

Automatic Identification of Locative Expressions from Informal Text

A thesis presented
by

Fei Liu

to

The Department of Computing and Information Systems
in total fulfillment of the requirements
for the degree of
Master of Software Systems Engineering

The University of Melbourne
Melbourne, Australia

May 2013

Declaration

This is to certify that:

- (i) the thesis comprises only my original work towards the PhD except where indicated in the Preface;
- (ii) due acknowledgement has been made in the text to all other material used;
- (iii) the thesis is fewer than 100,000 words in length, exclusive of tables, maps, bibliographies and appendices.

Signed: _____

Date: _____

©2013 - Fei Liu

All rights reserved.

Thesis advisor(s)
Timothy Baldwin
Maria Vasardani

Author
Fei Liu

Automatic Identification of Locative Expressions from Informal Text

Abstract

Language technology rules, OK.

Contents

Title Page	i
Abstract	iii
Table of Contents	iv
Citations to Previously Published Work	vi
Acknowledgments	vii
Dedication	viii
1 Introduction	1
1.1 Motivation	1
1.2 Research Question	1
1.3 Contribution	1
1.4 Definition of Locative Expression	1
1.5 Scope of the Thesis	1
1.6 Structure of the Thesis	1
2 Background	2
2.1 Related Work	2
3 Resources	3
3.1 Corpus	3
3.1.1 Tell Us Where	3
3.1.2 Data Preprocessing	4
3.1.3 Manual Annotation	5
3.2 Tools	7
3.2.1 Conditional Random Fields	7
3.2.2 CRF++	8
3.2.3 Data Reinterpretation	9
3.3 Identification of Locative Expressions	10
3.4 External Resources	11
3.4.1 Gazetteers	12
3.4.2 Dictionaries	13
3.5 Evaluation Tools	13
3.5.1 StanfordNER	13

3.5.2	Unlock Text	14
4	Methodology	15
4.1	Gold Standard Setup	15
4.1.1	Feature Set	15
4.2	Automatic Identification Setup	20
4.3	Evaluation Methodology	20
4.4	Baseline Systems	20
5	Results	22
6	Conclusion	23
A	Stuff that didn't belong in the thesis	25

Citations to Previously Published Work

Large portions of Chapter 1 have appeared in the following paper:

Kim Smith (2005) LT Stuff, In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, Ann Arbor, USA, pp. 1–8.

Acknowledgments

I want to thank my Mum, and my Dad, and my Uncle Bruce, and my Aunty Gertrude, and ...

Dedicated to youse all.

Chapter 1

Introduction

1.1 Motivation

1.2 Research Question

1.3 Contribution

1.4 Definition of Locative Expression

1.5 Scope of the Thesis

1.6 Structure of the Thesis

Chapter 2

Background

2.1 Related Work

Chapter 3

Resources

In order to verify the validity of our theories and assumptions, an appropriate dataset is needed either for the model to learn from or to run experiments with. Fortunately, such data was sourced from a web-hosted location-based mobile game project - *Tell Us Where*¹. Moreover, to predict sequences of labels for a given corpus, a machine learning model, namely conditional random field, is adopted as it takes contextual information into account, which plays an essential role in the prediction process. Lastly, external resources, such as gazetteers and dictionaries, have been put into use as well with the aim of providing additional features for the model to learn from, thereby, further improving the performance of our model.

3.1 Corpus

3.1.1 Tell Us Where

Tell Us Where is a crowd-sourcing human knowledge project that collects descriptions of places. It is essentially a location-based mobile game where participants were asked to submit descriptions about their locations through a web interface via their mobile phones.

The collected data is primarily used to support academic projects aimed at discovering how people describe locations in Victoria, Australia, which may ultimately enable the development of better web searching, mapping and navigation systems, and even emergency services.

Initially, 2,221 place descriptions were collected by the game. However, duplicate data entries existed. As the primary concern lies in the qualitative data rather than quantitative, duplicates were excluded from the collection. After the elimination of duplicates, 1,858 place descriptions remain valid and will be used as the original corpus of this research.

¹<http://telluswhere.net>

An example of the raw data collected from *Tell Us Where* is presented in Table 3.1. Each line represents a place description posted to *Tell Us Where* and the highlighted one is an example that will be used throughout Chapter 3.

