbon, Portugal*, 2004.

[41] Aline Villavicencio, Valia Kordoni, Yi Zhang, Marco Idiart, and Carlos Ramisch. Validation and evaluation of automatically acquired multiword expressions for grammar engineering. In *EMNLP-CoNLL*, pages 1034–1043. Citeseer, 2007.

[42] Xuerui Wang, Andrew McCallum, and Xing Wei. Topical n-grams: Phrase and topic discovery, with an application to information retrieval. In *Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on*, pages 697–702. IEEE, 2007.

[43] Wan Yu Ho, Christine Kng, Shan Wang, and Francis Bond. Identifying idioms in Chinese translations. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014)*. European Language Resources Association (ELRA), 2014.

[44] Yi Zhang, Valia Kordoni, Aline Villavicencio, and Marco Idiart. Automated multiword expression prediction for grammar engineering. In *Proceedings of*

the Workshop on Multiword Expressions: Identifying and Exploiting Underlying Properties, pages 36–44. Association for Computational Linguistics, 2006.