Table 3.1: Raw Data from *Tell Us Where*

...
@ the john cain memorial park, took ages to walk here tho
i am in north melbourne trying to find a car park xd
optus oval watching the footy
visiting caufield park and playing in the snow beats grusiem
kyaking in this little lake with my nephew
...

In this research, the corpus used for the model to learn from is the preprocessed version of the raw data (See Section 3.1.2) combined with manual annotations (See Section 3.1.3), such as granularity levels and toponym ambiguities of place descriptions within Victoria, Australia.

3.1.2 Data Preprocessing

Previous to being fed to the machine learning model, the raw data was preprocessed for the purpose of grammatical tagging and chunking. OpenNLP², an open-sourced toolkit built for the processing of natural language text, was used to automate this task. The outcome of OpenNLP is presented in Table 3.2.

As can be seen, a place description can be divided into several chunks separated by “[” and “]”. Each chunk starts with the the type of the chunk (chunk tag, e.g., “NP”, “PP” and etc.) and consists of one or more word(s). Each word is followed by a “_” and its part-of-speech tag (POS tag, e.g., “IN”, “NNP” and etc.). In some cases, a chunk neither has a chunk tag nor is surrounded by “[” and “]” (e.g., `and_CC`). Such chunks are recognised as chunks that contain only one word and have no chunk tag.

Table 3.2: Chunked Data

...
[PP @_IN] [NP the_DT john_NN cain_NN memorial_NN park_NN] ...
[NP i_PRP] [VP am_VBP] [PP in_IN] [NP north_NN melbourne_NN ...
[NP optus_NN oval_NN] [VP watching_VBG] [NP the_DT footy_NN]
[VP visiting_VBG] [NP caufield_NN park_NN] and_CC ...
[VP kyaking_VBG] [PP in_IN] [NP this_DT little_JJ lake_NN] ...
...

²<http://opennlp.apache.org/index.html>

3.1.3 Manual Annotation

The annotations were marked manually using Brat annotation tool³ by Igor Tytyk. Each place name was annotated with its granularity level and identifiability, both of which were marked with the assist of external gazetteers, namely OpenStreetMap⁴ and Google Maps⁵.

Each annotation clearly defines the boundary of a place description (See Figure 3.1), which helps us extract place descriptions. As can be seen in Figure 3.1, the coloured area is an example of a manual annotation. Manual annotations are vital to this research as they provide a means to locate place descriptions which can later be expanded to locative expressions according to the set of rules presented in Definition ??.

Figure 3.1: Manual Annotation through Brat GUI.

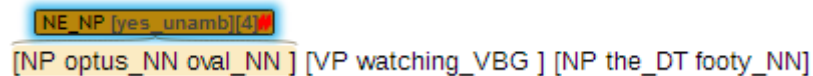


Table 3.3: Annotations of the Chunked Data

...				
#1067	AnnotatorNotes	T1517	Broadford	
T3245	NE_NP	67288 67310	[NP optus_NN oval_NN]	
A7043	normalised	T3245		
A7044	identifiability	T3245	yes_unamb	
A7045	gran_level	T3245	4	
A7046	vernNE	T3245		
#1068	AnnotatorNotes	T3245	Princes Park	
T1393	NE_NP	69593 69604	timbarra_NN	
...				

The detail information of annotations can be exported from Brat. As presented in Table 3.3, an annotation is interpreted into several compounds.

One annotation is likely to start with a row that contains the ID of the annotation (e.g., T3245), the entity type (e.g., NE_NP), the start position and end position of the description in the chunked corpus (e.g., 67288, 67310) and the text of the description (e.g., [NP optus_NN oval_NN]).

³<http://brat.nlplab.org/>

⁴<http://www.openstreetmap.org/>

⁵<http://maps.google.com/>

Table 3.4: Detail Information of Annotation

Item Name	Item Description	Example
Annotation ID	The ID of the annotation.	T3245
Entity Type	Information about whether the underlying description is a geospatial named entity (NE_NP) or a geospatial noun phrase (NP_NP). If the description is located outside of Victoria, Australia, it is marked as irrelevant with IRREL.	NE_NP or NE_NE or IRREL
Start Position	The start position of the description in the chunked corpus.	67288
End Position	The end position of the description in the chunked corpus.	67310
Annotation Text	The text of the description.	[NP optus_NN oval_NN]

Following the starting row of an annotation are: a flag of whether the annotation is normalised (e.g., A7043 normalised T3245), the identifiability of the annotation (e.g., A7044 identifiability T3245 yes_unamb), the granularity level of the annotation (e.g., A7045 gran_level T3245 4), a flag of whether the description is a vernacular/misspelt name of another place (e.g., A7046 vernNE T3245) and the canonical name/spelling of the description (e.g., #1068 AnnotatorNotes T3245 Princes Park).

Three different values can be assigned to the identifiability of an annotation as shown in Table 3.5.

Table 3.5: Possible Values of Identifiability

Identifiability	Description
yes_unamb	identifiable non-ambiguous
yes_amb	identifiable ambiguous
no	non-identifiable

The value of granularity level ranges from 1 to 7 with each value representing a specifically defined level of geospatial granularity (See Table 3.6).

With the help of manual annotations, 3,279 place descriptions were extracted. However, 218 place descriptions were marked irrelevant. Therefore, only 3,061 place descriptions were actually valid and can be used as seeds to be expanded to locative expressions.

Table 3.6: Granularity Level

Granularity Level	Description	Example
1	Furniture	<i>my bed, windows</i>
2	Room	<i>back porch, my bedroom</i>
3	Building	<i>the church, Swan St Optometrist</i>
4	Street	<i>Bell St, Tobruk Avenue</i>
5	District	<i>Templestowe, Parkville</i>
6	City	<i>Melbourne, Mornington</i>
7	Country	<i>australia, Victoria</i>

3.2 Tools

3.2.1 Conditional Random Fields

Conditional Random Fields (CRFs) are widely used for tasks of labelling sequential data. In this research, CRFs were brought in to predict the IOB-encoded label of a single word with regard to contextual information.

Feature Functions

A feature function takes the form shown in Equation 3.1 where s is the observation sequence (a sentence), l is a particular label sequence and i is the position of a word in the observation sequence s . Hence, l_i is the label of the i th word (current word) in the observation sequence s and l_{i-1} is the label of the $(i-1)$ th word (previous word).

$$f(s, i, l_i, l_{i-1}) \quad (3.1)$$

The output of a feature function is a real-valued number which is usually either 0 or 1.

Probabilities

Using a set of feature functions, the score of a label sequence l given a particular observation sequence s can be calculated as shown in Equation 3.2.

$$score(l|s) = \sum_{j=1}^m \sum_{i=1}^n \lambda_j f_j(s, i, l_i, l_{i-1}) \quad (3.2)$$

For each feature function f_j , a weight value λ_j is assigned. A large and positive λ_j suggests that the feature defined by function f_j has strong indications of the current word's label being l_i .

$$p(l|s) = \frac{\exp(score(l|s))}{\sum_{l'} \exp(score(l'|s))} \quad (3.3)$$

Learning Weights

In order to learn the optimal weight for a particular feature function, the Equation 3.4 is repeated until it reaches certain stopping conditions.

$$\lambda'_j = \lambda_j + \alpha \left[\sum_{i=1}^n f_j(s, i, l_i, l_{i-1}) - \sum_{l'} p(l'|s) \sum_{i=1}^n f_j(s, i, l'_i, l'_{i-1}) \right]$$

In Equation 3.4, λ'_j is the next weight for feature function $f_j(s, i, l_i, l_{i-1})$ while λ_j is the current weight. α is the learning rate which can be adjusted.

3.2.2 CRF++

CRF++⁶ is an open-sourced, highly-customizable implementation of the CRF model written in C++. It can be applied to a wide variety of NLP tasks thanks to its generic design which allows feature sets to be redefined. Both training and testing functions are provided and can perform their designated tasks respectively with optimised memory usage and minimum time consumption. Considering the merits mentioned above, we adopt CRF++ ~~to work~~ as our machine learning model to which we feed our data.

Training Data

An example of the format of the training data of CRF++ is presented in Table 3.7. Each line of the input file represents not only the word itself but its features as well. Sentences should be separated by empty lines. In this example, apart from the current word, one additional feature: POS tag, is listed in Table 3.7 as the second column and the last column is the correct label of the word. More features are allowed to be inserted into the feature table as long as the last column remains ~~to be~~ the correct label of the current word. The correct label column is IOB encoded with "B-NP" and "I-NP" representing the beginning and continuous part of a locative expression respectively and "O" being "outside" of a locative expression.

Testing Data

The format of the testing data is expected to follow the format of the training data. Thus, the feature set needs to remain exactly the same as the training data with the correct label column as the last column.

Based on the features of a word in the testing data and the knowledge obtained from the training data, *CRF++* predicts the sequence of labels of words and appends it as the last column.

⁶<http://crfpp.googlecode.com/svn/trunk/doc/index.html>

Table 3.7: Example Training Data for CRF++

Word	POS tag	Label
Off	IN	B-NP
Rathdowne	NNP	I-NP
St	NNP	I-NP
,	,	O
behind	IN	B-NP
the	DT	I-NP
Kent	NNP	I-NP
Hotel	NNP	I-NP
Parked	VBN	O
on	IN	B-NP
road	NN	I-NP
outside	IN	B-NP
primary	JJ	I-NP
school	NN	I-NP

The output format of *CRF++* is compatible with *CoNLL 2000* shared task, which enables the adoption of the perl script *conlleval.pl*⁷. The script is used to evaluate the performance of the learning model.

Feature Template

A feature is represented as a column in the training and testing data of *CRF++*. To make a feature visible to *CRF++*, however, a set of feature templates is required to be defined. Feature templates are defined in a similar fashion to two-dimensional coordinates (x, y) where x coordinate is the relative position to the current word and the y coordinate corresponds to the absolute position of a column which represents a feature.

Eventually, *CRF++* generates $L \times N$ features where L is the number of output classes (in this case $L = 3$ (“B-NP”, “I-NP”, “O”)) and N is the number of distinct features.

3.2.3 Data Reinterpretation

In order to use *CRF++*, as mentioned in Section 3.2.2, the input data needs to be transformed to comply with the format that *CRF++* requires. Considering ~~the fact~~ the amount of entries (1,858 entries), the transformation is non-trivial. Therefore, we adopt a programmatic approach to automate the transformation process.

⁷<http://www.cnts.ua.ac.be/conll2000/chunking/conlleval.txt>

The transformation is based on the exported data presented in Table 3.2 and Table 3.3.

In order to reinterpret the exported files to the format described in Section 3.2.2, we use ~~the algorithm described in~~ Algorithm 1 to match words with their annotations and then output to a external file one word a line.

Algorithm 1 Matching words against annotations

Precondition: The chunked corpus and detail information of annotations of the chunked corpus are already read and stored into lists *sentences* and *annotations* respectively with irrelevant annotations eliminated. Also, *annotations* list is sorted in ascending order of the start position of each annotation.

```

1 function MATCH_ANNOTATION(sentences, annotations)
2   result  $\leftarrow$  []
3   idx  $\leftarrow$  0
4   for each sentence in sentences do
5     for each chunk in sentence.chunks do
6       for each word in chunk.words do
7         if word matches annotations[idx] then
8           word  $\leftarrow$  annotations[idx]            $\triangleright$  set detail information of
annotation[idx] to word
9           idx  $\leftarrow$  idx + 1
10        end if
11        append word to result
12      end for
13    end for
14  end for
15  return result
16 end function

```

Having parsed the chunked corpus and the detail information of annotations of the chunked corpus, we are able to locate manually marked place descriptions in the chunked corpus, which enables us to move on to identifying locative expressions.

Being able to retrieve all place descriptions in the chunked corpus is an essential step to identifying locative expressions.

3.3 Identification of Locative Expressions

Since all informal place descriptions can be retrieved from the corpus using the method described in Section 3.2.3, we now try to identify locative expressions with the assist of place descriptions.

~~Since~~ a locative expression consists of at least one place description, the process of identifying locative expressions can be interpreted as the task of expanding place descriptions to locative expressions and concatenating multiple place descriptions to one locative expression.

Based on the definition of locative expression (See Section 1.4), a set of rules can be derived to identify locative expressions using place descriptions:

1. A locative expression contains at least one place description.
2. A prepositional phrase is considered a part of a locative expression if it precedes a place description.
3. Prepositional phrases, namely “of”⁸ and “and”, are considered semantic connectors that concatenate the two surrounding place descriptions, thereby, constituting a larger locative expression.
4. Punctuations, namely “,” and “s”, are considered semantic connectors that concatenate the two surrounding place descriptions, thereby, constituting a larger locative expression.

Note that it is not always true that a place description equals to a locative expression. In most cases, a locative expression consists of not only the place description itself but the prepositional phrases preceding it as well. In some cases where lexical connectors, such as “of”, “and” and “,”, are situated between two or more locative expressions, we essentially treat them as one single locative expression linked by lexical connectors rather than multiple separate ones.

Example 1. I am in my bedroom at home, on Rathmines Road, Hawthorn East.

As underlined in Example 1, three locative expressions can be retrieved while four place descriptions were annotated as highlighted. The third one, “*on Rathmines Road, Hawthorn East*” contains two place descriptions, “*Rathmines Road*” and “*Hawthorn East*”, connected by a “,”.

Ultimately, 2,757 locative expressions were identified. The number of words contained in a locative expression ranges from 1 to 22 (mean: 2.74, standard deviation: 0.18).

3.4 External Resources

Two types of external resources, gazetteers and dictionaries, are introduced as they provide unbiased data sources upon which we can improve the performance of our model.

⁸“of”s that are essentially tagged as particles (“PRT”) are excluded.

3.4.1 Gazetteers

A gazetteer provides mappings between actual place names and information about places. In this research, we use two gazetteers: GeoNames⁹ and VICNAMES¹⁰.

GeoNames

GeoNames is a geographical database with eight million place names across all countries. Its data can be accessed through various web services. Even though users are able to edit and improve GeoNames database through a Wiki interface, most data is provided by official public sources. Having said that, however, it is not guaranteed each source is of the same quality.

112,858 place names across Australia are stored in the GeoNames database. Apart from place names, other features are also included in the database, such as latitude, longitude, elevation, population, administrative subdivision etc.. In some cases, people may use various aliases while referring to the same place. To cope with such cases, alternative names are included as one feature of every entry as well.

In this research, we adopt GeoNames as an external gazetteer and extract useful information such as feature class and feature code¹¹. The general geospatial category of a place name is represented by its feature class property whereas its detailed geospatial category information is stored as the attribute feature code. Both properties can be used as features to feed to CRF++ with the aim of improving the performance of the system.

VICNAMES

VICNAMES consists of more than 200,000 places located in Victoria, Australia. A wide variety of places are included in the database ranging from landscape features (e.g., mountains and rivers) to bounded localities (e.g., suburbs, towns, cities and regions). Physical infrastructure such as roads, reserves and schools are stored as well.

Entries included in the VICNAMES database are created and maintained by the state government of Victoria. A total of 43,863 entries are included in the VICNAMES database. Compare with GeoNames the data of which is collected from a range of official public sources, the quality of the VICNAMES database is supposed to be better than GeoNames thanks to governmental maintenance and a localised coverage.

Since no web service is provided, data stored in the VICNAMES database can only be accessed by downloading ~~the data~~ and parsing it locally.

In this research, VICNAMES is used in a similar fashion to GeoNames except for feature class since it is not provided in the VICNAMES database.

⁹<http://www.geonames.org/>

¹⁰<http://services.land.vic.gov.au/vicnames/>

¹¹<http://www.geonames.org/export/codes.html>

3.4.2 Dictionaries

Of all the nouns and verbs, some, such as the underlined words shown in Example 2 and Example 3, have strong indication of representing places or are frequently used in conjunction with locative expressions. Hence, for the purpose of identifying such nouns and verbs, we employ external dictionaries.

Example 2. 103 hephman street

Example 3. visiting rocky point for the day

WordNet

*WordNet*¹² is a lexical database for English with four types of words (nouns, verbs, adjectives and adverbs) grouped hierarchically into a network structure according to their conceptual relations, which makes WordNet a perfect candidate for the task of identifying words related to locative expressions. For a word *A*, a set of hyponyms are chosen as long as the their semantic fields are included by *A*'s. For instance, *Red* and *Blue* are both hyponyms of the word *Colour*. Hence, for any word, "is-a" relationships exist between the word itself and all its hyponyms.

In this research, *Natural Language Toolkit*¹³ (nlk) is used to access *WordNet*.

3.5 Evaluation Tools

3.5.1 StanfordNER

The *Stanford Named Recognizer*¹⁴ (*StanfordNER*), developed by the Stanford Natural Language Processing Group at Stanford University, is a Java implementation of a named entity recognizer based on the conditional random field sequence model. It was developed to tackle the problem of named entity recognition by labelling sequences of words as locations, persons and organisation names or gene and protein names depending on the training data. Equipped with well-engineered features and the training data which can be downloaded from the the website¹⁵, the application takes text as input and identifies named entities in the input text, which is similar to our task of identifying locative expression. In most cases, however, it is not always true that a place description equals to a locative expression. Thus, given the uniqueness of the task, it is highly unlikely that using *StanfordNER* out-of-the-box is able to produce any competitive results.

¹²<http://wordnet.princeton.edu/>

¹³<http://nltk.org/>

¹⁴<http://nlp.stanford.edu/software/CRF-NER.shtml>

¹⁵<http://www-nlp.stanford.edu/software/CRF-NER.shtmlDownload>

Apart from the well-engineered features and the provided training data, a general implementation of linear chain Conditional Random Field sequence models is provided by StanfordNER as well, which allows us to retrain the model using our particular training data and therefore is employed as one of the baseline systems in this research.

3.5.2 Unlock Text

Unlock Text¹⁶, developed by the Language Technology group at the School of Informatics at the University of Edinburgh, is a geoparser based on GeoNames. It is able to identify possible place names from a given text. Unlock Text is used as the second baseline system.

¹⁶<http://unlock.edina.ac.uk/texts/introduction>

Chapter 4

Methodology

Having the annotated corpus and all the detailed information of each annotation at our disposal, we are able to locate locative expressions within a given corpus according to the definition of locative expression (as described in Section 3.3). We now move on to the task of automatic identification of locative expressions.

To get the best performance out of our model, two sets of features are defined for both the gold standard setup (Section 4.1), which makes use of the manual annotations, and the automatic identification setup (Section 4.2). Moreover, for each feature in both sets, a set of templates is ~~to~~ defined.


4.1 Gold Standard Setup

To help the learning model determine the sequence of labels of words, we make use of the manually annotated corpus to get the maximum performance.

4.1.1 Feature Set

Word

The text of a word is used as a feature as it provides the most basic information of a word. It is used as is without any modifications.

Apart from the text of the current word, additional information about neighbouring words is taken into account as well to help the learning model make  more informed prediction of the label of the current word.

An example is presented in Table 4.1. In this particular example, we assume “*home*” is the current word.

Assume the current word is the i th word in a sentence, and we define templates shown in Table 4.2 for this feature.


The interpretation of templates defined in Table 4.2 of Example 4.1 is presented in Table 4.3 with “*home*” being the current word .

Table 4.1: An Example of Word Feature

Word	POS Tag	Chunk Tag	Label
I	PRP	B-NP	O
am	VBP	B-VP	O
in	IN	B-PP	B-NP
my	PRP\$	B-NP	I-NP
bedroom	NN	I-NP	I-NP
at	IN	B-PP	B-NP
home	NN	B-NP	I-NP << current word
,	,	O	O
on	IN	B-PP	B-NP
Rathmines	NNP	B-NP	I-NP
Road	NNP	I-NP	I-NP
,	,	O	I-NP
Hawthorn	NNP	B-NP	I-NP
East	NNP	I-NP	I-NP
.	.	O	O

Table 4.2: Template Setup for Word Feature

Template	Description
Windows of neighbouring words	The text of the n th word ($i - 3 \leq n \leq i + 3$)
Combinations of two immediate neighbouring words	The combination of the text of the n th word and the $(n + 1)$ th word ($i - 2 \leq n \leq i + 1$)

Table 4.3: Example of Mapping between Template and Words

Template	Word
$i - 3$	<i>my</i>
$i - 2$	<i>bedroom</i>
$i - 1$	<i>at</i>
i	<i>home</i>
$i + 1$,
$i + 2$	<i>on</i>
$i + 3$	<i>Rathmines</i>
$[i - 2/i - 1]$	<i>bedroom/at</i>
$[i - 1/i]$	<i>at/home</i>
$[i/i + 1]$	<i>home/,</i>
$[i + 1/i + 2]$	<i>,/on</i>

POS Tag

In addition to the text of a word, the POS tag of a word is also used as a feature as it provides information about the grammatical role the word plays in a sentence. With such additional information, it is assumed that the learning model is able to figure out the pattern hidden in the sequence of POS tags which is somehow connected to the identification of locative expressions.

Assume the current word is the i th word in a sentence, and we define templates shown in Table 4.4 for this feature.

Table 4.4: Template Setup for POS Tag Feature

Template	Description
Windows of neighbouring POS tags	The POS tag of the n th word ($i - 2 \leq n \leq i + 2$)
Combinations of two immediate neighbouring POS tags	The combination of the POS tags of the n th word and the $(n + 1)$ th word ($i - 2 \leq n \leq i + 1$)
Combinations of three immediate neighbouring POS tags	The combination of the POS tags of the n th word and the $(n+1)$ th word and the $(n+2)$ th word ($i - 2 \leq n \leq i$)

We adopt the same example as displayed in Table 4.1. Again, we assume “*home*” is the current word, therefore the POS tag of the current is “*NN*”. Neighbouring POS tags that will be used to predict the label of “*home*” are listed in Table 4.5.

Table 4.5: Example of Mapping between Template and POS Tags

Template	POS Tag
$i - 2$	<i>NN</i>
$i - 1$	<i>IN</i>
i	<i>NN</i>
$i + 1$,
$i + 2$	<i>IN</i>
$[i - 2/i - 1]$	<i>NN/IN</i>
$[i - 1/i]$	<i>IN/NN</i>
$[i/i + 1]$	<i>NN/</i> ,
$[i + 1/i + 2]$	<i>,/IN</i>
$[i - 2/i - 1/i]$	<i>NN/IN/NN</i>
$[i - 1/i/i + 1]$	<i>IN/NN/</i> ,
$[i/i + 1/i + 2]$	<i>NN/,/IN</i>

Chunk Tag

Similar to POS tags, chunk tags also provide grammatical information about the constituents (e.g., noun groups, verb groups, prepositional groups, etc.) of a sentence.

Most place descriptions (2,232 out of 3,279 annotations) start with “[”, which is the start of a chunk, and end with “]”, which is the end of a chunk. In other words, 68.1% of the annotations are not words inside chunks but consist of undivided chunks. Therefore, chunk tags are supposed to have indications of boundaries of place descriptions. To discriminate the beginning and the rest of a chunk, IOB tags are employed. Hence, the beginning of a chunk is marked “B-” + chunk tag (e.g., “B-NP”) and words in the rest of the chunk are tagged “I-” + chunk tag (e.g., “I-NP”).

Assume the current word is the i th word in a sentence, and we define templates shown in Table 4.6 for this feature. 

Table 4.6: Template Setup for Chunk Tag Feature

Template	Description
Windows of neighbouring chunk tags	The chunk tag of the n th word ($i - 2 \leq n \leq i + 2$)
Combinations of two immediate neighbouring chunk tags	The combination of the chunk tags of the n th word and the $(n + 1)$ th word ($i - 2 \leq n \leq i + 1$)

Identifiability

The identifiability of a place description reflects the uniqueness of the place description in question within Victoria. Three possible values can be assigned as shown in Table 3.5.

To interpret identifiability as a feature, we assign the identifiability of a place description to each word within that place description. For words that are not contained by any place description, “None”s are assigned. Consequently, this feature does not only provide information about identifiabilities of words but potentially feeds knowledge on boundaries of place descriptions as well.

Example 4. I am in my bedroom at home, on Rathmines Road, Hawthorn East.

An example is displayed in Example 4 where four place descriptions were annotated (“*my bedroom*”, “*home*”, “*Rathmines Road*”, “*Hawthorn East*”) with their respective identifiabilities hovering over. Hence, the identifiability feature for the sentence presented in Example 4 is translated as shown in Table 4.7.

Assume the current word is the i th word in a sentence, and we define templates shown in Table 4.8 for this feature.

Table 4.7: An Example of Identifiability Feature

Word	Identifiability
I	None
am	None
in	None
my	no
bedroom	no
at	None
home	no
,	None
on	None
Rathmines	yes_unamb
Road	yes_unamb
,	None
Hawthorn	yes_unamb
East	yes_unamb
.	None

Table 4.8: Template Setup for Identifiability Feature

Template	Description
Windows of neighbouring identifiabilities	The identifiability of the n th word ($i - 2 \leq n \leq i + 2$)
Combinations of two immediate neighbouring identifiabilities	The combination of the identifiabilities of the n th word and the $(n + 1)$ th word ($i - 2 \leq n \leq i + 1$)

Granularity Level**Normalised****Vernacular****Preceding Prepositional Word**

According to the definition of locative expression, a locative expression may consist of not only one or multiple place description(s) but **prepositional words** as well. In fact, 1,103 out of 2757 locative expressions start with a prepositional word, not to mention locative expressions preceded by lexical connectors (e.g., “*of*”, “*and*”). As can be drawn from the statistics, prepositional words play an essential role in determining the beginnings of locative expressions. Therefore, we employ this as a feature to feed to the learning model. Moreover, since most prepositional phrases are used in conjunction with either ~~particular~~ place descriptions of particular granularity level

or identifiability, it is advisable to take the combinations of preceding prepositional words and features mentioned above into account as well.

Example 5. [ADVP Approximate_RB halfway_RB] [PP between_IN] [NP Lara_NNP and_CC Little_NNP River_NNP] [VP ...]

As displayed in Example 5, two place descriptions were annotated as underlined. The first place description “*Lara_NNP*” is preceded by a prepositional phrase (hence the chunk tag “*PP*”) and is therefore assigned “*between*” as its preceding prepositional word. The second one “*Little_NNP River_NNP*”, even though located inside the chunk, has “*and_CC*” as its predecessor which is assigned to both “*Little*” and “*River*”. For words that ~~is~~ neither annotated as place descriptions nor included in annotations not preceded by prepositional phrases, “*None*” is assigned. An example of the interpretation of this feature using Example 5 is presented in Table 4.9.

Table 4.9: An Example of Preceding Prepositional Word Feature

Word	Prepositional Word
Approximate	None
halfway	None
between	None
Lara	between
and	None
Little	and
River	and
.	None

Assume the current word is the i th word in a sentence, and we define templates shown in Table 4.10 for this feature.

Token Position

Geospatial Feature Class

4.2 Automatic Identification Setup

4.3 Evaluation Methodology

4.4 Baseline Systems

Table 4.10: Template Setup for Preceding Propositional Words Feature

Template	Description
Windows of neighbouring preceding propositional words	The preceding prepositional word of the n th word ($i - 2 \leq n \leq i + 2$)
Combinations of two immediate neighbouring preceding propositional words	The combination of the preceding prepositional word of the n th word and the $(n + 1)$ th word ($i - 2 \leq n \leq i + 1$)
Combinations of words and preceding propositional words	The combination of the text of the n th word and the preceding prepositional word of the n th word ($i - 1 \leq n \leq i + 1$)
Combinations of granularity levels and preceding propositional words	The combination of the granularity level of the n th word and the preceding prepositional word of the n th word ($i - 1 \leq n \leq i + 1$)
Combinations of identifiabilities and preceding propositional words	The combination of the identifiability of the n th word and the preceding prepositional word of the n th word ($i - 1 \leq n \leq i + 1$)

Chapter 5

Results

In order to uncover the optimal setup of features, we adopt the technique *feature ablation*.

Chapter 6

Conclusion

Bibliography

Appendix A

Stuff that didn't belong in the thesis

In this Appendix, I dump a whole bunch of stuff that didn't quite fit into the thesis proper.